Kontext-abhängige Personalisierung multimedialer Inhalte auf mobilen Endgeräten

Dissertation

eingereicht an der
Fakultät für Mathematik, Informatik und Statistik
der
Ludwig-Maximilians-Universität München

vorgelegt von

Diana Weiß

am 22. Juni 2009

Kontext-abhängige Personalisierung multimedialer Inhalte auf mobilen Endgeräten

Dissertation

eingereicht an der
Fakultät für Mathematik, Informatik und Statistik
der
Ludwig-Maximilians-Universität München

vorgelegt von

Diana Weiß

Datum der Einreichung: 22. Juni 2009
Datum des Rigorosums: 31. August 2009

1. Berichterstatter: **Prof. Dr. Claudia Linnhoff-Popien**,
Ludwig-Maximilians-Universität München
2. Berichterstatter: **Prof. Dr. Lars Wolf**,
Technische Universität Braunschweig

Danksagung

Diese Arbeit entstand im Rahmen meiner Tätigkeit als wissenschaftliche Mitarbeiterin am Lehrstuhl für Mobile und Verteilte Systeme des Instituts für Informatik an der Ludwig-Maximilians-Universität München. Viele Leute unterstützten mich fachlich, organisatorisch oder motivierend während des Entstehungsprozesses. All diesen Menschen möchte ich hiermit meinen herzlichen Dank ausdrücken.

Zunächst danke ich Frau Prof. Dr. Claudia Linnhoff-Popien. Sie ließ mir viele Freiheiten bei der Wahl und Bearbeitung des Themas und hatte stets ein offenes Ohr für meine Fragen und Probleme. Insbesondere bin ich ihr dankbar, dass sie neben der fachlichen Betreuung so engagiert meine persönliche Zeitplanung unterstützte. Dank gebührt auch Herrn Prof. Dr. Lars Wolf, der sich bereit erklärt hat, diese Arbeit als Zweitgutachter zu betreuen.

Für viele inspirierende Gespräche im Arbeitsalltag und bei den Teamworkshops, für die konstruktive Zusammenarbeit und die Hilfsbereitschaft bedanke ich mich bei meinen derzeitigen und ehemaligen Kollegen. Danke sagen möchte ich auch für die Unterstützung aller am Gang und natürlich für den vielen Spaß, den wir immer hatten.

Ein besonderer Dank geht an Caroline und Iris, die mir immer mit Rat zur Seite standen und für den nötigen Ausgleich sorgten. Nicht vergessen darf ich an dieser Stelle Bettina, mit der ich vermutlich Hunderte von Kilometern im Englischen Garten gelaufen bin, um den Kopf wieder frei zu bekommen.

Eine Doktorarbeit kann nicht ohne entsprechende Unterstützung im privaten Umfeld entstehen. Ich danke meinen Freunden für die Geduld und das Verständnis während dieser Zeit. Meinen Eltern und meinem Bruder möchte ich für ihre jahrelange Unterstützung und Motivation danken. Meinem Verlobten Gregor danke ich für seine Wärme und Liebe, die mich in dieser Zeit immer auffingen.

München im September 2009

Zusammenfassung

Die Bedeutung multimedialer Dienste hat in den letzten Jahren beträchtlich zugenommen. Ihre Nutzung ist heutzutage nicht mehr nur auf stationäre Geräte beschränkt: durch bessere mobile Endgeräte und leistungsfähigere Netze können diese Dienste immer mehr auch unterwegs eingesetzt werden. Gerade im mobilen Umfeld kämpfen sie aber mit zwei grundlegenden Problemen: zum einen ist es für den Nutzer schwierig, aus der riesigen Menge der Inhalte diejenigen zu finden, die für ihn wirklich relevant sind. Dieses Problem tritt auch bei stationärer Nutzung auf, die schlechteren Eingabemöglichkeiten und die geringere Bandbreite mobiler Endgeräte schränken aber gerade hier die Nutzung massiv ein. Zum anderen zeichnen sich mobile Geräte durch eine starke Heterogenität aus. Um multimediale Inhalte komfortabel nutzen zu können, muss deren optimale Darstellung für die unterschiedlichen Charakteristika dieser heterogenen Endgeräte gefunden werden. Eine Lösung für diese beiden Probleme ist die Personalisierung multimedialer Inhalte.

Im mobilen Bereich findet die Nutzung multimedialer Inhalte in einem wesentlich dynamischeren Kontext statt. Der Nutzer kann sich räumlich bewegen, die Umgebungslautstärke und die Lichtverhältnisse ändern sich häufig, und er ist der Witterung ausgesetzt. Wie diese Informationen genutzt werden, um die Personalisierung multimedialer Inhalte zu unterstützen, soll im Rahmen dieser Arbeit näher untersucht werden. Dafür wurde die *Multimedia Adaptation and Selection Language* (MASL) zur Beschreibung von Inhalten und Nutzern entwickelt. Die Informationen, die mit dieser Sprache erfasst werden, werden durch Nutzereingabe (explizit) oder automatisch (implizit) gewonnen. Exemplarisch wird dies in der vorliegenden Arbeit mit der expliziten und impliziten Gewinnung von Schlüsselwörtern zu Inhalten und Nutzerbewertungen gezeigt. Die Beschreibungen in MASL werden verwendet, um multimediale Inhalte Kontext-abhängig auszuwählen und die gewählten Inhalte in der Darstellung an den aktuellen Nutzungskontext anzupassen. Für die Auswahl werden Empfehlungssysteme eingesetzt, die Inhalte gemäß den aktuellen Kontextinformationen des Nutzers selektieren. Die entwickelten Ansätze werden in ein einheitliches Framework integriert, das flexibel konfigurierbar ist. Für die Anpassung der Darstellung wird eine Middleware entwickelt, die verteilt arbeitet: der Server-seitige Teil führt eine Voranpassung der Inhalte durch, der Client beendet den Anpassungsprozess.

Abstract

In recent years, multimedia services have become more and more important. Nowadays, their usage is not restricted to stationary devices anymore: with the upcoming of better mobile devices and high-capacity networks, these services can also be used while on the move. Especially in the mobile domain, they are facing two basic problems: on the one hand, it is difficult for the user to find relevant content out of the huge amount of content available. This problem influences stationary usage as well, but with the bad input capabilities and smaller bandwidth of mobile devices, their usage in the mobile environment is especially limited. On the other hand, mobile devices are much more heterogeneous. To easily use multimedia content, its presentation has to be optimized with respect to the different characteristics of these heterogeneous devices. A possible solution for these problems is the personalization of multimedia content.

In a mobile environment, multimedia is used in a much more dynamic context. The user can move to different places, the sound level and lighting conditions of the user's immediate vicinity are changing and she is exposed to the weather. This thesis examines how this additional information can be used to support personalization of multimedia content. For this, the *Multimedia Adaptation and Selection Language* (MASL) was developed which allows detailed descriptions of content as well as of users. The information contained in this language is derived by user input (explicitly) or automatically (implicitly). As an example, this work shows the explicit and implicit retrieval of key words of content and user ratings. These descriptions are used to select available multimedia content with respect to context and to adapt its presentation to the current context of use. For selection, recommendation systems are used that filter content according to the user's current context information. The developed approaches are integrated into a uniform framework that can flexibly be configured. For adaptation, a distributed middleware is developed: server-sided the content is pre-adapted; the client completes the adaptation process.

Inhaltsverzeichnis

Kapitel 1.

Einleitung

In der heutigen Zeit ist die Nutzung von Mobilfunk und mobilen Diensten nicht mehr aus dem Alltag wegzudenken. In den letzten 15 Jahren ging die Entwicklung von einem Nischenprodukt zu einem breiten Massenmarkt, dem man sich kaum noch entziehen kann. Durch die Verfügbarkeit von immer besseren mobilen Endgeräten und Funknetzen wird sich dieser Trend in Zukunft auch weiterhin fortsetzen.

Der ursprüngliche Markt für Sprachdienste erlangte schon vor einiger Zeit eine Sättigung. Deswegen bemühen sich die Mobilfunkanbieter immer stärker, auf Basis der vorhandenen Übertragungstechnologien zusätzliche Dienste und Anwendungen zu entwickeln, die gewinnträchtig sind. Analog zur stationären Nutzung multimedialer Inhalte haben sich zunehmend mobile Multimediadienste entwickelt. Angefangen von noch relativ einfachen Diensten wie dem *Mobile Multimedia Messaging Service* (MMS), ging die Entwicklung über *Mobile TV* und *Mobile Internet* beständig weiter. Durch immer leistungsfähigere Endgeräte mit Multimediaunterstützung wird diese Entwicklung zusätzlich vorangetrieben [73].

Die Nutzung multimedialer Inhalte bringt grundlegende Probleme mit sich. Durch die wachsende Bedeutung multimedialer Dienste, wächst die Anzahl der verfügbaren Inhalte zusehends. Laut der Studie „The Expanding Digital Universe: A Forecast of Worldwide Information Growth Through 2010“ [67] entsprach die Menge der vorhandenen digitalen Inhalte bereits 2006 ungefähr dem Dreimillionenfachen der Menge an Informationen aus allen Büchern, die jemals geschrieben wurden. Glaubt man den Hochrechnungen der Studie, wird sich diese Datenmenge bis 2010 noch versechsfachen. Durch diese große Menge an digitalen Informationen wächst der Aufwand, den Nutzer investieren müssen, um relevante Inhalte zu finden. Dies führt einerseits zu einer sinkenden Nutzerzufriedenheit, hat aber auch monetäre Konsequenzen. Laut der Studie kostet es ein Unternehmen mit 1000 Angestellten circa 5,3 Millionen Dollar jährlich, wenn Informationen nicht gefunden werden. Ein großer Teil dieser Inhalte sind die hier betrachteten multimedialen Inhalte [141, 37].

Ein zusätzliches Problem multimedialer Inhalte tritt bei mobiler Nutzung auf: mobile Endgeräte zeichnen sich durch eine starke Heterogenität aus. Diese bezieht sich auf

die verfügbare Konnektivität, die Bandbreite oder auf die Ausgabemöglichkeiten wie die Bildschirmgröße oder die vorhandenen Lautsprecher. Um die Nutzung dieser Inhalte dennoch komfortabel zu gestalten, muss die optimale Darstellung der Inhalte für die unterschiedlichen Endgeräte gewährleistet werden. Die oben erwähnte Studie fand heraus, dass das Unternehmen mit 1000 Angestellten jährlich weitere 5,7 Millionen Dollar in die Anpassung der Inhalte für bestimmte Anwendungen investieren muss.

Eine Lösung dieser beiden Probleme ist die Personalisierung multimedialer Inhalte und Dienste. Personalisierung ist ein in vielen Bereichen eingesetztes Mittel, um die Zufriedenheit des Nutzers zu erhöhen und seinen Aufwand zu senken. Sie basiert meist auf statischen Informationen wie Nutzerpräferenzen, demographischen Daten, dem Vergleich der Nutzer untereinander und der Bestimmung von ähnlichen Nutzern, oder auch auf den statischen Eigenschaften der Endgeräte. Gerade aber bei mobiler Nutzung ist der Kontext des Nutzers sehr dynamisch: er kann sich räumlich bewegen, ist unterschiedlicher Witterung ausgesetzt, die Umgebungslautstärke oder die Lichtverhältnisse ändern sich häufig. Im Rahmen dieser Arbeit wird untersucht, wie man den Vorgang der Personalisierung multimedialer Inhalte für mobile Endgeräte mit Hilfe dieser dynamischen Kontextinformationen unterstützen kann. Um dem Nutzer zu helfen für ihn interessante Inhalte zu finden, wird eine personalisierte Auswahl der Inhalte vorgenommen. Dies wird mit Hilfe von Kontext-abhängigen Empfehlungssystemen umgesetzt. Die optimale Darstellung der Inhalte findet man durch eine personalisierte Anpassung der Darstellung der ausgewählten Inhalte, die Kontext-abhängig und automatisch durchgeführt wird.

Die vorliegende Arbeit ist wie folgt gegliedert:

Zunächst werden in Kapitel 2 die für diese Arbeit grundlegenden Begriffe Multimedia, Kontext und Personalisierung erläutert. In diesem Zusammenhang werden die Dimensionen der Personalisierung untersucht und die Konzepte der Arbeit eingeordnet. Charakteristische Szenarien für die Kontext-abhängige Personalisierung multimedialer Inhalte werden untersucht und daraus ein Anforderungskatalog erstellt. Darauf basierend werden die in dieser Arbeit zu betrachtenden Problemstellungen abgeleitet.

Eine Grundannahme dieser Arbeit ist, dass für Nutzer und Inhalte passende Beschreibungen vorliegen. Zu diesem Zweck wird in Kapitel 3 die *Multimedia Adaptation and Selection Language* (MASL) definiert. Diese erlaubt u. a. die Beschreibung generischer Kontextinformationen nach dem ASC-Model von Strang et al. [159], Nutzerkommentare und erweiterte Nutzerpräferenzen bezüglich der personalisierten Auswahl und der Anpassung der Darstellung.

Wie diese Beschreibungen mit Informationen gefüllt werden, wird in Kapitel 4 diskutiert. Man unterscheidet zwischen Verfahren, die eine Nutzerinteraktion benötigen, und automatischen Verfahren. Exemplarisch für zwei Elemente aus dem gesamten Sprachschatz von MASL werden Konzepte zur Gewinnung von Schlüsselwörtern und von Nutzerbewertungen diskutiert.

In Kapitel 5 wird die Kontext-abhängige Auswahl der Inhalte mit Hilfe von Empfehlungssystemen untersucht. Es wird diskutiert, wie man Kontextinformationen im Emp-

fehlungsprozess berücksichtigen kann. Das Konzept beruht auf dem Lokal-Global-Prinzip, das die Gesamtähnlichkeit zweier Objekte auf lokale Ähnlichkeiten zurückführt und die Teilergebnisse zu einem Gesamtergebnis aggregiert. Es werden Verfahren zur Ähnlichkeitsmessung von einzelnen Kontextinformationen näher erläutert. Zusätzlich wird noch das Nutzerprofil-basierte Empfehlen als ein Verfahren vorgestellt, das ein häufiges Problem von Empfehlungsverfahren behebt, nämlich den *Curse of Dimensionality*, also den exponentiellen Anstieg des Volumens von mathematischen Räumen beim Hinzufügen von Attributen. Ein weiterer Ansatz ermöglicht strukturerhaltende Empfehlungen auf Datenhierarchien wie sie durch Inhalte und ihnen zugeordnete Nutzerkommentare entstehen. Alle vorgestellten Ansätze werden in ein generisches und konfigurierbares Framework integriert.

Die personalisierte Anpassung der Inhalte wird in Kapitel 6 näher untersucht. Kern des Konzepts bildet eine Middleware, die die Anpassung der Inhalte gemäß Gerätecharakteristika, Nutzerpräferenzen und dynamischer Kontextinformationen unterstützt. Diese erlaubt eine verteilte Anpassung: der Server nimmt eine Vorverarbeitung der Inhalte vor, der Client beendet den gesamten Anpassungsprozess. Um zu bestimmen, wo welche Anpassung durchgeführt werden soll, wird vom Client ein Anpassungsplan erzeugt. Bei der verteilten Anpassung kann jedoch das Problem auftreten, dass der Client einen Plan erzeugt, der vom Server nicht durch geeignete Methoden unterstützt wird. Deswegen werden bei der initialen Anmeldung des Clients die vom Server unterstützten Methoden übertragen und zusätzlich bei der Planerstellung berücksichtigt.

Die Anwendbarkeit der entwickelten Konzepte wird in Kapitel 7 anhand von prototypischen Implementierungen gezeigt. Zusätzlich wird eine abschließende Bewertung basierend auf dem in Kapitel 2 entwickelten Anforderungskatalog durchgeführt.

Kapitel 8 fasst die Arbeit zusammen und gibt einen Ausblick auf weiterführende Themen, die Gegenstand zukünftiger Untersuchungen sein können.

Kapitel 2.

Grundlagen

Im folgenden Kapitel werden zunächst die grundlegenden Begriffe erläutert, die im Rahmen dieser Arbeit relevant sind. Sie dienen als Basis für die weiteren Arbeiten und schaffen ein einheitliches Verständnis. Anschließend wird eine detaillierte Analyse von Szenarien durchgeführt, aus denen ein Rollenmodell und Anforderungen für die Kontextabhängige Personalisierung multimedialer Inhalte abgeleitet werden. Dieser Anforderungskatalog dient als Grundlage für die Konzepte, die in den anschließenden Kapiteln erarbeitet werden.

2.1. Begriffsdefinitionen

Im folgenden Abschnitt werden die wichtigsten Begriffe für das Verständnis dieser Arbeit definiert und eindeutig festgelegt.

2.1.1. Multimedia

Recherchiert man die Definition des Begriffes *Multimedia* in der Literatur, so erhält man viele unterschiedliche Erklärungen.

In seinem Buch *Multimediatechnologien* [158] versteht Ralf Steinmetz unter dem Begriff *Multimedia im engeren Sinne* die Kombination von mindestens einem diskreten, also zeitunabhängigen und einem kontinuierlichen, also zeitabhängigen Medium, während *Multimedia im weiteren Sinne* die parallele Verarbeitung von mehreren Medien beinhaltet.

Issing und Klimsa dagegen assoziieren mit dem Begriff in ihrem Buch *Information und Lernen mit Multimedia und Internet* [90] Hardware- und Softwaretechnologien für die Integration von digitalen Medien. Sie gehen auch näher auf Integrations- und Präsentationsaspekte ein und betonen die wichtige Rolle von Interaktivität, Multitasking und Parallelität (bezogen auf die Medienpräsentation) in Zusammenhang mit Multimedia.

Henning geht im *Taschenbuch Multimedia* [77] bei seiner Definition von zwei grundlegenden Aspekten von Multimedia aus. Zum einen ist für ihn eine Trennung der Infor-

mation und ihrer Darstellung wichtig, also auch, dass Multimediainhalte in abstrakter, digitaler Form vorliegen. Zum anderen ermöglicht diese Abstraktion eine Vernetzung der Informationsquellen und die Aufbewahrung an räumlich getrennten Orten.

Diese Definitionen greifen nur einen speziellen Aspekt von Multimedia heraus und liefern keine allgemeine Definition des Begriffes. Um für den Rahmen dieser Arbeit eine passende Definition zu finden, werden zunächst die beiden Bestandteile des Worts und ihre Bedeutung im Lateinischen betrachtet: *multi*: viel, zahlreich und *medium*: der Mittelpunkt, das Vermittelnde, aber auch im übertragenen Sinne ein Mittel zur Weitergabe oder Verbreitung von Informationen [119].

Dies kann man so interpretieren, dass ein Multimediasystem nicht nur einen einzelnen Medientyp wie z. B. Video berücksichtigt, sondern vielmehr eine Vielzahl von verschiedenen und auch unterschiedlichen Medientypen. Der Begriff Medientyp wird in dieser Arbeit wie folgt definiert:

Definition 1 (Medientyp) *Ein Medientyp beschreibt die Art, in der ein digitaler Inhalt vorliegen kann.*

Man unterscheidet zwischen diskreten digitalen Medientypen wie Fotografie, Text oder Grafik, die keine zeitlichen Abhängigkeiten haben, und kontinuierlichen digitalen Medientypen wie Audio, Video oder Animation, die einen zeitlichen Verlauf besitzen. Eine weitere Einteilung unterscheidet zwischen visuellen Medientypen, die über den Gesichtssinn erfasst werden, und auditiven, die über das Gehör wahrgenommen werden. Es ergibt sich die folgende Definition für Multimediasysteme:

Definition 2 (Multimediasystem) *Unter einem Multimediasystem versteht man ein Computersystem, das nicht auf einen Medientyp beschränkt ist, sondern unterschiedliche Medientypen und Kombinationen von Medientypen verarbeiten kann.*

Im Gegensatz dazu stehen z. B. reine Video-, Audio- oder Bildsysteme, die sich eben nur auf einen Medientyp beziehen. Unter Betrachtung obiger Definitionen wird Multimedia wie folgt definiert:

Definition 3 (Multimedia) *Multimedia ist der Überbegriff für kontinuierliche digitale Medientypen und diskrete digitale Medientypen sowie für beliebige Kombinationen aus diesen.*

Hardwareaspekte, die oft in engem Zusammenhang mit dem Begriff Multimedia gebracht werden, werden im Rahmen dieser Arbeit nicht berücksichtigt.

2.1.2. Kontext

Der Begriff der *Kontextsensitivität* (*Context-awareness*) ist erstmalig von Schilit und Theimer 1994 eingeführt worden [150]. In dieser Veröffentlichung wird Kontext als der Ort, die Identitäten von in der Nähe befindlichen Personen und Objekten und Veränderungen dieser Objekte bezeichnet. Auch andere versuchen den Begriff zu definieren, in dem sie Beispiele für Kontext anführen [42, 125]. Eine andere Herangehensweise ist es, den Begriff über Synonyme genauer zu erfassen. Oft werden dabei Begriffe wie Umgebung (environment) oder Situation (situation) verwendet [25, 179, 63]. Genauso wie die Definition über Beispiele kann diese Art der Definition zwar ein Gefühl dafür vermitteln, was mit dem Begriff Kontext gemeint ist, sie ist aber schlecht in der Praxis zu verwenden.

Deswegen wurde von verschiedenen Forschern der Versuch unternommen, allgemeine Definitionen für Kontext zu finden. Oft enthalten diese eine Kategorisierung von Kontextinformationen, die für die Überprüfung verwendet werden kann, ob eine Information auch wirklich eine Kontextinformation ist. Die wohl anerkannteste Definition ist die von Dey et al. [45, 43, 44]: 'Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.'[1]

Crowley et al. [38] nehmen an, dass Nutzer normalerweise ergebnisorientiert arbeiten und die Mehrheit der Nutzeraktionen mit einer Anzahl von Zielen in Zusammenhang stehen. In diesem Zusammenhang wird auch der Begriff der Aktivität eingeführt. Bei der Definition des Begriffes Kontext wird zwischen Nutzer- und Systemkontext unterschieden. Die Begriffe sind wie folgt definiert: 'The system's context is composed of a model of the user's context plus a model of its own internal context. The system's model of the user's context provides the means to determine what to observe and how to interpret the observations. The system's model of its own context provides a means to compose the federation of components that observe the user's context.'[2]

Auch im Rahmen des MUSIC-Projekts[3] wurde versucht, Kontext näher zu definieren: '[...] we use context to denote the circumstances and conditions under which services provided by applications and other software systems (e.g. middleware) are being used.'[4]

[1] Kontext ist jegliche Information, die zur Charakterisierung der Situation einer Entität verwendet werden kann. Eine Entität ist eine Person, ein Ort oder ein Objekt, das als relevant für die Interaktion zwischen Nutzer und Applikation, inklusive Nutzer und Applikation selbst, betrachtet wird.

[2] Der Systemkontext ist zusammengesetzt aus einem Modell des Nutzerkontexts und einem Modell seines eigenen internen Kontexts. Das Systemmodell des Nutzerkontexts stellt dabei ein Mittel zur Verfügung, um zu bestimmen, was beobachtet werden soll und wie diese Beobachtungen interpretiert werden können. Das Systemmodell des eigenen Kontexts liefert ein Mittel zur Komposition des Zusammenschlusses von Komponenten, die den Nutzerkontext überwachen.

[3] www.ist-music.eu

[4] ... wir verwenden den Begriff 'Kontext' um die Umstände und Bedingungen anzugeben, unter denen Dienste, die durch Applikationen und andere Software Systeme angeboten werden, verwendet werden.

Weiterhin liefern sie eine Kategorisierung von Kontext in die drei Kategorien Nutzerkontext (bezieht sich auf den Nutzer eines Diensts), Systemkontext (umfasst die Eigenschaften der Ausführungsumgebung der Applikation) und Umgebungskontext (repräsentiert Kontextinformationen, die die Umgebung eines Objekts betreffen, wie das Wetter oder den Ort).

Im Rahmen dieser Arbeit wird zwischen den beiden Begriffen Kontextinformation und Kontext unterschieden. Basierend auf der Definition von Dey wird der Begriff Kontextinformation wie folgt festgelegt:

Definition 4 (Kontextinformation) *Eine Kontextinformation ist eine Information, die zur Charakterisierung einer Entität oder deren Situation verwendet werden kann.*

Kontext wird darauf aufbauend wie folgt definiert:

Definition 5 (Kontext) *Kontext ist die Menge aller für die Entität relevanten Kontextinformationen.*

Kontext kann eingeteilt werden in statischen Kontext, der Kontextinformationen beinhaltet, die sich z. B. während einer Dienstausführung wenig oder gar nicht ändern (z. B. demographische Daten), und dynamischen Kontext, der Kontextinformationen beinhaltet, die sich oft ändern können (z. B. der Ort bei mobiler Nutzung).

2.1.3. Personalisierung

Der Begriff der Personalisierung hat je nach Fachgebiet viele Bedeutungen. Im Bereich der Geschichtsdidaktik z. B. bezeichnet man damit die Darstellung von geschichtlichen Ereignissen am Beispiel von historischen Persönlichkeiten [12]. Als Personalisierung bezeichnet man auch eine Strategie des Wissensmanagements, die den Schwerpunkt auf persönlichen Kontakt legt und die jeweiligen Individuen eines Unternehmens als Wissensträger in den Vordergrund stellt [72].

Doug Riecken betrachtet in [142] das Phänomen von einem marketingorientierten Standpunkt: 'personalization is about building customer loyalty by building a meaningful one-to-one relationship; [...] it is about the mapping and satisfying of a user's/customer's goal in a specific context with a service's/business's goal in its respective context.'[5].

Einen ähnlichen Blickwinkel auf das Thema hat auch Jeff Bezos, Gründer des US-amerikanischen Unternehmens amazon.com. Dieser sieht den Begriff als Synonym für Eins-zu-Eins-Marketing: 'If I have 3 million customers on the web, I should have 3 million

[5] Bei der Personalisierung geht es darum, die Loyalität des Kunden aufzubauen, indem man eine ernst zu nehmende eins-zu-eins Beziehung schafft; [...] es geht um die Abbildung und das Zufriedenstellen eines Nutzer- bzw. Kundenzieles in einem speziellen Umfeld durch ein Dienst- bzw. Geschäftsziel in seinem entsprechenden Umfeld.

stores on the Web'[6]. Dies wird durch die Strategie realisiert, die der Online-Versandhandel verfolgt, nämlich einzeln auf jeden Kunden und seine Präferenzen einzugehen.

Aus technischer Sicht wird der Begriff häufig im Web im Zusammenhang mit Nutzerschnittstellen verwendet. Er umfasst einerseits die Identifizierung eines einzelnen Nutzers und die Anpassung an diesen, andererseits kann aber auch die Anpassung an eine Nutzergruppe als Personalisierung verstanden werden.

Andere Formen der Personalisierung im Web beschäftigen sich mit der Anpassung von grafischen Oberflächen oder akustischen Hinweisen durch so genannte Themes, die helfen sollen eine individuelle Umgebung zu schaffen. Dies dient vor allem dem Zweck [161] 'to reduce the complexity and abundance of the information and service offers available on the internet to an appropriate level for the individual user'[7].

Niederée untersucht in [127] die Auswirkungen von Personalisierung im Zusammenhang mit digitalen Bibliotheken. Dabei sieht sie einen personalisierten Dienst als eine Sonderform eines adaptiven Systems und kommt zu folgender Definition: 'Personalisierung ist die dynamische Anpassung des Dienstleistungs- oder Informationsangebots eines Systems mit dem Ziel, die Präferenzen und Bedürfnisse von Individuen und Gruppen abzubilden.' Hierbei wird die Personalisierung genutzt, um einen Mehrwert zu schaffen.

Kobsa et al. [99] beschäftigen sich mit der Personalisierung von Hypermediasystemen. Darunter versteht man ein System, das unterschiedliche digitale Inhalte mit Hyperlinks verknüpft. In diesem Zusammenhang wird Personalisierung wie folgt definiert: 'We define a personalised hypermedia application as a hypermedia system which adapts the content, structure and/or presentation of the networked hypermedia objects to each individual user's characteristics, usage behaviour and/or usage environment.'[8].

Im Rahmen dieser Arbeit wird unter Berücksichtigung der angeführten Definitionen Personalisierung wie folgt definiert:

Definition 6 (Personalisierung) *Personalisierung ist die dynamische Anpassung von Inhalten und Diensten an ein gegebenes Nutzerprofil.*

Ein Nutzerprofil ist dabei folgendes:

Definition 7 (Nutzerprofil) *Ein Nutzerprofil entspricht der Beschreibung eines Nutzers durch ein Benutzermodell. Es umfasst statische Kontextinformationen wie seine demographischen Daten, persönliche Vorlieben und Abneigungen, Bedürfnisse und Fähigkeiten und das verwendete Endgerät, aber auch dynamische Kontextinformationen wie den Aufenthaltsort oder die Zeit. Weiterhin gehören hierzu Daten mit zeitlichem Verlauf wie die Nutzungshistorie.*

[6] wenn man 3 Million Kunden im Netz hat, sollte man 3 Millionen Geschäfte im Netz haben

[7] der Reduzierung der Komplexität und Fülle an Informationen und verfügbaren Dienstangeboten im Internet auf ein geeignetes Maß für den individuellen Nutzer

[8] Wir definieren eine personalisiertes Hypermedia-Anwendung als ein Hypermediasystem, das Inhalt, Struktur und/oder Präsentation von vernetzen Hypermedia-Objekten an die individuellen Nutzercharakteristika, Nutzerverhalten und/oder Nutzungsumgebung anpasst.

Personalisierung ist gerade bei mobilen Endgeräten interessant, weil hier meist nur ein Nutzer zur selben Zeit auf das Gerät zugreift und es zusätzlich eine eindeutige Zuordnung eines Geräts zu einem bestimmten Nutzer gibt. Im Folgenden werden hauptsächlich Ansätze zur Personalisierung für die mobile Nutzung von Multimedia vorgestellt. Diese lassen sich aber auch bei stationärer Nutzung anwenden.

In dieser Arbeit werden zwei Arten der Personalisierung näher betrachtet:

- Personalisierte Auswahl von multimedialen Inhalten basierend auf dem Nutzerprofil (siehe Kapitel 5). Hier spielen in Empfehlungssystemen eingesetzte Methoden eine wichtige Rolle.
- Personalisierte Anpassung der Darstellung multimedialer Inhalte basierend auf dem Nutzerprofil (siehe Kapitel 6).

Dimensionen der Personalisierung

Um Personalisierung besser erfassen zu können, kann man deren Dimensionen genauer betrachten. Jedes System zur Personalisierung kann eine beliebige Ausprägung in jeder der Dimensionen annehmen. Die in dieser Arbeit vorgestellten Konzepte werden in die jeweiligen Aspekte eingeordnet.

WIE wird personalisiert? Hierbei wird unterschieden, auf welche Art der Inhalt an das Benutzermodell angepasst wird. Im Rahmen dieser Arbeit werden die personalisierte Auswahl kompletter Inhalte sowie die personalisierte Anpassung der Darstellung genauer untersucht. Erstere kann mittels Methoden aus dem Bereich der Empfehlungssysteme umgesetzt werden. Hierzu gehören die klassischen Filtermethoden wie Content-based Filtering, User-based Filtering, Collaborative Filtering oder hybride Formen (siehe Abschnitt 5.1). Andere Arten befassen sich mit einer Anpassung des Inhalts an sich wie es in interaktiven Geschichten der Fall ist (siehe hierzu z. B. [107]). Dies ist aber nicht im Fokus dieser Arbeit.

Für die Personalisierung der Darstellung des Inhalts werden Methoden wie Skalierung (z. B. die Anpassung der Dateigröße oder das Anpassen eines Videos oder Bildes an eine Bildschirmgröße) und Konvertierung (Umwandlung in ein anderes Format oder sogar in einen anderen multimedialen Typ), aber auch das Entfernen, Ersetzen oder Hinzufügen einzelner Teile von zusammengesetzten multimedialen Inhalten eingesetzt.

WAS wird personalisiert? Was an das gegebene Nutzerprofil angepasst werden soll, muss genau festgelegt sein. Beispiele hierfür sind der eigentliche Inhalt, die Benutzerschnittstelle oder das Darstellungsformat. Im Rahmen dieser Arbeit werden die Anpassung des Inhalts oder präziser die Auswahl geeigneter Inhalte und die Anpassung der Darstellungsform genauer untersucht. Eine Anpassung der Benutzerschnittstelle wie in [161] wird nicht betrachtet.

WO findet die Personalisierung statt? Ein wichtiges Kriterium ist, wo genau die Personalisierung durchgeführt wird: auf dem Server, dem Client oder hybrid. Für eine Personalisierung auf dem Server (z. B. [144, 187]) spricht die dort meist vorhandene höhere Rechenleistung und eine u. U. geringere Belastung des Netzes, weil man nur die schon personalisierten Inhalte übertragen muss und nicht noch zusätzliche Informationen, die der Client für die Verarbeitung benötigt. Dies ist gerade im Bereich mobiler Endgeräte auf Grund der beschränkten Ressourcen der Luftschnittstelle wichtig. Vorteile einer Personalisierung auf Seite des Clients (z. B. [33]) ist ein besserer Schutz der Privatsphäre, da das Nutzerprofil, das in der Regel sensible Daten enthält, nicht auf den Server übertragen werden muss, und eine bessere Skalierbarkeit, da die Berechnungen nicht durch eine zentrale Instanz durchgeführt werden. Ein hybrider Ansatz versucht die Vorteile beider Verfahren zu kombinieren und wird im Rahmen dieser Arbeit verfolgt.

FÜR WEN findet die Personalisierung statt? Für wen die Personalisierung stattfindet, ist ein weiterer relevanter Aspekt. Gemäß der aufgestellten Definition von Personalisierung, werden hier Personalisierungsansätze für Einzelnutzer erarbeitet. Wie bereits erläutert, kann man auch für Benutzergruppen personalisieren, wie in [127] vorgeschlagen. Weiterhin kann für geeignete Gerätegruppen personalisiert werden wie in [152], indem die verschiedenen Charakteristika von Endgeräten wie z. B. Displaygröße, Eingabemöglichkeiten (Tastatur, Touchpad, Audio etc.) oder die Konnektivität berücksichtigt werden.

WANN wird personalisiert? Eine weitere Dimension umfasst den Zeitpunkt der Personalisierung. Manche Systeme stoßen die entsprechenden Bearbeitungsschritte erst nach einer expliziten Anfrage an, wie es der Fall bei einer Suchanfrage durch einen Nutzer ist, andere arbeiten proaktiv und reagieren auf das Eintreten gewisser Situationen wie der Anmeldung des Nutzers am System oder einer Veränderung im Nutzerprofil. Es kann auch sein, dass manche Systeme periodisch eine Prä-Personalisierung durchführen, um die Bearbeitungszeit bei der eigentlichen Personalisierung zu reduzieren. Die hier entwickelten Ideen sind für alle diese Varianten geeignet.

WARUM wird personalisiert? Eher im Forschungsfokus der Kommunikationswissenschaften liegt die Frage, warum personalisiert wird. Mögliche Gründe können z. B. die Informationsfindung oder die Vereinfachung der Nutzung, aber auch betriebswirtschaftliche Vorteile wie die Abgrenzung gegenüber Wettbewerbern sein. Dieser Dimension wird im Folgenden keine Beachtung geschenkt.

2.2. Analyse von Szenarien und Definition von Anforderungen

Nachdem im vorhergehenden Abschnitt ein Überblick über die Begriffe Multimedia, Kontext und Personalisierung gegeben wurde, werden im Folgenden unterschiedliche Szenarien untersucht und Anforderungen abgeleitet. Dabei werden auch die Rollen genauer untersucht, die bei der Kontext-abhängigen Personalisierung relevant sind.

2.2.1. Szenarien

Nachfolgend werden Anwendungsszenarien aufgezeigt, in denen Kontext-abhängige Personalisierung eine wichtige Rolle spielt und die Benutzerfreundlichkeit der einzelnen Systeme erheblich erhöht wird.

Personalisierter Touristenführer

Im ersten Szenario spielt Alice die Hauptolle. Sie ist zu Besuch in München und nutzt einen von *Perfect Travel* angebotenen mobilen Touristenführer, um die unbekannte Stadt zu erkunden. Sie kann wählen zwischen einer geführten Route gemäß ihrer Interessen, oder sie kann eigenständig durch die Stadt streifen und wird proaktiv informiert, sobald sich eine Sehenswürdigkeit in der Nähe befindet. Zu diesen Sehenswürdigkeiten wird ihr ein Video mit interessanten Informationen gezeigt.

Alice trägt ihre Interessen ein, entscheidet sich für die proaktive Benachrichtigung und schaltet beim Verlassen des Hotels den Dienst an. Während sie durch die Münchner Innenstadt streift, macht das System sie immer wieder auf interessante Sehenswürdigkeiten aufmerksam. Gegen Mittag wird Alice automatisch eine Liste von Restaurants in der Nähe empfohlen. Da das Wetter heute sonnig und warm ist, sind besonders Biergärten und Restaurants mit schönen Sonnenterassen dabei. Alice entscheidet sich dafür, in einen der Biergärten zu gehen. Die Nutzer von *Perfect Travel* können die mobilen Videos kommentieren. Wegen den beschränkten Eingabemöglichkeiten mobiler Endgeräte, werden insbesondere Video-, Audio- und Bildkommentare unterstützt. Da Alice das günstige Mittagsmenü des Biergartens besonders gefallen hat, entschließt sie sich dazu, einen kurzen Audiokommentar aufzuzeichnen.

Als Alice am Nachmittag in die Nähe des Münchner Kunstareals kommt, werden ihr auch Filme von der *Alten Pinakothek*, der *Pinakothek der Moderne* und dem *Museum Brandhorst* gezeigt, nicht jedoch von der *Neuen Pinakothek*, weil heute Dienstag ist und das Museum an diesem Tag nicht geöffnet hat. Da sie sich besonders für die alten Meister interessiert, beschließt sie in die *Alte Pinakothek* zu gehen. Für die einzelnen Museen ist jeweils eine geführte Videotour im System vorhanden, die man auswählen kann. Alice entschließt sich die Tour zu machen. Um die anderen Besucher im Museum durch den Ton der Videos nicht zu stören, wird automatisch die Audiospur ausgeschaltet

und dieselbe Information als Untertitel angezeigt. Da Alice lieber wieder den Ton hören will, schließt sie ihre Kopfhörer an ihr Handy an und das System stellt automatisch wieder von Untertitel auf Audio um.

Kontext-basierte, multimediale Nachrichten

Im zweiten Szenario geht es um den Soziologen Bob, der gerne über die aktuellen Geschehnisse in der Welt informiert wird. Leider ist er beruflich den ganzen Tag unterwegs und hat deswegen kaum Zeit, sich durch Zeitung, Internet oder Fernsehen zu informieren. Er entdeckt im Internet den neuen Dienst *MultiNewsPortal*, der personalisierte und Kontext-abhängige Nachrichten für mobile Endgeräte anbietet. Der Dienst kann je nach Präferenz das aktuelle Geschehen als Videofilm, Audiostream oder in Textform mit oder ohne Bilder zusammenfassen. Zusätzlich können zu den multimedialen Nachrichten auch noch Nutzerkommentare eingetragen und angezeigt werden. Das System wirbt damit, dass es die Nachrichten Kontext-abhängig anzeigt, also dass es zusätzlich den persönlichen Kontext bei der Bereitstellung der Beiträge berücksichtigt. Um dies zu unterstützen, kann der Dienst auch noch mit anderen Diensten wie einem Terminkalender verknüpft werden, was Bob ebenfalls nutzt.

Bob registriert sich bei dem Dienst und kann bei der Konfiguration zahlreiche Informationen über sich und seine Interessen angeben. Dazu kann er u. a. die einzelnen Kategorien *Politik*, *Wirtschaft*, *Sport*, *Technik* etc. nach seinen Interessen bewerten und Themenbereiche sogar ganz ausschließen. Zusätzlich kann er noch Schlüsselwörter angeben, an denen er besonderes Interesse hat. Als Soziologe interessiert er sich besonders für Beiträge zum Thema *Migration*. Im System kann er einstellen, wann und wie er informiert werden will. Außerdem trägt er ein, dass er zusätzlich zu den Nachrichten noch Kommentare von anderen Nutzern angezeigt haben möchte, wenn sie die Kommentare inhaltlich mit *Migration* beschäftigen oder die Kommentare für ihn anderweitig von Interesse sind.

Am nächsten Tag sitzt Bob in der Staatsbibliothek München um etwas zu einem Thema zu recherchieren, als das regelmäßige Nachrichtenupdate auf seinen PDA übertragen wird. Das System erkennt dabei seinen aktuellen Aufenthaltsort und sendet zusätzlich noch Informationen über weiterführende Literatur zu einigen der Themen, wenn die entsprechenden Bücher in der Bibliothek vorrätig sind. Außerdem sendet der Dienst automatisch nur Textbeiträge mit Bildern und keinerlei Audioinformationen, um die anderen Besucher der Bibliothek nicht zu stören. Bob schaut sich das aktuelle Nachrichtenupdate an und freut sich sehr darüber, gleich die entsprechenden Bücher ausleihen zu können.

Am Nachmittag ist Bob in einer Besprechung als das regelmäßige Nachrichtenupdate eingeht. Das System erkennt dies automatisch durch seinen Kalender und wartet mit einer Benachrichtigung bis zum Ende der Besprechung. Leider ist die momentane Konnektivität von Bobs PDA sehr schlecht, weswegen nur Textinformationen ohne Bilder übertragen werden. Bob wird später auf die neuen Nachrichten hingewiesen und greift

von seinem Auto aus auf diese zu. Da die Verbindung dort deutlich besser ist und niemand gestört werden kann, werden die Videos der Beiträge nachgeladen.

Das letzte Update des Tages bekommt Bob am Abend als er gerade nach Hause kommt. Er schaut sich die Videobeiträge sofort an. Nach einer kurzen Zeit werden die Videobeiträge durch eine Diashow ersetzt, da der Akku des PDAs langsam leer wird und das System automatisch versucht den Batterieverbrauch zu senken. Bob erkennt das, steckt es an das Ladekabel an und sofort wird aus der Diashow wieder ein Videobeitrag.

Personalisierte Werbung

Im letzten Szenario geht es um Chuck, der unterwegs gerne mobiles Fernsehen (Mobile TV) auf seinem Handy nutzt. Seit neuestem gibt es eine interessante Innovation: statt allen Zuschauern dieselben Werbespots zu zeigen, die diese u. U. nicht interessieren, werden die einzelnen Spots in Abhängigkeit von den Nutzerinteressen, aber auch von anderen Kontextinformationen wie der Uhrzeit oder dem aktuellen Aufenthaltsort des Nutzers ausgewählt. Zusätzlich wirbt das System damit, dass es basierend auf dem Nutzerverhalten Interessen automatisch ableiten kann. Ein kurzer Werbefilm erklärt, dass dafür beobachtet wird, welche Werbespots komplett angeschaut, welche nach einer Weile wieder angehalten und welche komplett ignoriert werden.

Chuck sitzt gerade in einem kleinen Cafe in der Innenstadt. Wie immer schaltet er sein Handy an und nutzt Mobile TV. In der Werbepause werden ihm speziell auf ihn zugeschnittene Spots gezeigt. Da das System weiß, dass Chuck gerne DVDs anschaut, macht einer der Spots ihn darauf aufmerksam, dass es eine Sonderverkaufsaktion bei einem Geschäft in der Nähe gibt, in dem drei DVDs zum Preis von zwei verkauft werden. Mit dem Spot wird Chuck auch mitgeteilt, warum gerade dieser für ihn ausgewählt wurde. Dadurch kann Chuck die Entscheidungen des Systems nachvollziehen. Zusätzlich wird Chuck noch ein Gutschein übermittelt, der ihm einen weiteren Rabatt gewährt. Chuck beschließt dort vorbeizuschauen, sobald er seinen Kaffee ausgetrunken hat.

Am Abend sitzt Chuck auf seinem Sofa und schaut Fernsehen. Auch hier werden die Spots wie beim mobilen Fernsehen gemäß seiner Präferenzen auf ihn zugeschnitten und Chuck schaut interessiert die Spots an. Später kommt seine Frau Doris nach Hause und setzt sich zu Chuck auf das Sofa. Das System bemerkt, dass jetzt nicht nur Chuck zusieht und wählt die Werbung so, dass sie für beide interessant ist.

2.2.2. Charakteristika der Szenarien

Im folgenden Abschnitt werden für die Szenarien typische Charakteristika aufgezeigt, die von einem System für Kontext-abhängige Personalisierung von multimedialen Inhalten berücksichtigt werden müssen. Aus diesen lassen sich ein Rollenmodell und die Anforderungen ableiten, die in den Abschnitten 2.2.3 und 2.2.4 aufgeführt werden.

1. *Professionelle und Nutzer-generierte Inhalte*
   In den Szenarien werden den Akteuren professionelle Inhalte wie die Nachrichten im zweiten Szenario, aber auch Nutzer-generierte, insbesondere Kommentare, präsentiert. Beide Arten unterliegen dem Vorgang der Personalisierung, indem sie passend gewählt bzw. bei der Darstellung angepasst werden.

2. *Explizite und implizite Profilgewinnung*
   Der Personalisierungsprozess setzt auf Nutzerprofilen auf. Diese können durch den Nutzer explizit mit Informationen gefüllt werden. Wie im dritten Szenario beschrieben, kann ein System aber auch basierend auf Beobachtungen des Nutzerverhaltens Rückschlüsse auf seine Präferenzen ziehen.

3. *Transparenz für den Nutzer*
   Wie die Szenarien zeigen, ist der eigentliche Personalisierungsprozess transparent für den Nutzer. Er erlebt zwar den höheren Komfort, die eine Personalisierung nach sich zieht, von den eigentlichen Schritten, die dafür benötigt werden, soll er jedoch nichts mitbekommen. Dennoch kann es je nach Anwendungsfall hilfreich sein, ihm Feedback über die Anpassung zu geben. So erhöht z. B. eine kurze Erklärung, warum ein Inhalt für einen Nutzer ausgewählt wurde, die Nutzerzufriedenheit erheblich.

4. *Personalisierung für einen und mehrere Nutzer*
   Die Personalisierung soll in dieser Arbeit für Einzelnutzer funktionieren und nicht basierend auf Nutzergruppen. Dennoch sollen wie im dritten Szenario beim gleichzeitigen Konsumieren multimedialer Inhalte durch mehrere Nutzer auch die Nutzerprofile all dieser Nutzer berücksichtigt werden und Inhalte empfohlen werden, die für alle von Interesse sind.

5. *Proaktivität*
   Die Personalisierung der Inhalte wird abhängig vom statischen und dynamischen Kontext durchgeführt. Dies soll vor allem proaktiv erfolgen, also ohne explizite Nutzerinteraktion, sondern automatisch, wenn sich die Situation entsprechend ändert. Die proaktive, personalisierte Anpassung wird z. B. im zweiten Szenario deutlich, wenn Nachrichtenbeiträge in der Bibliothek ohne Audio abgespielt werden. Eine proaktive, Kontext-abhängige Auswahl findet z. B. im ersten Szenario statt, als Alice Videos von Museen erhält, die in ihrer Nähe und zum entsprechenden Zeitpunkt geöffnet sind.

6. *Heterogenität der Endgeräte*
   In den unterschiedlichen Szenarien werden verschiedene Endgeräte verwendet. So nutzt Chuck z. B. im dritten Szenario denselben Dienst mobil, aber auch am heimischen Fernseher. Abhängig vom verwendeten Endgerät sollen die Inhalte passend

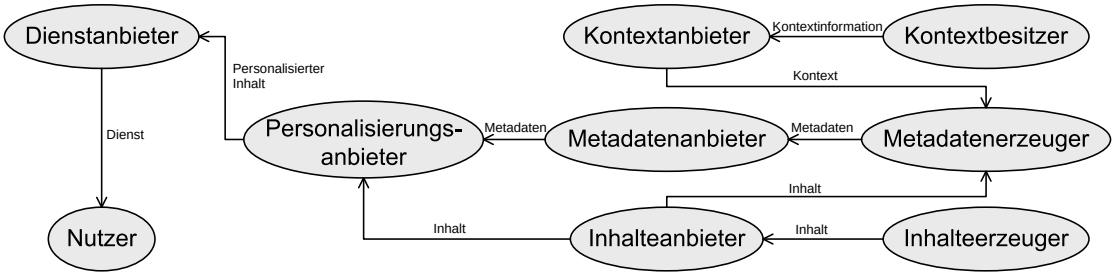

Abbildung 2.1.: Rollenmodell für Kontext-abhängige Personalisierung

personalisiert werden. Dies bezieht sich vor allem auf die Anpassung der Darstellung an die technischen Eigenheiten des jeweiligen Geräts, insbesondere da dieselben Dienste auf unterschiedlichen Geräten genutzt werden. Videos können z. B. nur abgespielt werden, wenn ein Gerät über einen Player verfügt, der das entsprechende Format unterstützt, und es soll an die Displaygröße angepasst werden. Andere Parameter können die Konnektivität (vgl. Szenario zwei), das Vorhandensein von Lautsprechern oder externen Geräten oder die Leistung des Prozessors sein.

7. *Ressourcenbeschränkte Endgeräte*
   Wie bei der Begriffsbildung bereits erwähnt, ist die Personalisierung gerade bei mobilen Endgeräten interessant, weil hier meist nur ein Nutzer zur selben Zeit auf das Gerät zugreift und es eine eindeutige Zuordnung eines Geräts zu einem bestimmten Nutzer gibt. Zusätzlich hat man hier einen deutlich dynamischeren Kontext und kann Inhalte passender auswählen. Diese Geräte haben aber geringere Ressourcen als stationäre Geräte, vor allem bzgl. Batterielebensdauer und Rechenkapazität. Zusätzlich kommt es zu weiteren Einschränkungen, da mobile Endgeräte die Inhalte nur über die eingeschränkten Ressourcen der Luftschnittstelle beziehen können.

2.2.3. Rollenmodell

Aus obigen Szenarien kann man die unterschiedlichen Rollen bei der Kontext-abhängigen Personalisierung identifizieren und daraus ein Rollenmodell ableiten. Es ergibt sich das Rollenmodell aus Abbildung 2.1. Die einzelnen Rollen und ihre jeweiligen Funktionen werden im Folgenden erläutert.

- **Nutzer/Konsument**
  Der Nutzer oder auch Konsument ist die eigentliche Zielperson der Personalisierung, die auf dessen Kontext basiert. Er kann explizit einen Dienst um Inhalte anfragen (reaktive Dienstnutzung) oder diesen proaktiv nutzen. Dafür muss er vorher beim Dienstanbieter registriert sein. In jedem Fall findet die eigentliche Personalisierung

automatisch und ohne Nutzerinteraktion statt. In den obigen Szenarien sind Alice, Bob, Chuck und Doris die Nutzer der jeweiligen Systeme.

- **Inhalteanbieter**
  Der Inhalteanbieter ist diejenige Rolle, der die eigentlichen multimedialen Inhalte vorhält und dem Nutzer bei Bedarf zur Verfügung stellt. Dafür werden die Inhalte z. B. in einer Datenbank unter einem eindeutigen Bezeichner abgespeichert. Im ersten Szenario kann z. B. die Alte Pinakothek die Videoinhalte für den Museumsführer vorhalten, die dann vom Reisedienst entsprechend verwendet werden. Im letzten Szenario ist eine Werbeagentur der Inhalteanbieter für die einzelnen Spots im Gegensatz zu den eigentlichen TV-Sendeanstalten.

- **Inhalteerzeuger**
  Der Inhalteerzeuger kreiert die Inhalte und stellt sie dem Inhalteanbieter zur Verfügung. In vielen Fällen werden die Rollen des Inhalteerzeugers und des -anbieters vom selben Akteur eingenommen, gerade aber im Bereich der Nutzer-generierten Inhalte sind diese Rollen getrennt. Auch im zweiten Szenario sind z. B. die jeweiligen Presseagenturen die Inhalteanbieter, ein Journalist hingegen der Inhalteerzeuger.

- **Dienstanbieter**
  Der Dienstanbieter stellt die Funktionalität eines Diensts bereit und verwendet dafür die vom Inhalteanbieter bereitgestellten Inhalte. Er ist für das Management des Diensts zuständig und erzielt direkt vom Nutzer Erlöse aus dessen Nutzung des Diensts. Im ersten Szenario ist der Dienstanbieter *Perfect Travel*, der die eigentlichen Inhalte in die Funktionalität eines mobilen Touristenführers einbettet.

Zusätzlich zu diesen Rollen, die auch in herkömmlichen multimedialen Diensten vorkommen, muss das Rollenmodell um passende Rollen für die Personalisierung erweitert werden. Im Rahmen dieser Arbeit wird davon ausgegangen, dass Inhalte und Nutzer geeignet durch Metadaten beschrieben sind, die für die Personalisierung verwendbar sind. Dies ergibt die folgenden weiteren Rollen im Rollenmodell:

- **Personalisierungsanbieter**
  Die eigentliche Personalisierung der Inhalte findet bei dieser Rolle statt. Sie übernimmt die Inhalte, wählt sie für den Nutzer passend aus, passt ihre Darstellung geeignet an und übergibt die so personalisierten Inhalte dem Dienstanbieter zur Nutzung im eigentlichen Dienst. In den Szenarien werden die Rollen des Personalisierungsanbieters und des Dienstanbieters vom selben Akteur eingenommen.

- **Metadatenanbieter**
  Analog zum Inhalteanbieter stellt der Metadatenanbieter die Metadaten zu den einzelnen Inhalten bereit. Es existiert eine eindeutige eins-zu-eins Zuordnung zwischen Inhalt und Metadaten, die z. B. über eine URI in der Beschreibung realisiert

werden kann. Aber nicht nur die Inhalte werden durch Metadaten beschrieben, auch für jeden Nutzer liegt als Nutzerprofil eine eindeutige Metadatenbeschreibung vor. In vielen Fällen werden die Rollen Inhalte- und Metadatenanbieter vom selben Akteur eingenommen, bzw. werden Profile beim Nutzer oder Dienstanbieter gehalten.

- **Metadatenerzeuger**
  Der Metadatenerzeuger ist diejenige Rolle, die die Metadaten zu einem Inhalt oder einem Nutzer erzeugt und dem Metadatenanbieter zur Verfügung stellt. Bei der Metadatenerzeugung unterscheidet man drei Fälle: die Erzeugung durch den Nutzer, die Erzeugung unterstützt durch Nutzer und die automatische Erzeugung. Im ersten Fall fügt ein Nutzer selbstständig Beschreibungen zu einem Inhalt dazu, indem er z. B. Schlüsselwörter oder Tags mit dem Inhalt assoziiert. Im zweiten Fall ist der Nutzer zwar immer noch selbst aktiv, bekommt aber computergestützte Hilfe z. B. bei der Auswahl passender Schlüsselwörter und der Erstellung von Beschreibungen. Im letzten Fall wird z. B. durch Text-, Bild-, Audio- oder Videoanalyse versucht, automatisch den Inhalt zu beschreiben. Handelt es sich bei den Metadaten um ein Nutzerprofil, kann dieses entweder explizit durch den Nutzer erzeugt worden sein, oder es wurde implizit aus dem Nutzerverhalten abgeleitet wie im dritten Szenario erläutert.

Diese Rollen genügen für ein herkömmliches Personalisierungssystem. Um die Personalisierung abhängig von Kontextinformationen zu machen, werden weitere Rollen benötigt:

- **Kontextanbieter**
  Der Zugriff auf Kontextinformationen erfolgt über die Rolle des Kontextanbieters. Wie genau die Kontextinformationen bestimmt werden, ist im Rahmen dieser Arbeit nicht von Interesse. Im Folgenden wird davon ausgegangen, dass diese in den Metadaten geeignet beschrieben vorliegen. Daher wurde auf eine umfangreichere Aufteilung der Rollen zur Kontextbereitstellung, wie sie in anderen Arbeiten zu finden ist (vgl. z. B. [146]), bewusst verzichtet.

- **Kontextbesitzer**
  Eine Kontextinformation bezieht sich auf eine Entität z. B. eine Person wie Alice im ersten Szenario. Diese Entität wird durch die Rolle des Kontextbesitzers repräsentiert.

Der Informationsfluss zwischen den Rollen ist wie folgt: Inhalte werden erzeugt und dem Inhalteanbieter zur Verfügung gestellt. Basierend auf den Inhalten werden automatisch oder durch den Nutzer die Metadaten vom Metadatenerzeuger erstellt und diese dem Metadatenanbieter übergeben. Dabei können auch Kontextinformationen wie der Erzeugungsort des Inhalts berücksichtigt werden. Diese werden beim Kontextbesitzer bestimmt

und von einem Kontextanbieter verwaltet. Ein Personalisierungsanbieter kann basierend auf den Metadatenbeschreibungen der Inhalte und der Nutzer, die jeweils Kontextinformationen beinhalten können, Inhalte auswählen und diese direkt beim Inhalteanbieter beziehen. Der Inhalt wird angepasst und an den Dienstanbieter übergeben, der diesen geeignet in den Dienst integriert.

2.2.4. Anforderungen

Aus den Szenarien, der Analyse der Charakteristika dieser Szenarien sowie dem Rollenmodell werden Anforderungen abgeleitet, die ein System zur Kontext-abhängigen Personalisierung multimedialer Inhalte erfüllen muss. Diese werden im Folgenden näher erläutert.

1. *Unterstützung unterschiedlicher Medientypen*
   Die hier erarbeiteten Konzepte sollen nicht nur für einen Medientyp anwendbar sein, sondern möglichst generisch Inhalte von unterschiedlichem Medientyp wie Audio, Bild, Video, Text, Webseiten, Animationen etc. berücksichtigen. Der Personalisierungsprozess darf also nicht abhängig sein vom konkreten Medientyp des Inhalts.

2. *Formale Beschreibung der Inhalte*
   Um Inhalte Kontext-abhängig personalisieren zu können, werden Beschreibungen der Inhalte benötigt. Diese können z. B. semantische Informationen über die zum Inhalt passenden Schlagwörter, das Genre oder komplexe semantische Beschreibungen über die Handlung enthalten. Weitere interessante Informationen sind Metainformationen über den Inhalt wie den Autor, den Erzeuger oder Mitwirkende wie Schauspieler oder Sprecher. Zusätzlich sollen Informationen über die Verwaltung der Inhalte wie auch Dateiformat, Auflösung etc. vorhanden sein.

3. *Formale Beschreibung des Nutzers*
   Neben den Beschreibungen der Inhalte werden die Beschreibungen der Nutzer des Systems benötigt. Diese beinhalten statische Kontextinformationen wie das Geburtsdatum, Geschlecht oder den Wohnort des Nutzers und dynamische Kontextinformationen wie seinen aktuellen Aufenthaltsort oder die Zeit. Neben Kontextinformationen enthält das Nutzerprofil Nutzerbewertungen von bereits zugegriffenen Inhalten und die gesamte Nutzungshistorie.

4. *Formale Beschreibung von situativen Anpassungsregeln*
   Neben der Beschreibung der Inhalte und der Nutzer soll das System die Spezifikation von situativen Anpassungsregeln erlauben. Diese legen fest, wann und auf welche Art ein Inhalt angepasst werden soll. Diese Regeln sind Teil des Nutzerprofils.

5. *Leichte Erweiterbarkeit der Anpassungsregeln, Nutzer- und Inhaltsbeschreibungen*
   Um zu einem späteren Zeitpunkt die Anforderungen erweitern, aber immer noch

die Konzepte nutzen zu können, sollen die Nutzer- und Inhaltsbeschreibungen leicht und einfach zu erweitern sein.

6. *Unterstützung unterschiedlicher Personalisierungsverfahren*
   Da sich die unterschiedlichen Empfehlungsverfahren wie Inhalts-basiertes oder kollaboratives Filtern (vgl. Abschnitt 5.1.2 – 5.1.4) je nach Anwendungsfall mehr oder weniger eignen, soll sich das Gesamtsystem nicht auf eines dieser Verfahren beschränken. Weiterhin soll es unterschiedliche Möglichkeiten zur Anpassung der Darstellung der Inhalte, also z. B. Skalieren von Bildern, Konvertieren von Formaten oder das Ersetzen von bestimmten Teilen, unterstützen.

7. *Unterstützung von Nutzer-generierten Inhalten*
   Nutzer-generierte Inhalte nehmen mit der zunehmenden Bedeutung von Web 2.0-Anwendungen eine stetig wachsende Bedeutung ein. Nutzer wollen immer mehr am Vorgang der Inhalteerzeugung und damit der Meinungsbildung teilnehmen. Im Rahmen dieser Arbeit handelt es sich bei diesen Inhalten in erster Linie um Nutzerkommentare. Sie sollen geeignet von dem in dieser Arbeit vorgestellten Personalisierungssystem unterstützt werden.
   Eine wichtige Annahme dieser Arbeit ist, dass zu allen Inhalten die entsprechenden Metadatenbeschreibungen vorliegen. Man kann von professionell erzeugten Inhalten annehmen, dass alle relevanten Informationen vorliegen. Bei Nutzer-generierten Inhalten wie den Kommentaren in den Szenarien sind die Inhaltsbeschreibungen aber nur rudimentär, wenn überhaupt vorhanden. Um dennoch bei diesen Daten auch eine Kontext-abhängige Personalisierung durchführen zu können, werden Mechanismen benötigt, die den Nutzer bei der Verschlagwortung seiner selbst erzeugten Inhalte unterstützen oder automatisch Schlagworte für Inhalte erzeugen.

8. *Implizites Profiling*
   Um die Nutzerinteraktion zu reduzieren und damit die Benutzbarkeit des Systems zu erhöhen, soll es automatisch aus dem Nutzerverhalten Präferenzen ableiten können. Man kann z. B. aus der Dauer, die ein Nutzer auf einen bestimmten Inhalt zugreift, Rückschlüsse auf sein Interesse an dem Inhalt ziehen und damit das Profil verfeinern. Dieser Prozess wird als implizites Profiling bezeichnet.

9. *Verfahren zur Ähnlichkeitsbestimmung von Kontextinformationen*
   Um Kontext-abhängig Personalisieren zu können, müssen Verfahren entwickelt werden, die bestimmen können, wann zwei Kontextinformationen, die u. U. auch in unterschiedlichen Formaten bzw. Granularitäten vorliegen, identisch oder ähnlich sind. Ein Beispiel sind zwei Ortsinformationen, wobei die eine als GPS-Koordinate, die andere als Adresse vorliegt.

10. *Anwendungsunabhängigkeit*
    Das entwickelte Gesamtkonzept soll möglichst generisch sein, um unterschiedliche

Anwendungen realisieren zu können. Dafür sollen zum einen Konfigurationsschnittstellen vorhanden sein, die eine Anpassung des Systems an das gewünschte Anwendungsszenario erlauben. Zum anderen sollen die einzelnen Teilfunktionalitäten stark modularisiert sein, um je nach Anwendungsfall die Module geeignet auswählen und kombinieren zu können.

11. *Schutz der Privatsphäre des Nutzers*
    Bei der Personalisierung müssen Daten über den Nutzer gesammelt, gespeichert und verarbeitet werden, die als Eingabewerte für den Personalisierungsprozess dienen. Dies kann der Nutzer als schwerwiegenden Eingriff in seine Privatsphäre empfinden. Die Problematik verstärkt sich noch bei der Kontext-abhängigen Personalisierung, da hier zusätzlich noch dynamische Kontextinformationen des Nutzers verwendet werden, die z. B. den aktuellen Aufenthaltsort des Nutzers preisgeben. Von daher muss ein geeignetes Mittelmaß gefunden werden zwischen dem Recht des Nutzers auf Privatsphäre und der Personalisierung von Inhalten basierend auf diesen Kontextinformationen.

12. *Geringe Belastung der Luftschnittstelle*
    Bei allen kabellosen Systemen ist die Luftschnittstelle ein limitierender Faktor. Es können nicht uneingeschränkt Daten zwischen einem Client- und einem Server-System übertragen werden, da dies mit einem hohen Zeitaufwand und mit hohen monetären Kosten verbunden ist, sofern keine Datenflatrate vorhanden ist. Deswegen sollen die Ressourcen der Luftschnittstelle effizient genutzt und nur die Daten übertragen werden, die unbedingt benötigt werden.

13. *Skalierbarkeit*
    Die Dienste, die auf der Kontext-abhängigen Personalisierung aufsetzen, sollen von einer großen Anzahl von Nutzern verwendbar sein. Das System soll deswegen bezüglich der Anzahl der Nutzer, für die eine Personalisierung vorgenommen werden soll, skalierbar sein.

14. *Effiziente Durchführung der Personalisierung*
    Bei der Bereitstellung von personalisierten Inhalten ist es für eine maximale Nutzerzufriedenheit relevant, wie lange der Personalisierungsprozess benötigt. Da gerade in mobilen Systemen die Übertragung der Daten einige Zeit in Anspruch nehmen kann, soll die eigentliche Personalisierung möglichst effizient und zeitnah erfolgen.

2.3. Betrachtete Problemstellungen

Aus der Analyse der Szenarien und den Anforderungen ergeben sich die verschiedenen Problemstellungen, die im Rahmen dieser Arbeit aufgegriffen werden und für die Lösungen entwickelt werden. Abbildung 2.2 zeigt diese Teilaufgaben im Überblick. Kernpunkt

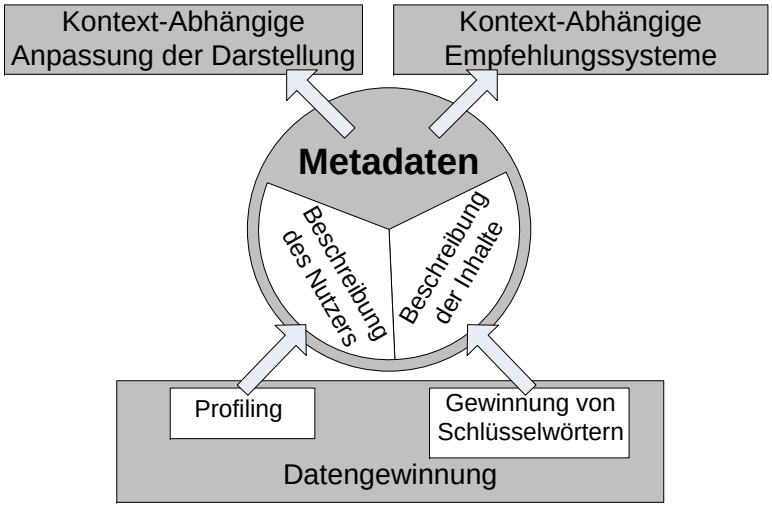

Abbildung 2.2.: Überblick über die einzelnen Teilgebiete dieser Arbeit.

der Arbeit ist dabei das Metadatenmodell, das alle relevanten Informationen über den Nutzer, dessen dynamischen Kontext und die Inhalte beschreiben kann (vgl. Kapitel 3). Dieses Modell wird über Mechanismen zur expliziten und impliziten Datengewinnung mit den entsprechenden Informationen gefüllt (vgl. Kapitel 4). In dieser Arbeit werden eine Methode zur Gewinnung von Profilinformationen, also zum Profiling, und eine Methode zur Gewinnung von Inhaltsbeschreibungen erläutert. Die Metadaten werden für die Ansätze zur Kontext-abhängigen Empfehlung (vgl. Kapitel 5) bzw. zur Kontext-abhängigen Anpassung der Darstellung (vgl. Kapitel 6) verwendet.

2.4. Zusammenfassung

In diesem Kapitel wurde eine Einführung in die Kontext-abhängige Personalisierung multimedialer Inhalte gegeben. Zunächst wurden die wichtigsten Begriffe in diesem Umfeld definiert, um ein einheitliches Verständnis zu schaffen. Basierend auf Szenarien wurde ein Rollenmodell entwickelt und Anforderungen für das Gesamtkonzept aufgestellt. Aus diesen wurden die Problemstellungen abgeleitet, die in den folgenden Kapiteln näher untersucht werden.

Kapitel 3.

Metadaten für die Kontext-abhängige Personalisierung multimedialer Inhalte

Eine Voraussetzung dieser Arbeit ist, dass geeignete Metadatenbeschreibungen für die multimedialen Inhalte und die Nutzer vorliegen, die es erlauben, multimediale Inhalte Kontext-abhängig zu personalisieren. Diese Metadatenbeschreibungen werden in einer formalen Sprache festgehalten, die in diesem Kapitel definiert wird. Sie spielt eine entscheidende Rolle für die Ansätze, die in den darauf folgenden Kapiteln vorgestellt werden. Nach einer Einführung in die Thematik der Metadaten werden die Anforderungen an Metadaten im Umfeld der Kontext-abhängigen Personalisierung aufgestellt, existierende Ansätze vorgestellt und anhand dieser Anforderungen analysiert. Basierend darauf wird die *Multimedia Adaptation and Selection Language* (MASL) für die Kontext-abhängige Personalisierung multimedialer Inhalte entwickelt.

3.1. Einführung Metadaten

Metadaten werden als *Daten über Daten* bezeichnet. Man versteht darunter meist strukturierte Daten, mit deren Hilfe eine Informationsressource wie z. B. ein bestimmter Inhalt, aber auch eine Person beschrieben und dadurch leichter und vor allem automatisch auffindbar gemacht wird [157].

Metadaten kann man grob in *inhaltsabhängige* und *inhaltsunabhängige* Metadaten einteilen. Erstere beziehen sich auf die Information, die ein Inhalt transportieren soll, also z. B. bestimmte, den Inhalt beschreibende Schlagworte, letztere dagegen sind nicht aus dem Inhalt an sich abzuleiten wie z. B. der Autor eines Buches.

Andere Einteilungen klassifizieren Metadaten nach ihrem Verwendungszweck [126, 162]. *Beschreibende oder deskriptive Metadaten* werden für das Auffinden und die Identifikation von Inhalten verwendet. Beispiele hierfür sind ein Titel, der Autor oder Schlüsselwörter. *Administrative Metadaten* enthalten Informationen zur Verwaltung des Inhalts wie das Datum der letzten Änderung oder nutzungsrechtliche Angaben. Informationen über die Hard- und Software, die für die Speicherung, die Konvertierung oder die An-

zeige der Inhalte verwendet werden, wie z. B. wann oder wie ein Inhalt erzeugt wurde, bezeichnet man als *technische Metadaten*. Die Struktur der Inhalte, z. B. wie Web-Seiten zusammenhängen, wird mit *strukturellen Metadaten* beschrieben.

Neben der eigentlichen Beschreibung der multimedialen Inhalte werden für die Filterung noch Metadaten über den Nutzer benötigt. Gemäß der Kategorisierung nach Kobsa et al. [99] kann man Nutzerdaten einteilen in *Benutzerdaten* (persönliche Daten über den Benutzer wie demographische Daten oder Präferenzen), *Benutzungsdaten* (Informationen über das Nutzungsverhalten) und *Umgebungsdaten* (Informationen über die Software- und die Hardwareumgebung sowie die räumliche Umgebung). Schubert teilt dagegen in [151] (vgl. auch [161]) Nutzerprofile in implizite und explizite Profile ein. Dadurch unterscheidet er Informationen, die dem System explizit zur Verfügung gestellt werden wie demographische Daten oder Nutzerbewertungen, und Informationen, die das System aus dem Verhalten des Nutzers ableiten kann.

3.2. Anforderungen an Metadaten für die Kontext-abhängige Personalisierung

Folgende Anforderungen werden an eine Metadatensprache gestellt, die für eine Personalisierung multimedialer Inhalte allgemein und insbesondere unter Berücksichtigung von Kontextinformationen verwendet werden kann. Ein Teil der Anforderungen ergibt sich direkt aus den allgemeinen Anforderungen aus Abschnitt 2.2.4.

1. *Beschreibung unterschiedlicher Medientypen*
   Die hier erarbeiteten Konzepte sollen nicht nur für einen Medientyp anwendbar sein, sondern möglichst generisch Inhalte von unterschiedlichem Medientyp wie Audio, Bild, Video, Text, Webseiten, Animationen etc. berücksichtigen können. Dafür muss die Sprache für die Beschreibung der unterschiedlichen Medientypen geeignet sein. Sie muss unabhängig vom Medientyp allgemeine Informationen bereitstellen, aber auch Typ-spezifische Daten unterstützen.

2. *Semantische Beschreibungen*
   Gerade für inhaltsbasierte Empfehlungstechniken (vgl. Abschnitt 5.1) ist es notwendig, dass die Inhalte genau durch Attribute beschrieben sind. Einen wichtigen Teil dieser Attribute nehmen die semantischen Beschreibungen ein, die die inhaltlichen Konzepte der multimedialen Datei ausdrücken, inklusive aller Beteiligten semantischen Akteure, Orte, Gegenstände etc. Eine solche Beschreibung kann z. B. die Namen der in einem Text vorkommenden Personen beinhalten oder die Handlung eines Filmes beschreiben. Neben diesen Informationen sind für die Kontext-abhängige Auswahl der Inhalte die semantischen Kontextinformationen des Inhalts relevant, also wie ein Inhalt z. B. zeitlich oder räumlich einzuordnen ist.

3. *Informelle Beschreibungen*
   Neben den semantischen Beschreibungen, können auch informelle Beschreibungen und Zusatzinformationen zu den Inhalten für eine personalisierte Auswahl relevant sein. Man unterscheidet zwischen relativ statischen Informationen wie den Öffnungszeiten eines Museums oder Restaurants, oder der Zielgruppe des Inhalts, oder dynamischen Informationen wie die aktuelle Anzahl der freien Plätze in einem Theater. Das Metadatenmodell muss beide Arten von informellen Beschreibungen unterstützen können.

4. *Beschreibungen der Erzeugung*
   Das Metadatenmodell soll Beschreibungen der Erzeugung eines Inhalts berücksichtigen. Dazu gehören Informationen über den Ort und Zeitpunkt der Aufnahme, die Produzenten oder beteiligte Personen wie der Regisseur oder Schauspieler in einem Film, oder Musiker in einem Musikstück. Man kann diese Informationen mit den Präferenzen des Nutzers abgleichen oder mit den aktuellen Kontextinformationen des Nutzers für die Auswahl der Inhalte in Beziehung setzen.

5. *Kategorisierung der Inhalte*
   Da Nutzer oft ähnliche Inhalte bevorzugen, soll das Metadatenmodell eine Kategorisierung gemäß dem Genre, unterschiedlicher Produktgruppen, Schlüsselwörter oder anderer inhaltlicher Charakteristika erlauben. Mit Hilfe dieser Informationen kann man die Inhalte gemäß den Nutzerpräferenzen auswählen.

6. *Dateiinformationen*
   Für die Kontext-abhängige Anpassung der Darstellung der Inhalte ist es notwendig, die Dateiinformationen der Inhalte geeignet zu beschreiben. Dazu gehören z. B. Angaben über das Format, die Dateigröße, die Auflösung oder auch die Dauer eines kontinuierlichen multimedialen Inhalts.

   Diese Informationen können auch für die eigentliche Auswahl der Inhalte geeignet sein, wenn der Nutzer z. B. ein bevorzugtes Format oder eine bevorzugte Auflösung eines Videos hat. Außerdem kann evtl. aus dem Kontext des Nutzer bestimmt werden kann, wie lange er für die Nutzung eines multimedialen Inhalts Zeit hat, und ein passender Inhalt ausgewählt werden.

7. *Nutzerdaten*
   Gemäß den allgemeinen Anforderungen aus Abschnitt 2.2.4 soll das System neben den Inhaltsbeschreibungen auch Nutzerbeschreibungen zur Verfügung stellen. Allgemein unterscheidet man zwischen relativ statischen Nutzerdaten und dynamischen Nutzungsdaten (siehe unten). Zu den Nutzerdaten zählen Beschreibungen des Nutzers, die sich relativ selten ändern, also z. B. demographische Daten wie das Geschlecht oder das Geburtsdatum, oder Präferenzen des Nutzers bezüglich der Auswahl der Inhalte (z. B. bevorzugte Musikgruppe).

8. *Gerätedaten*
   Eng mit den Nutzerdaten hängen die Beschreibung des verwendeten Endgeräts und dessen Hard- und Softwareeigenschaften (z. B. Displaygröße, Konnektivität oder installierte Multimedia-Software) zusammen. Diese Informationen werden vor allem für die Kontext-abhängige Anpassung der Darstellung der Inhalte benötigt.

9. *Nutzungsdaten*
   Im Gegensatz zu den relativ statischen Nutzerdaten, können sich die dynamischen Nutzungsdaten z. B. während der Nutzung eines Diensts häufig ändern. Zu diesen gehören die aktuellen Kontextinformationen des Nutzers wie der aktuelle Aufenthaltsort oder die aktuelle Zeit, oder die Nutzungshistorie, also wann welcher Inhalt vom Nutzer verwendet wurde. Diese Informationen werden für kollaboratives Filtern benötigt, können aber auch für inhaltsbasiertes Filtern verwendet werden (vgl. Kapitel 5).

10. *Beschreibung des Netzes*
    Wichtig für die Kontext-abhängige Anpassung der Darstellung der Inhalte sind Informationen über die Art und den aktuellen Zustand des Netzes. Die Beschreibungen sollen statische Informationen wie die in diesem Netz maximal mögliche oder garantierte Bandbreite und vorhandene Mechanismen zur Fehlerkorrektur bzw. Fehlererkennung enthalten, oder dynamische wie die aktuell verfügbare Bandbreite, die Latenzzeit oder die Bitfehlerrate. Basierend auf diesen Informationen, kann ein Inhalt geeignet angepasst werden (z. B. keine Übertragung von Full-HD-Videos bei schlechter Übertragungsrate).

11. *Annotationen*
    Das Metadatenmodell soll multimediale Annotationen oder Kommentare von Nutzern über einen Inhalt erlauben und diese auch in Beziehung zu dem Inhalt setzen. Während textuelle Kommentare auf stationären Endgeräten mit Tastatur noch für den Nutzer komfortabel sind, sind gerade auf mobilen Endgeräten mit ihren stark eingeschränkten Eingabemöglichkeiten Audio-, Video- oder Bildkommentare sinnvoll. Deshalb soll jeder Medientyp zur Kommentierung verwendet werden können. Zusätzlich soll deren Kontextinformation berücksichtigt werden. Eine Annotation gilt wiederum ebenfalls als Inhalt und kann auf die gleiche Art und Weise beschrieben und selber annotiert werden.

12. *Zeit-abhängige Bewertungen von Inhalten, Attributen und Attributkombinationen*
    Der Nutzer soll die Möglichkeit haben, konsumierte Inhalte geeignet zu bewerten. Diese Information kann für die Auswahl der Inhalte verwendet werden. Da sich aber Nutzerinteressen über die Zeit ändern, sollen die Bewertungen der Inhalte Zeit-abhängig in den Empfehlungsprozess einfließen: Bewertungen, die vom Nutzer vor kurzem abgegeben wurden, haben einen stärkeren Einfluss bei der Empfeh-

lung als Bewertungen, die der Nutzer vor z. B. einem Jahr gegeben hat. Daher müssen Bewertungen Zeit-abhängig im Profil hinterlegt werden. Zusätzlich soll es aber möglich sein, beliebte, aber vom Nutzer nicht oft konsumierte Inhalte zeitunabhängig zu erfassen. Aber nicht nur Inhalte sollen vom Nutzer bewertet werden können, auch einzelne Attributwerte (z. B. der Wert `Science Fiction` vom Attributtyp `Genre`) oder Attributkombinationen (z. B. der Schauspieler `Harrison Ford` in Filmen des `Genre Action`) sollen Zeit-abhängig berücksichtigt werden.

13. *Beschreibung von Anpassungspräferenzen*
    Für die Anpassung der Darstellung ist es wichtig, dass der Nutzer angeben kann, wie er die Inhalte angepasst haben will. Der Nutzer kann hierfür allgemeine Präferenzen angeben, wie seine bevorzugten Bildschirmeinstellungen oder die bevorzugte Lautstärke, oder Situations-abhängige Präferenzen. Diese Präferenzen können mittels Regeln angegeben werden, also *wenn* eine bestimmte Situation eintritt, *dann* soll eine Anpassung gemäß den Nutzervorgaben stattfinden. Beispiele für solche Regeln sind 'keine Auswahl von Inhalten während der Arbeitszeit' oder 'keine Anzeige von Audio in öffentlichen Verkehrsmitteln'.

14. *Erweiterbarkeit*
    Das Metadatenmodell muss erweiterbar sein und darf sich nicht auf vorher festgelegte Beschreibungen, insbesondere der Kontextinformationen beschränken. Da eine Standardisierung personalisierter multimedialer Dienste eher unwahrscheinlich ist und es in Zukunft neue Anwendungen geben wird, ist es notwendig, Erweiterungen des Datenmodells zuzulassen und zu erleichtern.

15. *Plattformunabhängigkeit*
    Die Beschreibungen der Inhalte sollen unabhängig von einer bestimmten Plattform sein. Dies ist besonders wichtig im Hinblick auf die starke Heterogenität mobiler Endgeräte und Betriebssysteme.

16. *Anwendungsunabhängigkeit*
    Da das im Rahmen dieser Arbeit entwickelte System möglichst generisch sein soll, um unterschiedliche Anwendungen realisieren zu können, muss auch die Sprache zur Beschreibung der Inhalte und Nutzer diese Anwendungsunabhängigkeit unterstützen.

3.3. Metadatenmodelle für multimediale Inhalte

Mittlerweile werden Metadaten in vielen unterschiedlichen Bereichen verwendet. Das Einsatzgebiet reicht von Bibliotheken über e-Learning Anwendungen bis zu Information Retrieval Systemen [157]. Im Rahmen dieser Arbeit sind jedoch nur solche relevant, die für die Beschreibung von multimedialen Daten und/oder deren Nutzer verwendbar sind.

Eine Vielzahl an Metadaten-Modellen für multimediale Inhalte wurde nur für einen sehr eingeschränkten Bereich entwickelt und ist kaum oder gar nicht anpassbar. Als Beispiel ist hier das *EXchangable Image File Format* (Exif) [94] zu nennen, das für die Beschreibung von Erzeugungsinformationen von Bild- und Audiodaten verwendet wird, oder *Identifier on mp3* (ID3) [128, 129] für Audiodaten, das zusätzlich die Nutzungshistorie der Datei abspeichert. Daher wird im Folgenden nur eine Auswahl der wichtigsten und vor allem allgemein einsetzbaren Multimedia-Metadatenstandards vorgestellt.

3.3.1. Tagging und Tag Clouds

In vielen Community-Portalen wie Flickr[1], MyVideo[2] oder Delicious[3] werden Inhalten so genannte *Tags* oder *Schlagwörter*, die von Nutzern angegeben werden, zugeordnet (siehe z. B. [131]). Um den Nutzer bei der Suche nach bestimmten Inhalten zu gegebenen Tags zu unterstützen, können *Tag Clouds* oder *Schlagwortwolken* eingesetzt werden. Dabei handelt es sich um eine Methode zur Informationsvisualisierung, die Tags alphabetisch sortiert und flächig und gewichtet darstellt. Diese Auflistung kann z. B. alle Tags beinhalten, nach denen Nutzer aktuell suchen, oder die Tags zu einem konkreten Inhalt. Je nach Häufigkeit sind die einzelnen Tags in unterschiedlicher Größe oder auch anders hervorgehoben dargestellt. Tag Clouds wurden wahrscheinlich 2002 erstmalig von Jim Flanagan in dem Plug-in *Zeitgeist* eingesetzt, das die eingegebenen Suchworte von unterschiedlichen Suchmaschinen visualisierte, die die Besucher zu einer Web-Seite geführt haben [75]. Darauf basierend wurde eine Version für Flickr als ersten Dienst mit hohen Nutzerzahlen implementiert.

Die Verwendung von Tags und Tag Clouds ermöglicht theoretisch eine vom konkreten Medientyp unabhängige semantische und informelle Beschreibung der Inhalte und deren Erzeugung, wenn die Tags geeignet gewählt werden. Selbiges gilt für Dateiinformationen. Grundsätzlich sind Tags aber nicht für diesen Zweck gedacht, sondern lediglich für eine grobe Kategorisierung. Da die Menge der Tags nur bis zu einer praktikablen Obergrenze erweitert werden kann, wird das System durch die Verwendung von Tags für die unterschiedlichen Arten von Beschreibungen unübersichtlich. Außerdem kann man so z. B. nicht mehr effizient zwischen einem semantischen Ort und dem Erzeugungsort unterscheiden. Derart ausführliche Beschreibungen, wie sie hier benötigt werden, können in der Praxis mit Tags kaum umgesetzt werden.

Tag Clouds beziehen sich rein auf den Inhalt, jegliche Art von nutzerbezogenen Daten wie Nutzerbeschreibungen, Nutzerpräferenzen, Netzbeschreibungen, Gerätedaten, Annotationen und Bewertungen werden nicht unterstützt. Sie sind für eine Verwendung im

[1] www.flickr.com
[2] www.myvideo.de
[3] www.delicious.com

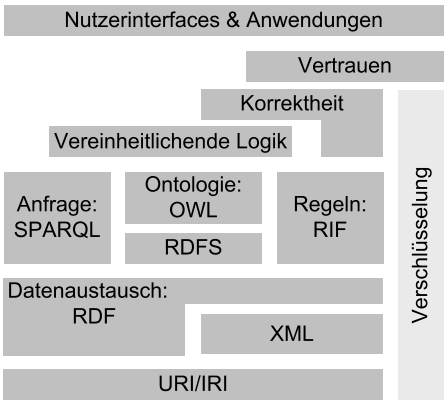

Abbildung 3.1.: Schichtenmodell des Semantic Web nach [174]

Web ausgelegt und können deshalb nicht für beliebige Anwendungsfälle genutzt werden.

3.3.2. Metadaten und das Semantic Web

Im Zuge der Weiterentwicklung des World Wide Web zum Semantic Web werden Metadaten benötigt, um Inhalte automatisch les- und interpretierbar zur Verfügung stellen zu können. Tim Berners-Lee fasst 2001 diese Bestrebungen in einem Modell zusammen (siehe Abbildung 3.1). Die einzelnen Schichten werden im Folgenden basierend auf [132], [133] und den jeweiligen Standards erläutert.

URI & IRI

Das gesamte Semantic Web basiert auf dem Namensschema, das vom *Uniform Resource Identifier* (URI) [14, 13] zur Verfügung gestellt wird. Er besteht aus einer Zeichenfolge aus einer Teilmenge des ASCII-Zeichensatzes, die zur Identifizierung einer physischen oder abstrakten Ressource dient. Der *Internationalized Resource Identifier* (IRI) [46] ist die internationalisierte Form der URI und erweitert den erlaubten Zeichensatz in URIs um fast alle Zeichen des Universal Character Set (Unicode).

XML und RDF

Auf der Identifikationsschicht der Ressourcen setzt die XML-Schicht auf, die die Basis des so genannten Dokumenten-Web bildet. Sie bietet eine grundlegende Syntax, um Ressourcen beschreiben zu können. Zusätzlich zu der Sprache XML [22, 23] beinhaltet diese Schicht weiterhin das XML Schema [175, 164, 15], die XML Namespaces [21], XPath [11], XPointer [71] und XLink [41]. Das derzeitige Web verwendet die syntaktischen Regeln

dieser Schicht, um selbstbeschreibende und von menschlichen Lesern verständliche Dokumentensprachen wie XHTML [49], die *Synchronized Multimedia Integration Language* (SMIL) [26] (vgl. Abschnitt 6.2.1) und *Scalable Vector Graphics* (SVG) [61] zu definieren.

Das *Resource Description Framework* (RDF) [116] ist eine Auszeichnungssprache für Metadaten im Web und erlaubt die Definition eines Datenmodells, das für den Datenaustausch verwendet werden kann. Es basiert auf so genannten RDF-Tripeln aus Subjekt, Prädikat und Objekt. Subjekte sind die eigentlichen Ressourcen wie Web-Seiten oder Bilder, die über eine URI (oder IRI) identifiziert werden. Ein Prädikat oder Eigenschaftselement gibt Auskunft über eine Ressource und stellt die Beziehung zu Objekten her. Sie werden durch das *Dublin-Core-Schema* [62] vorgegeben. Objekte beschreiben den Wert eines Prädikats. Dies kann ein Literal, also ein konkreter Wert, eine andere Ressource oder eine leere Ressource sein. Ein RDF-Datenmodell kann als XML-Dokument oder als Graph dargestellt werden.

RDF Schema, SPARQL, OWL und RIF

Um den in RDF definierten Metadaten eine Semantik zuordnen zu können, wurde die Sprache *RDF Schema* [24] spezifiziert, die RDF um hierarchische Strukturen für Prädikate und um die Klassifizierung von Ressourcen erweitert. Durch die Struktur von RDF Schema werden Anfragen auf semantischen Daten und Beziehungen ermöglicht.

Die *Simple Protocol and RDF Query Language* (SPARQL) [139] ist eine Anfragesprache für semantische Webressourcen. Sie erlaubt komplexe Anfragen, die aus Tripel-Pattern, Konjunktionen und Disjunktionen bestehen können.

Die *Web Ontology Language* (OWL) [136] bietet zusätzliches Vokabular um Prädikate und Klassen, Beziehungen von Klassen untereinander, Kardinalitäten, Gleichheit, erweiterte Typen und Charakteristika für Prädikate und nummerierte Klassen anzugeben. Es erlaubt Ausdrücke ähnlich der Prädikatenlogik zu definieren.

Das *Rule Interchange Format* (RIF) [143] soll zum Austausch von Regeln in Regelbasierten Systemen im Semantic Web dienen. RIF-Regeln beschreiben, wie man neue Informationen aus einer Ontologie ableiten, geeignet kombinieren oder manipulieren kann.

Unifying Logic, Proof und Trust

Die obersten Level zeigen das wichtigste Ziel des Semantic Web: Maschinen sollen nicht nur in der Lage sein, relevante Informationen zu finden und zu verarbeiten, sondern auch bewerten können, wie korrekt und vertrauensvoll die Daten sind. Dazu werden eine einheitliche Logik und Mechanismen zur Sicherstellung der Korrektheit und des Vertrauens bereit gestellt.

Prinzipiell können alle benötigten Arten der Beschreibungen der Inhalte unabhängig vom konkreten Medientyp, der Nutzer und des verwendeten Geräts und Netzes mittels

Web-Standards umgesetzt werden, da deren Konzepte sehr generisch, erweiterbar und für unterschiedliche Anwendungsfälle einsetzbar sind. Die gemäß den Anforderungen benötigten Informationen kann man relativ leicht mittels Erweiterung von XML definieren, semantische Informationen über RDF Schema beschreiben und mittels OWL verfeinern. Dennoch unterstützen die Semantic Web Standards per se keine multimedialen Inhalte und eine Verwendung dieser Standards in diesem Bereich erfordert einen erheblichen Entwicklungsaufwand.

3.3.3. Composite Capability/Preference Profiles und UAProf

Das vom W3C entwickelte Composite Capability/Preference Profile (CC/PP) [98] dient der Beschreibung von Geräteeigenschaften und Nutzerpräferenzen, die für die Anpassung der Darstellung von Inhalten verwendbar sind. Der Standard wurde entwickelt, um der steigenden Anzahl unterschiedlicher Geräte Rechnung zu tragen, die mittlerweile Zugriff auf das Web besitzen. Er soll als Beschreibungsformat für die Inhaltsanpassung und Kontextualisierung verwendet werden. CC/PP basiert auf dem Resource Description Framework (RDF) [116] (vgl. Abschnitt 3.3.2). RDF ist u. a. als Standard für den Datenaustausch im Web gedacht, weswegen es sich für den Austausch von Profilinformationen zwischen einem Nutzer oder seinem Nutzeragenten und einem Inhalteanbieter wie bei CC/PP vorgesehen gut eignet.

Ein CC/PP-Profil ist in einer Zwei-Schichten-Hierarchie aufgebaut: jedes Profil enthält mindestens eine Komponente (z. B. Browser oder Hardware) und jede Komponente hat mindestens ein zugehöriges Attribut, dem ein Wert zugeordnet ist (z. B. Browser-Version oder Displaygröße). Diese Information kann von einem Server verwendet werden, um die für den Client am besten geeignete Form einer Ressource zu bestimmen. Durch diese hierarchische Struktur entsteht ein Graph, der durch RDF gut dargestellt werden kann. Komponenten können auch durch eine Referenz auf ein externes Standardprofil angeben werden, auf das der Server zugreifen kann.

Die Menge von Attributnamen, erlaubten Werten und deren Semantik bilden das verfügbare CC/PP-Vokabular. Es wird davon ausgegangen, dass unterschiedliche Applikationen unterschiedliche Vokabulare verwenden. Damit unterschiedliche Anwendungen zusammenarbeiten können, müssen ein einheitliches Vokabular oder geeignete Konvertierungsmöglichkeiten vorhanden sein.

Bei der Entwicklung von CC/PP wurde auf Kompatibilität mit dem von der OMA standardisierten User Agent Profile (UAProf) [188] geachtet, das als Erweiterung des WAP 2.0 Standards gilt. UAProf ist eine durch das WAP Forum entwickelte Beschreibung von Geräte-Fähigkeiten, die speziell für mobile Endgeräte gedacht ist. Es basiert auf CC/PP, d.h. jede gültige UAProf-Profil ist auch ein gültiges CC/PP-Profil.

CC/PP und UAProf eignen sich gut für die Beschreibung von Geräteeigenschaften

und Nutzerpräferenzen bezüglich der Anzeige der Inhalte unabhängig vom Medientyp. Beschreibungen des Inhalts, der Dateiinformationen, des Netzes, detailliertere Nutzerbeschreibungen und Nutzerkommentare fehlen jedoch gänzlich. Beide Standards sind im Rahmen von RDF erweiterbar, aber nur bedingt anwendungsunabhängig, weil sie speziell für Webanwendungen definiert wurden. Da sie entwickelt wurden, um der steigenden Anzahl unterschiedlicher Geräte Rechnung zu tragen, die mittlerweile Zugriff auf das Web besitzen, wurden sie plattformunabhängig spezifiziert.

3.3.4. DVB-SI

Das 1993 gegründete Digital Video Broadcasting (DVB) Projekt hat sich zum Ziel gesetzt, einheitliche Richtlinien und offene Standards für die digitale Übertragung insbesondere von Fernseh- und Rundfunkangeboten zu erarbeiten. Im Rahmen dieser Bemühungen wurden Standards für unterschiedliche Übertragungsarten wie über Satellit (DVB-S), terrestrisch (DVB-T) oder für mobile Endgeräte (DVB-H) spezifiziert. Daneben wurde ein Standard zur Beschreibung von Zusatzdaten, die mit den eigentlichen Inhalten übertragen werden, festgelegt, die so genannte *Digital Video Broadcasting - Service Information* (DVB-SI) [47]. DVB-SI stützt sich auf vier Tabellen, die die Informationen über die elementaren Ströme des Programms (*Program Map Table*), die Liste aller Programme im Transportstrom (*Program Association Table*), die Organisation des Transportstroms im Netz (*Network Information Table*) und für Conditional Access notwendige Informationen für das Bezahlfernsehen (*Conditional Access Table*), beinhalten. Diese Informationen fasst man als *Program Specific Information* (PSI) zusammen. Neben diesen hauptsächlich für das Multiplexen benötigten Daten werden zusätzliche Informationen über Dienste und Ereignisse für den Empfänger bereitgestellt.

Die Informationen, die elektronischen Programmführern (Electronic Program Guides, EPGs) zugrunde liegen, befinden sich in der *Event Information Table* (EIT). Sie enthält Informationen wie die Anfangszeit und Dauer von Programmen oder verschiedene Deskriptoren. Zu den wichtigsten Deskriptoren gehören der *Short_ event_ descriptor* für eine Kurzbeschreibung bzw. der *Extended_ event_ descriptor* für eine ausführliche Beschreibung des Events. Der *Component_ descriptor* dient zur Beschreibung der wichtigsten charakteristischen Daten des Inhalts z. B. ob es sich um Video oder Audio handelt, die Auflösung, das Seitenverhältnis und die Bildwiederholfrequenz. Mit dem *Content_ descriptor* wird die Programmart bzw. das Genre der Sendung angegeben, wie Krimi, Unterhaltung oder Dokumentation. Der *Parental_ rating_ descriptor* enthält Beschreibungen über die Altersfreigabe. Grundsätzlich sieht der Standard vor, dass zwei EITs übertragen werden, wobei die erste alle Informationen über das aktuell laufende und über das folgende Event beinhaltet, und die zweite weitere Sendungen umfasst. Dabei ist ein Vorlauf für Sendungen von bis zu 64 Tagen möglich, aber in der Praxis wird lediglich nur bis zu sieben Tage im Voraus übertragen, da sich jeder zusätzliche Tag auf die Übertragungsrate auswirkt [59].

Die Deskriptoren innerhalb der EIT unterstützen rudimentär einige der Anforderungen wie die Kategorisierung, informelle Beschreibungen wie die Altersgruppe oder grobe Dateiinformationen über den Inhalt. Semantische und informelle Beschreibungen, Kontextinformationen, sowie Beschreibungen über die Erzeugung können zwar innerhalb des *Extended_ event_ descriptor* als Fließtext fest gehalten, jedoch nicht effizient verarbeitet werden. Unterstützt werden alle mittels DVB übertragenen Medientypen wie Fernseh- und Radioinhalte, Teletext und EPGs. Nutzerbeschreibungen, Anpassungspräferenzen, Beschreibungen des Netzes und Gerätedaten fehlen gänzlich, ebenso wie Feedbackmöglichkeiten des Nutzers, die für Annotationen oder Bewertungen benötigt werden. Erweiterungsmöglichkeiten für DVB-SI sind nicht vorgesehen und es ist nur für die Verwendung in Kombination mit DVB gedacht. Es kann auf allen DVB unterstützenden Plattformen verwendet werden.

3.3.5. MPEG-7

Das von der Moving Pictures Expert Group entwickelte *Multimedia Content Description Interface*, auch *MPEG-7* (ISO/IEC 15398) [87, 83, 115, 88] genannt, dient im Gegensatz zu den bekannteren Kompressionsstandards MPEG-1, -2 oder -4 zur formalen Beschreibung multimedialer Inhalte. Der Standard konzentriert sich ausschließlich auf die Beschreibung der Daten, wie diese erzeugt bzw. verwendet werden ist nicht von Interesse. MPEG-7 ist nicht auf eine bestimmte Applikation zugeschnitten sondern generisch gehalten, soll eine große Menge von Anwendungsgebieten unterstützen können und einfach erweiterbar sein. Ebenso ist der Standard unabhängig von einem bestimmten Übertragungsmedium.

Zu den Hauptelementen von MPEG-7 gehören die *Beschreibungswerkzeuge (Description Tools)*, die *Description Definition Language (DDL)* und die *Systemwerkzeuge (System Tools)*. Der Zusammenhang dieser Elemente ist in Abbildung 3.2 dargestellt. Die Beschreibungswerkzeuge bestehen aus *Deskriptoren (Descriptors, D)* und *Beschreibungsschemata (Description Schemes, DS)*. Ein D repräsentiert die Merkmale eines audiovisuellen Inhalts und legt dessen Syntax und Semantik fest. Merkmale werden in MPEG-7 durch Ds auf eine Menge von Werten reduziert. Die Struktur und Semantik von Beziehungen zwischen MPEG-7 Komponenten wird von DSs festgelegt. Die so verbundenen MPEG-7 Komponenten können Ds oder weitere DSs sein. Durch DSs wird das Datenmodell einer Beschreibung festgelegt. Die Syntax von Ds und DSs wird von der DDL definiert. Diese erlaubt auch die Erweiterung und Modifikation von Standard-Ds oder DSs. Eine Instanziierung der Ds und der zugehörigen DSs auf Basis der DDL ergibt eine Beschreibung in textueller Form basierend auf XML. Die System Tools unterstützen ein binär kodiertes Format der Beschreibung für die effiziente Speicherung bzw. Übertragung.

Der MPEG-7 Standard ist in mehrere Teile gegliedert. Im Hinblick auf die Anforderungen relevant ist Teil 5, das *MPEG-7 Multimedia Description Scheme* (MDS), weswegen es im Folgenden näher vorgestellt wird.

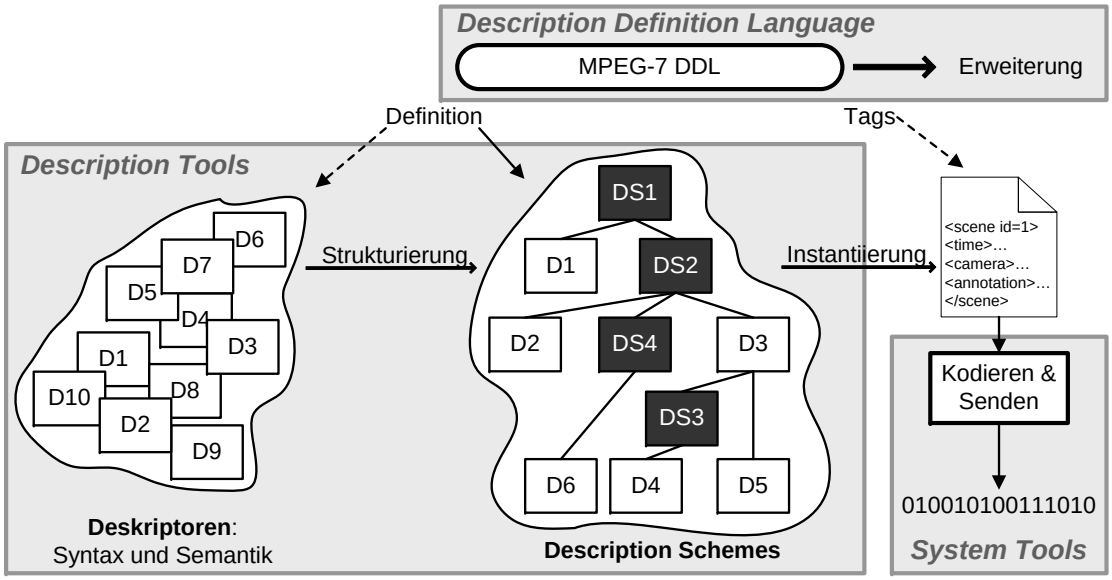

Abbildung 3.2.: Hauptelemente von MPEG-7 nach [87]

Im MDS werden die Beschreibungswerkzeuge zusammengefasst, die generische, also nicht speziell nur für auditive oder visuelle Medien relevante, multimediale Eigenschaften definieren. Die vordefinierten Beschreibungstools können gemäß ihrer Funktionalität den Klassen *Content Description*, *Content Management*, *Content Organization*, *Navigation & Access* und *User Interaction* zugeordnet werden (vgl. Abbildung 3.3). Diese setzen auf der Menge der Basiselemente auf, die grundlegende Datentypen (Vektoren, Matrizen, Zahlen etc.), Konstrukte zur Verlinkung und Lokalisierung von Inhalten und weitere, nicht Multimedia-spezifische Beschreibungselemente für Informationen wie z. B. Zeit, Ort, Personen, Organisationen und textuelle Anmerkungen enthalten. Weiterhin gehören hierzu Schema-Werkzeuge, die die Möglichkeit zur Formatierung, Verpackung und Annotation von MPEG-7 Beschreibungen bieten. Sie bilden eine Art Wrapper für die Beschreibungen und erlauben die Gruppierung von ähnlichen Beschreibungswerkzeugen.

Die *Content Description* enthält alle wahrnehmbaren Informationen der multimedialen Daten, also Beschreibungen der Struktur und der Semantik. Die Struktur kann unter Berücksichtigung des entsprechenden Medientyps entweder durch räumliche, zeitliche oder auch raum-zeitliche Segmente angegeben werden, die hierarchisch angeordnet werden (z. B. Videosegmente und Frames). Das erlaubt die Erstellung von Indexen oder Inhaltsverzeichnissen für den leichteren Zugriff auf die Daten. Die Semantik des multimedialen Inhalts kann durch Objekte, Ereignisse und Konzepte beschrieben werden.

Mit Hilfe der *Content Management* Werkzeuge können Beschreibungen über die Medienerzeugung angegeben werden. Dazu gehören das Datum und der Ort der Erzeugung

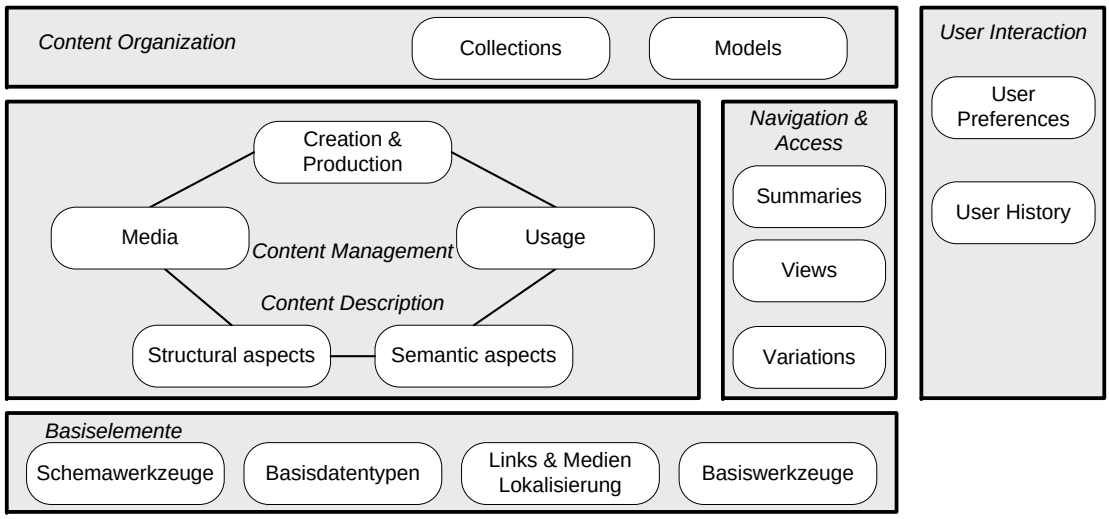

Abbildung 3.3.: Überblick über MPEG-7 Multimedia DSs inklusive der Basiselemente nach [87]

oder der Erzeuger selbst. Durch Informationen wie Genre, Sprache oder die erlaubte Altersklasse des Zuschauers kann der Inhalt klassifiziert werden. Weiterhin kann man auf verwandte Mediendateien verlinken. Zum Content Management gehören auch allgemeine Informationen über die Medieneigenschaften wie Qualität, Kompression oder verwendeter Codec. Um Hinweise zur Verwendung der Inhalte zu geben, können Nutzungsinformationen und die Nutzungsrechte angegeben werden. Die eigentlichen Nutzungsrechte werden aber nicht in MPEG-7 spezifiziert, sondern durch eine Referenz auf den Rechteinhaber. Die Sicherstellung der Einhaltung dieser Rechte ist nicht im Fokus von MPEG-7.

Die *Content Organization* erlaubt die Erzeugung und Modellierung von Sammlungen von verschiedenen Medieninhalten bzw. Medienbeschreibungen.

Kompakte Zusammenfassungen multimedialer Daten, die durch Werkzeuge aus *Navigation & Access* erzeugt werden, ermöglichen u. a. ein leichteres Auffinden von Inhalten oder die Navigation innerhalb der Mediendateien. Weiterhin können durch Partitionierungen bzw. Dekompositionen von audio-visuellen Signalen in Raum, Zeit oder Frequenz verschiedene Sichten auf die Daten geliefert werden. Ebenso kann hier angegeben werden, wenn für den selben Inhalt verschiedene Variationen zur Verfügung stehen, z. B. in unterschiedlichen Sprachen oder Auflösungen.

Mittels der Werkzeuge der *User Interaction* können Nutzerpräferenzen zur personalisierten Filterung und die Nutzungshistorie des Inhalts definiert werden. Dies erlaubt z. B. den Abgleich zwischen Präferenz und MPEG-7 Beschreibungen um Inhalte personalisiert dem Nutzer zur Verfügung zu stellen.

Da MPEG-7 speziell für die Beschreibung multimedialer Daten unterschiedlichen Typs spezifiziert wurde, erfüllt es einen großen Teil der aufgeführten Anforderungen. Semantische Beschreibungen können mit den umfangreichen Werkzeugen des `Semantic DS` angegeben werden, Beschreibungen der Erzeugung eines Inhalts werden durch das `Creation DS` ermöglicht und eine umfangreiche Kategorisierung ist unter Verwendung des `Classification DS` umsetzbar. Informationen über die Datei, in der ein Inhalt vorliegt, können umfassend definiert werden. Dazu gehören auch Informationen über die Qualität, in der ein Inhalt verfügbar ist.

MPEG-7 erlaubt umfangreiche Nutzerbeschreibungen. Diese umfassen demographische Daten, Nutzerpräferenzen bezüglich der Auswahl der Inhalte und die Nutzungshistorie, da der Standard explizit für die Verwendung in Empfehlungssystemen gedacht ist. Beschreibungen des Endgeräts oder von Anpassungspräferenzen fehlen hingegen vollständig.

Einige informelle Beschreibungen werden ebenfalls vom Standard unterstützt, darunter Angaben über die Nutzungsrechte des Inhalts oder die Altersfreigabe, allerdings keine dynamischen Informationen wie die aktuelle Auslastung eines Kinosaals.

MPEG-7 reserviert eigene Werkzeuge zur Verwaltung von Annotationen, bei diesen handelt es sich aber ausschließlich um allgemeine Annotationen, die für alle Nutzer gültig sind. Ebenso verhält es sich mit Bewertungen. Die Kommentierung oder Bewertung von Inhalten durch die Nutzer ist in MPEG-7 nicht vorgesehen.

Der Standard sieht die Möglichkeit vor, dass zu einem Inhalt (und damit einer Beschreibung) mehrere Dateien vorliegen, die je nach Netzwerk und dessen aktuellem Zustand verwendet werden können. Es werden allerdings ausschließlich die unterschiedlichen Dateien beschrieben, nicht jedoch das Netz.

Innerhalb der semantischen Beschreibung und der Beschreibung der Erzeugung der Inhalte erlaubt MPEG-7 bereits die Angabe grundlegender dynamischer Kontextinformationen der Inhalte wie den Ort der Handlung oder den Erzeugungsort. Die dynamischen Kontextinformationen des Nutzers werden nicht berücksichtigt.

Der Standard wurde bewusst sehr generisch spezifiziert und soll per se unterschiedliche Anwendungsfälle unterstützen. Ebenso wurde auf Plattformunabhängigkeit geachtet. Da das MPEG-7 Schema als Erweiterung von XML entwickelt wurde, kann es leicht erweitert werden.

3.3.6. MPEG-21

Der ebenfalls von der Moving Pictures Expert Group entwickelte Standard MPEG-21 [86, 84, 85] realisiert ein einheitliches Framework zur Erstellung, Manipulation, Verteilung und Konsumierung von Multimediainhalten sowie zur Integration von Inhalten in verschiedenen Formaten. Ein wichtiger Bestandteil ist die digitale Rechteverwaltung (Digital Rights Management, DRM).

MPEG-21 basiert auf zwei fundamentalen Konzepten: der Definition des Digital Item (DI; das *what* des Frameworks), einer grundlegenden Einheit für die Verbreitung und

Verarbeitung, und dem Nutzer (das *who* des Frameworks), der mit diesem interagiert. Das DI ist ein strukturiertes digitales Objekt mit einer standardisierten Repräsentation und Identifikation inklusive der zugehörigen Metadaten. Ressource und Metadaten werden in einer Struktur, der *Digital Item Declaration* (DID), eingebettet, die diese in Beziehung zueinander setzt. Ursprünglich sah der Entwicklungsprozess des MPEG-21-Standards eine weitgehende Automatisierung der Verwaltung und Nutzung multimedialer Inhalte vor. Da MPEG-21 aber eine Vielzahl an Anwendungsszenarien unterstützt, führt dies zu einer geringen Akzeptanz für sehr spezifische Einsatzgebiete. Aus diesem Grund wurde dieser letzte Schritt nicht mehr umgesetzt [165].

Der Standard ist in mehrere Teile gegliedert; für die Themenstellung relevant ist Teil 7, *Digital Item Adaption* (DIA). Diese setzt sich aus drei Hauptkategorien zusammen: *Usage Environment Description*, *Digital Item Resource Adaptation* und *Digital Item Declaration Adaptation*.

Innerhalb der *Usage Environment Description* werden Werkzeuge zur Beschreibung der Nutzungsumgebung angegeben. Dazu gehören die Fähigkeiten des Endgeräts, die Charakteristika des verwendeten Netzwerks, die Beschreibung der natürlichen Umgebung während der Nutzung und eine Beschreibung des Nutzers inklusive seiner Präferenzen. MPEG-21 integriert hierfür die Nutzerbeschreibungen aus dem MPEG-7-Standard.

Die *Digital Item Resource Adaptation* zielt auf die Anpassung von Ressourcen in einem DI ab. Dafür stellt es u. a. Werkzeuge zur Unterstützung von Dienstgüte-Management und zur Anpassung der Metadaten zur Verfügung.

Die *Digital Item Declaration Adaptation* stellt Werkzeuge zur Anpassung eines kompletten DID zur Verfügung. Dazu gehört die Übertragung von Sitzungen auf andere Endgeräte (Session Mobility) und die Parameter, die von der Komponente, die die eigentliche Anpassung der Inhalte durchführt, hierfür benötigt werden.

MPEG-21 hat einen starken Fokus auf den Zugriff und die Anpassung von multimedialen Inhalten unabhängig von ihrem Medientyp, nicht jedoch auf deren genaue Beschreibung. Daher findet man auch umfangreiche Informationen über die Gerätedaten, das Netz und Dateiinformationen, während jegliche Beschreibungen der Inhalte und auch die Möglichkeit der Annotation oder Bewertung gänzlich fehlen. Zur Beschreibung des Nutzers verwendet MPEG-21 die von MPEG-7 spezifizierten Beschreibungen und erweitert diese um eine Beschreibung der Nutzungsumgebung.

MPEG-21 erlaubt einige recht umfangreiche Beschreibungen von dynamischen Kontextinformationen wie die aktuelle Position, die Zeit, vorherrschende Lichtverhältnisse oder die momentane Umgebungslautstärke. Dennoch müssen diese erweitert werden, um alle gewünschten Kontextinformationen berücksichtigen zu können. Weiterhin sind Beschreibungen der Eigenschaften der Netzinfrastruktur möglich. Bei den Anpassungspräferenzen werden nur allgemeine und keine Situations-spezifischen unterstützt.

Ähnlich wie MPEG-7 wurde MPEG-21 sehr generisch und anwendungsunabhängig

spezifiziert. Ein primäres Ziel der Entwicklung war die Plattformunabhängigkeit. Da viele Teile von MPEG-21 ebenfalls als Erweiterung von XML entwickelt wurden, sind Erweiterungen des Standards leicht umsetzbar.

3.3.7. TV-Anytime

Im Jahr 1999 wurde von unterschiedlichen Rundfunkorganisationen, Telekommunikationsanbietern und Elektronikherstellern das TV-Anytime Forum gegründet, das Standards für eine kontrollierte Übertragung von Fernsehinhalten auf die persönlichen Endgeräte des Nutzers wie z. B. einen Personal Videorecorder (PVR) schaffen wollte. Es versucht die raschen Entwicklungen im Bereich der Nutzung und Speicherung von digitalen Informationen auszuschöpfen und dem Zuschauer ein stark personalisiertes TV-Erlebnis zu bieten. Solche Systeme nehmen eine immer wichtigere Rolle ein: so werden z. B. PVRs in den USA mit dem System TiVo (Television Input / Video Output)[4] der Firma TiVo Inc. bereits breitflächig eingesetzt und sind mittlerweile auch schon in anderen Ländern wie Kanada, Korea oder Taiwan verfügbar. TiVo bietet u. a. die Möglichkeit zum zeitversetzten Fernsehen und eine intelligente Lernfunktion, die Fernsehsendungen abgestimmt auf den Geschmack des Nutzers aufzeichnet. Im Rahmen des TV-Anytime Forums entstand ein Standard, der von der ETSI in mehreren Teilen veröffentlicht wurde [52, 54, 55, 56, 57, 51, 53, 58, 50].

Das vom TV-Anytime-Forum entwickelte Metadatenformat wird insbesondere bei der kontrollierten Übertragung von multimedialen Inhalten auf einen so genannten digitalen Videorekorder (digital video recorder, DVR) verwendet oder für die Anzeige in einem personalisierten, elektronischen Programmführer (electronic program guide, EPG) und Inhaltsfinder. Die Metadaten von TV-Anytime sind in vier Elemente eingeteilt. *Content Description Metadata* enthält allgemeine Informationen über den eigentlichen Inhalt. *Instance Description Metadata* repräsentiert die unterschiedlichen Instanzen desselben Inhalts. *Consumer Metadata* beschreibt den Nutzer inklusive seiner demographischen Daten, seiner Präferenzen und seiner Nutzungshistorie. *Segmentation Metadata* umfasst die einzelnen Segmente, die bei der Übertragung der Inhalte entstehen können. Da MPEG-7 (siehe Abschnitt 3.3.5) bereits umfangreiche Beschreibungen audiovisueller Daten bietet, wurden viele Teile des Standards für TV-Anytime wiederverwendet und um eigene Sets z. B. zur Beschreibung der Atmosphäre eines Inhalts und zur umfangreichen Kategorisierung der Inhalte anhand des Genres erweitert. Einige Veränderungen sind jedoch in der Syntax und Darstellung der einzelnen Elemente gemacht worden, weswegen TV-Anytime nicht voll kompatibel zu MPEG-7 ist. Zusätzlich übernimmt TV-Anytime auch noch Beschreibungen aus dem DVB-SI-Standard (vgl. Abschnitt 3.3.4).

Da TV-Anytime Metadaten-Elemente direkt aus MPEG-7 übernimmt, verhält es sich

[4] www.tivo.com

ähnlich in Bezug auf die Anforderungen: Beschreibungen der Erzeugung eines Inhalts, informelle Beschreibungen und Nutzungsdaten sind zumindest teilweise vorhanden. Für eine Kategorisierung entwickelte TV-Anytime ein umfangreiches hierarchisches Schema. Semantische Beschreibungen und jegliche dynamische Kontextinformationen fehlen hingegen vollends.

TV-Anytime wird meist auf Plattformen ohne Rückkanal verwendet, weswegen umfangreiche Nutzerbeschreibungen oder Feedbackmechanismen nicht unterstützt werden. Die einzelnen Nutzer können über einen lokalen Identifikator leicht unterschieden werden.

Da der Standard ausschließlich Fernsehinhalte berücksichtigt, stehen nur Dateiinformationen für diese Art von multimedialen Inhalten und keine Beschreibungen des Netzes zur Verfügung. Die Geräte und Anpassungspräferenzen können nicht beschrieben werden. Es ist nicht vorgesehen, den Standard für andere Anwendungsbereiche zu erweitern. Da die Metadaten jedoch in XML spezifiziert wurden, können sie zumindest theoretisch erweitert werden und sind plattformunabhängig.

3.3.8. Fazit

Wenn man die Anforderungen an ein Metadatenmodell für die Kontext-abhängige Personalisierung von multimedialen Inhalten aus Abschnitt 3.2 mit den hier vorgestellten Ansätzen vergleicht, erkennt man, dass sie bei keinem ausreichend erfüllt werden. Tabelle 3.1 fasst die Beurteilungen zusammen, wobei ein + anzeigt, dass eine Anforderung erfüllt wird, ein –, dass sie nicht erfüllt wird, und ein o bedeutet, dass eine Anforderung nur teilweise oder nur mit Anpassungen des jeweiligen Standards erfüllt wird.

Während eine mehr oder weniger grobe Kategorisierung unabhängig vom Medientyp mit den meisten Ansätzen möglich ist, werden detaillierte Beschreibungen der Inhalte gemäß semantischer Informationen oder der Erzeugung meist nicht unterstützt. MPEG-7 bietet hierfür einen umfangreichen Ansatz. Informelle Beschreibungen sind bei keinem Standard vollständig vorhanden und insbesondere dynamische Daten fehlen.

Das Vorhandensein von umfangreichen Nutzerbeschreibungen hängt jeweils vom angedachten Interaktionsgrad des Nutzers mit dem System zusammen. Sollen Metadaten in Systemen mit umfassenden Feedbackmöglichkeiten eingesetzt werden, so sind Nutzerbeschreibungen vorhanden (z. B. MPEG-7), ansonsten sind sie nur rudimentär bis gar nicht verfügbar (z. B. TV-Anytime). Selbiges gilt für Gerätebeschreibungen, wobei diese umso umfassender ausfallen, je mehr ein System die Inhalte an diese Daten anpasst (z. B. CC/PP oder MPEG-21). Die Nutzerpräferenzen für diese Anpassung werden, wenn überhaupt, nur sehr allgemein unterstützt, insbesondere Situations-abhängige Anpassungsregeln fehlen vollständig. Beschreibungen des Netzes unterstützt ausschließlich MPEG-21. Feedbackmöglichkeiten wie sie für Annotationen oder Bewertungen durch einen Nutzer benötigt werden, werden dagegen überhaupt nicht unterstützt.

Die eher allgemeineren Ansätze wie Tagging bzw. Tag Clouds oder die Standards des Semantic Web sind auf Grunde ihrer mangelnden Berücksichtigung von multimedialen

Anforderung	Tag Clouds	Semantic Web	CC/PP & UAProf
1. Beschreibung unterschiedlicher Medientypen	+	o	+
2. Semantische Beschreibungen	o	o	–
3. Informelle Beschreibungen	o	o	–
4. Beschreibung der Erzeugung	o	o	–
5. Kategorisierung der Inhalte	+	o	–
6. Dateiinformationen	o	o	–
7. Nutzerdaten	–	o	–
8. Gerätedaten	–	o	+
9. Nutzungsdaten	–	o	–
10. Beschreibung des Netzes	–	o	–
11. Annotationen	–	–	–
12. Zeit-abhängige Bewertungen	–	o	–
13. Beschreibung von Anpassungspräferenzen	–	o	o
14. Erweiterbarkeit	o	+	o
15. Plattformunabhängigkeit	o	+	+
16. Anwendungsunabhängigkeit	–	+	o

Anforderung	DVB-SI	MPEG-7	MPEG-21	TV-Anytime
1. Beschreibung unterschiedlicher Medientypen	o	+	+	–
2. Semantische Beschreibungen	o	+	–	–
3. Informelle Beschreibungen	o	o	–	o
4. Beschreibung der Erzeugung	o	+	–	+
5. Kategorisierung der Inhalte	+	+	–	+
6. Dateiinformationen	+	+	–	o
7. Nutzerdaten	–	+	+*	–
8. Gerätedaten	–	–	+	–
9. Nutzungsdaten	–	+	+*	+
10. Beschreibung des Netzes	–	–	+	–
11. Annotationen	–	–	–	–
12. Zeit-abhängige Bewertungen	–	–	–	–
13. Beschreibung von Anpassungspräferenzen	–	–	o	–
14. Erweiterbarkeit	–	+	+	o
15. Plattformunabhängigkeit	+	+	+	+
16. Anwendungsunabhängigkeit	–	+	+	–

* ganz oder teilweise aus dem MPEG-7 Standard übernommen

Tabelle 3.1.: Vergleich von verwandten Metadatenmodellen

Inhalten für die Verwendung in dieser Arbeit nicht geeignet. CC/PP und UAProf, DVB-SI und TV-Anytime sind zu stark auf ihren Anwendungsfall spezialisiert und würden zu umfangreiche Änderungen nach sich ziehen, was z. T. nicht einmal unterstützt wird.

Eine gute Ausgangsbasis für die weitere Entwicklung eines Metadatenmodells für die Kontext-abhängige Personalisierung von multimedialen Inhalten bieten hingegen MPEG-7 und MPEG-21. Bei genauerem Blick auf die Tabelle zeigt sich, dass Schwächen des einen Standards durch den anderen ausgeglichen werden. Beide Standards sind auch dank XML leicht erweiterbar. Deshalb werden für die Metadatensprache, die im Folgenden vorgestellt wird, Teile von MPEG-7 und MPEG-21 wiederverwendet, wo nötig die beiden Standards geeignet erweitert und zusätzlich benötigte Teile neu entwickelt.

3.4. Multimedia Adaptation and Selection Language (MASL)

Im Folgenden wird die Multimedia Adaptation and Selection Language (MASL) vorgestellt, die sowohl für eine personalisierte Auswahl der Inhalte als auch für eine personalisierte Anpassung der Darstellung multimedialer Inhalte entwickelt wurde [182]. Zum Sprachumfang von MASL gehören neben Sprachelementen aus dem MPEG-7- und MPEG-21-Standard auch passende Erweiterungen. Im Folgenden wird der Aufbau der Inhaltsbeschreibungen und des Nutzerprofils in MASL erläutert. Dafür werden die aus MPEG-7 und MPEG-21 übernommenen Elemente und die für die Kontext-abhängige Personalisierung benötigten Erweiterungen näher vorgestellt. Die hinter den beschriebenen MPEG-7 bzw. MPEG-21 Elementen und Typen in Klammern angegebenen Zahlen dienen als Referenz auf die entsprechenden Teile des MDS [83] bzw. DIA [84], in dem sie ursprünglich definiert wurden, die Listings beziehen sich auf die jeweiligen Definitionen im Anhang.

3.4.1. Aufbau der Inhaltsbeschreibungen in MASL

Eine Inhaltsbeschreibung in MASL wird mit dem Wurzel-Element `Metadata` begonnen. Unter diesem können sich sieben unterschiedliche Sub-Typen befinden (vgl. Abbildung 3.4 und Listing A.1), die im Folgenden näher erläutert werden. Die Schema-Definition von MASL sowie zwei Beispielbeschreibungen von Inhalten sind in Anhang A.1 bzw. B.1 zu finden.

ProgramIdentifier

Mit dem `ProgramIdentifier`-Element wird in MASL der eindeutige Bezeichner des beschriebenen Inhalts angegeben. Er dient zur Unterscheidung der einzelnen Inhalte und zum Herstellen von Querbezügen innerhalb einer Beschreibung und zum Nutzerprofil.

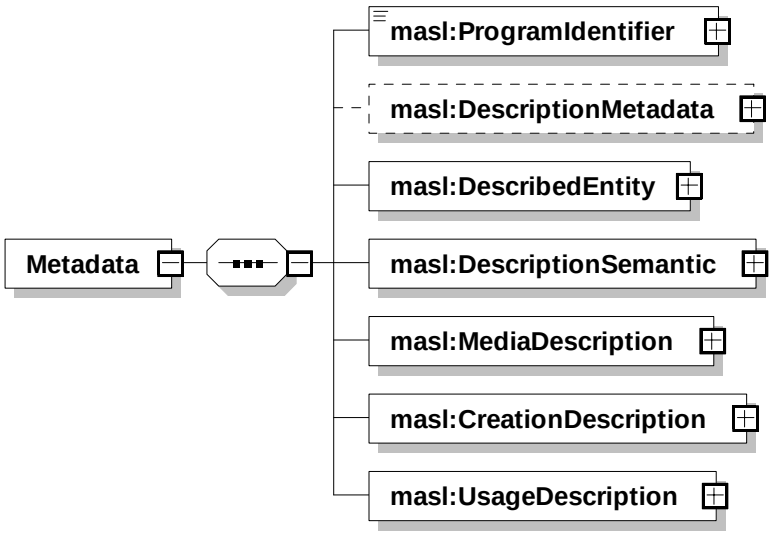

Abbildung 3.4.: Aufbau der Inhaltsbeschreibungen in MASL

Das Element verwendet den MPEG-7-Typ `UniqueIDType` (6.3.1), der die Angabe von Identifikatoren als Text oder Binärwert in den Formaten `base16` oder `base64` erlaubt.

DescriptionMetadata

Das optionale `DescriptionMetadata`-Tag gibt die Metadaten der Beschreibung an. Dafür wurde der `DescriptionMetadataType` (4.5.3) aus dem MPEG-7-Standard in MASL übernommen. Auf diese Weise können z. B. die Versionsnummer der Beschreibung, die Korrektheit, das Datum der letzten Änderung und Kommentare zur Beschreibung angeführt werden. Außerdem kann beschrieben werden, wie und von wem die Beschreibung erzeugt wurde und welche Nutzungsrechte ihr zugeordnet wurden. Dieselben Informationen können größtenteils auch für die Beschreibung des eigentlichen Inhalts verwendet werden; sie werden in den folgenden Abschnitten näher erläutert.

DescribedEntity

Der Medientyp des beschriebenen Inhalts wird in MASL innerhalb des `DescribedEntity`-Tags angegeben. Es übernimmt den `MultimediaContentType` (4.4.5) aus MPEG-7, der abstrakt eine Multimedia-Entität beschreibt. Dieser Typ wird um Medien-spezifische Charakteristika für z. B. Audio, Video, Bilder oder Kombinationen einzelner Medientypen erweitert. Innerhalb der Beschreibungen wird die räumliche und zeitliche Dekomposition der Inhalte, also die Zerlegung eines kompletten Inhalts in unterschiedliche Teilelemen-

te vom Typ `SegmentType` (11.2.2) berücksichtigt. Jedem Teilelement können einzelne Beschreibungen zugeordnet werden, wie semantische Beschreibungen, Dateiinformationen oder Beschreibungen des Erzeugungsprozesses und der Verwendung. Dieselben Teile werden auch für den kompletten multimedialen Inhalt verwendet und in den folgenden Abschnitten näher erläutert.

Als Medientyp erlaubt MPEG-7 die Beschreibung einer umfangreichen Menge an Medientypen, nicht jedoch Texte an sich. Aus diesem Grund wurde in MASL der `TextType` (Listing A.28) neu definiert. Dieser enthält ein `Text`-Element vom Typ `TextSegmentType` (Listing A.27) und erlaubt die räumliche Dekomposition von Inhalten u. a. auch in Texte. Hier können analog zu den anderen Typen, die vom `SegmentType` erben, neben den oben aufgeführten Beschreibungsmöglichkeiten auch Text-spezifische Informationen wie die Schriftgröße (`FontSize`), die Schriftfarbe (`FontColor`) und die Schriftart (`FontType`) angegeben werden. Texte wurden weiterhin in das `Content CS` (Listing A.35) aufgenommen, das zur Beschreibung des Medientyps des Inhalts verwendet werden kann.

SemanticDescription

MPEG-7 bietet mit den *Semantics Description Tools* (12) ein umfangreiches Werkzeug, das die semantische Beschreibung von Inhalten erlaubt. Dies ermöglicht die Erfassung von Konzepten der realen oder beschriebener Welten mit allen zugehörigen Objekten, Ereignissen, Orten, Zeiten etc. Semantische Elemente können auch Beziehungen zwischen den Elementen beschreiben, was zu einer komplexen Ontologie führt. Zu den wichtigsten Konzepten gehören die Begriffe *Erzählwelten* (narrative worlds) und *Abstraktion* (Abstraction). Eine Erzählwelt beinhaltet alle Informationen, die eine Erzählung oder die reale Welt ausmachen. Dazu gehören u. a. der Kontext, der Hintergrund und die Mitwirkenden. Ein multimedialer Inhalt kann mehrere Semantiken beinhalten und umgekehrt. Da es möglich ist, dass eine Entität ihren Status oder ihre Funktionalität innerhalb einer Erzählwelt ändert, kann sie durch verschiedene Semantiken beschrieben werden. So kann z. B. ein Bild ein Objekt sein (wenn es an einer Wand hängt), ein Ort (bei der Beschreibung von Objekten im Bild), oder selbst zu einer Erzählwelt werden (bei der Beschreibung der Szene eines bestimmten Bildes). Die Abstraktion bezieht sich auf die Generalisierung einer semantischen Beschreibung einer Instanz. Damit kann man aus einer konkreten Beschreibung eine abstrakte Beschreibung machen, die sich z. B. nicht auf eine konkrete Instanz bezieht, sondern als Beschreibung für mehrere Dateien steht oder in denen ein konkreter Ausdruck (‘Der Mann mit Namen Alex‘) durch eine Variable ersetzt wird (‘Ein beliebiger Mann‘ oder ‘ein beliebiger Mensch‘).

Das `Semantic DS` beschreibt die Erzählwelten, die mit einem multimedialen Inhalt in Zusammenhang stehen. Die `SemanticBase DS` ist ein abstraktes Werkzeug, das als Basis für die Werkzeuge zur Beschreibung der semantischen Entitäten dient. Dieses kann spezialisiert werden um z. B. Objekte, Ereignisse, Konzepte, Zustände etc. zu beschreiben. Jeder Entität wird ein Abstraktionslevel zugeordnet. Semantische Entitäten und

ihre Beziehungen (`SemanticRelation CS`) werden mittels `SemanticBag DS` zusammengefasst. Elemente des `SemanticRelation CS` sind fest durch den MPEG-7-Standard vorgegeben und erlauben die Bildung von komplexen Beschreibungsgraphen, die die Inhalte repräsentieren.

Die komplette Untermenge des `Semantic DS` (12.3.4) ist für die personalisierte Auswahl von Inhalten geeignet und wurde deswegen in MASL übernommen. So können z. B. Inhalte basierend auf den favorisierten fiktiven Charakteren ausgewählt werden, auf Grund des Orts, an dem die Handlung spielt (der nicht mit dem eigentlichen Drehort übereinstimmen muss, welcher über die `CreationDescription` angegeben ist, siehe unten) oder der fiktiven Zeit. Vorstellbar ist auch eine Auswahl unter Berücksichtigung von Konzepten (verwendet als Schlüsselwörter oder Tags für die Inhalte), bestimmten Zuständen der Entitäten wie z. B. Trauer oder Freude bei fiktiven Personen, vorkommenden Objekten oder bestimmten Ereignissen. Probleme beim Abgleich können allerdings auftreten, weil nicht wie bei anderen DS eine fest vorgegebene Menge an zu verwendenden Literalen zur Verfügung steht, sondern frei formuliert werden kann. Daher werden Mechanismen benötigt, die Homonyme und Synonyme geeignet erkennen können. Beispiel hierfür werden in Abschnitt 4.1 diskutiert, eine nähere Betrachtung ist jedoch nicht Teil dieser Arbeit.

MediaDescription

Für die personalisierte Darstellung ist es wichtig, dass Dateiinformationen vorliegen. Diese Informationen werden in MPEG-7 mittels `MediaInformation DS` (8) beschrieben, das in MASL integriert wurde. Es enthält die Sub-Elemente `MediaIdentification` und `MediaProfile`. Über `MediaIdentification` kann die Ressource identifiziert werden. `MediaProfile` beinhaltet Informationen wie die Dateigröße und das Dateiformat, und Medien-abhängige Informationen wie die Auflösung bei Bildern oder die Framerate bei Videos. Da das MDS bereits 2003 spezifiziert wurde, fehlte im `Medium CS`, das das physische Speichermedium angibt, die Unterstützung von BluRay, weswegen es um diese erweitert wurde (Listing A.40). Ein wichtiger Aspekt bei der Auswahl der Inhalte ist die Qualität. Sie kann als Teil der `MediaInformation` in MPEG-7 spezifiziert werden. In der Regel werden Inhalte mit höherer Qualität im Vergleich mit Inhalten von niedriger Qualität vom Nutzer bevorzugt.

Um die in MASL zusätzlich unterstützten Texte (s.o.) berücksichtigen zu können, musste der von MPEG-7 definierte `MediaIdentificationType` geeignet zum `MediaIdentificationExtType` (Listing A.15) erweitert werden. Dafür wurde das in Listing A.41 aufgeführte `TextDomain CS` entwickelt. Dieses beinhaltet die Spezifikation der Quelle des Texts (`Ink` für Eingabe über ein Touchpad, `Keyboard` für Eingaben über eine Tastatur, `Scanned Document` für eingescannte Texte). Weiterhin kann noch die Art des Texts angegeben werden (`Acquisition`), die sich an den Literaturangaben von BibTex[5] orientiert.

[5] www.bibtex.org/de/

Die Werte für die Anwendungsdomäne des Inhalts (`Use`) wurden aus den analog definierten `ImageDomain CS`, `AudioDomain CS` und `VideoDomain CS` aus MPEG-7 übernommen.

CreationDescription

Neben dem semantischen Inhalt sind gerade Beschreibungen der Erzeugung und die Kategorisierung von Inhalten relevant für die personalisierte Auswahl. Diese Informationen werden in MPEG-7 mit dem `CreationInformation DS` (9.2.1) beschrieben, das u. a. das `Creation DS` (9.2.2) und das `Classification DS` (9.2.3) enthält. Außerdem kann es Material beschreiben, das weiterführende Informationen enthält wie z. B. zugehörige Webseiten oder Coupons, die zu Werbezwecken mit versandt werden. `CreationInformation DS`, `Creation DS` und `Classification DS` werden im Folgenden näher erläutert.

`CreationInformation DS` Das `CreationInformation DS` enthält neben wichtigen Informationen über die Erzeugung eines Inhalts und seine Kategorisierung (s.u.) noch ein Element zur Angabe von weiteren Informationen durch das `RelatedMaterial`-Element. Mit diesem kann man auf externe Zusatzinformationen verweisen, wie z. B. eine Web-Seite oder eine zugehörige DVD. Um anzugeben, wie dieser Inhalt verteilt wird, beinhaltet MPEG-7 das `Dissemination CS`. Da aber das MDS bereits 2003 entwickelt wurde, fehlen heute geläufige Verteilungsmöglichkeiten. Deswegen wurde das CS um BluRay und die Luftschnittstelle erweitert (Listing A.36). Mit dem `RelatedMaterial` DS kann man keine weiterführenden Informationen wie Öffnungszeiten oder die Kapazität einbetten. Das `CreationInformation DS` wurde deshalb in MASL um das `RelatedInformation`-Element vom Typ `RelatedInformationType` (Listing A.22) erweitert, das die Anforderung nach informellen Beschreibungen erfüllt. Hierbei können keine bis beliebig viele Zusatzinformationen direkt eingebettet werden. Ein Beispiel sind Öffnungszeiten wie in Listing B.2 zu sehen.

Die Definition des `RelatedInformationType` ist an die Definition des `RelatedMaterial-Type` angelehnt. Die Elemente `CreationInformation` bzw. `CreationInformationRef` zur Beschreibung der Erzeugung der Zusatzinformation (bzw. der Verweis auf diese Beschreibung), sowie `UsageInformation` bzw. `UsageInformationRef` zur Beschreibung von Nutzungsinformationen wurden beibehalten. Das neu definierte Element `InformationType` dient zur einfachen Spezifizierung, um welche Art von Information es sich handelt. Die Klassifikation erfolgt gemäß dem neu definierten `Information CS` (Listing A.39). Es definiert alle möglichen Typen von informellen Beschreibungen und unterscheidet zwischen statischen und dynamischen Informationen. Dieses CS kann nach Bedarf entsprechend erweitert werden. Die eigentlichen Inhalte der informellen Beschreibungen werden mittels neu definiertem `Information`-Element als textuelle Beschreibung definiert.

Durch diese Erweiterung wird eine umfangreiche Menge an informellen Beschreibungen ermöglicht. Diese Informationen können die für den Nutzer bestimmte Auswahl an Inhalten einschränken, indem Inhalte von vornherein aussortiert werden, die z. B. Museen

behandeln, die geschlossen sind, Kinofilme, die bereits ausverkauft sind, oder Restaurants mit einer nicht vom Nutzer bevorzugten Kleiderordnung.

MPEG-7 erlaubt im `CreationInformation DS` weiterhin die Annotation von Inhalten mittels Fließtext oder Schlüsselworten. Bei diesen Informationen handelt es sich ausschließlich um Annotationen, die für alle Nutzer gültig sind. Eine umfangreiche Möglichkeit der Kommentierung von Inhalten mit unterschiedlichen multimedialen Annotationen ist im Standard bisher nicht vorgesehen. Dafür wurde das `CreationInformation DS` in MASL um das `RelatedAnnotation` Element (Listing A.10) erweitert und der `Content-AnnotationType` (Listing A.5) definiert. Innerhalb einer solchen Annotation können der Typ der multimedialen Kommentars, also `audio`, `video`, `audiovisual`, `image`, `text` und `multimedia` und eine Referenz auf die eigentliche multimediale Kommentardatei und auf die dem Kommentar zugeordnete Metadatendatei angegeben werden. Diese Metadatenbeschreibungen können wiederum alle in diesem Abschnitt beschriebenen Informationen für die Kommentare enthalten, inklusive Kontextinformationen und weitere Kommentare zu diesem Kommentar. So entstehen komplexe Baumstrukturen mit dem Inhalt als Wurzel und den Kommentaren als Nachfolger. Die Metadatenbeschreibungen der Kommentare erfüllen den Zweck, dass das System nicht erst die komplette Kommentardatei laden muss, sondern vorher an Hand der Metadaten die Relevanz eines Kommentars beurteilen kann. So wird vermieden, dass eine große Anzahl unpassender Kommentare aufs Endgerät übertragen wird, was besonders bei Mobiltelefonen auf Grund der geringen Ressourcen der Luftschnittstelle wichtig ist. Ein Beispiel für die Verwendung von Kommentaren ist ein Touristenführer, der es dem Nutzer erlaubt, die vorhandenen Informationen um Kommentare zu ergänzen, die entweder für alle oder nur für eine Teilmenge der Nutzer verfügbar sind.

`Creation DS` Im `Creation DS`, einem Sub-Element des `CreationInformation DS`, werden Informationen hinterlegt, die in Zusammenhang mit der Erzeugung stehen. Dazu gehören der Regisseur, die Schauspieler bzw. Interpreten, der Autor etc. Üblicherweise können die Informationen nur sehr schwer aus den multimedialen Dateien an sich extrahiert werden, sie entstehen vielmehr im Laufe des Produktionszyklus und müssen manuell erfasst werden. Diese Informationen eignen sich gut für eine personalisierte Auswahl. So können diese z. B. basierend auf dem Titel des Inhalts, auf Personen, die an der Erzeugung beteiligt sind wie Schauspieler oder Regisseure, oder auf dem Erzeugungsort oder der -zeit selektiert werden.

MPEG-7 unterstützt dadurch die Möglichkeit, den Erzeugungsort und die Erzeugungszeit des Inhalts anzugeben. Für eine generische Definition von Kontextinformationen wurde in MASL der neue `ContextInformationType` eingefügt (Listing A.7), der gemäß dem Aspect-Scale-Context (ASC) Modell definiert wurde [159]. Der Typ erlaubt, die Kontextinformation durch Angabe der Art der Kontextinformation (`type`; z. B. temperature), die verwendete Skala (`scale`; z. B. degreeCelsius für die Temperatur gemessen in Grad

Celsius) und den Wert der Kontextinformation (`value`) als Attribute zu spezifizieren. Dadurch wird ein Matching von zusätzlichen Kontextinformationen wie die Temperatur oder die allgemeine Wetterlage, die bei der Erzeugung des Inhalts relevant waren, ermöglicht. Dies ist z. B. für die multimedialen Kommentare von Relevanz, wenn diese nur in einem bestimmten Kontext gültig sein sollen.

Schlüsselwörter, die sich auf den Inhalt beziehen, können im `Creation DS` innerhalb des `Abstract`-Tag als `KeywordAnnotation` angegeben werden. Dieses erlaubt eine umfangreiche Liste aller mit dem Inhalt verknüpften Schlüsselwörter über das Subelement `Keyword`. Im folgenden Kapitel wird in Abschnitt 4.1 ein Verfahren näher erläutert, das es erlaubt, diesen Bereich mit Informationen zu füllen. Dies kann durch den Nutzer passieren oder automatisch. Um den automatischen Prozess zu unterstützen wird das neue Konzept der Tag-Bewertung eingeführt. Dem Nutzer wird also erlaubt, die zu einem Inhalt vergebenen Tags nach ihrer Relevanz zu bewerten. Dies liefert wertvolles Feedback für den automatischen Prozess. Um diese Bewertungen in die Metadaten-Beschreibungen aufzunehmen, wurde in MASL innerhalb des `Abstract`-Tag eine zusätzliche `RatedKeywordAnnotation` (Listing A.26) eingeführt. Diese definiert über ein `rating`-Attribut die Durchschnittsbewertung des Schlüsselworts und über das Attribut `ratingCount` die Anzahl der Bewertungen, auf denen dieser Mittelwert beruht (Listing A.14).

`Classification DS` MPEG-7 erlaubt mit dem `Classification DS` eine umfangreiche Kategorisierung von Inhalten nach verschiedenen Kriterien wie dem Genre, dem Format, dem Alter der Zielgruppe, der Sprache, Hinweisen für Erziehungsberechtigte oder dem eigentlichen Zweck, für den der Inhalt erzeugt wurde. Diese Informationen sind für die personalisierte Auswahl relevant.

Ein besonders wichtiges Kriterium ist das `Genre` des Inhalts. Prinzipiell können die Werte für das Genre beliebig gewählt werden, MASL orientiert sich jedoch an dem in ISO/IEC15938-5 [83] spezifizierten `Genre CS`. Dieses beinhaltet u. a. 91 verschiedene Sportarten und 25 verschiedene Arten von Unterhaltungsshows. MPEG-7 bietet mit dem `Genre` bereits ein umfangreiches Werkzeug zur Kategorisierung der Inhalte an. Um auch die vorgestellten Szenarien vollständig umsetzen zu können, müssen noch Tools für die Klassifizierung von Restaurants sowie von Museen definiert werden (Listing A.38).

Dafür wird zum einen die vorgegebene Klassifizierung um die entsprechenden Terme für Restaurants erweitert. Diese wird unter dem Term `Information` → `Information/tabloid` → `Gourmet eating/cooking` eingeordnet. Zunächst werden die Restaurants nach den entsprechenden Erdteilen eingeteilt; Restaurants, die sich mehreren Kochstilen verschrieben haben, können natürlich auch mehrere Genre-Tags haben. Restaurants, die sich nicht in eine bestimmte Länderküche einordnen lassen, können einen Tag aus 1.2.6.6 `Other` verwenden, wo Restaurants wie vegetarische Restaurants oder Cocktailbars einsortiert sind. Für MASL wurde das `Genre CS` ebenfalls um eine Klassifizierung für Museen erweitert. Diese wird unter dem Term `Information` → `Arts and Media` → `Art` eingeordnet.

Mit Hilfe dieser Informationen können noch gezielter Inhalte mit den entsprechenden Nutzerpräferenzen abgeglichen werden. Ein Anwendungsfall für die Verwendung der umfangreichen Restaurantklassifizierung ist die personalisierte Werbung, die einem Nutzer auf das Endgerät übertragen wird, evtl. in Kombination mit einem Coupon. Je mehr die Werbung mit den Nutzerinteressen übereinstimmt, umso wahrscheinlicher wird er auch das Restaurant besuchen. Ein Beispiel für die Verwendung der Museums-Kategorisierung ist der bereits beschriebene Touristenführer.

Das von MPEG-7 definierte `Format CS`, das das Format des Inhalts angibt, wurde neu definiert. Die vorhandene Klassifikation war im Vergleich zu der entsprechenden aus TV-Anytime bei der Einteilung der Inhalte nicht so detailreich. Deswegen wurde basierend auf TV-Anytime eine neues `Format CS` entwickelt (Listing A.37).

UsageDescription

Mit Hilfe der `UsageInformation DS` (10.2.1) gestattet es MPEG-7, nutzungsrechtliche Informationen über einen Inhalt in seiner Beschreibung anzugeben. Mit diesem Werkzeug kann detailliert angegeben werden, wie ein Inhalt verwendet werden darf, welche Kosten durch einen multimedialen Inhalt erzeugt werden und wie ein Inhalt publiziert wird, bzw. wo und wann er genutzt werden darf. Diese Informationen schränken die personalisierte Auswahl der Inhalte ein, da sie die Ergebnismenge um diejenigen Inhalte reduzieren, die zwar für den Nutzer von Interesse sind, auf die von Anbieter-Seite aus jedoch nicht zugegriffen werden darf. Das DS wird in MASL übernommen.

Innerhalb der `Availability DS` des `UsageInformation DS` kann die Anzahl der bisherigen Abrufe eines Inhalts angegeben werden. Diese Information kann für die Auswahl verwendet werden, da ein oft aufgerufener Inhalt u. U. auf einen guten Inhalt hinweisen kann.

In MASL wird das `UsageInformation DS` um ein Element des Typs `ContextInformationType` (Listing A.7) erweitert. Dieser erlaubt die Angabe von Kontextinformationen gemäß dem ASC-Modell (siehe oben). Der Typ wird zusätzlich zum `ContextInformationRestrictionType` (Listing A.6) erweitert, um Vergleichsoperationen auf Kontextinformationen zu erlauben. Definiert wurden hier `max`, `maxEqual`, `min`, `minEqual` und `equal`. Damit kann man angeben, dass ein Inhalt erst ab einer bestimmten Temperatur oder bei einer bestimmten Wetterlage relevant ist.

3.4.2. Aufbau des Nutzerprofils in MASL

Eine Nutzerbeschreibung in MASL wird mit dem Wurzel-Element `Profile` begonnen. Unter diesem können sich sieben unterschiedliche Sub-Typen befinden (vgl. Abbildung 3.5 und Listing A.2), die im Folgenden näher erläutert werden. Die komplette Schema-Definition von MASL sowie ein Beispiel Profil sind in Anhang A.1 bzw. B.2 zu finden.

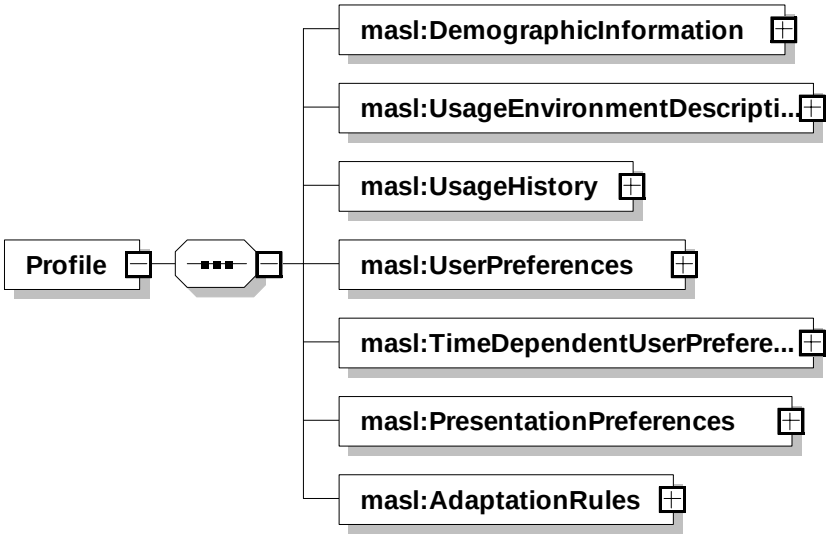

Abbildung 3.5.: Aufbau des Nutzerprofils in MASL

DemographicInformation

MPEG-7 unterstützt neben den reinen Beschreibungen der Inhalte umfangreiche Werkzeuge zur Beschreibung des Nutzers, die in MASL übernommen werden. Eine Person wird durch das `Person DS` (7.5.3) näher beschrieben, das Informationen wie den Namen, die zugehörige Organisation oder Kontaktinformationen enthält. Bei der Beschreibung von demographischen Daten weißt MPEG-7 aber Lücken auf. Die Beschreibung von so grundlegenden Informationen wie das Geschlecht (`Gender`) oder das Geburtsdatum (`DateOfBirth`) werden bisher nicht unterstützt und deswegen für MASL zum `Person DS` (Listing A.16) hinzugefügt. Die erweiterten Informationen können für ein Matching von demographischen Daten verwendet werden, wie das in der Praxis durch die genaue Analyse der Zielgruppe und deren Interessen durchgeführt wird.

UsageEnvironmentDescription

Die `UsageEnvironmentDescription` dient in MASL zur Beschreibung von überwiegend dynamischen Kontextinformationen, die bei der Nutzung relevant sind. Dazu gehören Beschreibungen des verwendeten Endgeräts, des Netzes und der Nutzungsumgebung.

MPEG-21 besitzt mit den `TerminalCapabilities` (5.4) umfangreiche Werkzeuge zur Beschreibung der Fähigkeiten eines Endgeräts, die in MASL übernommen wurden. Dazu gehören allgemeine Informationen über das Endgerät wie den Typ des Endgeräts (z. B. PC oder PDA), die Energieverwaltung (z. B. Ladestand, Batterieverbrauch), Speicherbezogene Informationen (z. B. Speichergröße und Zugriffsrate) oder I/O-bezogene Werte

(z. B. Transferrate). Basierend auf diesen Informationen kann eine personalisierte Anpassung der Darstellung durchgeführt werden. Die Beschreibungen beinhalten Informationen über die Ein- und Ausgabemöglichkeiten des Endgeräts wie Display, Tastatur, Kopfhörer oder Mikrophon. Weiterhin können vom Endgerät unterstützte Formate zum Kodieren und Dekodieren von audio-visuellen Ressourcen beschrieben werden oder dafür benötigte Zusatzinformationen wie die Puffergröße, die ebenfalls für die Anpassung der Darstellung verwendbar sind.

Da bei der Beschreibung der Fähigkeiten eines Endgeräts in MPEG-21 ausschließlich statische Kontextinformationen berücksichtigt werden, kann nicht bestimmt werden, ob z. B. die Batterie gerade geladen wird oder ob ein externes Gerät zur Wiedergabe von Audio (z. B. ein Kopfhörer) oder Video (z. B. ein Beamer) angeschlossen ist. Aus diesem Grund wurden für MASL die entsprechenden Typen geeignet erweitert. Der `PowerCharacteristicsType` (Listing A.17) wurde um ein bool'sches Attribut `batteryBeingCharged` erweitert, das angibt, ob der Akku des mobilen Endgeräts aktuell geladen wird. `AudioOutputCapabilitiesType` (Listing A.3) und `DisplayCapabilityType` (Listing A.12) wurden um das Attribut `outputDevice` erweitert. Dieses spezifiziert, welches visuelle oder auditive Ausgabegerät (z. B. Beamer oder Kopfhörer) momentan zur Verfügung stehen.

Mit den integrierten MPEG-21-Werkzeugen zur Beschreibung von Netzwerkcharakteristika (`NetworkCharacteristics` (5.5)) erlaubt MASL die Beschreibung von Informationen über die momentane Lage des Netzes. Dazu gehören einerseits die statischen Eigenschaften des Netzes wie die maximal verfügbare Bandbreite oder das Vorhandensein von Mechanismen zur Fehlererkennung und –korrektur, und dynamische Eigenschaften wie die aktuell verfügbare Bandbreite, oder die Latenzzeit. Diese Beschreibungen liefern wertvolle Informationen, die für die Anpassung der Darstellung verwendbar sind, da z. B. bei schlechter Netzverbindung keine hochauflösenden Bilder übertragen werden sollen.

Eine wichtige Rolle spielen der Ort bzw. die Zeit der Nutzung eines multimedialen Inhalts, die mit den Werkzeugen der `NaturalEnvironmentCharacteristics` (5.6) aus MPEG-21 beschrieben werden. Sie wurden in MASL übernommen. Zur Beschreibung von Orten wird der von MPEG-7 übernommene `PlaceType` verwendet, der sowohl geographische Koordinaten als auch Klassen von Orten unterstützt. Der `TimeType` wird aus MPEG-7 übernommen, der zeitbezogene Informationen mittels Zeitpunkten oder der Dauer erlaubt. Basierend auf diesem Informationen können Inhalte ausgewählt (z. B. ein zu diesem Zeitpunkt und an diesem Ort relevantes Video) und angepasst werden (z. B. keine Audioausgabe in öffentlichen Verkehrsmitteln). Zu den von MPEG-21 unterstützten Kontextinformationen gehören Beschreibungen der natürlichen Umgebungscharakteristika. Dazu gehören Informationen, die die Wiedergabe der Inhalte beeinflussen. Für Audio-Inhalte sind Informationen wie die aktuelle Umgebungslautstärke wichtig, für visuelle Inhalte werden Informationen über die Lichtintensität oder die Farbtemperatur des Lichts benötigt. Um dem Nutzer eine optimale Darstellung der Inhalte zu gewährleisten, müssen diese Informationen berücksichtigt werden. Um die Ausdrucksmöglichkeit des Nut-

zerprofils in MASL zu erweitern, wurde der neu eingeführte `ContextInformationType` (s.o., Listing A.7) zur generischen Beschreibung der Nutzungsumgebung hier intergiert. Dadurch können z. B. Inhalte basierend auf der Temperatur ausgewählt werden, wie es bei einem Werbevideo über einen Biergarten für einen Touristenführer sinnvoll ist.

UsageHistory

Basierend auf bereits konsumierten Inhalten (Nutzungshistorie) kann auf Nutzerpräferenzen geschlossen werden. Diese können am Nutzerendgerät aufgezeichnet und zur Filterung der Inhalte verwendet werden. Das `UsageHistory DS` (15.3; Listing A.32) wird verwendet, um eine Menge von `UserActionHistory`-Elementen zusammenzufassen, die jeweils einen eigenen Beobachtungszeitraum umfassen. Ein `UserActionHistory`-Element besteht aus `UserActionList`-Elementen, die wiederum `UserAction`-Elemente enthalten. Das `UserAction`-Element wird verwendet, um eine einzelne Nutzerinteraktion wie das Vorspulen oder Stoppen eines Videos durch den Zeitpunkt, die Dauer, den entsprechenden Programm-Identifikator und Verweise auf verwandtes Material zu beschreiben. Dabei entstehen komplexe Beschreibungen und Nutzungsstatistiken.

Da Nutzerbewertungen in MASL zeitabhängig sein sollen (siehe hierzu Abschnitt 4.2), wird hier zusätzlich die `UsageHistorySummary` (Listing A.31) integriert. Diese erlaubt die Angabe eines bewerteten Inhalts über dessen Identifikator (`ProgramIdentifier`) und Titel (`Title`), sowie die Bewertung des Inhalts (`Rating`). Die Bewertung soll im Intervall [0,10] liegen. Man kann spezifizieren, ob die Bewertung explizit oder implizit erzeugt wurde (`explicit`) und ob der Bewertungswert fest oder zeitabhängig sein soll (`fixed`). Zu diesem Zweck wurde der in MPEG-7 vorhandene `RatingType` geeignet zum `RatingExtType` (Listing A.20) erweitert. Zusätzlich wird für einen Eintrag in der `UsageHistorySummary` ein Zeitstempel (`TimeStamp`) vergeben, wenn der Inhalt eingefügt wurde, und ein Gültigkeitsfaktor (`Validity`) vom Typ `ValidityType` (Listing A.34) festgelegt. Dieser wird beim Einfügen des Inhalts auf 1 gesetzt und konvergiert mit der Zeit gegen 0.

UserPreferences

Mit den Werkzeugen des `FilteringAndSearchPreferences DS` (15.2.5) stellt MPEG-7 eine grundlegende Möglichkeit zur personalisierten Auswahl von Inhalten zur Verfügung. Beschreibungen von Inhalten können mit den Präferenzen abgeglichen und die passenden Inhalte dem Nutzer präsentiert werden. Das Werkzeug erlaubt es anzugeben, wie die Vorlieben und Abneigungen des Nutzers für bestimmte Arten von Inhalten sind und wie die bevorzugten Werte für Informationen über die Erzeugung (`CreationPreferences DS` (15.2.6)), die Kategorisierung (`ClassificationPreferences DS` (15.2.7)) und den Ursprung der Inhalte sind (`SourcePreferences DS` (15.2.8)). Dabei können `FilteringAndSearchPreference`-Elemente weitere `FilteringAndSearchPreference`-Elemente enthal-

ten, was die Definition von komplexen Hierarchien erlaubt. MASL integriert alle diese DSs. Da in der vorliegenden Arbeit das Intervall [0,10] für Präferenzen und Bewertungen von Inhalten gewählt wurde (vgl. Abschnitt 4.1.3), MPEG-7 aber das Intervall [-100, 100] erlaubt, musste dieses entsprechend angepasst werden. Dafür wurde der Wertebereich des `preferenceValueType` (Listing A.19) geeignet eingeschränkt.

Um dem Nutzer in MASL die Angabe von Präferenzen bezüglich der semantischen und informellen Beschreibungen zu erlauben, wurden die `FilteringAndSearchPreferences` (Listing A.13) um die Typen `SemanticPreference` (Listing A.24) und `RelatedInformationPreference` (Listing A.21) erweitert. Die `SemanticPreference` verknüpft alle Elemente des zuvor beschriebenen `Semantic DS` mit einem Präferenzwert. In der `RelatedInformationPreference` kann zu einer Kombination aus `InformationType` und `Information` ein Präferenzwert angegeben werden. Ein Beispiel hierfür ist eine Präferenz für einen bestimmten Dresscode oder bevorzugte Öffnungszeiten.

Mit dem `PreferenceConditionType` (Listing A.18) kann in MPEG-7 spezifiziert werden, zu welchem Zeitpunkt und an welchem Ort eine bestimmte Präferenz gelten soll. Der Typ wurde um den `ContextInformationType` (s.o., Listing A.7) erweitert, um generisch eine Vielzahl von Kontextinformationen unterstützen zu können.

TimeDependentUserPreferences

Nutzerbewertungen sollen in MASL zeitabhängig sein. Dabei sind Bewertungen von ganzen Inhalten, von einzelnen Attributwerten und von Kombinationen aus Attributwerten möglich. Inhalte und ihre Bewertungen werden in der `UsageHistory` verwaltet (s.o.), für Attributwerte und ihre Kombinationen wird das Element `TimeDependentUserPreferences` mit den beiden Sub-Elementen `TimeDependentAttributes` (Listing A.29) und `TimeDependentAttributeCombination` (Listing A.30) eingefügt. Diese enthalten die Liste aller Inhalte (`ProgramIdentifier`) aus der `UsageHistorySummary`, die den Wert des betrachteten Attributs bzw. der betrachteten Kombination enthalten. Weitere Elemente sind der Gültigkeitsfaktor (`Validity`), der im Intervall [0,1] liegt, sowie die durchschnittliche Bewertung des Attributwerts bzw. der Werte der Attributkombination (`AverageRating`) gemittelt über alle Inhalte, die diese enthalten. Der betrachtete Attributtyp wird über das Attribut `typeName` (bzw. `firstType` und `secondType` bei Attributkombinationen) angegeben, die jeweiligen Werte über das Attribut `value` (bzw. `firstValue` und `secondValue` bei Kombinationen). Ein Ansatz, wie dieser Teil des Profils mit Daten gefüllt werden kann, wird in Abschnitt 4.2 erläutert.

PresentationPreferences

In MPEG-21 können mit den Werkzeugen der `UserCharacteristics` Präferenzen definiert werden, wie ein multimedialer Inhalt angezeigt werden soll. Diese Informationen können für eine personalisierte Anpassung der Darstellung verwendet werden und

wurden deswegen in MASL integriert. Es wird zwischen den unterschiedlichen Arten von Multimediadateien und deren Anpassung unterschieden. So kann bei Audio-Dateien die Lautstärke, der Frequenzbereich oder das bevorzugte Ausgabegerät (Kopfhörer oder Lautsprecher) spezifiziert werden und bei visuellen Medien die präferierten Einstellungen des Anzeigegeräts wie z. B. die Sättigung, die Helligkeit oder der Kontrast. Ebenso können die Eigenschaften von graphischen Elementen wie die Dichte der Textur oder die Zerlegung von Animationen in Einzelbilder definiert werden.

Wenn ein Netzwerk oder Endgerät den Transport bzw. die Verwendung einer bestimmten Datei nicht unterstützt, muss sie entsprechend konvertiert werden. Wie diese Konvertierung durchgeführt werden soll, kann durch Nutzerpräferenzen angegeben werden. Man unterscheidet zwischen Konvertierungsregeln für allgemeine (z. B. alle mp3-Dateien) und für spezielle Ressourcen (für die Ressource "example.mp3"), wobei die genaue Reihenfolge der Konvertierung festgelegt werden kann.

Manche Beeinträchtigungen des Nutzers, die sich auf die Wiedergabe von Audiovisuellen Medien auswirken können, wie ein Hörverlust, Farbschwächen oder ein Verlust der Sehkraft sollen beim Anpassungsprozess berücksichtigt werden, weswegen MPEG-21 und damit auch MASL solche Nutzerbeschreibungen unterstützen. Der Nutzer kann für die unterschiedlichen Anpassungsprozesse Prioritäten angeben und genau definieren, wie die Qualität der angepassten Ressource sein soll.

AdaptationRules

In MASL können Situations-abhängige Anpassungsregeln spezifiziert werden, die im Folgenden näher erläutert werden. Die Situation, an die angepasst wird, ist abhängig von den Nutzerpräferenzen, den Gerätecharakteristika und den dynamischen Kontextinformationen wie der Aktivität des Nutzers oder Personen in seiner Umgebung.

Ein `Situation`-Tag (Listing A.25) beinhaltet eine oder mehrere `SituationInformation`-Tags und ein oder mehrere `PrivacyPreferences`-Tags, mit denen die Nutzerpräferenzen bezüglich des Schutzes der Privatsphäre als Name der Elemente angegeben werden, die in der beschreiben `Situation` nicht vom Client übertragen werden dürfen (vgl. Kapitel 6).

`SituationInformations` können entweder Gerätecharakteristika (`DeviceParameters`), Nutzerpräferenzen (`UserPreferences`) oder dynamische Kontextinformationen (`Context`) enthalten. Diese übernehmen jeweils die zuvor erläuterten Beschreibungen aus MASL. Unterschiedliche `SituationInformation`-Tags können über das Attribut `connective` verknüpft werden. Dieses Konzept ist aus der Schema-Definition von APPEL [103] entnommen und erlaubt die folgenden Werte:

- `or`: Eine `Situation` tritt ein, wenn mindestens einer der enthaltenen `SituationInformation`-Tags aktuell gegeben ist.

- `and`: Eine `Situation` tritt ein, wenn alle der enthaltenen `SituationInformation`-Tags aktuell gegeben sind.

- `non-or`: Eine `Situation` tritt ein, wenn keine der enthaltenen `SituationInformation`-Tags aktuell gegeben ist.

- `non-and`: Eine `Situation` tritt ein, wenn nicht alle der enthaltenen `SituationInformation`-Tags aktuell gegeben sind.

Um geschachtelte bool'sche Ausdrücke zu erlauben, kann außerdem rekursiv ein weiteres `Situation`-Tag innerhalb der `SituationInformation` angegeben werden. Das `SituationInformation`-Tag hat zusätzlich das optionale Attribut `constraint`, das verwendet werden kann, um Werte der `SituationInformation` einzuschränken. Erlaubt sind die Werte `max`, `maxEqual`, `min`, `minEqual` und `equal`. Ist das `constraint`-Attribut auf `max` gesetzt tritt die Situation nur dann ein, wenn z. B. der aktuelle Batteriestand einen Höchstwert von (exklusiv) dem in der Beschreibung gegebenen Wert hat. Damit tritt in dem Beispielprofil aus Listing B.3 die Situation `lowBatteryPower` dann ein, wenn der aktuelle Batteriestand unter 4200 Sekunden ist *und* die Batterie nicht geladen wird *oder* der aktuelle Batteriestand unter 1800 Sekunden ist *und* die Batterie aufgeladen wird.

Das `PrivacyPreferences`-Tag enthält ein oder mehrere `lockedData`-Tags. Mit diesen werden die Datentypen angegeben, die nicht an den Server übertragen werden dürfen. Durch die Angabe eines bestimmten Elementnamens werden dieses Element und alle sich darunter befindlichen Subelemente gesperrt.

Inhalte sollen basierend auf unterschiedlichen Situationen unterschiedlich verarbeitet werden. Wie diese Anpassung geschehen soll, wird über ein `Behavior`-Tag (Listing A.4) festgelegt. Jedes Verhalten beinhaltet unterschiedliche `Function`-Tags, die jeweils eine tatsächlich vorhandene Anpassungsmethode repräsentieren. Diese werden über einen einheitlichen Namen identifiziert. Die Entwicklung dieser Methoden lag jedoch nicht im Fokus dieser Arbeit und wird deswegen nicht näher diskutiert.

Die Methoden bekommen ein oder mehrere `Objects` übergeben, die den Teil des Inhalts beschreiben, auf den die Funktion angewendet werden soll. Werden noch weitere Parameter für die Anpassung benötigt, können diese über optionale `Parameter`-Tags übergeben werden. Die Funktionen werden der Reihe nach von oben nach unten durchgeführt. Im Beispiel im Anhang B.2 wird von einem Video die Audiospur entfernt, ein Untertitel hinzugefügt und anschließend das gesamte Video in MPEG-4 konvertiert.

Eine konkrete Anpassungsregel (`Rule`, Listing A.23) setzt eine `Situation` mit einem Verhalten in Beziehung. Die Priorität der entsprechenden Regel kann über das `priority`-Attribut mit Werten im Intervall $[0, 5]$ angegeben werden, wobei ein höherer Wert einer höheren Priorität entspricht. Dadurch kann entschieden werden, welche Anpassungsregel im Konfliktfall zu bevorzugen ist.

3.5. Bewertung des eigenen Ansatzes

Um die hier vorgestellte Sprache MASL zu bewerten, werden die Anforderungen verwendet, die in Abschnitt 3.2 identifiziert wurden. MASL übernimmt Teile aus den beiden Standards MPEG-7 und MPEG-21, die einen Teil der Anforderungen erfüllen und erweitert diese geeignet. Daher wird im Folgenden nur erläutert, wie MASL die Anforderungen erfüllt, die weder von MPEG-7 noch von MPEG-21 bedient werden.

3. *Informelle Beschreibungen*
   MPEG-7 und MPEG-21 unterstützen kaum bzw. keine informellen Beschreibungen, weswegen diese für MASL neu definiert wurden. Dafür wurden das `RelatedInformation`-Element und der `RelatedInformationType` eingeführt, die die Angabe dieser Daten in textueller Form erlauben. Beim zugehörigen `Information CS` wird auch zwischen relativ statischen und dynamischen Informationen unterschieden, bei fehlenden Beschreibungstypen kann dieser Teil je nach Anwendungsfall leicht erweitert werden.

11. *Annotationen*
    Annotationen in MPEG-7 sind immer für alle Nutzer des Systems gedacht, nicht nur für einzelne Nutzer oder Nutzergruppen. Außerdem unterstützt der Standard nur die Annotation mittels Fließtext oder Schlüsselwörter, nicht jedoch durch beliebige multimediale Inhalte. Um diese in MASL zu unterstützen, wurde der Standard um das `RelatedAnnotation` Element und den `ContentAnnotationType` erweitert. Dieser enthält die Art des Kommentars sowie Verweise auf die Kommentar-Ressource und die zugehörige Metadatenbeschreibung, die dann die Ziel-Nutzer des Kommentars enthält. Da Beschreibungen von Kommentaren identisch sind mit den Kommentaren der Inhalte, können sie wiederum selbst annotiert werden. Auf diese Weise können komplexe Kommentarstrukturen erzeugt werden.

12. *Zeit-abhängige Bewertungen von Inhalten, Attributen und Attributkombinationen*
    Bewertungen durch den Nutzer werden weder von MPEG-7 noch MPEG-21 unterstützt. MASL integriert mit den `TimeDependentUserPreferences` und der `UsageSummary` eine Möglichkeit, wie man diese umsetzen kann. Dafür wird für die Inhalte neben den Bewertungen ein Zeitstempel in MASL abgelegt, der angibt, wann der entsprechende Wert eingetragen wurde. Aus den Bewertungen aller Inhalte, die mit einem entsprechenden Attribut oder einer Attributkombination beschrieben werden, kann der Bewertungswert der Attribute und Attributkombinationen bestimmt werden (siehe Abschnitt 4.2.5).

13. *Beschreibung von Anpassungspräferenzen*
    Der Nutzer soll die Möglichkeit haben, allgemeine und spezielle Präferenzen für die Anpassung anzugeben. Die von MPEG-21 unterstützten allgemeinen Regeln wurden in MASL übernommen, spezielle fehlen aber bisher völlig. Aus diesem Grund

wurden in MASL Situations-abhängige Anpassungsregeln spezifiziert, die es dem Nutzer erlauben, spezielle Vorlieben bei der Anpassung anzugeben. Dies ermöglicht eine größere Personalisierung der Anpassung gemäß den individuellen Vorlieben des Nutzers.

3.6. Zusammenfassung

Nach der Definition von Anforderungen und der Analyse von bereits existierenden Metadatenmodellen wurde in diesem Kapitel MASL, eine Sprache zur Beschreibung von multimedialen Inhalten und deren Nutzer vorgestellt, die es erlaubt, multimediale Inhalte Kontext-abhängig zu personalisieren. MASL basiert auf den beiden Standards MPEG-7 und MPEG-21, die an sich schon einen Teil der Anforderungen erfüllen. Die für diese Arbeit relevanten Teile von MPEG-7 und MPEG-21 wurden vorgestellt und die benötigten Erweiterungen erläutert. Dadurch kann MASL alle Anforderung an eine Sprache zur Kontext-abhängigen Personalisierung multimedialer Inhalte bedienen. Hierzu zählen insbesondere:

- Generische Beschreibung von Kontextinformationen
- Beschreibung von informellen Zusatzinformationen zu einem Inhalt
- Erweiterte Interaktionsmöglichkeiten des Nutzers durch individuelle Bewertungen und Nutzerkommentare
- Unterstützung von erweiterten Nutzerpräferenzen bezüglich der Auswahl der Inhalte wie z. B. Präferenzen für bestimmte fiktive Personen oder Orte
- Unterstützung von erweiterten Nutzerpräferenzen bezüglich der Anpassung der Darstellung der Inhalte durch Situations-abhängige Anpassungsregeln

Im folgenden Kapitel werden zwei Möglichkeiten diskutiert, wie man die für eine MASL-Beschreibung benötigten Daten gewinnen kann. Daran schließen sich die Mechanismen für die eigentliche Kontext-abhängige Auswahl und Anpassung der Darstellung der Inhalte an. Diese Mechanismen arbeiten mit Hilfe der durch MASL gegebenen Beschreibungen.

Kapitel 4.

Datengewinnung

Im folgenden Kapitel werden zwei Ansätze zur expliziten und impliziten Datengewinnung diskutiert, die die in Kapitel 3.4 vorgestellte Sprache zur Beschreibung multimedialer Inhalte und Profile für die Kontext-abhängige Personalisierung mit Daten füllen können. Zwei Arten an Informationen werden hierbei betrachtet: Schlüsselwörter zur Kategorisierung der beschriebenen Inhalte und Inhaltsbewertungen, die aus dem Nutzerverhalten gewonnen werden.

Gerade bei Nutzer-generierten Inhalten (User-generated Content) ist es meist nicht realistisch anzunehmen, dass umfangreiche Beschreibungen vorliegen. Dadurch können diese Inhalte nicht im Personalisierungsprozess berücksichtigt werden. In Abschnitt 4.1 wird ein Ansatz diskutiert, der es erlaubt, Schlüsselwörter oder Tags für Inhalte zu erzeugen, die für eine personalisierte Auswahl genutzt werden. Dieser Prozess soll entweder durch den Nutzer gestützt oder automatisch durchgeführt werden.

Da Nutzer nur bis zu einem gewissen Grad bereit sind, umfangreiche Nutzerprofile in ein Personalisierungssystem einzubringen, werden Ansätze benötigt, automatisch Profilinformationen generieren zu können. In Abschnitt 4.2 wird ein Profiling-Mechanismus vorgestellt, der aus Beobachtung des Nutzerverhaltens Rückschlüsse auf Nutzerpräferenzen ermöglicht. Dadurch kann das Personalisierungssystem umfangreiche Daten über den Nutzer ohne explizite Nutzereingaben beziehen.

Für die anderen Informationen, die mittels MASL beschreibar sind, müssen eigene Verfahren zur impliziten Datengewinnung bereitgestellt werden, dies ist jedoch nicht im Fokus dieser Arbeit. Für diese Informationen wird angenommen, dass sie geeignet erfasst und in Beschreibungen konvertiert wurden.

Anzumerken ist, dass im Gegensatz zu Kapitel 3, 5 und 6 hier kein Schwerpunkt auf die Anforderungsanalyse und Bewertung verwandter Arbeiten gelegt wurde, da die hier vorgestellten Ideen nicht im Kernbereich der Kontext-abhängigen Personalisierung liegen. Die wichtigste Anforderung war, dass die Ansätze auf den in MASL gegebenen Beschreibungen arbeiten können. Da die Sprache im Rahmen dieser Arbeit entwickelt wurde, gibt es keine verwandte Arbeit, die diese Anforderung erfüllen kann. Durch eine umfangreiche Literaturrecherche wurden einige Arbeiten identifiziert, die auch Ideen für

den eigenen Ansatz brachten. Diese werden im Überblick vorgestellt. Die wichtigsten im Folgenden verwendeten Bezeichnungen sind in Anhang C aufgelistet.

4.1. Gewinnung von Schlüsselwörtern

Im folgenden Abschnitt sollen Verfahren für die Gewinnung von Schlüsselwörtern, das so genannte Tagging, vorgestellt werden. Grob werden zwei Verfahren unterschieden: das Taggen durch einen Nutzer (*Nutzergestütztes Taggen*) und die automatische Gewinnung von Schlüsselwörtern (*Automatisches Taggen*).

Zunächst wird ein Überblick über verwandte Arbeiten aus den Bereichen des nutzergestützten Taggens, der Informationsanalyse und des Wissenserwerbs gegeben. Dann werden die im Rahmen dieser Arbeit entwickelten Konzepte vorgestellt, die mit MASL-Inhaltsbeschreibungen verwendet werden.

4.1.1. Überblick über verwandte Ansätze

Beim nutzergestützten Taggen werden vom Nutzer manuell Tags zu einem Inhalt erzeugt. Gestaltet man diesen Prozess intuitiver und interessanter für den Nutzer, erhöht man automatisch seine Bereitschaft zum Taggen.

Mit dem von Abel et al. entwickelten System *GroupMe!* [1] können beliebige Ressourcen aus dem Internet manuell zu Gruppen zusammengefasst und getaggt werden. Dadurch werden strukturierte semantische Beschreibungen der Inhalte erzeugt. Diese können für die Navigation in den Inhalten und für Suchfunktionalitäten verwendet werden. Unterstützt werden von diesem System die Suche nach Schlüsselwörtern und die Suche unter Berücksichtigung von Gruppenzugehörigkeiten.

Das von Hotho et al. entwickelte *BibSonomy* System [80] bietet die Möglichkeit, URLs zu Webseiten und Publikationsreferenzen im Web zu verwalten und mit anderen Nutzern zu teilen. Es handelt sich bei dem System um eine so genannte Folksonomie, eine Web-2.0-Technik, die es Nutzern gestattet, kollaborativ Tags zu Ressourcen im Web zu vergeben. BibSonomy versucht durch ein durchdachtes URL-Schema die Graphstruktur von Folksonomies auf hierarchische Ordnerstrukturen abzubilden. Dies erlaubt eine vereinfachte Navigation innerhalb der Seite und eine klare Strukturierung der Inhalte. Außerdem unterstützt BibSonomy die Bildung von Tag-Hierarchien und umfangreichen Taxonomien und die automatische Erkennung von identischen Ressourceneinträgen.

Beim automatischen Taggen multimedialer Inhalte spielen die Analyse des Inhalts und der damit einhergehende Wissenserwerb eine entscheidende Rolle. Für die Analyse von Inhalten gibt es eine Menge an spezialisierten Verfahren, die versuchen aus Texten, Audio, Bildern oder Videos die semantische Bedeutung des Inhalts abzuleiten. Bei der Textanalyse werden die Inhalte lexikalisch untersucht und die gefundenen Tokens auf geeignete Klassen abgebildet [109]. Dabei gibt es Verfahren, die mit Hilfe von Regelsätzen

arbeiten (regelbasierte Algorithmen; z. B. [65]) oder die statistische Methoden verwenden (statistische Algorithmen; z. B. [89]). Die Audioanalyse stützt sich auf Mechanismen der Spracherkennung, die versuchen, Wörter aus Audiosignalen zu extrahieren. Dafür werden neben der reinen Worterkennung Sprachmodelle eingesetzt, mit denen man z. B. eine Wortfolgeanalyse durchführen kann [121]. Ein Ansatz für die automatische Bild-Analyse ist es, mittels statistischer Algorithmen Beziehungen zwischen Bildausschnitten und Schlüsselwörtern herzustellen. Dafür werden mögliche Bildausschnitte bestimmt, deren Charakteristika wie Farben, Formen oder die Position im Gesamtbild untersucht und mit statistischen Methoden die Semantik abgeleitet [95]. Für die Analyse von Videos können ähnliche Verfahren angewandt werden, da es sich hier im Grunde genommen um eine Folge von Bildern handelt. Es werden hierfür die Keyframes eines Videos bestimmt und mit Hilfe von Bildanalyseverfahren die Semantik abgeleitet [105, 68]. Alle Ansätze aus dem Bereich der Inhaltsanalyse benötigen eine umfangreiche Trainingsphase, um zu zufriedenstellenden Ergebnissen zu gelangen.

Basierend auf der eigentlichen Inhaltsanalyse können Mechanismen zum Wissenserwerb arbeiten. Das von Adrian et al. entwickelte System *ConTag* [2] definiert zu diesem Zweck komplexe Ontologien, die die dem System bekannten Daten verwalten. Als Tags werden Entitäten verwendet, die einer Ontologieklasse zugeordnet werden und über einen eindeutigen textuellen Bezeichner und eine URI verfügen. Durch Normalisierung eines zu taggenden Texts und einen Abgleich dieser Normalisierung mit Ontologien und vorhandenen Tags werden dem Nutzer passende Tags empfohlen.

Ein Ansatz zum Taggen von überwiegend textuellen Webressourcen durch semantische Informationen wird von Vargas-Vera et al. im Rahmen des *MnM*-Projekts [171] verfolgt. Das automatische und semi-automatische Taggen von Inhalten wird unterstützt und versucht, die Tags aus den Ressourcen an sich abzuleiten. Dafür stellt das System eine umfangreiche Sammlung von Ontologien zur Verfügung. Vom Nutzer kann eine der Ontologien ausgewählt werden, um Eingabedokumente manuell mit den in der Ontologie definierten Begriffen zu versehen. Dadurch wird das System trainiert.

Das von Mishne entwickelte System *AutoTag* [124] verwendet die Grundidee des kollaborativen Filterns (siehe Abschnitt 5.1.3), um Blogeinträgen automatisch Tags zuzuordnen. Dafür werden zu einem neu erstellten Blogeintrag ähnliche, bereits vorhandene Einträge gesucht, indem die vorkommenden Wörter analysiert werden. Aus deren Tags wird eine Liste mit möglichen Tags für den neuen Inhalt erstellt. Diese werden basierend auf der Häufigkeit, mit der ein Tag verwendet wurde, sortiert und dem Nutzer als Vorschlag angezeigt.

Die durch Folksonomies erzeugten Sammlungen von Schlüsselwörtern können nicht effizient maschinell verarbeitet werden, um daraus Wissen zu gewinnen. Mit dem *Folk 2 Onto*-System [3] versuchen Almeida et al. Ontologien zu verwenden, um die Daten, die in Folksonomies unstrukturiert vorliegen, in ein einheitliches und strukturiertes Datenmodell zu überführen. Dafür werden die Daten der Folksonomies in das Folk 2 Onto-System geladen und mit Wortdefinitionen und Synonymlisten abgeglichen. Basierend darauf kön-

nen dann die Tags auf ein Ontologie-Schema abgebildet und eine Beschreibung des Inhalts erzeugt werden.

4.1.2. Überblick über die Gewinnung von Schlüsselwörtern

Nach Smith [154] unterscheidet man zwei unterschiedliche Arten des Taggens durch den Nutzer: der Nutzer kann eine Ressource in das System einbringen und mit Schlüsselwörtern versehen (*Adding and Tagging*) oder eine bereits vorhandene Ressource taggen (*Just Tagging*). Weiterhin kann man zwischen dem Taggen eines einzelnen Inhalts (*Individual Tagging*) und dem Taggen mehrerer Inhalte (*Bulk Tagging*) unterscheiden.

Fügt ein Nutzer einen Inhalt ein, ohne ihn geeignet mit Schlüsselwörtern zu versehen, soll beim hier vorgestellten Ansatz der automatische Prozess angestoßen werden. Außerdem kann periodisch nach spärlich getaggten Inhalten gesucht werden, die ebenfalls automatisch mit einer größeren Menge an Schlüsselwörtern versehen werden. Die beiden Prozesse des nutzergestützten und des automatischen Taggens laufen im System also parallel ab. Der automatische greift immer dann ein, wenn durch die Nutzer keine oder nur eine geringe Menge an Tags vergeben wurden.

Als neue Idee wird hier das Konzept der *Tagbewertung* eingeführt. Es soll dadurch dem Nutzer ermöglicht werden, die zu einem Inhalt (durch den Nutzer oder automatisch) vergebenen Tags zu bewerten. Durch diese Bewertung von Tags kann neuen Nutzern der Einstieg in das System erleichtert werden, da sie direkt ablesen können, was eine gute oder eine schlechte Verschlagwortung eines Inhalts ausmacht (Best Practices). Außerdem wird dem System so Feedback für das automatische Taggen gegeben. Ein weiterer Ansatz um den automatischen Prozess zu unterstützen ist die Einteilung der Inhalte in vorgegebene Kategorien basierend auf den vergebenen Tags. Da den Kategorien ebenfalls Tags zugeordnet sind, können diese Tags auch den Inhalten zugeordnet werden.

4.1.3. Nutzergestütztes Taggen

Vom Nutzer erzeugte Tags bilden die präziseste Möglichkeit, Inhalte geeignet mit Schlüsselwörtern zu versehen. Das liegt daran, dass der Nutzer im Gegensatz zu einem automatischen System die Bedeutung eines Inhalts besser erfassen und beurteilen kann. Er kann Formen erkennen, Muster und Farben deuten, den Kontext erfassen und interpretieren etc. Durch die so aufgenommenen Informationen ist es ihm möglich, Inhalten einen Sinn zuzuweisen und dies semantisch auszudrücken. Ein wichtiges Kriterium bei der Vergabe der Schlüsselwörter besteht darin, dass der Nutzer diese beliebig wählen kann. MASL gibt also keine Grundmenge an Schlüsselwörtern vor, aus denen die passenden gewählt werden müssen.

Den Nutzern soll zusätzlich zu der einfachen Vergabe von Schlüsselworten noch die Möglichkeit gegeben werden, die zu einem Inhalt bereits vergebenen Tags zu bewerten. Dafür muss festgelegt werden, welche Werte die Bewertungen annehmen können. Eine

Möglichkeit ist eine binäre Klassifikation in *Tag passend zum Inhalt* und *Tag nicht passend zum Inhalt*. Diese erleichtert dem Nutzer zwar das Eintragen seiner Bewertung („Ist der Tag passsend?"– „Ja/Nein"), aber sie erlaubt dem System keine Unterscheidung von Bewertungen. Diese ist aber notwendig, um darauf basierend das System zu trainieren. Eine Bewertung soll daher im Folgenden eine stetige Variable im Bereich $[0, 10]$ sein, wobei der Wert 10 für die bestmögliche Bewertung steht. Dieser Bereich wurde gewählt, da eine Bewertung in einem kleineren Intervall eine unzureichendere Unterscheidung der Bewertungen zulässt, ein wesentlich größeres Intervall hingegen den Nutzer bei der expliziten Bewertung überfordert.

4.1.4. Automatisches Taggen

Die Vorgehensweise des Nutzers beim Taggen soll beim automatischen Taggen nachgebildet werden. Das hier vorgestellte Verfahren gliedert sich in drei Arbeitsschritte. Zunächst wird versucht, über die *Inhaltsanalyse* den zu taggenden Inhalt maschinell zu erfassen und zu interpretieren. Das Ergebnis ist eine erste Liste mit möglichen Tags. Diese wird in einem zweiten Schritt einer *semantischen Analyse* unterzogen, die versucht, die bisher gefundenen Tags auf eine Konzeptualisierung abzubilden und dadurch zu erweitern. Im letzten Schritt, der *Tagzuweisung*, werden die Tags einem Inhalt zugeordnet. Parallel dazu können über Kategorien Rückschlüsse auf Tags gezogen werden. Die einzelnen Schritte werden im Folgenden näher erläutert.

Inhaltsanalyse

Ein kurzer Überblick über Methoden der Informationsanalyse von Texten, Audio, Bildern und Videos ist bereits in Abschnitt 4.1.1 gegeben worden. Je nach Typ des zu taggenden Inhalts werden diese durchlaufen, um eine Analyse und Deutung der Ressourcenbeschaffenheit durchführen zu können. Um akzeptable Ergebnisse zu erzielen, müssen die Verfahren zunächst einer aufwändigen Trainingsphase durch einen Menschen unterzogen werden. Die Qualität eines Verfahrens ist abhängig von der Anzahl der Trainingsläufe und der Anzahl und Qualität der Trainingsdokumente.

Die Umsetzung dieser Inhaltsanalyse ist nicht Teil der vorliegenden Arbeit. Es können hierfür vorhandene Konzepte wie die oben vorgestellten oder spezialisierte externe Web-Dienste wie Alipr[1] oder OpenCalais[2] verwendet werden. Der Prozess kann jedoch durch das zuvor vorgestellte Konzept der Tag-Bewertung unterstützt werden. Erzeugt der automatische Prozess zur Tagerzeugung unpassende Tags, können diese durch die Nutzer mit einer entsprechend negativen Bewertung versehen werden. Wird zu einem späteren Zeitpunkt ein neuer Inhalt automatisch mit denselben oder zumindest ähnlichen Tags versehen, ist die Wahrscheinlichkeit groß, dass sich die Inhalte ähneln. In diesem Fall

[1] www.alipr.com
[2] www.opencalais.com

können die Bewertungen der Tags des bereits vorhandenen Inhalts bei der Berechnung der Tags des neuen berücksichtigt und Tags mit schlechter Bewertung verworfen werden. Auf diese Weise hat der Nutzer einen direkten Einfluss auf die Arbeitsweise der automatischen Komponente und trainiert automatisch das System. Dieser Ansatz wurde bisher bei keinem anderen System zur automatischen Erzeugung von Schlüsselwörtern verfolgt.

Semantische Analyse

Nach Beendigung der Inhaltsanalyse liegt eine Liste mit Schlüsselwörtern für den zu taggenden Inhalt vor. In einem zweiten Schritt soll diese untersucht, interpretiert und weiter verfeinert werden. Zunächst wird in einer *Diagnose*-Phase versucht, die semantischen Bedeutungen und Verwandtschaften der bisher gefundenen Tags aufzulösen. Dazu werden zu den Schlüsselwörtern folgende Wörter bestimmt:

- Synonyme: verschiedenen Ausdrücke bei der gleichen semantischen Bedeutung.
- Meronyme: Ist-Teil-Von-Beziehungen von Ausdrücken.
- Hypernyme und Hyponyme: Ober- bzw. Unterbegriffe zu einem Ausdruck.
- Homonyme: gleicher Ausdruck bei unterschiedlicher semantischer Beziehung.

Hierfür bietet sich die Verwendung von Thesauri an, da sie die enthaltenen Wörter zueinander in Beziehung setzen können. Diese Arbeit stützt sich auf bereits existierende Arbeiten wie *WordNet*[3], das jedoch nur für die englische Sprache ausgelegt ist. Eine Erweiterung bestehender Thesauri oder eine komplette Neuentwicklung ist vorstellbar, wird jedoch hier nicht näher betrachtet.

Durch die Diagnose wird die Liste, die aus der Inhaltsanalyse entstanden ist, um weitere Tags angereichert und um unpassende Wörter reduziert. Bei diesem Vorgang fließen ebenfalls die Tag-Bewertungen mit ein, indem Tags mit schlechter Bewertung nicht berücksichtigt werden. Zusätzlich werden die Tags ihren Oberbegriffen zugeordnet bzw. die Liste mit den Oberbegriffen angereichert. Es entsteht dadurch eine erweiterte Liste mit Schlüsselwörtern und Hierarchien.

Um die bisher bestimmten Tags semantisch erfassen zu können, werden sie in einem zweiten Schritt, der *Projektion*, Konzepten zugeordnet. Dadurch erhalten die Inhalte Tags, die in ihnen vielleicht nur indirekt vorhanden sind. Für diesen Schritt eignen sich Ontologien, die Domänenwissen in maschinenlesbarer Form explizit und formal modellieren. Auf diese Weise können die gefundenen Tags um domänenspezifische Tags angereichert werden. Auch hier ist es vorstellbar, je nach Anwendungsfall umfangreiche Ontologien zu entwickeln und einzusetzen. Im Rahmen dieser Arbeit wurden jedoch vorhandene Ontologien verwendet: *YAGO* [160], eine am Max-Planck-Institut entwickelte

[3] wordnet.princeton.edu

Ontologie, hat Informationen aus der Wikipedia extrahiert und durch *WordNet*-Anfragen vereinheitlicht. Die Daten werden in RDF modelliert. Einen ähnlichen Ansatz verfolgt *DBpedia* [4]. Beide Ontologien basieren auf Englisch-sprachigen Daten.

Tagzuweisung

In einem letzten Schritt werden die gefundenen Tags den Ressourcen zugewiesen. Dafür werden sie unter dem `RatedKeywordAnnotation` Element in die MASL-Beschreibung des Inhalts eingefügt. Eine Beispielbeschreibung eines Inhalts, die dieses Element enthält, ist in Listing B.1 im Anhang B.1 zu finden.

Kategorisierung der Inhalte

Durch das Zuweisen von Schlüsselwörtern zu Inhalten und deren Bewertung entsteht von selbst eine grobe Kategorisierung dieser Inhalte. Dieser Effekt soll zusätzlich verstärkt werden, indem feste Kategorien im System erzeugt werden, denen die Inhalte gemäß ihren Schlüsselworten zugewiesen werden. Den Kategorien werden ebenfalls Schlüsselwörter zugeordnet. Dies kann z. B. durch einen Administrator geschehen. Die verwendeten Kategorien und deren Tags sind abhängig von der Anwendung.

Um eine Zuordnung eines Inhalts zu den vorhandenen Kategorien zu erlauben, werden vorhandene Tags des Inhalts und ihre Bewertung verwendet. Um einen Inhalt c einer Kategorie cat aus der Menge aller verfügbaren Kategorien Cat zuordnen zu können, muss die Relevanz $rel_{cat}(c)$ einer Kategorie $cat \in Cat$ für einen Inhalt c bestimmt werden. Diese wird gemäß Formel 4.1 berechnet. $\tilde{R}_c(tag)$ repräsentiert die durchschnittliche Bewertung eines Schlüsselworts tag für einen Inhalt c im Intervall $[0, 10]$. Je höher dieser Wert ist, desto passender ist das Schlüsselwort. $tag_{c,cat}(j)$ steht für das $j$te Tag, das in der Metadatenbeschreibung $md(c)$ des Inhalts c und Kategorie cat vorkommt, und $cnt_{tag_{cat}}$ für die Gesamtzahl der in der Kategorie cat vorkommenden Tags. Ein Inhalt wird derjenigen Kategorie zugeordnet, die bei der Berechnung den höchsten Relevanzwert im Intervall $[0, 10]$ aufweist.

$$rel_{cat}(c) = \frac{\sum_{j=1}^{n} \frac{\tilde{R}(tag_{c,cat}(j))}{10}}{cnt_{tag_{cat}}} \tag{4.1}$$

Durch diese Kategorisierung aller Inhalte können Rückschlüsse auf ähnliche Inhalte und damit weitere passende Tags gezogen werden. Außerdem werden die Tags der Kategorie dem Inhalt zugeordnet. Diese werden jedoch in der MASL-Beschreibung gesondert markiert, da sie bei der Berechnung der Kategorie nicht berücksichtigt werden sollen. Beim Einfügen neuer Tags zu einem Inhalt muss die Zuordnung zu einer Kategorie neu berechnet werden. Wird ein Inhalt einer neuen Kategorie zugeordnet, werden die Tags der alten Kategorie aus der Inhaltsbeschreibung gelöscht.

4.2. Profiling

Im folgenden Abschnitt werden Verfahren für das Profiling, also für die Erzeugung von Profilinformationen vorgestellt [186]. Grundsätzlich unterscheidet man zwei Ansätze: das *explizite* und das *implizite* Profiling.

Explizites Profiling bedeutet, dass der Nutzer dem System Informationen zu seiner Person oder seinen Vorlieben aktiv mitteilt. Dies kann über Formulareingaben oder über Bewertungen von Inhalten durch den Nutzer geschehen. Implizites Profiling hingegen benötigt keine Interaktion des Nutzers mit dem Profil, sondern das System versucht basierend auf der Beobachtung des Nutzerverhaltens Rückschlüsse auf dessen Vorlieben zu ziehen. Ziel ist es, eine durch den Nutzer durchgeführte Interaktion oder eine Kombination mehrerer Interaktionen geeignet zu bewerten. Prinzipiell liefert explizites Profiling immer die passendsten Profildaten, da der Nutzer diese selbstständig einträgt. Auf der anderen Seite soll ein Nutzer nicht zu viel Zeit mit diesen Aufgaben verbringen müssen, da dies die Nutzerzufriedenheit drastisch senkt. Laut einer von Choicestream an 1100 Internetnutzern durchgeführten Befragung wollen die meisten Nutzer maximal 5 Minuten in die Beantwortung von Fragen zu Geschmack und Interesse investieren [34].

Im Folgenden werden Ansätze für eine Kombination aus explizitem und implizitem Profiling vorgestellt, das ein in MASL gegebenes Profil mit Daten füllen kann. Die Ansätze basieren auf Bewertungen von Inhalten, beschrieben in MASL. Zunächst wird ein Überblick über verwandte Arbeiten aus dem Bereich des expliziten und impliziten Profiling gegeben. Im Anschluss daran werden die im Rahmen dieser Arbeit entwickelten Konzepte zum Profiling vorgestellt.

4.2.1. Überblick über verwandte Ansätze

Mechanismen zum expliziten Profiling arbeiten alle auf ähnliche Weise: über Formulareingaben können nach der Registrierung beim System die wichtigsten demographischen Daten eingetragen und/oder über ein Bewertungsformular die Attributwerte der wichtigsten Attributtypen bewertet werden. Welche die wichtigsten sind, ist von der Anwendung abhängig. So eine Bewertung für einen gegebenen Wertebereich $[0, 10]$ kann z. B. wie folgt aussehen: dem Nutzer wird eine Liste aller möglichen Werte des Attributtyps `Genre` vorgelegt, die alle standardmäßig mit einem Mittelwert von 5 belegt sind. Einzelne Bewertungen können dann manuell entsprechend nach oben oder unten angepasst werden, wobei eine höhere Bewertung einer höheren Präferenz entspricht. Dieser Ansatz wird z. B. bei dem von Zimmermann et al. für das Fernsehen konzipierten Personalisierungssystem [192] verfolgt und ist eine gängige Praxis bei vielen Web-Angeboten wie dem Film-Portal MovieLens[4]. Wichtig ist immer, den Aufwand für den Nutzer so gering wie möglich zu halten und die Belohnung des Nutzers, also eine möglichst gute und zutreffende Personalisierung, zu maximieren [74].

[4] movielens.umn.edu

Im Bereich des impliziten Profilings existieren unterschiedliche Ansätze aus einer beobachteten Nutzerinteraktion eine Nutzervorliebe oder –abneigung abzuleiten. Matevz et al. schlagen in [117] einen *inkrementellen Aufbau* des Profils vor. Auf eine Nutzeranfrage wird dem Nutzer eine Liste mit zukünftig ausgestrahlten Fernsehsendungen präsentiert. Wählt der Nutzer eine Sendung aus, werden alle Attribute dieser Sendung beschrieben in TV-Anytime um drei Punkte erhöht. Wird das Angebot ignoriert, werden die Attribute um einen Punkt reduziert. Die Unterschiede im Erhöhen bzw. Erniedrigen treten deswegen auf, weil die Autoren in einem ausdrücklichen Auswählen eine höhere Aussagekraft sehen als die Nicht-Beachtung einer Sendung. Auf diese Weise entsteht inkrementell eine Unterteilung der Inhalte in positive und negative, eine feinere Abstufung ist jedoch nicht vorgesehen.

Ein *statistischer Ansatz* zum Erstellen der Nutzerpräferenzen wird von Lee et al. ebenfalls für die TV-Domäne vorgeschlagen [106]. Die Präferenzen werden basierend auf einer Kategorisierung anhand von Wahrscheinlichkeiten implizit berechnet. Die Annahme dieses Ansatzes ist, dass die Vorliebe z. B. für ein bestimmtes Genre dadurch gezeigt wird, wie häufig Inhalte dieses Genres konsumiert wurden. Die Präferenz für einen bestimmten Attributwert wird berechnet, indem der prozentuale Anteil der Zeit bestimmt wird, in der Inhalte mit diesem Attributwert im Verhältnis zur Gesamtdauer aller genutzten Inhalte konsumiert wurden. Eine interessante Idee dieses Ansatzes ist es, dass mittels einer Dichtefunktion bestimmt werden soll, was die bevorzugte Fernsehzeit des Nutzers ist. Durch Kombination beider Ansätze wird dann die Empfehlung für einen Nutzer erzeugt. Auf ähnliche Weise arbeiten die Ansätze von Parsons et al. [135] und Pigeau et al. [137]. Jedoch werden hier nicht die Zeiten der Inhalte aufsummiert, sondern ihnen anhand der Dauer, die ein Inhalt bezogen auf seine Gesamtzeit angesehen wurde, ein Wert im Intervall $[0, 1]$ zugewiesen, wobei ein komplett konsumierter Inhalt den Wert 1 erhält.

Von Lee et al. [106] wird weiterhin ein *regelbasierter Ansatz* vorgeschlagen. Dieser basiert auf der Beobachtung des Nutzerverhaltens während dem Konsum kontinuierlicher Inhalte, die vom System aufgenommen wurden. Wird dieser vermehrt vorgespult, lässt das auf ein starkes Desinteresse des Nutzers schließen, während ein Zurückspulen und erneutes Konsumieren derselben Stelle für ein starkes Interesse spricht.

Innerhalb des AVATAR-Systems [18, 19, 17, 16] (siehe auch Abschnitt 5.3.4) werden Rückschlüsse auf die Nutzerpräferenzen aus dem Umgang des Nutzers mit berechneten Empfehlungen gezogen. Dafür wird untersucht, welche Empfehlungen angenommen und welche abgelehnt werden. Dies hat Einfluss darauf, wie ein Inhalt in das Profil übernommen wird. Insbesondere wird berücksichtigt, an welcher Stelle in der Empfehlungsliste der Inhalt stand. So wird ein in den Empfehlungen weit oben stehender Inhalt, der vom Nutzer abgelehnt wurde, mit einer schlechteren Bewertung übernommen, als ein weiter unten stehender. Außerdem wird überwacht, wie lange ein Nutzer einen Inhalt konsumiert: je länger, desto besser hat ihm dieser vermutlich gefallen.

4.2.2. Überblick über den Profilingmechanismus

Wie bei der Tagbewertung, wird für die Bewertung der Inhalte ebenfalls der Bereich $[0, 10]$ gewählt, wobei der Wert 10 für die bestmögliche Bewertung steht. Dieser Bereich eignet sich, da eine Bewertung in einem kleineren Intervall eine unzureichendere Unterscheidung der Bewertungen zulässt, ein wesentlich größeres Intervall hingegen den Nutzer bei der expliziten Bewertung überfordert.

Prinzipiell kann MASL eine umfangreiche Menge an Attributtypen erfassen. Es macht jedoch nur Sinn, solche Typen über das Profiling zu erfassen, die auch tatsächlich für die Personalisierung verwendet werden. Dadurch kann der gesamte Prozess effizienter und auch deutlich schneller durchgeführt werden. Eine mögliche Auswahl dieser Attributtypen wird in Abschnitt 5.4.1 diskutiert.

Der Profiling-Mechanismus soll wie folgt arbeiten: für einen vom Nutzer konsumierten Inhalt kann explizit vom Nutzer eine Bewertung eingetragen werden. Nutzt dieser diese Möglichkeit nicht, wird eine implizite Bewertung zu dem Inhalt erzeugt, basierend auf dem zuvor beobachteten Nutzerverhalten. Die Attribute dieses Inhalts, gegeben durch die MASL-Beschreibung, bekommen alle einzeln den Bewertungswert des Inhalts zugewiesen. Ist bereits zuvor ein Inhalt mit demselben Attributwert bewertet worden, wird über allen Bewertungen der Attribute der gewichtete Mittelwert gebildet. Diese Gewichtung soll zeitabhängig sein, d. h. je länger die Bewertung zurück liegt, desto weniger fällt sie bei der Berechnung der Gesamtbewertung des Attributs ins Gewicht. Dadurch kann das Profil schnell sich ändernde Nutzerpräferenzen erfassen. Außerdem wird so sicher gestellt, dass die Bewertung eines aktuell konsumierten Inhalts bei bereits umfangreich gefüllten Profilen noch einen Einfluss haben kann. Alle so bestimmten Daten und die Beschreibung der Inhalte werden anschließend in eine MASL-Beschreibung überführt.

4.2.3. Explizites Profiling

Die einfachste Art eine Bewertung für einen Inhalt zu erhalten, ist explizites Nutzerfeedback. Diese Daten sind für das System auch besonders wertvoll, da sie am besten die tatsächlichen Nutzerpräferenzen repräsentieren. Es sollen zwei Arten der expliziten Bewertung erlaubt sein: die Bewertung eines kompletten Inhalts und die Bewertung der Ausprägung eines Attributtyps.

Explizite Bewertung eines Inhalts

Während oder nachdem ein Nutzer einen Inhalt konsumiert hat, ist es ihm erlaubt, eine Bewertung abzugeben. Dem Nutzer wird die Angabe von Werten auf einer diskreten, ganzzahligen Skala im Intervall $[0, 10]$ erlaubt. Er muss also nur einen von elf verfügbaren Werten auswählen und keine Komma-genaue Bewertung eintragen.

Um den Nutzeraufwand für eine explizite Bewertung minimal zu halten, wurde bewusst darauf verzichtet, dass die durch die MASL-Beschreibungen gegebenen Attribute

eines gerade konsumierten Inhalts einzeln bewertet werden müssen. Oft ist es auch nicht möglich ein beschreibendes Attribut zu bewerten, wie es z. B. bei Schlüsselwörtern der Fall ist. Deswegen wird die Gesamtbewertung für den Inhalt als Bewertungswert für alle Attributtypen des Inhalts gegeben in MASL verwendet. Dadurch ist der Aufwand für den Nutzer gering und die Profile werden schneller aufgebaut, da mit der Bewertung eines einzigen Inhalts eine Vielzahl an Attributen bewertet wird.

Explizite Bewertung eines Attributwerts

Neben der Bewertung ganzer Inhalte, soll der Nutzer einzelne Attributwerte beurteilen können. Dadurch kann er dem System seine allgemeinen Präferenzen mitteilen, ohne dass ein bestimmter Inhalt konsumiert werden muss. Dies wird in der Regel zu Beginn der Nutzung des Personalisierungssystems sein und kann über ein Formular erfolgen, das alle Werte einzeln aufführt. Auch hier wird die bereits erwähnte Bewertungsskala, ganzzahlige Werte im Intervall $[0, 10]$, verwendet. Auf diese Weise kann der Nutzer z. B. seine Präferenz für bestimmte Ausprägungen des Attributtyps `Genre` wie `Adventure` oder `Thriller` angeben.

4.2.4. Implizites Profiling

Bei der Entwicklung von Personalisierungssystemen ist die geringe oder gar gänzlich fehlende Bereitschaft des Nutzers, dem System seine Daten zur Verfügung zu stellen, immer ein kritischer Punkt. Obwohl die zuvor vorgestellten Methoden zum expliziten Profiling für den Nutzer sehr einfach gestaltet sind, zieht es der Nutzer vor, dass ein Großteil der Arbeit vom System unsichtbar im Hintergrund ausgeführt wird [34]. Deswegen wurde ein Ansatz entwickelt, der die zuvor erläuterten expliziten Bewertungen von Inhalten automatisch erzeugen kann. Die in den folgenden Formeln auftretenden Parameter wurden größtenteils empirisch ermittelt. Zu diesem Zweck wurden in Kooperation mit dem Institut für Rundfunktechnik (IRT) umfangreiche Nutzerstudien durchgeführt.

Berechnung der impliziten Bewertung

Wie die explizit erzeugten Bewertungen soll die implizite Bewertung einen Wert im Intervall [0,10] annehmen. Dadurch wird eine Vergleichbarkeit der beiden Werte ermöglicht. In der Regel bewertet ein Nutzer jedoch selten einen Inhalt mit einem Wert nahe der Extrema, weswegen dies auch bei der impliziten Bewertung nicht passieren soll.

Mittels impliziter Bewertung werden Inhalte basierend auf dem Nutzerverhalten vom System bewertet. Im hier vorgestellten Ansatz werden dafür empirisch erstellte Verhaltensregeln verwendet. Die Dauer, über die ein Nutzer einen Inhalt konsumiert hat, wird betrachtet, um eine möglichst treffende Bewertung zu erzeugen. Der Grundgedanke ist folgender: je länger ein Nutzer einen Inhalt konsumiert, desto besser gefällt ihm dieser. Bei kontinuierlichen Medien scheint der konsumierte prozentuale Anteil des Inhalts ein

passender Ausgangspunkt zu sein, die implizite Bewertung zu berechnen. Zusätzlich soll aber die Gesamtlaufzeit des Inhalts berücksichtigt werden: je länger die Gesamtdauer eines kontinuierlichen Inhalts, desto aussagekräftiger wird der prozentuale Anteil, der vom Nutzer konsumiert wurde. So soll z. B. beim Konsum zweier Videos, wobei das eine eine Laufzeit von 15 Minuten, das andere eine Laufzeit von 120 Minuten hat, die Tatsache höher bewertet werden, dass der Nutzer den langen Inhalt zu 100% konsumiert. Dieselben Überlegungen kommen bei diskreten Medien zum Tragen. Bei diesen muss nur ein Standard-Wert angegeben werden, wie lange ein Nutzer in der Regel benötigt, um ihn voll zu erfassen. Dieser kann bei Texten z. B. aus der durchschnittlichen Lesegeschwindigkeit und der Anzahl der Wörter errechnet werden. Damit ist die Berechnung der impliziten Bewertung bei kontinuierlichen und diskreten Medien abhängig von der Gesamtdauer eines Inhalts und dem prozentualen Anteil, den der Nutzer diesen konsumiert hat.

In der Regel benötigen Nutzer eine gewisse Zeit, um sich mit einem Inhalt vertraut zu machen. Daher soll der Bewertungsmechanismus einen Inhalt erst berücksichtigen, wenn der Nutzer mindestens einen bestimmten Prozentsatz des Inhalts konsumiert hat. Empirische Untersuchungen haben gezeigt, dass ein Wert von 10% gute Ergebnisse liefert. Vorstellbar ist aber auch ein absoluter zeitlicher Wert wie z. B. 10 Minuten, wobei dann kürzere Inhalte nicht berücksichtigt werden, oder eine Kombination aus beidem (nach 10 Minuten oder nach Konsum von 10% des Inhalts). Dieser Wert muss je nach Anwendungsfall geeignet ausgewählt werden.

Diese Überlegungen werden in Formel 4.2 mathematisch ausgedrückt. $R(c)$ steht für die implizit berechnete Bewertung von Inhalt c im Intervall $[0, 10]$, $cd(c)$ für die Dauer in Minuten, die ein Nutzer den Inhalt konsumiert hat und $d(c)$ für die Gesamtdauer des Inhalts in Minuten. Da längere Inhalte einen stärkeren Einfluss auf die Bewertungen im Profil haben sollen, wird zusätzlich noch der Gewichtungsfaktor $Wf(d(c))$ und der Spreizungsexponent $Se(d(c))$ eingeführt. Durch den Spreizungsexponenten erhalten die Bewertungen längerer Inhalte eine größere mögliche Abweichung von der mittleren bzw. neutralen Bewertung mit dem Wert 5. Der Gewichtungsfaktor hingegen steigt mit der Länge der Inhalte und erlaubt für diese höhere Bewertungen.

$$R(c) = \begin{cases} \left(\frac{cd(c)}{d(c)}\right)^{Se(d(c))} * Wf\left(d(c)\right), & if\ \frac{cd(c)}{d(c)} \geq 0,1; \\ \\ keine\ Bewertung, & else. \end{cases} \tag{4.2}$$

Die genauen Definitionen von $Wf(d(c))$ bzw. $Se(d(c))$ wurden empirisch ermittelt und sind in Formel 4.3 bzw. 4.4 zu sehen. Je nach Anwendungsfall können sie geeignet angepasst werden. Durch beide Variablen wird es ermöglicht, die Varianz der Bewertung und deren maximal mögliche Ausprägung von der Gesamtdauer des Inhalts abhängig zu machen. Dies lässt sich an einem Beispiel verdeutlichen: hört ein Nutzer ein dreiminütiges Musikstück zu 80%, ergibt dies eine implizite Bewertung von gerundet 6,4. Schaut er

hingegen ein Video mit einer Gesamtlänge von 120 Minuten zu 80%, wird dieser mit 7,5 bewertet. Auf diese Weise wird ein kurzer Inhalt zwar auch noch als gut bewertet, jedoch mit einem geringeren Wert als ein deutlich längerer.

$$Wf(d(c)) = \begin{cases} 7 + \frac{d(c)}{60\ min}, & if \quad \text{d(c)} < 180 \text{ min ;} \\ \\ 10, & else \end{cases} \tag{4.3}$$

$$Se(d(c)) = \begin{cases} 0,4 + \frac{d(c)}{300\ min}, & if \quad \text{d(c)} < 180 \text{ min;} \\ \\ 1, & else \end{cases} \tag{4.4}$$

Berücksichtigung von Zapping-Verhalten und Werbepausen

Wird ein Inhalt zu einem bestimmten Zeitpunkt ausgestrahlt wie z. B. beim Fernsehen oder Radio und ist nicht permanent verfügbar, kann beim Nutzer folgendes Verhalten auftreten: ein Inhalt wird oft nur für eine kurze Zeit konsumiert und dann zu einem anderen gewechselt. Dies wird solange durchgeführt, bis ein passender Inhalt gefunden wurde. Diese Verhalten bezeichnet man als *Zapping* oder auch *Channel Surfing*. Da dies dazu führen kann, dass viele Inhalte fälschlicherweise schlecht bewertet werden, wird ein zusätzlicher Schwellwert th eingeführt. Die Dauer $cd(c)$, die ein Nutzer mit dem Konsum eines Inhalts c verbringt, muss nicht zusammenhängend sein, sondern wird aus i verschiedenen Zeitintervallen $cd_i(c)$ zusammengesetzt. Um Zapping-Verhalten zu berücksichtigen, werden in der gesamten Konsum-Dauer nur die $cd_i(c)$ berücksichtigt, die den Schwellwert th übersteigen (siehe Formel 4.5). Empirisch wurde 30 Sekunden als ein guter Wert für th ermittelt.

$$cd(c) = \sum cd_i(c), \qquad \forall\ cd_i(c) > th \tag{4.5}$$

Einige Inhalte wie Fernseh- oder Radioshows werden regelmäßig durch Werbung unterbrochen. Die meisten Nutzer verwenden diese Zeiten, um durch die andere Kanäle zu schalten. Dies kann aber fälschlicherweise zu schlechten Bewertungen des ursprünglichen Inhalts und den in der Werbepause konsumierten alternativen Inhalten führen. Um dies zu vermeiden, sollen Inhalte, die als Alternativen in den Werbepausen konsumiert werden, nicht implizit bewertet werden, und die Bewertung des ursprünglich konsumierten Inhalts nicht auf Grund der Abwesenheit des Nutzers in der Werbepause schlechter bewertet werden. Zu diesem Zweck werden zwei weitere Parameter eingeführt. $sc(c)$ bezieht sich auf die Anzahl, wie oft ein Nutzer zu Inhalt c umgeschaltet hat. Die Konstante ad gibt die maximale Dauer einer Werbepause in Minuten an. Normalerweise wird dieser

Wert gesetzlich festgelegt. In Deutschland z. B. ist hierfür der *Rundfunkstaatsvertrag*[5] zuständig. Dieser reguliert die Werbepausen in Abhängigkeit des Senders, der Sendung, der Art der Werbung und der aktuellen Uhrzeit. Mit Hilfe dieser beiden Parameter wird eine zusätzliche Bedingung formuliert, die eintreten muss, damit eine implizite Bewertung durchgeführt wird. Inhalte sollen nur dann berücksichtigt werden, wenn die einzelnen Inhaltsstücke $cd_i(c)$ durchschnittlich länger konsumiert wurden als die maximale Werbedauer. Dies wird wie in Formel 4.6 zu sehen ausgedrückt, wobei $cd(c)$ wie in Formel 4.5 definiert ist.

$$\frac{cd(c)}{sc(c)} > ad \tag{4.6}$$

Da $d(c)$ die Dauer eines Inhalts c inklusive aller Werbepausen angibt, müssen alle Abwesenheiten $ab_i(c)$ vom Inhalt c von $d(c)$ subtrahiert werden, wenn diese kürzer sind als die maximale Dauer einer Werbepause ad. Dies wird in Formel 4.7 dargestellt. Wenn $ab_i(c) \geq ad$ gilt, wird dies nicht berücksichtigt, da man davon ausgehen kann, dass der Nutzer einen anderen interessanten Inhalt während des Zappens gefunden hat, der ihm mehr zusagt.

$$d(c)^* = d(c) - \sum ab_i(c), \qquad \forall ab_i(c) < ad \tag{4.7}$$

Mit diesen Überlegungen müssen die Formeln 4.2, 4.3 und 4.4 zu Formel 4.8, 4.9 bzw. 4.10 angepasst werden. Dadurch werden Inhalte nur berücksichtigt, wenn sie länger als die durchschnittliche Dauer einer Werbepause konsumiert wurden.

$$R(c) = \begin{cases} \left(\frac{cd(c)}{d(c)^*}\right)^{Se(d(c)^*)} * Wf\left(d(c)^*\right), & if\ \frac{cd(c)}{d(c)^*} \geq 0,1 \wedge \\ & \wedge \frac{cd(c)}{sc(c)} > ad; \\ keine\ Bewertung, & else. \end{cases} \tag{4.8}$$

$$Wf(d(c)^*) = \begin{cases} 7 + \frac{d(c)^*}{60}, & if\ d(c)^* < 180; \\ 10, & else. \end{cases} \tag{4.9}$$

$$Se(d(c)^*) = \begin{cases} 0,4 + \frac{d(c)^*}{300}, & if\ d(c)^* < 180; \\ 1, & else. \end{cases} \tag{4.10}$$

[5] Staatsvertrag für Rundfunk und Telemedien (Rundfunkstaatsvertrag - RStV) vom 31.08.1991, zuletzt geändert durch Artikel 1 des Neunten Staatsvertrages zur Änderung rundfunkrechtlicher Staatsverträge vom 31.07. bis 10.10.2006 (GBl. BW 2007 S. 111), in Kraft getreten am 01.03.2007.

4.2.5. Profilerzeugung

Die explizit oder implizit erzeugten Bewertungen müssen in einem zweiten Schritt geeignet in das in MASL gegebene Profil eingetragen werden. Dabei ist zu beachten, dass bei einer expliziten Bewertung durch den Nutzer auch immer eine implizite Bewertung erzeugt wurde, da das System zu Beginn des Konsums des Inhalts durch den Nutzer nicht wissen kann, ob er ihn explizit bewerten wird. Es gibt verschiedene Möglichkeiten mit diesen beiden Bewertungen zu verfahren. So kann im Profil ein Mittelwert aus beiden hinterlegt werden. Dadurch wird aber der Stellenwert der expliziten Bewertung geschmälert und u. U. bei sehr unterschiedlichen expliziten und impliziten Bewertungen ein nicht den Nutzerpräferenzen entsprechender Wert hinterlegt. Daher ist die explizite Bewertung der impliziten vorzuziehen und wird im Profil gespeichert. Die implizite wird verworfen. Es ist vorstellbar, die implizite Bewertung mit der expliziten zu vergleichen und daraus Rückschlüsse auf notwendige Anpassungen des impliziten Bewertungsverfahrens zu schließen. Dies ist jedoch nicht im Rahmen dieser Arbeit näher untersucht worden.

Profiling-Mechanismen beschäftigen sich in der Regel damit, unbekannte Inhalte in das Profil aufzunehmen. Es kann aber auch vorkommen, dass ein Nutzer einen Inhalt wiederholt konsumiert. Im Profil liegt dann eine explizite oder implizite Bewertung dieses Inhalts vor. Oft verhalten sich Nutzer bei der Mediennutzung eines bereits bekannten Inhalts anders als bei komplett unbekannten Inhalten. So kann es sein, dass er den Inhalt interessant findet, ihn aber nicht bis zum Ende konsumiert, da er das Ende bereits kennt. Ein solches Verhalten führt fälschlicherweise zu einer schlechten impliziten Bewertung des Inhalts. Von daher sollen bei dem hier vorgeschlagenen Ansatz bereits vorhandene Bewertungen nur von expliziten Bewertungen ersetzt werden. Dahinter steht die Annahme, dass bei einer expliziten Bewertung der Nutzer sich selbst eine Meinung gebildet, bzw. diese geändert hat und den Inhalt aktuell bewertet.

Basierend auf den expliziten oder impliziten Bewertungen der Inhalte, werden die Bewertungen der Attributwerte $a_i(j)$ aller in der Beschreibung $md(c)$ des Inhalts c gegebenen und berücksichtigten Attributtypen a_i errechnet. Zusätzlich wird der Zeit-abhängige Gültigkeitsfaktor $T(c)$ eingeführt. Dieser soll verhindern, dass die Profile zu statisch werden, und erlaubt eine schnelle Reaktion auf geänderte Nutzerpräferenzen. Wurde ein Inhalt c gerade von einem Nutzer konsumiert, wird $T(c)$ auf den Wert 1 gesetzt. Im Laufe der Zeit konvergiert er gegen 0. Bewertungen können so z. B. nach einem Jahr ungültig und damit gelöscht werden, indem der Bewertungswert wöchentlich neu berechnet wird gemäß $\frac{52-t}{52}$, wobei t die Anzahl der Wochen angibt, die seit der Bewertung verstrichen sind. Je nach Anwendungsfall kann dieser Wert angepasst werden. Basierend darauf werden die Bewertungen einzelner Attributwerte gemäß Formel 4.11 berechnet, wobei $R(c_k)$ für die Bewertungen aller Inhalte c_k steht, die den Attributwert $a_i(j)$ in der Beschreibung enthalten. Auf diese Weise haben ältere Bewertungen einen geringeren Einfluss auf den Gesamtwert eines Attributs. Diese Berechnung wird nur durchgeführt,

wenn keine gültige explizite Bewertung für diesen Attributwert vorliegt.

$$R(a_i(j)) = \frac{\sum_{k=1}^{N} R(c_k) * T(c_k)}{\sum_{k=1}^{N} T(c_k)}, \qquad \forall\ md(c_k) \ni a_i(j) \tag{4.11}$$

Nicht nur die Bewertungen von Inhalten können ihre Gültigkeit verlieren, das soll auch für einzelne Attributwerte gelten. Der zu einem Attributwert $a_i(j)$ vom Attributtyp a_i zugehörige Gültigkeitsfaktor ergibt sich wie in Formel 4.12 zu sehen aus der Summe aller Gültigkeitsfaktoren $T(c_k)$ aller Inhalte c_k, die den Attributwert $a_i(j)$ enthalten.

$$T(a_i(j)) = \sum_{k=1}^{N} T(c_k), \qquad \forall\ md(c_k) \ni a_i(j) \tag{4.12}$$

Um einzelne Inhalte oder Attributwerte vor dem Löschen zu schützen, die für den Nutzer von spezieller Wichtigkeit sind, aber nicht regelmäßig von ihm konsumiert werden, können diese markiert werden. Für alle markierten Inhalte c_k ist der zugehörige Wert von $T(c_k)$ permanent auf 1 gesetzt.

Damit wird in die in MASL gegebenen Nutzerprofile die Bewertung des Inhalts in die `UsageHistorySummary` übernommen und die relevanten Attribute mit derselben Bewertung in die `TimeDependentUserPreferences`. Ein Beispiel-Profil ist in Listing B.3 in Anhang B.2 zu finden.

4.3. Bewertung der Ansätze

In diesem Kapitel wurden Ansätze diskutiert, wie man die Inhaltsbeschreibungen und Profile, die in MASL beschrieben werden, mit Werten füllen kann. Es wurde ein Ansatz für die Gewinnung von Schlüsselwörtern und für das Profiling vorgestellt. Der ganze Themenkomplex der automatischen Datengewinnung ist sehr umfangreich und bedarf der Entwicklung weiterer Ideen, um den vollen Sprachumfang von MASL unterstützen zu können. Dies konnte im Rahmen dieser Arbeit nur exemplarisch im vorgestellten Umfang untersucht werden, da dies nicht im Kernbereich der Kontext-abhängigen Personalisierung liegt.

Die hier beschriebene Idee zur Gewinnung von Schlüsselwörtern gliedert sich in einen nutzergestützten und einen automatischen Teil. Neben der reinen Angabe von Tags für Inhalte durch den Nutzer wurde zusätzlich das Konzept der Tagbewertung eingeführt. Dieses erlaubt den Nutzern, dem automatischen Prozess Feedback zu geben, was die Qualität der erzeugten Tags erhöht. Dadurch, dass schlecht bewertete Tags nicht weiter berücksichtigt werden, wird das System nebenbei durch die Nutzer trainiert.

Der automatische Teil verwendet Technologien wie Werkzeuge zur Inhaltsanalyse, Thesauri und Ontologien, um die Inhalte erfassen und interpretieren zu können. Da diese nur als Hilfsmittel verwendet werden, nicht jedoch im Mittelpunkt der Kontext-abhängigen

Personalisierung stehen, wurden keine eigenen Ansätze entwickelt, sondern es wurde sich auf bereits vorhandene Systeme wie WordNet oder YAGO gestützt. Durch Verwendung dieser umfangreichen und gut untersuchten Ansätze, kann die semantische Analyse qualitativ hochwertige Ergebnisse berechnen.

Zusätzlich wird bei der Gewinnung von Schlüsselwörtern die Idee verfolgt, Inhalte gemäß ihrer Tags zu kategorisieren. Durch die Zuteilung der Inhalte zu vorhandenen und getaggten Kategorien können sie sinnvoll gruppiert und ähnliche Inhalte schneller gefunden werden. Indem die Tags der Kategorien zusätzlich zu den Inhalten hinzugefügt werden, können auf einfache Weise umfangreiche Mengen an Tags zu Inhalten gefunden werden.

Der Profiling-Mechanismus erzeugt umfangreiche Profilinformationen basierend auf Inhaltsbewertungen, die entweder explizit oder implizit gegeben wurden. Aus den Bewertungen kompletter Inhalte können Rückschlüsse auf die Präferenz des Nutzers für einzelne Attributwerte der in MASL gegebenen Attributtypen gezogen werden. Eine Besonderheit ist, dass diese Bewertungen zeitabhängig sind, dass also eine kürzlich abgegebene Bewertung einen stärkeren Einfluss hat, als eine Bewertung, die z. B. vor einem Jahr gegeben wurde. Dadurch sind die Profile deutlich flexibler und spiegeln schneller geänderte Nutzerpräferenzen wider.

Der vorgestellte implizite Ansatz berücksichtigt die Unterschiede von Inhalten, die permanent oder nur zu einem bestimmten Zeitpunkt verfügbar sind. Außerdem werden Zappingverhalten oder ein Umschalten in Werbepausen geeignet beachtet. Das verhindert, dass Inhalte zu Unrecht schlecht bewertet werden.

4.4. Zusammenfassung

In diesem Kapitel wurden zwei Ansätze zur expliziten und impliziten Datengewinnung vorgestellt: die nutzergestützte und automatische Gewinnung von Schlüsselwörtern sowie das explizite und implizite Profiling. Beide Ansätze liegen nicht im Kernbereich der Kontext-abhängigen Personalisierung, zeigen aber Möglichkeiten, die MASL-Beschreibungen des Inhalts und des Profils mit Inhalten zu füllen.

Das nutzergestützte und automatische Tagging liefert Ansätze, Inhalte schnell und unkompliziert zu verschlagworten. Es zeichnet sich durch die Idee der Tagbewertung und der Kategorisierung zur Erzeugung von Tags aus. Durch die Integration existierender Systeme kann es schnell qualitativ hochwertige Tags erzeugen.

Der vorgestellte Ansatz zum expliziten und impliziten Profiling verwendet Nutzerbewertungen, um auf Präferenzen schließen zu können. Eine Besonderheit ist die zeitlich begrenzte Gültigkeit der Bewertungen, die eine schnelle Reaktion auf geänderte Nutzervorlieben erlaubt. Der in Abschnitt 5.6 vorgestellte Ansatz zur Nutzerprofil-basierten Empfehlung von Inhalten kann gut mit den durch dieses Profiling-Verfahren erzeugten Daten arbeiten.

Kapitel 5.

Kontext-abhängige Empfehlungssysteme

Wie in Abschnitt 2.1.3 beschrieben, werden im Rahmen dieser Arbeit zwei Arten der Kontext-abhängigen Personalisierung multimedialer Inhalte genauer untersucht: die Kontext-abhängige Auswahl der Inhalte sowie deren Kontext-abhängige Anpassung der Darstellung. Im folgenden Kapitel soll auf die Auswahl der Inhalte näher eingegangen werden. Dafür werden Empfehlungssysteme verwendet, deren grundlegende Techniken zunächst näher erläutert werden. Im Anschluss daran werden Anforderungen an ein Kontext-abhängiges Empfehlungssystem aufgestellt und verwandte Ansätze unter Berücksichtigung dieser Anforderungen näher untersucht. In diesem Kapitel werden unterschiedliche Konzepte erarbeitet, die die Nachteile geläufiger Algorithmen beseitigen (Abschnitt 5.6) oder die Struktur zwischen Inhalten und zugehörigen Kommentaren bei der Empfehlung bewahren (Abschnitt 5.7). Der Schwerpunkt dieses Kapitels liegt jedoch auf den Empfehlungsalgorithmen, die insbesondere den dynamischen Kontext berücksichtigen. Dafür werden Ansätze zum Ähnlichkeitsvergleich von Kontextinformationen vorgestellt, die beim Abgleich zwischen Inhaltsbeschreibungen und Nutzerprofilen verwendet werden (Abschnitt 5.5). Alle vorgestellten Konzepte werden in ein Framework integriert, das es erlaubt, Inhalte in einem verteilten Ablauf dem Nutzer zu empfehlen (Abschnitt 5.8).

Die wichtigsten im Folgenden verwendeten Bezeichnungen sind in Anhang C aufgelistet.

5.1. Grundlegende Techniken von Empfehlungssystemen

Die in Empfehlungssystemen eingesetzten Methoden spielen im Rahmen dieser Arbeit eine wichtige Rolle, weswegen sie im Folgenden kurz erläutert werden. Terveen und Hill haben in [163] ein allgemeines, schematisches Modell für den Empfehlungsprozess in einem computergestützten System entwickelt (siehe Abbildung 5.1). Das Modell unterstützt sowohl so genannte Pull-Dienste, bei denen der Empfehlungssuchende von sich aus das System explizit nach möglichen Vorschlägen anfragen kann als auch Push-Dienste, bei denen das System unabhängig von einer Nutzerinteraktion Empfehlungen generiert. Dem System liegen die Nutzerprofile vor, oder sie können vom Nutzer oder einem Profil-

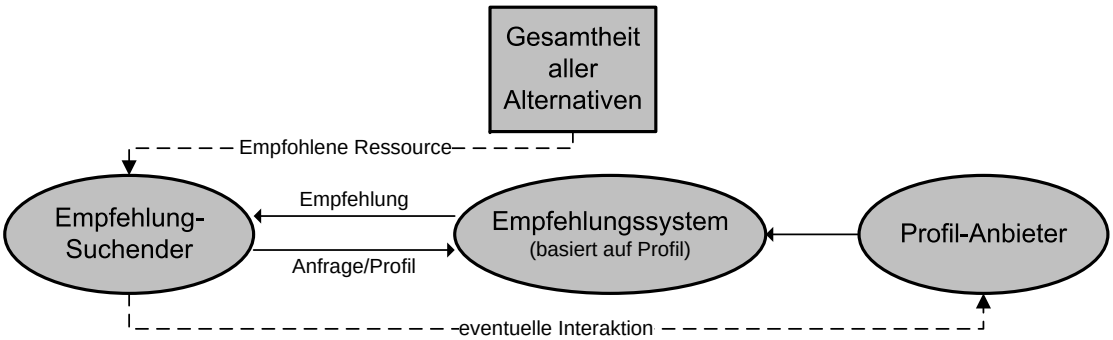

Abbildung 5.1.: Grundlegende Arbeitsweise von Empfehlungssystemen nach [163]

anbieter angefragt werden. Viele dieser Systeme arbeiten mit Ähnlichkeitsmetriken, die entweder die Ähnlichkeit von Nutzern oder die Ähnlichkeit von Inhalten bestimmen. Die gängigsten Filterverfahren werden im Folgenden näher erläutert.

5.1.1. Regelbasiertes Filtern

Beim Regelbasierten Filtern (Rule-based Filtering) stehen Regeln im Vordergrund, die auf die Informationen des Nutzerprofils angewendet werden. Regeln haben die Form *WENN(Bedingung) DANN Aktion*. Sie beziehen sich hauptsächlich auf demographische Daten wie Alter, Geschlecht oder Staatsangehörigkeit, können aber auch auf beobachtete Informationen, wie die Anzahl der Besuche oder die bisher betrachteten oder erworbenen Produkte angewendet werden. Das Erstellen von Regeln ist zeitintensiv und kann entweder manuell durch Marketingexperten und Analysten, oder computergestützt durch die Methoden der Forschungsbereiche des *Data Mining* und der *Knowlegde Discovery in Databases* durchgeführt werden.

Obwohl diese Art des Filterns relativ einfach zu implementieren ist und auch bei geringen vorhandenen Datenmengen zu einem guten Ergebnis führt, kann die Komplexität des Systems durch eine Vielzahl an neuen, sich u. U. sogar widersprechenden Regeln im Laufe der Zeit erheblich zunehmen. Das führt zu einem hohen Aufwand um das System zu warten und die Regeln an die aktuellen Trends anzupassen. Deshalb wird Regelbasiertes Filtern im Folgenden nicht weiter betrachtet.

5.1.2. Inhaltsbasiertes Filtern

Das inhaltsbasierte Filtern (Content-based Filtering) leitet Wissen direkt aus der Bestimmung von Ähnlichkeiten von Objekten ab. Es bezieht sich ausschließlich auf die Daten eines einzelnen Nutzers, was zu einer hohen Empfehlungsgüte führt. Die Ähnlichkeit zwischen Objekten (z. B. zwischen einem dem Nutzer bekannten Objekt aus

seiner Nutzungshistorie und einem unbekannten Objekt) oder einem Benutzerprofil und einem Objekt (z. B. zwischen Nutzerpräferenzen und einem unbekannten Objekt) wird über die jeweiligen Eigenschaften bestimmt. Dies setzt voraus, dass die Eigenschaften eines Objekts geeignet beschrieben sind, bzw. aus dem Objekt extrahiert werden können. Beispiele für diese Eigenschaften sind die häufigsten in einem Text vorkommenden Wörter, die Audioeigenschaften eines Musikstücks oder semantische Informationen wie Genre oder Handlung eines Spielfilms. Im Rahmen dieser Arbeit wird vorausgesetzt, dass alle relevanten Eigenschaften der Inhalte und Nutzer in einer Metadaten-Beschreibung vorliegen (vgl. Kapitel 3).

Zur Bestimmung der Ähnlichkeit existieren verschiedene grundsätzliche Ansätze, die für das inhaltsbasierte Filtern verwendet werden können [9, 5]:

- Ansätze basierend auf dem **bool'schen Modell**: Hierbei handelt es sich um ein Exact Match Verfahren. Die Existenz einer bestimmten Eigenschaft wird durch einen Binärwert ausgedrückt. Die gesamte Ähnlichkeit wird durch Auswertung eines bool'schen Ausdrucks bestimmt. Der Ansatz ist zwar relativ einfach umsetzbar, Gewichtungen von einzelnen Attributen können jedoch nicht berücksichtigt werden.

- Ansätze basierend auf dem **Vektormodell**: Hierbei handelt es sich um ein Best Match Verfahren. Eigenschaften von Nutzerprofil und Objekten werden als Vektoren in einem mehrdimensionalen Featureraum dargestellt und deren Ähnlichkeit mittels Distanzfunktionen bestimmt. Beispiel für die Distanzfunktion sind die Euklidische Distanz oder die Manhattan-Distanz. Dieser Ansatz erlaubt die Gewichtung einzelner Eigenschaften und liefert auch dann ein brauchbares Ergebnis, wenn es keine perfekte Ähnlichkeit gibt.

- Ansätze basierend auf einem **Wahrscheinlichkeitsmodell**: Dieses Verfahren arbeitet gemäß dem Prinzip der Maximum Likelihood. Die Vektoren des Featureraumes, die aus dem Nutzerprofil erstellt werden, werden verwendet, um unbekannte Objekte in Klassen einzuordnen. Die Klassen werden als statistische Prozesse beschrieben und die Regeln zur Klassifikation mit Hilfe des Satzes von Bayes als bedingte Wahrscheinlichkeit definiert. Ein vereinfachter Ansatz, auch naiver Bayes-Ansatz genannt, setzt voraus, dass jedes Attribut nur vom Klassenattribut abhängt. Diese Bayes'schen Klassifikatoren sind aber nur bei kleinen Datenmengen effizient.

  Ähnlich arbeiten Bayes'sche Netzwerke. Diese stellen mittels azyklischer Graphen in kompakter Form die gemeinsame Wahrscheinlichkeitsverteilung aller Variablen dar. Knoten sind die Variablen und Kanten die bedingten Abhängigkeiten. Die Erstellung dieser Graphen ist meist recht zeitaufwändig. Daher werden sie nur bei relativ statischen Präferenzen verwendet.

  Beide Ansätze werden meist mittels eines Trainingsdatensatzes, also einer Teilmenge der Nutzer-Objekt-Matrix, vorberechnet, um bei der tatsächlichen Prognose

deutlich schneller zu arbeiten. Durch diese Informationsverdichtung bei der Modellbildung kann allerdings ein erheblicher Informationsverlust eintreten.

- Ansätze basierend auf einem **Entscheidungsbaum**: Dieser Ansatz arbeitet mittels Entscheidungsbäumen, bei denen die inneren Knoten die Merkmale, die Kanten einen Test auf einem Merkmal des Elternknotens und die Blätter die Klassen darstellen, in die die Datensätze einzuteilen sind. Dieser Baum kann ebenfalls mit Gewichten verfeinert werden.

5.1.3. Kollaboratives Filtern

Die Grundidee des kollaborativen oder auch kooperativen Filterns (Collaborative Filtering) besteht darin, dass Nutzerprofile mehrerer Nutzer verwendet werden, um eine gemeinschaftliche Informations-Filterung durchzuführen. Im Unterschied zum inhaltsbasierten Filtern werden nicht unbedingt Informationen über die Inhalte benötigt. Grundsätzlich unterscheidet man zwei Arten: bei dem einen bestimmt die Korrelation zwischen den Objekten (oder Inhalten), bei dem anderen die Korrelation zwischen den Nutzern die Empfehlung [176].

- **Item-basiertes kollaboratives Filtern**: Das Nutzerprofil wird nach einem noch nicht bewerteten Inhalt durchsucht. Für diesen soll eine Vorhersage berechnet werden, wie gut er dem Nutzer wahrscheinlich gefällt. Dies wird dadurch erreicht, dass man für jeden vom Nutzer bereits bewerteten Inhalt die Ähnlichkeit zu dem noch unbekannten Inhalt abschätzt. Dazu werden die Daten von Nutzern verwendet, die beide Inhalte bereits bewertet haben, und die Inhalte in Gruppen eingeteilt. Zur Bestimmung der Ähnlichkeit gibt es mehrere Verfahren. Bei der Kosinus-basierten Ähnlichkeit werden zwei Inhalte als Vektoren in einem m-dimensionalen Nutzerraum dargestellt und die Ähnlichkeit durch den Kosinus zwischen diesen beiden Vektoren bestimmt. Bei dem Korrelations-basierten Ansatz hingegen beruht die Ähnlichkeit auf der Pearson-r Korrelation. Dazu müssen alle Nutzer bestimmt werden, die den bekannten und den unbekannten Inhalt bewertet haben.

- **Nutzer-basiertes kollaboratives Filtern**: Dieses Verfahren versucht gemäß den Bewertungen ähnliche Benutzer, auch Nachbarn genannt, zu identifizieren, also Nutzer, die eine ähnliche Meinung haben. Aus den Bewertungen der Nachbarn für Inhalte, die dem Nutzer noch unbekannt sind, können Rückschlüsse auf die wahrscheinliche Bewertung des Nutzers gezogen werden. Man kann auch hier die Algorithmen zur Erstellung der Empfehlung in unterschiedliche Klassen einteilen. Bei den Speicher-basierten Verfahren werden alle Berechnungen auf einer Nutzer-Objekt-Matrix durchgeführt und Nutzer identifiziert, die dieselben Inhalte ähnlich bewerten. Beim Modell-basierten Ansatz wird in einer Offline-Phase ein Modell erstellt auf dem dann in einer Online-Phase gearbeitet wird. Die Erstellung des

Modells ist zwar deutlich zeitaufwändiger, in der Online-Phase geht die Berechnung hingegen schneller als beim Speicher-basierten Verfahren.

5.1.4. Hybrides Filtern

Um die verschiedenen Vorteile der einzelnen Verfahren zu nutzen und deren Nachteile abzumildern, wurden verschiedene Ansätze entwickelt, die die inhaltsbasierten und kollaborativen Verfahren geeignet kombinieren. Einige dieser Ansätze werden im Folgenden näher erläutert:

- **Gewichtete Kombination** [36]: Bei diesem Ansatz wird die Bewertung eines unbekannten Inhalts getrennt mit verschiedenen Verfahren errechnet und das Gesamtergebnis anschließend gewichtet zusammengefügt. Die Gewichte können dabei variiert werden: während die Gewichte zu Beginn noch identisch sind, können sie zu einem späteren Zeitpunkt gemäß den Nutzerbewertungen der Empfehlungen angepasst werden.

- **Sequentielle Kombination** [19]: Eine Möglichkeit, inhaltsbasiertes und kollaboratives Filtern zu kombinieren ist, die Verfahren hintereinander zu schalten. Zunächst werden in einer inhaltsbasierten Phase die semantischen Ähnlichkeiten eines unbekannten Inhalts mit dem Profil der Nutzer berechnet. Für alle Nutzer, für die dieses Verfahren zu einem Ergebnis führt, wird der Inhalt entsprechend empfohlen, alle anderen Nutzer werden in einer anschließenden kollaborativen Phase weiterverarbeitet. Die Nachbarn werden basierend auf den Nutzerprofilinformationen gesucht.

- **Parallele Kombination** [155]: Eine weitere Möglichkeit ist, die Verfahren parallel Empfehlungen berechnen zu lassen. Inhaltsbasiertes Filtern wird auf Basis von textuellen Beschreibungen oder Metadaten und kollaboratives Filtern basierend auf dem Geschmack anderer Nutzer verwendet. Treten dabei Konflikte zwischen generierten Empfehlungen auf, wird die mit inhaltsbasierter Filterung generierte Empfehlung präferiert.

- **Inhaltsverstärktes kollaboratives Filtern (Content-boosted Collaborative Filtering)** [118]: Dieser Ansatz löst gezielt die Probleme der ersten Nutzerbewertung, also dass ein Inhalt erst empfohlen werden kann, wenn er von mindestens einem Nutzer bewertet wurde, und der Seltenheit, die besagt, dass es bei riesigen Datenmengen schwer ist ähnliche Nutzer zu finden. Dafür wird mittels inhaltsbasierter Filterung der vorhandenen Nutzerprofile versucht, eine vollständige Nutzer-Objekt-Matrix zu erzeugen. Dieses Verfahren zeigt eine deutliche Qualitätssteigerung bei der Identifizierung der Nachbarn eines Nutzers.

- **Inhaltsbasierte, kollaborative Empfehlungen (Content-based, Collaborative Recommendation)** [6]: Die Nutzerprofile eines inhaltsbasierten Filterverfahrens werden hier verwendet, um ähnliche Benutzer zu identifizieren. Inhalte werden entweder dann empfohlen, wenn sie eine hohe Ähnlichkeit mit dem Nutzerprofil erzielen (inhaltsbasierter Ansatz), oder wenn sie von einem Nutzer mit einem ähnlichen Profil hoch bewertet wurden (kollaborativer Ansatz).

Ob und in welcher Form die einzelnen Verfahren kombiniert werden, ist sehr stark abhängig von dem jeweiligen Einsatzgebiet. So können Algorithmen für ein Szenario gut geeignet sein, für ein anderes hingegen nicht. In der Regel liefern aber hybride Verfahren deutliche Vorteile gegenüber der Verwendung von einzelnen Verfahren und kombinieren gezielt deren Stärken. Dies geht jedoch mit einem deutlich vergrößerten Nutzerprofil und erhöhtem Aufwand für die Berechnung der Empfehlung einher.

5.2. Anforderungen an ein Kontext-abhängiges Empfehlungssystem

Im folgenden Abschnitt werden Anforderungen an ein Empfehlungssystem aufgestellt, das eine personalisierte Auswahl multimedialer Inhalte unter Berücksichtigung von Kontextinformationen unterstützt. Ein Teil der Anforderungen ergibt sich direkt aus den allgemeinen Anforderungen aus Abschnitt 2.2.4.

1. *Generierung passender Empfehlungen*
   Der wichtigste Punkt bei Empfehlungssystemen ist das Generieren von treffenden, also für den Nutzer interessanten Empfehlungen. Ohne präzise Algorithmen zur Empfehlungsfindung sinkt die Zufriedenheit des Nutzers und damit seine Bereitschaft, das System zu nutzen, wodurch es unbrauchbar wird.

2. *Empfehlung bekannter und unbekannter Objekte*
   Ziel eines Empfehlungssystems ist es, dem Nutzer aus der Menge der bisher unbekannten Objekte diejenigen zu empfehlen, die wahrscheinlich von Interesse für ihn sind. Dennoch sollen auch bereits bekannte, positiv bewertete Objekte in die Liste der Empfehlungen aufgenommen werden, weil dies das Vertrauen des Nutzers in das System erhöht. Objekte, mit denen der Nutzer in der Vergangenheit positive Erfahrungen gemacht hat, stehen in engem Zusammenhang mit der Nutzerzufriedenheit, auch wenn sie dem Nutzer keine neuen Informationen liefern. Außerdem ist es für manche Anwendungen unabdingbar, nochmals auf bereits bekannte Inhalte hinzuweisen. Ein Beispiel ist der mobile Touristenführer, der bei einer zweiten Reise an dasselbe Ziel dieselben Informationen bereitstellen soll.

3. *Empfehlung von Kommentaren*
   Wie im Szenario in Abschnitt 2.2.1 beschrieben, soll das System nicht nur die Inhal-

te empfehlen, sondern auch die zugehörigen Kommentare geeignet filtern. Dabei ist zu beachten, dass sich ein Kommentar immer auf einen Inhalt bezieht, also nicht gesondert vom Inhalt betrachtet oder empfohlen werden soll. Ebenso soll die Struktur zwischen Inhalt und Kommentar, sowie zwischen den Kommentaren untereinander, bei der Berechnung der Empfehlungen berücksichtigt werden.

4. *Berücksichtigung von Kontextinformationen*
   Um die Empfehlungen auch Kontext-abhängig erzeugen zu können, werden Ansätze benötigt, um die aktuellen Kontextinformationen des Nutzers mit den in den Beschreibungen der Inhalte enthaltenen Kontextinformationen, insbesondere den dynamischen, abgleichen zu können. Dafür müssen Verfahren entwickelt werden, die bestimmen können, wann zwei Kontextinformationen, die u. U. in unterschiedlichen Formaten bzw. Granularitäten vorliegen, überhaupt identisch oder ähnlich sind. Ein Beispiel sind zwei Ortsinformationen, wobei die eine als GPS-Koordinate, die andere als Adresse vorliegt. Die Ähnlichkeit wird vom Anwendungsfall bestimmt. So liegen bei einer auf Fußgänger ausgelegten Anwendung im Vergleich zu einer Anwendung für Autofahrer zwei ähnliche Orte räumlich meist deutlich näher aneinander.

5. *Berücksichtigung von Kreuzkorrelationen der Attribute*
   Ein wichtiger Aspekt, der außerdem die Nutzerzufriedenheit zusätzlich erhöhen kann, ist, Attribute der Inhaltsbeschreibungen nicht unabhängig voneinander zu betrachten. Das bedeutet, ein Nutzer kann bestimmte Vorlieben nicht nur für Einzelattribute, sondern auch für Attributkombinationen haben. So kann z. B. ein Nutzer einen Schauspieler nur in Filmen eines bestimmten Genres mögen, in anderen Genres aber nicht. Diese Kreuzkorrelation der Attribute muss im Empfehlungsprozess berücksichtigt werden.

6. *Reaktive und proaktive Empfehlungen*
   Empfehlungen können prinzipiell auf zwei Arten erzeugt werden: reaktiv, also als explizite Antwort auf eine Nutzeranfrage, oder proaktiv, also ohne jegliche Nutzerinteraktion mit dem System. Während im ersten Fall das System einfach nur die Kontextinformationen zur Anfragezeit auswerten muss, erfordern proaktive Empfehlungen eine Überwachung von Kontextveränderungen. Dies kann nach verschiedenen Strategien geschehen, wie Zeit-basiert (z. B. alle 5 Minuten), Distanz-basiert (z. B. alle 100 m) oder Zonen-basiert (beim Verlassen oder Betreten einer vordefinierten Zone wie z. B. eines Fußballstadiums). Die Konzeptionierung dieser Strategien ist nicht Teil der vorliegenden Arbeit, aber das hier entwickelte System soll reaktive und proaktive Empfehlungen unterstützen können.

7. *Systemfeedback*
   Die Nutzerzufriedenheit wird zusätzlich durch explizites Systemfeedback erhöht.

Das System soll dem Nutzer nicht nur die Empfehlungen präsentieren, sondern zusätzlich die Gründe für diese Empfehlungen nennen. Dies ist allgemein eine gängige Praxis in Empfehlungssystemen und gibt dem Nutzer das Gefühl, dem System nicht ausgeliefert zu sein, was das Vertrauen in das System erhöht.

8. *Anwendungsunabhängigkeit*
   Wie schon in Abschnitt 2.2.4 beschrieben, soll der in dieser Arbeit vorgestellte Ansatz zur Kontext-abhängigen Personalisierung anwendungsunabhängig sein. Dafür muss diese Anforderung natürlich auch für den Teilaspekt der Kontext-abhängigen Empfehlung gelten.

9. *Unterstützung unterschiedlicher Empfehlungsverfahren*
   Da sich die unterschiedlichen Empfehlungstechniken, also z. B. inhaltsbasiertes oder kollaboratives Filtern (vgl. Abschnitt 5.1) je nach Anwendungsfall mehr oder weniger für die Erzeugung von passenden Empfehlungen eignen, soll sich das Empfehlungssystem nicht auf eines dieser Verfahren beschränken, sondern eine möglichst umfangreiche Menge und auch Kombinationen dieser Verfahren unterstützen. Dies ergibt sich auch aus der oben bzw. in Abschnitt 2.2.4 gestellten Anforderung der Anwendungsunabhängigkeit.

10. *Unterstützung unterschiedlicher Endgeräte*
    Um unabhängig von einem bestimmten Anwendungsfall zu sein, muss das Empfehlungssystem unterschiedliche Endgeräte unterstützen können. Für eine personalisierte Auswahl von Fernsehinhalten oder einen personalisierten EPG wird z. B. eine Set-Top-Box benötigt, während personalisierte Online Videos von einem PC abgerufen werden. Und auch wenn das Empfehlungssystem nur für einen einzelnen Anwendungsfall verwendet werden soll, will der Nutzer dieselben Empfehlungen und dieselbe Empfehlungsqualität haben, wenn er nicht sein übliches Endgerät verwendet, weil er z. B. auf Reisen ist. Dafür muss das für die Empfehlung benötigte Nutzerprofil auch über Anwendungsgrenzen hinweg exportierbar sein.

11. *Unterstützung unterschiedlicher Metadatenformate*
    Das hier vorgestellte Empfehlungssystem ist speziell für die Anwendung mit der in Kapitel 3 definierten Sprache MASL gedacht. Dennoch sollen auch andere Metadatenformate unterstützt werden. MPEG-7 und MPEG-21 können leicht integriert werden, da MASL darauf aufbaut. Aber auch für andere Sprachen wie z. B. TV-Anytime für Fernsehinhalte muss es eine explizite Möglichkeit zur Konvertierung in MASL geben. Zu beachten ist, dass ohne Unterstützung von Kontextinformationen in den Metadaten die speziell für die Kontext-abhängige Auswahl entwickelten Mechanismen nicht verwendet werden können, sondern nur die Standard-Techniken wie z. B. das inhaltsbasierte Filtern bzw. deren hier vorgestellte Anpassung.

12. *Konfigurationsschnittstelle*
    Um das Empfehlungssystem möglichst unabhängig vom jeweiligen Anwendungsfall zu machen, muss eine entsprechende Konfigurationsschnittstelle vorhanden sein, die eine Anpassung des Systems an das gewünschte Anwendungsszenario erlaubt.

13. *Schutz der Privatsphäre des Nutzers*
    Bei jedem Empfehlungssystem müssen auch immer Daten über den Nutzer gesammelt, gespeichert und verarbeitet werden, die als Eingabewerte für die Filtertechniken dienen. Dies kann als schwerwiegender Eingriff in die Privatsphäre des Nutzers gewertet werden. Die Problematik verstärkt sich noch bei der Kontext-abhängigen Empfehlung, da zusätzlich die dynamischen Kontextinformationen des Nutzers verarbeitet werden, die z. B. den aktuellen Aufenthaltsort des Nutzers oder seine derzeitige Aktivität verraten. Von daher muss ein geeignetes Mittelmaß gefunden werden zwischen dem Recht des Nutzers auf Privatsphäre und der effizienten und passenden Filterung und Empfehlung von Inhalten basierend auf diesen Kontextinformationen.

14. *Geringe Belastung der Luftschnittstelle*
    Bei allen kabellosen Systemen ist die Luftschnittstelle ein limitierender Faktor. Im Gegensatz zu kabelgebundenen Systemen können nicht beliebig viele Daten zwischen einem Client- und einem Server-System übertragen werden, da dies mit einem hohem Zeitaufwand und, wenn keine Datenflatrate vorhanden ist, mit hohen monetären Kosten verbunden ist. Deswegen sollen die Ressourcen der Luftschnittstelle effizient genutzt und die Client-Server-Kommunikation auf ein Minimum beschränkt werden. Insbesondere soll vermieden werden, für den Nutzer irrelevante Inhalte auf den Client zu übertragen.

15. *Skalierbarkeit*
    Eine gute Empfehlung für einen Nutzer in einer guten Antwortzeit zu generieren, ist im Allgemeinen kein Problem. Schwieriger wird es jedoch, zeitnah für eine größere Menge von Nutzern Empfehlungen zu berechnen. Von daher muss das System auch bei großen Nutzerzahlen gut skalieren können.

16. *Effiziente Berechnung der Empfehlungen*
    Neben der Erzeugung von brauchbaren Empfehlungen ist es für eine maximale Nutzerzufriedenheit natürlich relevant, wie lange dieser auf diese Empfehlung warten musste. Da gerade in mobilen Systemen die Übertragung der Daten einige Zeit in Anspruch nehmen kann, soll die eigentliche Berechnung der Empfehlung möglichst effizient und schnell erfolgen.

5.3. Verwandte Ansätze

Im folgenden Abschnitt werden einige verwandte Arbeiten näher vorgestellt und mit den aufgestellten Anforderungen abgeglichen. Es wird unterschieden zwischen Empfehlungssystemen, die auf den Standards MPEG-7 und/oder MPEG-21 und verwandten Ansätzen wie TV-Anytime basieren, da diese als Basis für die Definition von MASL gewählt wurden (vgl. Kapitel 3), und Systemen, die explizit bereits Kontextinformationen bei der Generierung von Empfehlungen berücksichtigen.

5.3.1. P-News

Im Rahmen des von Wang, Balke et al. durchgeführten Projekts P-News [177, 8], wurde ein Empfehlungssystem entwickelt, das unbekannte Nachrichten basierend auf Nutzerpräferenzen filtert und den Nutzer Situations-orientiert auf diese hinweist. Zur Beschreibung der Präferenzen wurde ein umfangreiches Modell entwickelt, mit dessen Hilfe man genau spezifizieren kann, welche Inhalte für den Nutzer relevant sind und wie er darüber informiert werden soll. Zur Beschreibung der multimedialen Inhalte werden bei P-News Teile des MPEG-7 Standards verwendet. Im Wesentlichen sind Beschreibungen über die Strukturierung der Inhalte sowie über ihre Dekomposition in Bezug auf Zeit und Medienquelle enthalten, jedoch keinerlei low-level Beschreibungen wie Farbhistogramme oder Texturen.

Die zentrale Komponente der P-News-Architektur (vgl. Abbildung 5.2) ist der *Personalizer*, der sich wiederum aus den beiden Sub-Komponenten *Composer* und *Synthesizer* zusammensetzt. Aufgabe des Composers ist, die Nutzerprofile mit neuen, unbekannten Inhalten abzugleichen. Dies kann bei jedem neu eintreffenden Dokument, periodisch oder nach k eingetroffenen Dokumenten geschehen. Für den Abgleich extrahiert der Composer die Präferenzen aus dem *Preference Repository*, in dem die Nutzerprofile permanent gespeichert werden. Die Profile werden durch *Preference Patterns* spezifiziert, die basierend auf dem entwickelten Präferenzmodell konstruiert wurden. Durch diese Patterns werden die möglichen Interaktionen der einzelnen Nutzergruppen beschrieben und die Präferenzen auf grundlegende Verhaltensmuster zurückgeführt. Diese Methode erhöht gerade bei proaktiven Diensten wie P-News die Effizienz des Systems, da die Menge der möglichen Präferenzen auf eine einfach anwendbare Untermenge aller möglichen Profile herunter gebrochen wird. Aus den einzelnen Präferenzen des Nutzers wird vom Composer eine Gesamtpräferenz erzeugt. Zusätzlich kann er vom *Domain Repository* unterstützt werden, das zusätzliches Wissen bereit hält, z. B. mittels Domänen-Ontologien. Der Zugriff auf die Präferenzen und das zugehörige Domänen-Wissen erfolgt einheitlich über den *Preference Manager*.

Die vom Composer erzeugte Gesamtpräferenz wird verwendet, um eine Preference XPATH-Anfrage zu erzeugen. Diese Anfragesprache wurde entwickelt, um personalisierte Anfragen und den Zugriff auf die Merkmale hierarchisch strukturierter Dokumente wie XML- oder MPEG-7-Dokumente zu ermöglichen. Die Anfrage wird von der *Search En-*

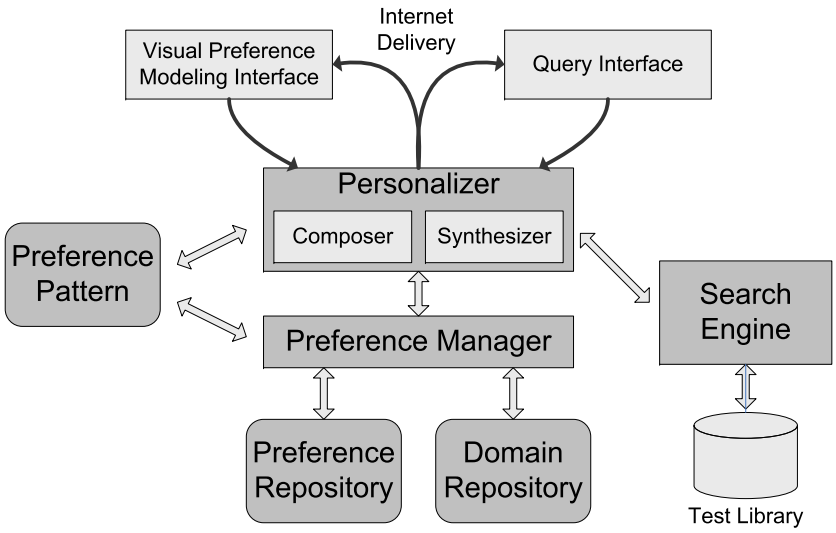

Abbildung 5.2.: P-News Architektur nach [177]

gine verarbeitet. Es wird ein Best-Matches-Only Ansatz verfolgt: zunächst wird versucht ein perfektes Ergebnis zu finden. Existiert ein solches nicht, wird nach der am besten passendsten Alternative gesucht, aber keinesfalls nach schlechteren Ergebnissen [96]. Neben harten können weiche Bedingungen angegeben werden, die für ein passendes Ergebnis nicht zwingend erfüllt sein müssen, sondern nur für ein perfektes Ergebnis. Die Antwort auf die Preference XPATH-Anfrage wird auf ihre Qualität hin untersucht, da bei dem Best-Matches-Only Ansatz durchaus qualitativ schlechte Ergebnisse zustande kommen können, und entschieden, ob und wie der Nutzer benachrichtigt werden soll. Die *Synthesizer* Komponente passt dazu die an den Nutzer zu übermittelnden Informationen und Benachrichtigungen gemäß den Nutzerpräferenzen an.

Der Ansatz des P-News-Projekts wurde für einen sehr speziellen Anwendungsfall entwickelt, weswegen es viele der Anforderungen aus Abschnitt 5.2 nicht erfüllt. So können bei diesem System keine unterschiedlichen Empfehlungstechniken verwendet werden, als Metadatenmodell ist ausschließlich MPEG-7 vorgesehen und die vorhandenen Konfigurationsschnittstellen sind nur für die Anpassung des Nutzermodells vorgesehen. Dennoch ist es für eine Verwendung auf unterschiedlichen Endgeräten geeignet, wobei bei mobiler Nutzung auf eine Ressourcen-schonende Nutzung der Luftschnittstelle geachtet wird.

Das System erzeugt proaktiv oder nach Nutzeranfrage passende Empfehlungen basierend auf den Nutzerpräferenzen. Es werden nur unbekannte Objekte berücksichtigt, da Empfehlungen von bekannten Objekten bei einem Nachrichtendienst nicht sinnvoll sind. Warum ein Inhalt ausgewählt wurde, wird dem Nutzer nicht mitgeteilt. Kontextinfor-

mationen werden vom System verwendet, jedoch nicht für eine Auswahl, sondern nur für eine rudimentäre Anpassung der Benachrichtigung und Inhalte an die momentane Nutzersituation. Eine Berücksichtigung von Kreuzkorrelationen der Attribute und von Kommentaren ist in diesem System nicht vorgesehen.

Da die Profile aller Nutzer gesammelt auf einem zentralen Server liegen, ist der Schutz der Privatsphäre des Nutzers in P-News nicht gegeben. Die Anforderung der Skalierbarkeit ist nur begrenzt erfüllt, da das System zentral für alle Nutzer Empfehlungen erzeugt. Der von P-News verfolgte Best-Matches-Only Ansatz wurde optimiert, um eine sichere Antwort zu finden ohne die Anzahl an Datenbank-Anfragen zu erhöhen.

5.3.2. TV-Trawler Projekt

Das TV-Trawler Projekt wurde von Rogers et al. entwickelt [144]. Es dient zur Filterung, Aufnahme und Auslieferung von digitalen Video Broadcast Inhalten über Satellit. Dafür werden aus dem Datenstrom extrahierte Inhaltsbeschreibungen mit vordefinierten Mengen aus persönlichen MPEG-7-Nutzerpräferenzen abgeglichen. Das System unterstützt die automatische Analyse der ausgestrahlten Videokanäle, die Auswahl von relevanten Programmen und die auf den Nutzer zugeschnittene Auslieferung der Inhalte.

Zu den Hauptzielen des Projekts gehörte, die Brauchbarkeit von MPEG-7 für die Nutzerpräferenz-basierte Filterung von relevanten audio-visuellen Inhalten, die über viele synchrone Broadcast-Kanäle versendet werden, zu beurteilen. Dafür wurde eine Vielzahl an Funktionalitäten bereit gestellt, wie die geeignete Speicherung von Nutzerpräferenzen, die automatische Extraktion von MPEG-7-Beschreibungen der Inhalte basierend z. B. auf DVB-SI (vgl. Abschnitt 3.3.4) und effiziente Algorithmen, die die Inhalte mit den Nutzerpräferenzen abgleichen. Zusätzlich werden die im MPEG-2-Format empfangenen Inhalte in MPEG-4 konvertiert, da dieses Format eine bessere Komprimierung bietet und eine Beschreibung einzelner Segmente mittels Metadaten erlaubt.

Die im Rahmen des Projekts entwickelte Architektur ist in Abbildung 5.3 zu sehen. Die *Filter Engine* empfängt das DVB-Signal, extrahiert die Inhaltsbeschreibungen und konvertiert diese in ein MPEG-7 konformes Format. Alle jemals empfangenen Metadaten werden im *Complete Metadata Archive* abgelegt. Die Beschreibungen werden mit den Nutzerpräferenzen abgeglichen und die zugehörigen Inhalte, sofern sie mit den Präferenzen übereinstimmen, in MPEG-4 konvertiert. TV-Trawler verfolgt einen inhaltsbasierten Filter-Ansatz (vgl. Abschnitt 5.1.2) nach klassischen Attributwerten wie Genre, Titel oder Autor. Anschließend wird basierend auf dem Nutzerprofil entschieden, ob das Programm unmittelbar oder erst zu einem späteren Zeitpunkt übertragen werden soll und ob der Nutzer über die neuen Inhalte informiert werden soll (*Notification & Delivery Engine*). Die *JukeBox Engine* ist für das Streamen von aufgezeichneten Inhalten zu einem ausgewählten Zeitpunkt zuständig, der ebenfalls im Nutzerprofil hinterlegt ist. Alle relevanten Einstellungen und Daten des Systems werden von einem zentralen *Metadata Store* verwaltet. Dieser beinhaltet u. a. die Nutzerpräferenzen, das persönliche Archiv

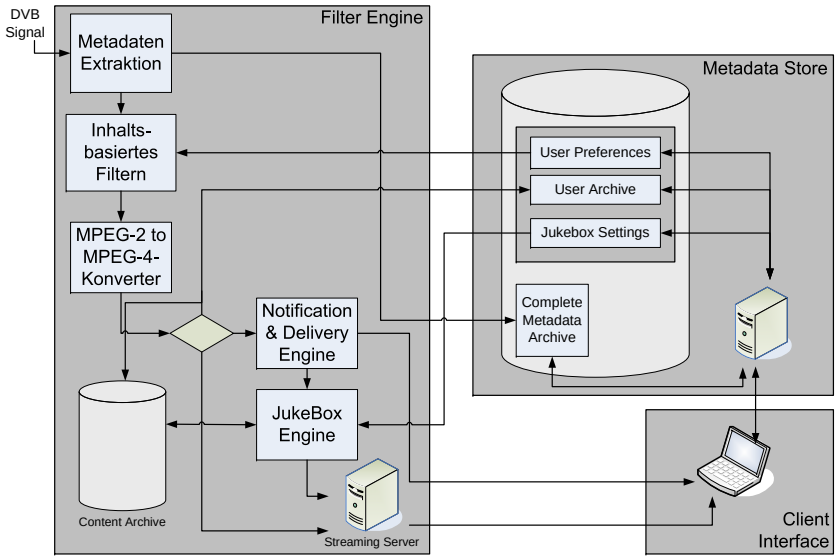

Abbildung 5.3.: TV-Trawler Architektur, vereinfacht nach [8]

des Nutzers, und des Gesamtarchivs des TV-Trawler-Systems, das aus der Summe der einzelnen Nutzerarchive besteht.

Das *Client Interface* bildet die Webschnittstelle zum Nutzer und dient zur Darstellung der Inhalte. Zusätzlich erlaubt es dem Nutzer seine Präferenzen einzutragen und Einstellungen des Systems anzupassen. Außerdem wird auf diesem Weg dem Nutzer Feedback vom System gegeben, z. B. warum ein bestimmter Inhalt ausgewählt wurde. Ein reaktiver Zugriff auf die Inhalte wird über eine Suche im persönlichen oder im Gesamtarchiv ermöglicht.

TV-Trawler wurde als Dienst zur Filterung, Aufnahme und Auslieferung von digitalen Videos, insbesondere von Fernsehinhalten, entwickelt, und ist nicht für andere Anwendungsfälle einsetzbar. Es unterstützt ausschließlich inhaltsbasiertes Filtern auf einer kleinen Attributmenge, was die Effizienz der Berechnung erhöht, aber die Empfehlungsqualität sinken lässt. Kreuzkorrelationen der Attribute werden bei der Filterung nicht berücksichtigt. Es unterscheidet nicht zwischen neuen und bekannten Inhalten, sondern betrachtet jeden vom System empfangenen Inhalt. Als Nutzerschnittstelle werden ausschließlich Web-basierte Systeme betrachtet und es gibt keine explizite Unterstützung von anderen wie z. B. mobilen Systemen. Kontextinformationen und Kommentare sind ebenfalls nicht vorgesehen.

Empfehlungen werden bei TV-Trawler proaktiv oder reaktiv erzeugt. Neben den eigentlichen Empfehlungen der Inhalte gibt das System dem Nutzer Feedback, warum ein

bestimmter Inhalt ausgewählt wurde. Da die Metadaten beim Eintreffen in das System in MPEG-7 konvertiert werden, könnte das System leicht um weitere als nur die vorgesehenen Beschreibungsmodelle ergänzt werden. Eine solche Erweiterung ist jedoch nicht explizit vorgesehen und auch andere Anpassungen sind auf Grund von fehlenden Konfigurationsschnittstellen nicht durchführbar.

Da die Profile aller Nutzer auf einem zentralen Server gesammelt werden, ist der Schutz der Privatsphäre des Nutzers nicht gegeben. Die Anforderung der Skalierbarkeit ist nicht erfüllt, da das System zentral für alle Nutzer Empfehlungen erzeugt. Für einige Teilkomponenten werden Ansätze diskutiert, die für eine bessere Skalierbarkeit sorgen.

5.3.3. MPEG-7/-21-basierte Video-Personalisierung

Von Tseng et al. wurde ein Framework entwickelt, das die Zusammenfassung und Personalisierung von Videos erlaubt [167]. Dafür sucht das System basierend auf Nutzerprofilbeschreibungen relevante Inhalte und erstellt für diese eine Zusammenfassung, die auf die aktuelle Nutzungsumgebung angepasst ist. Die Zusammenfassung soll den Nutzer beim Zugriff auf den eigentlichen Inhalt unterstützen.

Das Framework kann in unterschiedlichen Anwendungsumgebungen eingesetzt werden und besteht aus einer 3-Tier-Architektur aus Server, Client und Middleware (Abbildung 5.4), die reaktiv mit Nutzeranfragen arbeitet. Über die Client-Komponente können die Nutzer ihre Anfragen absetzen und ihre Nutzungsumgebung wie das verwendete Endgerät und dessen Eigenschaften spezifizieren (1.). Der Server beinhaltet eine Datenbank, die alle Multimediadateien und zugehörige Informationen wie Inhaltsbeschreibungen, Informationen zur Rechteverwaltung und zur Anpassung der Inhalte an das Endgerät in MPEG-7 bzw. MPEG-21 enthält.

Die Beschreibungen der Inhalte und Informationen zu deren Rechteverwaltung werden von der Middleware in der *Personalization Engine* mit den in der Nutzeranfrage angegebene Parametern abgeglichen (2., 3.) und Inhalte personalisiert ausgewählt. Dafür wird ein semantischer Clustering Algorithmus verwendet, der die Segmente der Inhalte geeignet kategorisiert. Dann werden Ähnlichkeiten und logische Zusammenhänge bestimmt.

Nach Auswahl der entsprechenden Segmente, werden die Beschreibungen der Inhalte an die *Adaptation Engine* übergeben (4.), die die für die Übertragung optimalste Version auswählt (6.) und die Inhalte an die Nutzungsumgebung des Nutzers anpasst (5.). Die so verarbeiteten Inhalte werden an den Client übertragen, der sie dem Nutzer anzeigt (7.).

Da das System ausschließlich für die Personalisierung von durch MPEG-7/21 beschriebenen Videos optimiert wurde, kann es nur für diese Metadaten eingesetzt werden, andere Formate werden nicht unterstützt. Im Videobereich kann es anwendungsunabhängig im Rahmen seiner Möglichkeiten verwendet werden, aber es stellt keine Konfigurationsschnittstellen zur Verfügung. Das System arbeitet ausschließlich reaktiv auf Nutzeran-

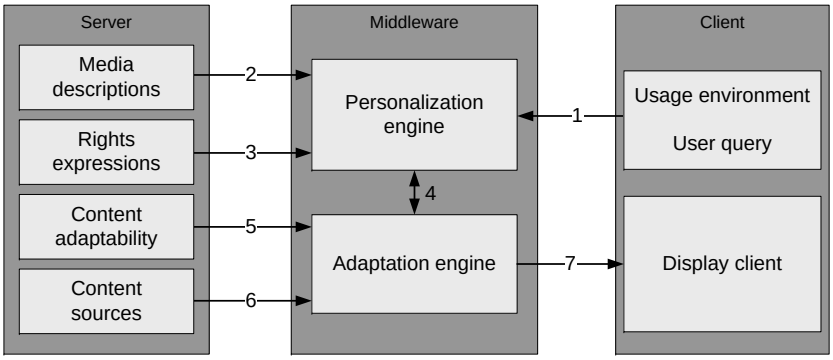

Abbildung 5.4.: Personalisierungs-Architektur nach [167]

frage. Es unterscheidet bei der Empfehlungsgenerierung nicht zwischen unbekannten und bekannten Video-Objekten. Kommentare sind nicht vorgesehen. Für die Generierung der Empfehlung wird ein eigener, effizienter Filteralgorithmus eingesetzt, der jedoch keine Kreuzkorrelationen der Attribute berücksichtigt. Warum ein bestimmter Inhalt für einen Nutzer ausgewählt wurde, wird dem Nutzer nicht mitgeteilt.

Kontextinformationen werden bei der Anfrage an das System erfasst, allerdings dienen diese ausschließlich der Anpassung der Inhalte. Eine Nutzung auf unterschiedlichen Endgeräten wird unterstützt. Bei der Übertragung werden die Einschränkungen bezüglich der Luftschnittstelle berücksichtigt. Da das System für alle Nutzer Empfehlungen zentral erzeugt, ist es nur bedingt skalierbar. Allerdings werden die Nutzerprofile nicht permanent an einer zentralen Stelle verwaltet, sondern nur bei einer Anfrage übertragen, was zumindest den Schutz der Privatsphäre im Vergleich zu rein zentralen Ansätzen verbessert.

5.3.4. AVATAR

Das von Blanco-Fernández, Arias et al. entwickelte System AVATAR dient als automatisches Empfehlungssystem insbesondere für Inhalte der TV-Domäne [18, 19, 17, 16]. Es arbeitet unter Verwendung von grundlegenden Technologien des Semantic Web und syntaktischen Suchmaschinen und nutzt den TV-Anytime Standard (vgl. Abschnitt 3.3.7) zur Beschreibung der Inhalte. Für AVATAR wurde eine Ontologie in OWL (vgl. Abschnitt 3.3.2) entwickelt, die die unterschiedlichen Attribute von Fernsehinhalten umfasst. Kern dieser Ontologie ist eine Klassenhierarchie, die basierend auf den in TV-Anytime spezifizierten Genre-Tags unterschiedliche Arten von Fernsehinhalten identifiziert (vgl. Abbildung 5.5). Die TV-Anytime-Beschreibungen der Inhalte werden von AVATAR automatisch in die interne Ontologie-Repräsentation überführt, um im System weiter verarbeitet werden zu können. Querbeziehungen aus den Beschreibungen werden extrahiert, die es erlauben, semantische Beziehungen zwischen den unterschiedlichen Klassen der

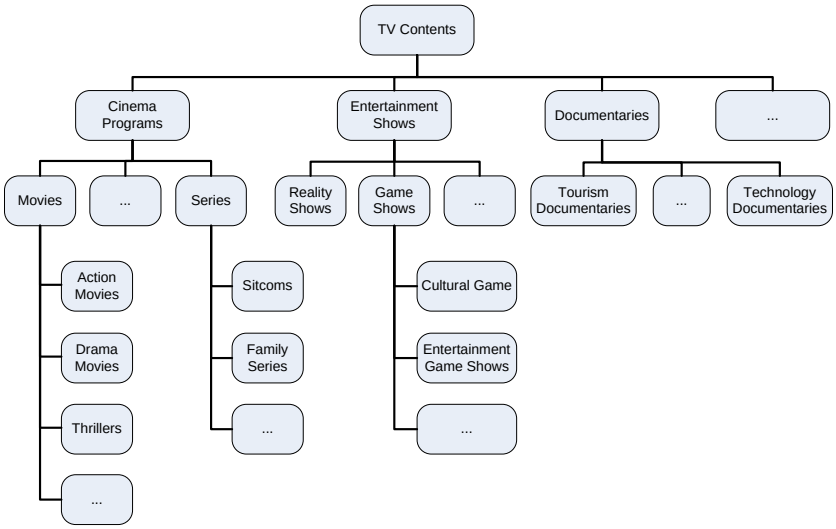

Abbildung 5.5.: Ausschnitt aus der für AVATAR entwickelten Hierarchie für Fernsehinhalte nach [19]

Ontologie herzustellen.

Zur Beschreibung der Nutzerpräferenzen werden ebenfalls Ontologien verwendet. In Abbildung 5.6 sieht man eine konkrete Ausprägung dieser Profil-Ontologie. Die Profile werden progressiv aufgebaut, in dem während der Nutzung zu einem Profil die Instanzen und Klassen hinzugefügt werden, die einer positiven oder negativen Präferenz des Nutzers entsprechen. Wenn ein neuer Inhalt eingefügt werden soll, fügt AVATAR die Instanz, die den Inhalt repräsentiert, die Klasse der Instanz und alle Elternknoten bis zur Wurzel gemäß der Hierarchie aus Abbildung 5.5 und die semantischen Beziehungen gemäß der TV-Anytime-Beschreibung ein. Diese Beziehungen sind wichtig, um mehrere Inhalte des Profils verlinken zu können. Wie sehr ein Nutzer einen Inhalt mochte, wird zusätzlich über einen Indexwert abgespeichert.

Für die eigentliche Empfehlung wird ein hybrider Ansatz aus inhaltsbasiertem und kollaborativem Filtern in einem zweistufigen Ansatz verwendet. Zunächst wird in der ersten Stufe ein unbekannter Inhalt mit allen verfügbaren Nutzerprofilen inhaltsbasiert verglichen. Dieser Berücksichtigt die Tiefe der jeweiligen Knoten in der Hierarchie. Je mehr Übereinstimmungen der Inhalt mit dem Profil hat, desto höher ist der entsprechende Relevanzwert. Alle Nutzer, deren berechneter Relevanzwert einen Schwellwert übersteigt, bekommen den Inhalt empfohlen. Für alle anderen wird in der zweiten Stufe unter Berücksichtigung der nächsten Nachbarn kollaborativ ein neuer Relevanzwert berechnet.

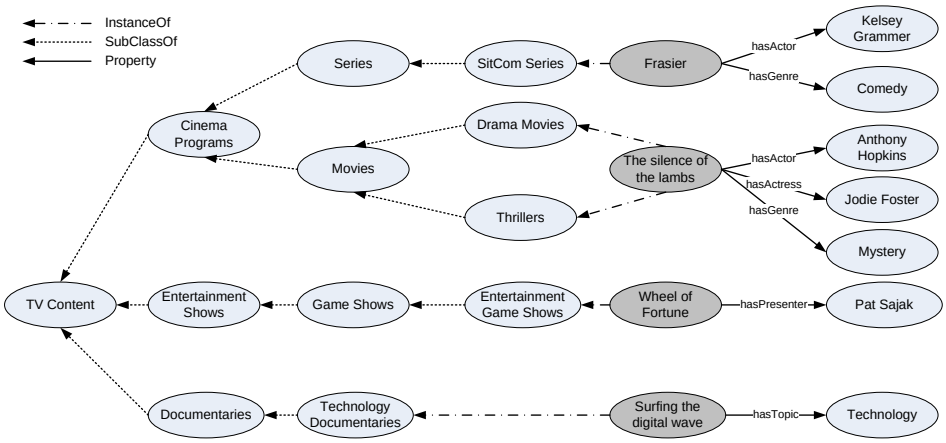

Abbildung 5.6.: Beispiel für ein Nutzerprofil nach [19]

Das AVATAR-System ist speziell für die Verwendung mit Fernsehinhalten konzipiert worden und nicht auf andere Anwendungsbereiche anwendbar. Konfigurationsschnittstellen sind nicht vorhanden. Auf dem TV-Anytime-Standard werden sämtliche für die Berechnung benötigten Ontologien aufgebaut, andere Metadatenformate sind nicht vorgesehen. Unterschiedliche Endgeräte wie mobile Geräte werden nicht unterstützt.

AVATAR erzeugt ausschließlich proaktiv in einem zweistufigen Ansatz gute Empfehlungen unter Berücksichtigung von Kreuzkorrelationen der Attribute, jedoch nur für unbekannte Objekte, bekannte Objekte oder Kommentare sowie jegliche Kontextinformationen werden nicht berücksichtigt. Obwohl AVATAR zwei unterschiedliche Empfehlungsverfahren einsetzt, können diese nicht unabhängig voneinander verwendet werden, sondern sind fester Bestandteil der Empfehlungsgenerierung. Andere Verfahren können nicht genutzt werden. Systemfeedback, das dem Nutzer mitteilt, warum ein Inhalt gewählt wurde, wird ebenfalls nicht zur Verfügung gestellt.

Bei AVATAR wird ein kollaboratives Filterverfahren eingesetzt, das auf mehreren, zentral gespeicherten Nutzerprofilen arbeitet, was im Widerspruch zum Schutz der Privatsphäre steht. Auf Grund der zentralen Berechnung der Empfehlung und dem Einsatz von Ontologien für Inhaltsbeschreibungen und Nutzerprofile, die schnell sehr umfangreich werden und dadurch schwer zu verwalten sind, ist das System nur bedingt effizient und skalierbar.

5.3.5. SmartRotuaari

Ojala, Korhonen et al. stellen in [130] das SmartRotuaari Service System vor, das zur einfachen Bereitstellung von mobilen, Kontext-sensitiven Multimedia-Diensten dient. Die Anwendungsbereiche umfassen personalisiertes, mobiles Marketing sowie Informations-,

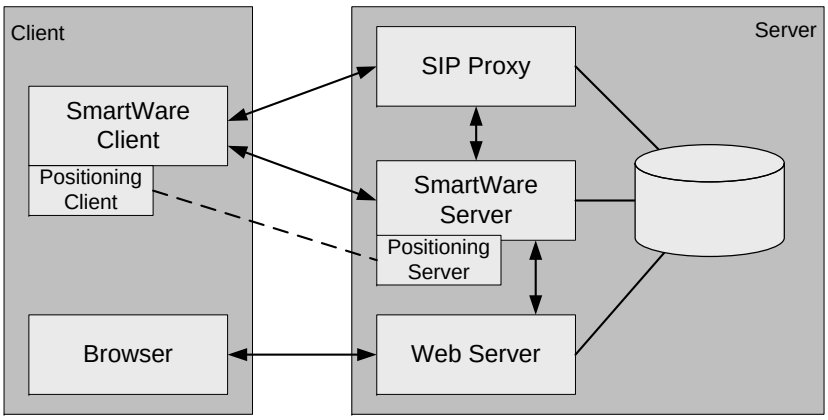

Abbildung 5.7.: Überblick über die SmartWare Architketur nach [130]

Kommunikations- und Bezahldienste.

SmartRotuaari beinhaltet drei wesentliche technische Komponenten. Zum einen arbeitet es basierend auf einem drahtlosen Netz, das in der Innenstadt von Oulu, Finnland installiert wurde und sich aus dem Rotuaari WLAN, einem GPRS-Netz (seit 2003 auch EDGE), das für die schnelle Umsetzung und den Test von mobilen Anwendungen in Oulu aufgebaut wurde, und Bluetooth an einigen ausgewählten Orten zusammensetzt. Für die Bereitstellung der mobilen Dienste wurde die SmartWare Architektur entwickelt (Abbildung 5.7). Der *SmartWare Client* wurde speziell für die Verwendung auf PDAs implementiert und wird mittels Stift bedient. Um vom Client auf die multimedialen Inhalte der Dienste zugreifen zu können, bietet der *SmartWare Server* geeignete Schnittstellen an. Außerdem kontrolliert er das IP Gateway, mit dem Nutzer auf Webinhalte aus dem öffentlichen Internet zugreifen können, und den SIP Server, der für Instant-Messaging-Funktionen verwendet wird. Mittels Benachrichtigungen werden die für den Dienst und den Nutzer relevanten Informationen übertragen. Wenn z. B. eine neue mobile Werbung in das System eingebracht wird, werden alle Nutzer, die diese Werbung erhalten sollen, benachrichtigt. Als Kontextinformation werden der aktuelle Aufenthaltsort, die Zeit, das Wetter, allgemeine demographische Nutzerdaten, die Stimmung des Nutzers und sein aktueller Status verwendet. Der Nutzer wird mit Unterstützung des *Positioning Client*s über WLAN durch den *Positioning Server* lokalisiert. Alle für das System relevanten Daten wie Nutzerprofile, Werbungen und Nutzerlogs werden Server-seitig in einer Datenbank abgelegt. Der *Web Server* stellt Schnittstellen zur Verfügung, die den Nutzern die Verwaltung ihrer persönlichen Profile mittels Browser und den Inhalteanbietern den Zugriff auf das System erlauben.

SmartRotuaari wurde als flexibles System zur Kontext-abhängigen Bereitstellung von Multimedia-Diensten konzipiert, aber nicht zur eigentlichen Empfehlung von multimedialen Inhalten. Deswegen erfüllt es die für Empfehlungssysteme relevanten Anforderungen nur teilweise. Es werden keine unterschiedlichen Empfehlungsverfahren unterstützt, Kommentare oder Kreuzkorrelationen der Attribute bleiben unberücksichtigt und es wird kein Feedback gegeben. Auf Effizienz wird im System nicht geachtet. Bei der Kontext-sensitiven Bereitstellung von Werbungen wird versucht, passende Werbungen auszuwählen, die u. U. dem Nutzer bereits bekannt sein können. Die reaktive oder proaktive Auswahl basiert auf einer relativ geringen Menge an Nutzerinformationen und ist nicht auf einen einzelnen Nutzer zugeschnitten. Metadatenmodelle zur Beschreibung der Inhalte werden nicht verwendet.

Eine wichtige Rolle bei der Dienstbereitstellung in SmartRotuaari spielt der Kontext des Nutzers. Es werden umfangreiche Informationen wie der aktuelle Ort, die Zeit und die Wetterverhältnisse berücksichtigt. Da die Profilinformationen Server-seitig abgespeichert werden, ist kein ausreichender Schutz der Privatsphäre vorhanden.

Das System ist Anwendungsunabhängig entwickelt worden. Unterschiedliche Dienste können integriert werden, die Dienste an sich sind nicht konfiguriert. SmartRotuaari wurde für ein spezielles Testszenario in Oulu umgesetzt, wobei auf die technische Machbarkeit, nicht jedoch auf Skalierbarkeit oder effiziente Nutzung der Luftschnittstelle geachtet wurde. Es können nur J2ME-fähige Endgeräte als Client eingesetzt werden.

5.3.6. Context-Aware Collaborative Filtering System

Annie Chen stellt in [28] ein System vor, das den klassischen kollaborativen Filteransatz (vgl. Abschnitt 5.1.3) um Kontextinformationen erweitert. Das System berücksichtigt bei der Empfehlungsgenerierung, wie sich dem aktiven Nutzer ähnliche Nutzer in einem ähnlichen Kontext verhalten haben. Darauf basierend werden mögliche Aktivitäten vorgeschlagen.

In einem normalen kollaborativen System bestehen Nutzerprofile aus einer Menge von Einträgen (Items), die von den Nutzern bewertet wurden. Ein Item kann ein Produkt, ein Ort oder auch eine gewisse Aktion sein und die Bewertung repräsentiert die Nutzerpräferenz bezüglich des Items. In einer dynamischen Umgebung, wie sie im Rahmen des vorgeschlagenen Systems betrachtet wird, kann sich die Nutzerpräferenz je nach Kontext verändern. Von daher muss diese Kontextinformation zusätzlich im Profil abgespeichert werden. In Abhängigkeit der vorhandenen Infrastruktur können unterschiedliche Kontextinformationen zur Verfügung stehen, die unabhängig voneinander verwaltet werden und deren kombinierter Einfluss berechnet werden muss.

Um eine Empfehlung zu erzeugen, müssen die verfügbaren Kontextinformationen mit den in den Profilen anderer Nutzer abgespeicherten verglichen werden. Dafür muss für jeden Typ von Kontextinformation eine quantifizierbare Messgröße verfügbar sein, die die Ähnlichkeit der Kontextinformationen bestimmen kann. Man kann für Kontextin-

formationen, die kategorisierbar, kontinuierlich oder hierarchisch sind, eine allgemeine Vergleichsfunktion aufstellen, für alle andern Typen muss eine entsprechende Funktion definiert werden.

Chen trifft zusätzlich die Annahme, dass, wenn die Nutzerpräferenz bezüglich eines Items sich in unterschiedlichen Kontexten kaum unterscheidet, die Bewertungen, die in einem Kontext abgegeben wurden, auch für den anderen Kontext gültig sind. Das bedeutet also, dass bei zwei ähnlichen Bewertungen gegeben in zwei unterschiedlichen Kontexten, die Kontexte ähnlich sind. Dies wird mittels der Pearson Korrelation berechnet (vgl. Formel 5.1). Die Bewertung, die ein Nutzer u in Kontext x (bzw. y) vom Typ t für Item i gegeben hat, wird als r_{u,i,x_t} bezeichnet. Dieser Wert kann als Gewicht bei der endgültigen Berechnung des Empfehlungswerts für ein unbekanntes Item verwendet werden. $\bar{r_i}$ steht für die durchschnittliche Bewertung des Items i und σ_{x_t} bzw. σ_{y_t} für die Standardabweichung des Kontexts x bzw. x vom Typ t.

$$rel_t(x, y, i) = \frac{\sum_{u=1}^{n} (r_{u,i,x_t} - \bar{r_i})(r_{u,i,y_t} - \bar{r_i})}{\sigma_{x_t} * \sigma_{y_t}} \tag{5.1}$$

Das hier vorgestellte Verfahren liefert einen sehr allgemeinen Ansatz, um kollaboratives Filtern um Kontextinformationen zu erweitern. Es erfüllt neben der Kontext-Unterstützung dieselben Anforderungen wie klassisches kollaboratives Filtern, also die Generierung passender Empfehlungen, ggf. von bereits bekannten Objekten, Schutz der Privatsphäre wird hingegen nicht realisiert, da der Algorithmus auf mehreren Nutzerprofilen arbeiten muss. Auf eine effiziente Berechnung oder Skalierbarkeit der Empfehlung oder die Berücksichtigung der Kreuzkorrelationen der Attribute wird in dem Ansatz nicht geachtet.

Die Umsetzung der Anforderungen nach Systemfeedback, reaktiven und proaktiven Empfehlungen, die Empfehlung von Kommentaren, eine effiziente Nutzung der Luftschnittstelle, die Unterstützung unterschiedlicher Endgeräte und Metadatenformate sowie die Möglichkeit zur Konfiguration ist stark abhängig von der Anwendung, in die dieser Ansatz integriert wird. Die unterschiedlichsten Arten von Anwendungen sind vorstellbar. Eine Kombination mit weiteren Filterverfahren ist möglich, jedoch nicht explizit vorgesehen.

5.3.7. Matching User's Semantics with Data Semantics in Location-Based Services

Yu et al. stellen in [190] einen Ansatz vor, wie man Ortsbasierte Dienste mit weiterem Kontext veredeln kann. Indem nicht nur die reinen Ortsinformationen berücksichtigt werden, sondern weitere Kontextinformationen wie die Zeit oder das Wetter, Nutzerprofile inklusive demographischer Daten und der Nutzungshistorie und informelle Beschreibungen

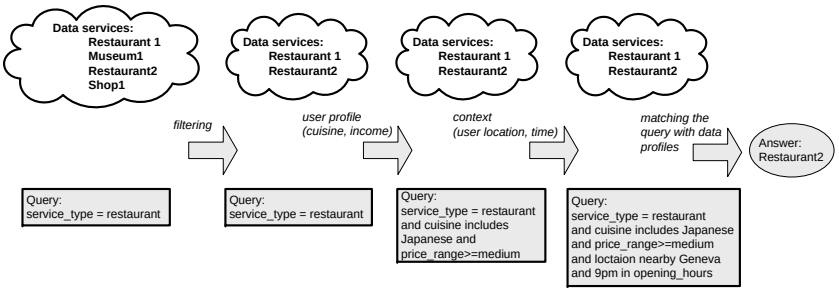

Abbildung 5.8.: Überblick über den Matchingansatz nach [190]

über verfügbare Dienste wie die Öffnungszeiten eines Restaurants, kann man wesentlich gezielter Informationsdienste auf einen Nutzer ausrichten. Hierfür wird ein semantischer Matching-Ansatz vorgeschlagen (Abbildung 5.8). Bei einer eingehenden Anfrage werden in Abhängigkeit der momentan verfügbaren Datendienste diejenigen Dienste herausgefiltert, die nicht dem gewünschten Diensttyp entsprechen. So werden z. B. bei der Suche nach Restaurants Museen nicht berücksichtigt. Beim reinen Abgleich von Schlüsselwörtern werden einige Dienste fälschlicherweise nicht berücksichtigt wie Cafés bei der Suche nach Restaurants. Dieses Problem wird mittels Wörterbüchern oder Ontologien gelöst.

In einem zweiten Schritt wird die gestellte Anfrage durch das Nutzerprofil verfeinert. Dafür müssen die Eigenschaften bestimmt werden, die berücksichtigt werden sollen. Dafür werden die Eigenschaften verwendet, die eine Entsprechung in den informellen Beschreibungen der Dienste finden (z. B. das Einkommen des Nutzers und die Preiskategorie des Restaurants).

In einem dritten Schritt werden die aktuellen Kontextinformationen verwendet, um die Anfrage weiter zu verfeinern z. B. die Öffnungszeiten. Hier muss wieder entschieden werden, welche Kontextinformationen verwendet werden. Abschließend wird die Anfrage mit den Diensten in der verbleibenden Antwortmenge verglichen und dem Nutzer eine Antwort bzw. Antwortliste präsentiert.

Das vorgestellte System erzeugt passende Empfehlungen für Datenprofile von unbekannten oder bekannten Datendiensten wie einem Restaurant unter Berücksichtigung von umfangreichen Kontextinformationen. Kommentare werden nicht betrachtet, der Nutzer erhält kein Feedback, warum ein Dienst gewählt wurde. Empfehlungen werden basierend auf Ontologien erzeugt, wobei keine Kreuzkorrelationen der Attribute berücksichtigt werden. Ontologien bieten zwar die Möglichkeit des semantischen Reasonings, unter deren Größe leiden jedoch die Effizienz der Berechnung und die Skalierbarkeit des Systems. Unterschiedliche Filterverfahren werden nicht unterstützt. Der Ansatz arbeitet rein reaktiv, eine proaktive Nutzung ist nicht vorgesehen.

Bei jeder Nutzeranfrage wird eine Teilmenge des Profils vom Client an den Server übertragen, um dort mit den Datenbeschreibungen abgeglichen werden zu können. Dadurch wird die Privatsphäre des Nutzers zumindest teilweise geschützt, da die Nutzerprofile nicht zentral verwaltet werden. Dies wirkt sich auch positiv auf die Skalierbarkeit aus. Außerdem entfällt dadurch die permanente Aktualisierung von hochdynamischen Kontextinformationen auf dem Server, die die Luftschnittstelle stark belastet. Zur Beschreibung der Nutzer und der Datendienste wird ein proprietäres Format vorgestellt, andere Metadatenbeschreibungen können nicht verwendet werden.

Das System wurde stark auf Ortsbasierte Dienste ausgelegt und erlaubt keine Anpassung an beliebige Anwendungsfälle. Eine Konfigurationsschnittstelle wird ebenfalls nicht angeboten. Dafür kann es auf jedem LBS-fähigen Endgerät eingesetzt werden.

5.3.8. Context-aware multimedia-based recommendation system

Kwon verfolgt in [102] den Ansatz, die Gedanken des Nutzers beim Navigieren durch Hypermedia-Seiten zu erraten und ihm geeignete Kaufempfehlungen zu präsentieren. Diese Funktionalität wird durch ein Agenten-basiertes System umgesetzt, das die Semantik von Hypermedia-Inhalten erkennen, daraus kontextuelle Information ableitet und voraussagen kann, was die Kunden benötigen.

Der genaue Ablauf ist in Abbildung 5.9 zu sehen. Ein Nutzer-Agent überwacht das Surfverhalten des Nutzers. Betrachtet der Nutzer einen Hypermedia-Inhalt, werden dessen URL und die dem Inhalt zugeordneten Daten an den Agenten übertragen. Unter Verwendung der URL werden öffentliche Ontologien durchsucht, um zu identifizieren, was die Bedeutung des Inhalts ist und aus welchen Komponenten er sich zusammensetzt. Neben Ontologien werden für diesen Schritt spezialisierte Datenbanken verwendet, die eine Menge von bereits identifizierten Inhalten besitzen, die mit dem unbekannten Inhalt verglichen werden. Aus diesen Informationen wird ein Attributed Relational Graph erzeugt, wobei die identifizierten Objekte des Inhalts durch Knoten und deren Beziehung durch Kanten repräsentiert werden. Dadurch kann der Agent verstehen, was die Semantik des Inhalts ist, der vom Nutzer gerade betrachtet wird.

Im nächsten Schritt wird die persönliche Ontologie des Nutzers mit einbezogen, die das Nutzerprofil, Präferenzen und weitere persönliche Nutzerinformationen wie die persönliche Einkaufsliste beinhaltet. Diese Informationen sind im Netz verfügbar und werden vom *e–Wallet* System verwaltet. Basierend auf den identifizierten Objekten im Inhalt und den Informationen über den Nutzer, können die Objekte identifiziert werden, an denen der Nutzer interessiert ist und die er evtl. kaufen möchte. Abschließend werden in einem Verzeichnis von Web-Diensten geeignete ausgewählt und dem Nutzer empfohlen.

Das System erzeugt passende Empfehlungen von bekannten und unbekannten Diensten und Inhalten, jedoch nicht von Kommentaren. Einige Kontextinformationen wie z. B.

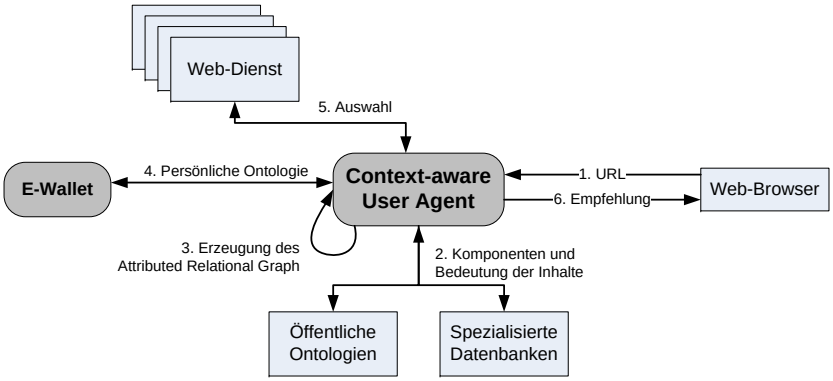

Abbildung 5.9.: Ablauf des Hypermedia-basierten, Kontext-abhängigen Empfehlungssystems nach [102]

die Nutzerpräferenzen werden berücksichtigt, nicht jedoch dynamische wie der Ort. Das liegt u. a. daran, dass das System für die Verwendung im Web konzipiert wurde. Ebenso arbeitet es rein proaktiv, Nutzeranfragen sind nicht vorgesehen. Dem Nutzer wird bei der Empfehlung kein Feedback gegeben, warum ein Dienst oder Inhalt ausgewählt wurde.

Nutzer werden mittels Ontologien und Inhalte mittels Graphen und Ontologien beschrieben, andere Formate werden nicht unterstützt. Es kann nur ein vorgegebenes Verfahren zur Erzeugung der Empfehlung eingesetzt werden, das keine Kreuzkorrelationen der Attribute berücksichtigt. Auf Grund der Komplexität der betrachteten Ontologien und Graphen kann die Berechnung der Empfehlung nicht effizient und auch nur bedingt skalierbar durchgeführt werden. Das Profil wird zentral gespeichert, jedoch sind Schutzmechanismen vorhanden, die bestimmen, wer darauf Zugriff hat. Ein optimaler Schutz der Privatsphäre wird dadurch aber nicht erzielt.

Der Ansatz ist für einen sehr eingeschränkten Anwendungsfall entwickelt worden und kann nicht anderweitig eingesetzt werden. Konfigurationsschnittstellen sind nicht vorhanden. Da das System Web-basiert arbeitet, ist es nur für Verwendung mittels Browser gedacht. Aufgrund der Inhalte sind nur Browser stationärer Geräte, aber keine mobilen Browser vorgesehen.

5.3.9. Mobile Tourist Application COMPASS

Van Setten, Pokraev et al. entwickelten die COMPASS (COntext-aware Mobile Personal ASSistant) Plattform zur Realisierung von Kontext-abhängigen Touristenführern [170, 138]. Ursprünglich konzipiert um automatisch angebotene Dienste an die Nutzerinteressen und an Kontextinformationen anzupassen, wurde sie um eine Komponente für Kontext-abhängige Empfehlungen erweitert. COMPASS ist ein offenes System, dass Drittanbietern erlaubt, ihre Dienste in die Plattform zu integrieren.

Die COMPASS-Architektur ist in Abbildung 5.10 zu sehen. Für die Ausführung der unterschiedlichen Dienste werden Dienste von Drittanbietern benötigt. Dazu gehören die 3G-Netzdienste, die alle für den Zugriff auf ein Netz notwendigen Funktionen (z. B. Nutzeridentifikation, Abrechnung oder Verbindungsaufbau) realisieren. Der Kontext des Nutzers wird mit Hilfe von Kontextdiensten identifiziert. Ein Teil dieser Informationen kann über Webservices direkt vom 3G-Netzdienst bezogen werden. Andere Dienste stellen vom Nutzer unabhängige Informationen zur Verfügung wie die aktuellen Wetterdaten oder die Zeit. Für die Umsetzung von Anwendungen auf der COMPASS-Plattform werden über Business Services grundlegende Dienste und Informationen zur Verfügung gestellt. Beispiele hierfür sind ein Dienst zur Verwaltung von Points-of-Interests (POIs), Listen von Restaurants oder ein Hotelreservierungsdient.

Die *WASP Plattform* ist für die Veredelung der zugrundeliegenden Dienste und deren Kommunikation mit der COMPASS-Plattform zuständig. Über einen Request Dispatcher werden die Nutzeranfragen zu den 3G-Netzen weitergeleitet, was dem Nutzer einen transparenten Zugriff auf unterschiedliche Anbieter ermöglicht. Änderungen werden vom Context Manager verwaltet, der auf die zugrundeliegenden Context Services zugreift, die Daten aggregiert oder höherwertige Kontextinformationen ableitet. Über den Notification Manager können sich Anwendungen für Benachrichtigungen bei Änderungen der Kontextinformationen registrieren. Alle Dienste von Drittanbietern müssen sich mittels detaillierter Beschreibungen ihrer Dienste bei der Service Registry anmelden. Diese arbeitet OWL-basiert und erlaubt dadurch eine erweiterte Suchfunktionalität unter Berücksichtigung von Beziehungen oder semantischen Informationen. Über den Matchmaker werden die von den Applikationen benötigten Dienste ausgewählt. Dabei können harte und weiche Kriterien von den Anwendungen definiert werden.

Die eigentliche *COMPASS-Plattform* besteht aus den beiden Komponenten Interaction Manager und POI Retriever. Über den Interaction Manager wird entschieden, wie das System bei einer Nutzeranfrage reagieren soll. Wenn sich der Nutzerkontext ändert, wird vom POI Retriever eine Suchanfrage an den Matchmaker der WASP Plattform gesendet, der mit einer neuen Liste von POIs antwortet, die den harten Kriterien entsprechen müssen. Diese Liste wird an die auf dem Client laufende Applikation gesendet.

Der *Recommendation Service* ist für die Realisierung der Kontext-abhängigen Empfehlungen von POIs zuständig. In der Recommendation Engine wird entschieden, welche der verfügbaren Filterverfahren bzw. welche Kombinationen für die gewünschte Anfrage, den entsprechenden Nutzer und den Kontext geeignet sind. Die POIs sind über eine Ontologie beschrieben. Die Recommendation Engine kennt die Hierarchie der POIs. Gibt es eine geeignete Filter-Strategie für eine bestimmte Klasse von POIs wird diese verwendet, andernfalls solange in der Hierarchie nach oben gesucht, bis eine gefunden wurde. Dem Wurzelknoten ist eine Default-Methode zugeordnet. Die Filterverfahren verwenden Profilinformationen die vom User Profiler verwaltet werden.

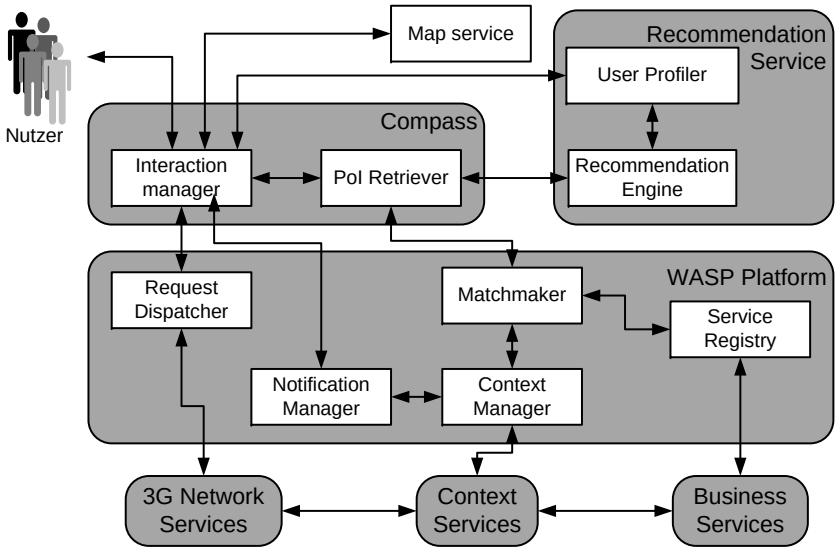

Abbildung 5.10.: COMPASS Plattform Systemarchitektur nach [170]

COMPASS liefert eine dem eigenen Ansatz recht nahe Methode zur Kontext-abhängigen Empfehlung. Allerdings handelt es sich bei den empfohlenen Objekten um bekannte und unbekannte POIs, multimediale Inhalte oder Nutzerkommentare werden nicht berücksichtigt. Auf Grund des Interaction Managers ist eine proaktive und reaktive Nutzung des System möglich. Bei der Auswahl der POIs werden unterschiedliche Filterverfahren eingesetzt. Gründe für die Auswahl werden dem Nutzer nicht mitgeteilt. Bei der Auswahl der Filterverfahren und POIs werden umfangreiche Ontologien eingesetzt, was zwar die Empfehlungsqualität erhöht, weswegen aber auch die Effizienz der Berechnung leidet. Kreuzkorrelationen der Attribute werden nicht unterstützt.

Das System wurde Anwendungsunabhängig als offene Plattform konzipiert und erlaubt die Einbindung externer Dienste, nicht jedoch die Konfiguration. Innerhalb des Systems werden Dienste und Nutzer beschrieben. Die Dienstbeschreibungen sind Ontologiebasiert, für die Nutzer wird ein internes Format verwendet. Andere Beschreibungen werden nicht unterstützt. Das Nutzerprofil wird zentral gespeichert, was zwar eine umfangreiche Übertragung von Profildaten über die Luftschnittstelle verhindert und besser skalierbar ist, sich aber negativ auf den Schutz der Privatsphäre auswirkt. Die Anwendung ist für das Sony Ericson P800 und ein externes GPS-Modul entwickelt worden, jedoch nicht auf anderen Endgeräten einsetzbar. Eine breite Unterstützung anderer Endgeräte ist bei weiteren Entwicklungen vorgesehen.

5.3.10. GUIDE

Das von Cheverst, Mitchell, Daviesan et al. der Universität von Lancaster entwickelte GUIDE-System soll den Nutzer beim Erkunden von unbekannten Städten unterstützen [33, 32, 31]. Dafür stellt es anhand der aktuellen Position und Präferenzen geeignete Informationen zur Verfügung. Es erlaubt eine Kontext-sensitive Informationssuche, Karten-basierte Navigation und die Erstellung von Touren unter Berücksichtigung der Entfernung, Öffnungszeiten und den besten Besuchszeiten. Die Tour passt sich automatisch an das Verhalten und auch das Tempo des Nutzers an.

Die GUIDE Architektur beinhaltet mehrere verteilte Basisstationen, die an für Touristen relevanten Stellen positioniert sind und die Inhalte auf Nutzeranfragen hin ausliefern, sowie mobile Clients, die die GUIDE-Anwendung enthalten, in die ein Web-Browser integriert ist. Über ein gesendetes Signal können die mobilen GUIDE-Einheiten die Basisstationen erkennen. Wird ein solches Signal bemerkt, wird ein auf der mobilen GUIDE-Einheit laufender *Client Agent* davon in Kenntnis gesetzt.

Alle wichtigen Informationen wie Kontextinformationen, geographische Daten oder die eigentlichen Inhalte werden mit einem Informationsmodell beschrieben und lokal gespeichert. So entsteht ein Modell der Stadt aus den einzelnen Objekten. Bei einer neu erkannten Basisstation sucht der Agent nach zugehörigen Objekten, die momentan lokal gespeichert sind, und erzeugt (wenn noch nicht vorhanden) ein neues Ortsobjekt für den aktuellen Ort. Basierend auf diesen Informationen können Methoden aufgerufen werden, die sich auf den aktuellen Aufenthaltsort beziehen, wie das Abfragen aller interessanten Attraktionen in der Nähe. Über den lokalen Browser können Nutzeranfragen eingetragen werden, die von einem lokalen *Proxy* bearbeitet werden, der lokale und entfernte Inhalte beziehen kann. Über eine *Filter*-Komponente werden diese Inhalte gemäß den Nutzerprofilen angepasst.

GUIDE ist ein auf den speziellen Anwendungsfall des Kontext-abhängigen Touristenführers ausgelegtes Informationssystem und bietet keinerlei Möglichkeiten der Anpassung. Es arbeitet mit speziell dafür zur Verfügung gestellten Endgeräten. Als geschlossenes System werden nur die internen Formate unterstützt. Richtige Empfehlungen werden durch das System nicht erzeugt, sondern nur basierend auf Kontextinformationen, insbesondere dem Ort, reaktiv relevante Informationen bereitgestellt. Eine Kommentierung dieser Informationen ist nicht vorgesehen, Korrelationen nicht berücksichtigt und es wird nicht unterschieden, ob dem Nutzer diese Informationen schon bekannt oder noch unbekannt sind.

Die Nutzerinformationen befinden sich ausschließlich auf dem Client, was als ein guter Schutz der Privatsphäre des Nutzers dient. Die wichtigen Informationen werden gebroadcastet und vom Endgerät gemäß dem Profil selektiert. Da es keine individuelle Verbindung des Endgeräts zum Server gibt, ist eine gute Nutzung der Luftschnittstel-

le gewährleistet. Innerhalb des Empfangsradius der GUIDE-Basisstationen können sich beliebig viele Endegeräte aufhalten, eine Erweiterung des Systems auf andere Gebiete benötigt hingegen einen umfangreichen Ausbau der Basisstationen.

5.3.11. Fazit

Vergleicht man die Anforderungen an ein Empfehlungssystem, das eine personalisierte Auswahl multimedialer Inhalte unter Berücksichtigung von Kontextinformationen unterstützt, aus Abschnitt 5.2 mit den hier vorgestellten Ansätzen, erkennt man, dass sie bei keinem ausreichend erfüllt werden. Tabellen 5.1 und 5.2 fasst die Beurteilungen zusammen, wobei ein + anzeigt, dass eine Anforderung erfüllt wird, ein –, dass sie nicht erfüllt wird, und ein o bedeutet, dass eine Anforderung nur teilweise oder nur mit umfangreichen Anpassungen des jeweiligen Ansatzes erfüllt wird.

Fast alle Ansätze erfüllen die Forderung nach passenden Empfehlungen von unbekannten, teilweise auch von bekannten Objekten. SmartRotuaari und GUIDE sind nicht als eigentliches Empfehlungssystem entwickelt worden und wurden auf Grund anderer interessanter Aspekte in dieser Arbeit erwähnt. Meist wurde bei der Entwicklung der Empfehlungs-Algorithmen auch auf Effizienz geachtet. In den Fällen, wo die Beurteilung dieses Punkts negativ ausfiel, lag das am Einsatz von Ontologien wie bei AVATAR und den Ansätzen aus Abschnitt 5.3.7 und 5.3.8. Ontologien bilden ein umfangreiches Werkzeug zum Abgleich von Informationen. Dies kann aber mit deren Anwachsen nicht mehr effizient durchgeführt werden. Erstaunlich hingegen ist, dass bis auf das TV-Trawler-Projekt, keines der untersuchten Systeme dem Nutzer mitteilt, warum ein Objekt oder Inhalt für ihn ausgewählt wurde.

Die meisten vorgestellten Systeme unterstützen zumindest rudimentär Kontextinformationen, was u. a. daran liegt, dass sie nach diesem Kriterium ausgewählt wurden. Einigen Systemen wie den in den Abschnitten 5.3.3 und 5.3.7 vorgestellten oder dem GUIDE-System, fehlen ein proaktives Element. Bei einer proaktiven Empfehlung muss der Kontext geeignet überwacht werden. Eine reaktive Umsetzung ist meistens bedeutend einfacher und wird deswegen breitflächiger unterstützt. AVATAR und der Ansatz aus Abschnitt 5.3.8 hingegen unterstützen ausschließlich proaktive Empfehlungen, explizite Nutzeranfragen sind hier nicht vorgesehen. Allein AVATAR berücksichtigt Kreuzkorrelationen der Attribute bei der Erzeugung von Empfehlungen.

Ein ausreichender Schutz der Privatsphäre ist eng gekoppelt mit dem Speicherort des Nutzerprofils. Wird dieses zentral also Server-seitig abgelegt wie bei AVATAR, P-News oder dem in Abschnitt 5.3.6 beschriebenen Ansatz, führt dies zu einer negativen Beurteilung, da der Nutzer nicht mehr so viel Kontrolle über seine Daten hat. Dafür können auf diese Weise kollaborative Ansätze eingesetzt werden. Auf diese Weise wird eine Übertragung von umfangreichen, für den Nutzer nicht relevanten Daten vermieden, da diese bereits auf dem Server gefiltert werden. Dies führt zu einer geringeren Belastung der

Anforderung	P-NEWS	TV-Trawler	5.3.3	Avatar	SmartRotuaari
1. Generierung passender Empfehlungen	+	+	+	+	o
2. Empfehlung bekannter und	–	o	+	–	o
unbekannter Objekte					
3. Empfehlung von Kommentaren	–	–	–	–	–
4. Berücksichtigung von	o	–	o	–	+
Kontextinformationen					
5. Berücksichtigung von Kreuzkorrelationen	–	–	+	–	
der Attribute					
6. Reaktive und proaktive Empfehlungen	+	+	–	–	+
7. Systemfeedback	–	+	–	–	–
8. Anwendungsunabhängigkeit	–	–	+	–	+
9. Unterstützung unterschiedlicher	–	–	–	o	–
Empfehlungsverfahren					
10. Unterstützung unterschiedlicher	+	–	+	–	–
Endgeräte					
11. Unterstützung unterschiedlicher	–	o	–	–	–
Metadatenformate					
12. Konfigurationsschnittstelle	o	–	–	–	o
13. Schutz der Privatsphäre des Nutzers	–	–	o	–	–
14. Geringe Belastung der Luftschnittstelle	+	n/a	+	n/a	–
15. Skalierbarkeit	o	o	–	–	–
16. Effiziente Berechnung	+	+	o	+	–
der Empfehlungen					

Tabelle 5.1.: Vergleich verwandter Arbeiten mit Anforderungen

Anforderung	5.3.6	5.3.7	5.3.8	COMPASS	GUIDE
1. Generierung passender Empfehlungen	+	+	+	+	o
2. Empfehlung bekannter und unbekannter Objekte	+	+	+	+	o
3. Empfehlung von Kommentaren	–	–	–	–	–
4. Berücksichtigung von Kontextinformationen	+	+	o	+	+
5. Berücksichtigung von Kreuzkorrelationen der Attribute	–	–	–	–	–
6. Reaktive und proaktive Empfehlungen	–	–	–	+	–
7. Systemfeedback	–	–	–	–	–
8. Anwendungsunabhänggkeit	+	–	–	+	–
9. Unterstützung unterschiedlicher Empfehlungsverfahren	–	–	–	+	–
10. Unterstützung unterschiedlicher Endgeräte	–	o	–	–	–
11. Unterstützung unterschiedlicher Metadatenformate	–	–	–	–	–
12. Konfigurationsschnittstelle	–	–	–	–	–
13. Schutz der Privatsphäre des Nutzers	–	o	o	–	+
14. Geringe Belastung der Luftschnittstelle	–	+	n/a	+	+
15. Skalierbarkeit	–	o	–	+	o
16. Effiziente Berechnung der Empfehlungen	–	–	–	o	o

Tabelle 5.2.: Vergleich verwandter Arbeiten mit Anforderungen (fortgesetzt)

Luftschnittstelle. Die Ausnahme bildet hier P-News, das explizit Mechanismen zur Minimierung des Datenverkehrs auf der Luftschnittstelle bereitstellt. Nachteil einer Serverseitigen Speicherung der Profile ist hingegen die geringere Skalierbarkeit, da die Profile zentral an nur einer Stelle gehalten werden.

COMPASS, SmartRotuaari und die Ansätze aus Abschnitt 5.3.3 und 5.3.6 wurden anwendungsunabhängig spezifiziert, die anderen Ansätze für einen speziellen Anwendungsfall. Dadurch werden bei diesen die Anforderungen der Unterstützung unterschiedlicher Empfehlungsverfahren und unterschiedlicher Metadatenformate, sowie einer Konfigurationsschnittstelle nur teilweise oder gar nicht erfüllt. Es wurden proprietäre Verfahren bzw. Formate entwickelt (vgl. Abschnitt 5.3.7) oder die Systeme wurden nur für spezielle Metadatenformate konzipiert wie AVATAR für TV-Anytime oder der in Abschnitt 5.3.3 beschriebene Ansatz für MPEG-7/21. Viele der Systeme wurden nur für eine bestimmte Plattform (z. B. SmartRotuaari) z. T. sogar nur für ein bestimmtes Endgerät (z. B. GUIDE oder COMPASS) entwickelt.

Kommentare werden bei keinem der hier vorgestellten Systeme unterstützt, da die untersuchten Systeme keinerlei Interaktion zwischen den Nutzern vorsehen.

5.4. Vorüberlegungen zur Kontext-abhängigen Auswahl

Im folgenden Abschnitt werden einige grundsätzliche Gedanken über die auf MASL basierende Kontext-abhängige Auswahl multimedialer Inhalte diskutiert. Die in MASL vorhandenen Attributtypen werden klassifiziert und eine Auswahl möglicher und bei der Empfehlung zu berücksichtigender Attribute gezeigt. Es wird diskutiert, wie man basierend auf den Beschreibungen die Inhalte klassifizieren und die Attribute darauf basierend unterschiedlich gewichten kann.

5.4.1. Auswahl geeigneter Attributtypen

Da MASL auf dem MPEG-7/21-Standard beruht, liefert es eine extrem umfangreiche Menge an Attributtypen. Eine Berücksichtigung aller vorhandenen Attributtypen in Attributs-basierten Empfehlungsprozessen wie inhaltsbasiertem Filtern führt jedoch zu einer Verschlechterung der Berechnungszeit des prognostizierten Empfehlungswerts eines Inhalts, weil eine große Anzahl an Attributwerten berücksichtigt werden muss. Außerdem sind nicht alle Attributtypen für jeden Anwendungsfall oder jeden Medientyp gleich relevant. So ist z. B. die Sprache bei einem Text oder einem Film deutlich wichtiger als bei einem Musikstück, da bei Musik nur ein Teil der Information über den Text transportiert wird und ein Nutzer u. U. auch Musik in einer Sprache hören will, die er nicht versteht.

In Kapitel 3 wurde bereits eine Einteilung von Attributtypen in u. a. semantische Beschreibungen, informelle Beschreibungen und Beschreibungen über die Erzeugung vorgestellt. Diese ist zwar für die Auswahl der Attributtypen interessant, die in den An-

wendungsfällen generell benötigt werden, jedoch nicht für die allgemeine Algorithmen-Entwicklung für Empfehlungssysteme.

Zunächst kann man unterscheiden, was der Wertebereich eines bestimmten Attributtyps ist. So gibt es Attributtypen, die *numerische* Werte annehmen können. Die numerischen Werte kann man in `Integer`, also ganzzahlige Werte, und `Real`, also Kommazahlen, einteilen Ein Beispiel für einen Attributtyp mit ganzzahligem Wert in MASL ist die Dateigröße, für einen Attributtyp mit Kommazahl die GPS-Koordinaten des Entstehungsorts. Andere Attribute können als *Vektor* oder *Matrix* modelliert werden. So liegen z. B. unterschiedliche Frequenzbereiche eines Equalizers als Vektor und der Frequenzbereich von Rauschen als Matrix vor. Schließlich können manche Attributtypen *textuelle* Werte annehmen wie das Attribut `Keyword`.

Bei genauer Betrachtung der Attributtypen von MASL erkennt man, dass manche eine beliebige und auch offene Menge an Werten annehmen. Ein Beispiel hierfür ist der Attributtyp `Keyword`, da dieser vom `TextualType` erbt und damit beliebige Texte als Schlüsselwörter angegeben werden können, wie `Rebellion` oder `Jedi Knight` aus dem Beispiel in Listing B.1 aus Anhang B.1. Diese Klasse wird als *nicht-klassifizierbar* bezeichnet. Andere Typen erlauben nur eine eingeschränkte Menge an Werten, vorgegeben durch die Schema-Definition. In MASL werden diese Werte über eine `enumeration` oder ein CS angegeben. Dabei ist zu unterscheiden, ob diese Werte hierarchisch angeordnet werden können wie beim `Genre CS` (`Action` ist z. B. ein Sub-Genre von `Drama`), oder nicht, wie bei der Altersfreigabe im `MPAAParentalRatingCS`, das nur die Werte `TVY`, `TVY7`, `TVG`, `TVPG`, `TV14`, `TVMA` und `None` erlaubt. Die erste Klasse wird als *hierarchisch klassifizierbar* bezeichnet, die zweite als *klassifizierbar, aber nicht-hierarchisch klassifizierbar*.

Eine weitere Einteilung der Attributtypen ergibt sich aus der Anzahl der Werte, die einem Inhalt tatsächlich zugewiesen werden. So können manche nur einen einzelnen Wert beinhalten wie z. B. den Erzeugungsort eines Inhalts (`CreationCoordinates`), andere Attributtypen erlauben die Zuweisung von beliebig vielen Werten zu einem Inhalt wie z. B. das `Genre`-Element. Die erste Klasse wird als *univalent*, die zweite als *multivalent* bezeichnet.

Diese Überlegungen sind in Abbildung 5.11 dargestellt und spielen eine wichtige Rolle bei der Auswahl der Attributtypen für die Empfehlung in einem bestimmten Anwendungsbereich. Prinzipiell sollen möglichst nur (hierarchisch oder nicht hierarchisch) klassifizierbare Attributtypen verwendet werden, da bei nicht-klassifizierbaren Attributtypen beliebig viele Werte im Nutzerprofil oder der Inhaltsbeschreibung vorhanden sein können und es bei einer theoretisch unendlich großen Menge an Werten vorkommen kann, dass die in der Beschreibung des Inhalts vorkommenden Werte nicht im Profil liegen. Es spricht natürlich nichts dagegen, nicht-klassifizierbare Attributtypen vereinzelt in die Empfehlungsberechnung mit einzubeziehen. Handelt es sich bei einem Attributtyp um ein multivalentes Attribut, so wird bei der Empfehlungsgenerierung nicht jeder in der Beschreibung vorkommende Wert einzeln berücksichtigt, sondern über alle vorhandenen Werte des Attributtyps gemittelt.

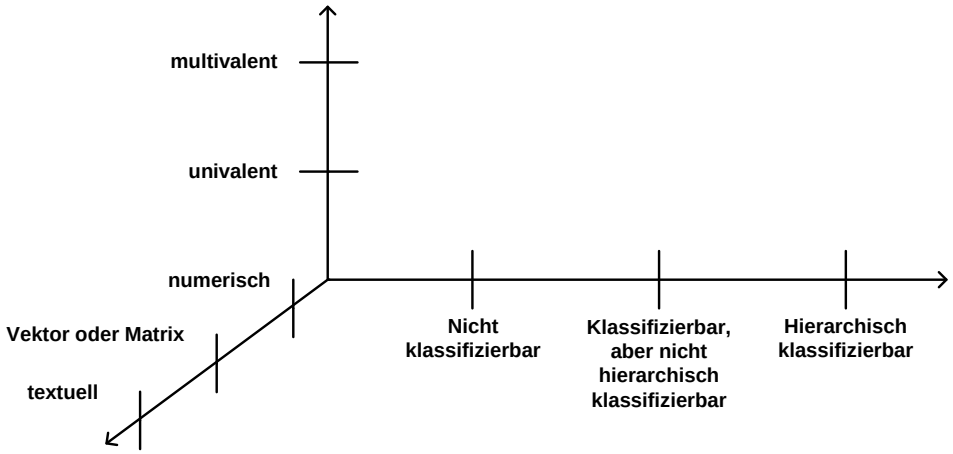

Abbildung 5.11.: Klassifizierung der Attributtypen in MASL

Einige multivalente Attributtypen treten in spezieller Form in den Beschreibungen auf und müssen daher gesondert betrachtet werden. Bei den `Genre`- und `Keyword`-Elementen kann unterschieden werden, ob es sich bei dem angegebenen Wert um einen Primärwert (`main`) oder eine Alternative handelt (`secondary`). Bei diesen Attributtypen kann für die Berechnungen ein größeres Gewicht für die Primärwerte als für die Sekundärwerte verwendet werden.

Beim Attributtyp `Titel` kann zwischen semantisch verschiedenen Titelwerten unterschieden werden. Diese Einteilung umfasst den Haupttitel, alternative Titel, den Originaltitel und weitere Unterteilungen. Da in der Regel der Titel nicht für die Berechnung des Empfehlungswerts für unbekannte Inhalte verwendet wird, sondern zur Identifikation der bekannten Inhalte dient, muss der Attributtyp nicht gesondert betrachtet werden.

Einen besonderen Stellenwert nehmen die Attributtypen `Creator` zur Beschreibung des Erzeugers oder mit der Erzeugung im Zusammenhang stehenden Personen wie Schauspieler, und `Disseminator`, der die Personen oder Organisationen beschreibt, die mit der Verbreitung des Inhalts betraut sind, wie der Filmverleih, ein. Bei diesen unterscheidet man nicht nur die einzelnen Werte der Attribute sondern zusätzlich noch die Rolle, die die entsprechenden Werte einnehmen können. So sind einige Rollen wichtiger als andere, da z. B. die Rolle eines an einem Film beteiligten Schauspielers oder Regisseurs für die Entscheidung eines Nutzers, ob er ihn sehen will, wichtiger ist als z. B. die Rolle des Requisiteurs. Von daher werden einigen Attributwerten des Attributs `Role` ein stärkeres Gewicht zugeordnet als anderen.

Eine sinnvolle Auswahl von Attributtypen aus MASL ist z. B. die folgende, je nach Anwendungsfall kann die Liste natürlich über die Konfigurationsschnittstelle ergänzt bzw. modifiziert werden. In Klammern befinden sich die jeweiligen Elemente aus MASL.

- Eindeutiger Identifikator des Inhalts (`ProgramIdentifier`)
- Genre des Inhalts (`Genre`)
- Sendungsformat (`Format`)
- Mitwirkende inkl. ihrer Rolle, insbesondere Schauspieler, Sprecher, Regisseure etc. (`Creator`)
- Qualität des Inhalts (`MediaQuality`)
- Aktuelle Zeit (`Time`)
- Inhalteanbieter (`Disseminator`)
- Schlüsselwörter, die den Inhalt beschreiben (`Keyword`)
- Sprache des Inhalts (`Language`)
- Beziehung zu anderen Inhalten (z. B. bei Kommentaren) (`RelatedAnnotation`)
- Zielgruppe (`Target`)
- Derzeitige Aktivität des Nutzers (`Context`)
- Aktueller Aufenthaltsort des Nutzers (`Location`)
- Personen in der Nähe des Nutzers (`Context`)

5.4.2. Inhaltsklassen und variable Gewichtungen

Im Empfehlungsprozess leistet jeder Attributtyp je nach Anwendungsfall und Nutzer einen unterschiedlichen Beitrag dazu, ob die Empfehlung passend ist oder nicht. Dies soll durch einen unterschiedlich gewichteten Einfluss des prognostizierten Empfehlungswerts des Attributtyps bei der Berechnung des Empfehlungswerts des Inhalts realisiert werden. Die grundsätzliche Idee, jedem Attributtyp ein anderes Gewicht zuzuordnen, ist nicht neu und wird auch in einigen Systemen praktisch umgesetzt (z. B. in [28], vgl. hierzu auch [156]). In dem hier vorgestellten Ansatz sollen diese Gewichte jedoch in Abhängigkeit der Art des Inhalts variiert werden. Dies bedeutet, dass die Gewichtung des Attributtyps nicht über alle Arten von Inhalten festgesetzt wird, sondern in Abhängigkeit von geeigneten Inhaltsklassen.

Für die Klassifizierung der Inhalte werden die Werte der Attributtypen aus den Beschreibungen verwendet. Dazu müssen geeignete Attributtypen aus dem Sprachschatz von MASL identifiziert werden, die eine solche Klassifizierung ermöglichen. Der Attributtyp `Genre` liefert bereits eine erste Klassifizierung der Inhalte und ist deswegen gut geeignet für diese Zwecke. Er ist stark hierarchisch aufgebaut und wird teilweise bis auf

eine vierte Ebene spezifiziert. Da er im `Genre CS` ca. 340 Subklassen umfasst, ist eine Verwendung des `Genre CS` bis auf das letzte Level nicht praktikabel. Im Folgenden wird ausschließlich die erste Ebene für die Zuteilung in die Inhaltsklassen verwendet, da dies ein guter Kompromiss zwischen feiner Aufteilung und Minimierung der Inhaltsklassen auf eine vernünftige Menge ist. Damit können bzgl. des Attributtyps `Genre` folgende Inhaltsklassen identifiziert werden:

1. `Information`: Aktuelle Informationen zur Wissensbildung oder zur Unterhaltung.
2. `Drama`: Inhalte in Prosa oder Versen, die eine Geschichte erzählen, insbesondere für die Bühne geschrieben.
3. `Entertainment`: Inhalte, die dem Zweck der Unterhaltung dienen.
4. `Music`: Musikalische Inhalte.
5. `Enrichment`: Inhalte, die den Zweck erfüllen, Wissen über nicht-tagesaktuelle Informationen zu schaffen.
6. `Movies`: Spielfilme allgemein.
7. `Animations/special effects`: Inhalte, die animiert oder stark auf Special Effects ausgelegt sind.

In den Beschreibungen kann mehr als ein `Genre` aufgeführt sein, für die Einteilung in Inhaltsklassen wird aber nur das `Genre` mit dem Attribut `main` verwendet.

Ein weiteres Attribut, das sich für die Klassifizierung eignet, ist das `Format`. MASL definiert dafür basierend auf TV-Anytime das `Format CS`, das verwendet wird, um Inhalte bezüglich ihrer formalen Struktur klassifizieren zu können (vgl. Abschnitt 3.4.1). Das Format des Inhalts wird ohne Berücksichtigung des Genres oder der mit dem Inhalt übertragenen Informationen angegeben. Folgende Werte stehen zur Verfügung:

1. `Proprietary`: Nicht standardisierte Inhalte.
2. `Non-fiction`: Inhalte, die sich mit Fakten, Meinungen, Vorhersagen, Nachrichten etc. beschäftigen.
3. `Sports`: Inhalte mit sportlichem Hintergrund.
4. `Fiction/Drama`: Gespielte Inhalte, Geschichten, Erzählungen etc.
5. `Entertainment`: Inhalte, die dem Zweck der leichten Unterhaltung dienen und nicht in die anderen Kategorien fallen.
6. `Music`: Musikalische Inhalte.

7. `Interactive`: Interaktive Inhalte.
8. `Lifestyle`: Inhalte, die einem Hobby, einer Freizeitbeschäftigung etc. zugeordnet werden wie Ratgeber, Kochsendungen etc.
9. `Adult`: Inhalte für Erwachsene.

Wichtig für eine Klassifizierung ist, von welchem Medientyp der Inhalt ist. Diese Informationen werden im `Content CS` beschrieben. Dieses beinhaltet die folgenden Werte:

1. `Audio`
   1.1 `Natural`
   1.2 `Synthetic`
2. `Audiovisual`
3. `Scene`
4. `Visual`
   4.1 `Image`
   4.2 `Video`
   4.3 `Graphics`
5. `Textual`

Basierend auf diesen drei Attributtypen werden im System die Inhaltsklassen erstellt. Es ergeben sich $7 \cdot 9 \cdot 10 = 630$ unterschiedliche Inhaltsklassen, die zumindest theoretisch den einzelnen Attributtypen jeweils andere Gewichtungen zuweisen können. In der Regel wird sich jedoch eine Vielzahl dieser Klassen überschneiden bzw. dieselben Gewichte aufweisen. Die unterschiedlichen Gewichte können einheitlich für einen Anwendungsfall von einem Dienstanbieter definiert oder von den Nutzern selbst angegeben werden. Bevor ein prognostizierter Empfehlungswert für einen Inhalt errechnet wird, werden erst die Werte dieser drei Attributtypen betrachtet und die entsprechende Inhaltsklasse mit den zugeordneten Gewichten bestimmt, auf deren Basis die weiteren Berechnungen durchgeführt werden. Kommen die entsprechenden Werte in der Beschreibung des Inhalts nicht vor, können Standard-Werte verwendet werden.

5.5. Berücksichtigung von Kontextinformationen im Empfehlungsprozess

Im Folgenden wird untersucht, wie Kontextinformationen bei der Empfehlung multimedialer Inhalte berücksichtigt werden [181, 183]. Dazu werden zunächst einige Herausforderungen und Schwierigkeiten beim Abgleich von Kontextinformationen diskutiert

und daraus relevante Schritte abgeleitet. Anschließend werden allgemeine und für spezielle Kontextinformationen geeignete Ähnlichkeitsfunktionen vorgestellt. Diese werden schließlich mittels Aggregatfunktion zu einer Gesamtähnlichkeit zusammengefasst.

5.5.1. Herausforderungen und Schwierigkeiten beim Abgleich von Kontextinformationen

Die Bestimmung der Ähnlichkeit von zwei Kontextinformationen ist mit einigen Herausforderungen und Schwierigkeiten verbunden, die im Folgenden diskutiert werden.

Verschiedene Charakteristika der Kontextelemente Für den Abgleich von Kontextinformationen ist es schwierig, generische Mechanismen zu entwickeln. Der Grund dafür ist, dass die Elemente unterschiedliche Charakteristika bezüglich der Datentypen, der Gültigkeit, der Genauigkeit und der Spezifikation besitzen. So kann eine Kontextinformation wie ein Ort als numerischer Wert (z. B. als GPS-Koordinate), als strukturierter Datentyp (z. B. bei der Strukturierung der Orte als Baum) oder als nominaler Wert (z. B. Adresse) vorkommen. Je nach Datentyp müssen andere Ansätze zum Abgleich verwendet werden.

Abstraktionslevel Prinzipiell werden Kontextinformationen über Sensoren gemessen und stehen als Rohdaten zur Verfügung. Diese Rohdaten können jedoch aggregiert und veredelt werden, was zu Kontextinformationen in unterschiedlichen Abstraktionsstufen führt. Ein Beispiel hierfür ist die Abstrahierung von GPS-Koordinaten zu einem symbolischen Ort mit Adresse oder einer Raumnummer in einem Gebäude.

Abhängigkeiten der Kontextinformationen Bei Kontextinformationen unterscheidet man grob zwischen low-level Kontextinformationen, die direkt aus den Rohdaten gewonnen werden, und den high-level Kontextinformationen, die durch die Aggregation und Kombination mehrerer Kontextinformationen bestimmt werden. Dies kann natürlich nur geschehen, wenn die entsprechenden Kontextinformationen vorhanden sind.

Darstellung Die tatsächlichen Ausprägungen der Kontextinformationen können auf unterschiedliche Art und Weise dargestellt werden. So modellieren einige Ansätze Kontextinformationen mittels Ontologien, Graphen, semantischen Netzen oder logischen Regelwerken. Die Werte der Ausprägungen können stetig ineinander über gehen oder auch einer diskreten Ordnung folgen.

Distanz- bzw. Ähnlichkeitsfunktion Üblicherweise können Kontextinformationen mit unterschiedlichen Datentypen ausgedrückt werden, die textuell, numerisch oder symbolisch sein können. Daher sind die verwendeten Skalen der Kontextinformationen unterschiedlich, und es ergeben sich Schwierigkeiten bei der Bestimmung von eindeutigen Distanz- bzw. Ähnlichkeitswerten.

Interpretation der Kontextinformationen Wie der Einfluss einer Kontextinformation auf die Auswahl eines Inhalts ist, kann relativ unterschiedlich sein. So kann in einem Anwendungsfall dieselbe Kontextinformation eine größere Auswirkung auf die Auswahl haben als in einem anderen.

Wie aus obigen Überlegungen hervorgeht, ist es nicht möglich, ein generisches Verfahren zur Verfügung zu stellen, das alle verfügbaren Kontextinformationen vergleichen kann. Deswegen eignet sich beim Abgleich von Kontextinformationen die Verwendung des *Lokal-Global-Prinzips* [156]. Dieses besagt, dass nicht für eine komplette Beschreibung, sondern für jeden Attributtyp einzeln lokale Ähnlichkeitswerte bzw. Teilergebnisse bestimmt werden, die mittels einer Aggregationsfunktion zu einem globalen Wert bzw. zu einem Gesamtergebnis zusammengefasst werden.

Unter einem *lokalen Ähnlichkeitsmaß* versteht man eine Funktion wie in Formel 5.2 definiert, wobei D_{a_i} den Wertebereich des Attributtyps a_i repräsentiert. Durch Parametrisierung kann das lokale Ähnlichkeitsmaß individuell angepasst werden. Beispiele für lokale Ähnlichkeitsmaße, die allgemein oder für spezielle Attributtypen einsetzbar sind, werden in Abschnitt 5.5.3 bzw. 5.5.4 vorgestellt.

$$sim_{a_i} : D_{a_i} \times D_{a_i} \rightarrow [0,1] \tag{5.2}$$

Ein *globales Ähnlichkeitsmaß* ist eine Funktion wie in Formel 5.3 dargestellt. $\vec{md(c_i)}$ und $\vec{md(c_j)}$ sind die jeweils relevanten Attribute der Beschreibung $md(c_i)$ bzw. $md(c_j)$ zweier Inhalte c_i und c_j (oder eines entsprechenden Inhalts und eines Profils), gegeben in MASL. $\vec{\omega} = (\omega_1, ..., \omega_n)$ stellt einen Vektor mit Gewichten dar, $\omega_i \in [0,1]$, $\sum_{i=1}^{n} \omega_i = 1$. Die Funktion $\pi : [0,1]^n \rightarrow [0,1]$ definiert die *Aggregatfunktion* für die gilt: $\forall\, \vec{\omega} : \pi(0, ..., 0, \vec{\omega}) = 0$. Für den Fall, dass die lokalen Ähnlichkeiten alle gleich 0 sind und damit keinerlei Ähnlichkeit der Attributtypen besteht, muss also auch die globale Ähnlichkeit gleich 0 sein, unabhängig des Gewichtsvektors $\vec{\omega}$. Außerdem wird angenommen, dass die Aggregatfunktion in jedem Argument monoton steigend ist. sim_{a_i}, $i = 1, ..., n$ ist eine Ähnlichkeitsfunktion für den Attributtyp a_i, definiert wie oben. Beispiele für Aggregatsfunktionen werden in Abschnitt 5.5.5 gegeben.

$$SIM_\pi(\vec{md(c_i)}, \vec{md(c_j)}) = \pi(sim_{a_1}(md(c_i)_1, md(c_j)_1), ..., sim_{a_n}(md(c_i)_n, md(c_j)_n), \vec{\omega}) \tag{5.3}$$

5.5.2. Mathematische Grundlagen von Ähnlichkeitsfunktionen

Die Summe aller Unterschiede zwischen zwei Objekten (wie einem Nutzerprofil und einem unbekannten Inhalt) kann mittels Distanzmaß ausgedrückt werden. Dies wird über eine

Funktion $dist : D \times D \rightarrow [0,1]$ auf einem Definitionsbereich D definiert. Eine Distanz von 0 steht dabei für eine geringe, 1 für eine hohe Distanz.

Unter einer Ähnlichkeitsfunktion oder einem Ähnlichkeitsmaß versteht man im Allgemeinen eine Funktion $sim : D \times D \rightarrow [0,1]$, die für zwei Objekte aus dem Wertebereich D die Ähnlichkeit berechnet. Eine Ähnlichkeit von 0 steht für eine geringe, von 1 für eine hohe Ähnlichkeit. In die Ähnlichkeitsfunktion kann das oben definierte Distanzmaß einfließen.

Ähnlichkeitsfunktionen nehmen einen wichtigen Stellenwert z. B. im Bereich der Ähnlichkeitssuche in Datenbanksystemen oder beim Vergleich von Diensten bei der Service Discovery ein. Üblicherweise müssen sie die folgenden Eigenschaften erfüllen:

- **Reflexivität**: Ein Ähnlichkeitsmaß *sim* wird als reflexiv bezeichnet, wenn für alle Objekte x gilt: sim(x,x) = 1. Wird außerdem noch die Eigenschaft $sim(x,y) = 1 \rightarrow x = y$ erfüllt, bezeichnet man das Maß als streng reflexiv.
- **Symmetrie**: Ein Ähnlichkeitsmaß *sim* wird als symmetrisch bezeichnet, wenn gilt: $sim(x,y) = sim(y,x)$.
- **Monotonie**: Ein Ähnlichkeitsmaß *sim* wird als monoton bezeichnet, wenn $sim(x,y) \geq sim(x,z)$ für $x <_D y <_D z$ oder $z <_D y <_D x$ gilt, wobei $<_D$ als Ordnungsrelation auf dem Wertebereich D definiert ist.

5.5.3. Allgemeine Ähnlichkeitsfunktionen

Im Folgenden werden allgemeine Ansätze für Ähnlichkeitsfunktionen vorgestellt. Dem Lokal-Global-Prinzip folgend werden immer nur zwei konkrete Ausprägungen oder Werte eines Attributtyps a_i, also $a_i(x)$ und $a_i(y)$ miteinander verglichen. Die Gesamtähnlichkeit wird mittels Aggregation wie in Abschnitt 5.5.5 beschrieben berechnet. Die aufgeführten Funktionen basieren auf den Arbeiten von Stahl [156] und Linnhoff [110].

Ähnlichkeitsfunktionen für numerische Attribute

Der Vorteil numerischer Attributtypen ist, dass auf ihnen bereits eine Ordnung definiert ist und somit ohne große Probleme die Distanz berechnet werden kann. Beispiele hierfür sind die lineare Differenz

$$dist_{lin}(a_i(x), a_i(y)) = a_i(x) - a_i(y) \tag{5.4}$$

oder die logarithmische Differenz

$$dist_{log}(a_i(x), a_i(y)) = \begin{cases} \ln(a_i(x)) - \ln(a_i(y)), & falls\, a_i(x), a_i(y) \in \mathbb{R}^+ \\ -\ln(-a_i(x)) + \ln(-a_i(y)), & falls\, a_i(x), a_i(y) \in \mathbb{R}^- \\ Undefiniert, & sonst \end{cases} \tag{5.5}$$

Basierend auf einer gewählten Distanzfunktion *dist* können die Ähnlichkeitsfunktionen definiert werden. Man nimmt an, dass ein Sinken der Ähnlichkeit in Beziehung zu steigenden Werten der Distanzfunktion steht. Man unterscheidet zwischen der symmetrischen Ähnlichkeit

$$sim_{a_i}(a_i(x), a_i(y)) = f(|dist(a_i(x), a_i(y))|) \tag{5.6}$$

und der asymmetrischen Ähnlichkeit

$$sim_{a_i}(a_i(x), a_i(y)) = \begin{cases} f_1(dist(a_i(x), a_i(y))), & falls\, a_i(x) > a_i(y) \\ 1, & falls\, a_i(x) = a_i(y) \\ f_2(dist(a_i(x), a_i(y))), & falls\, a_i(x) < a_i(y) \end{cases} \tag{5.7}$$

f_1 ist üblicherweise eine monoton steigende, f_2 eine monoton fallende Funktion, die $\mathbb{R}$ auf $[0, 1]$ abbildet. Sie beschreiben das Abnehmen der Ähnlichkeit mit zunehmender Distanz, $f_1(0) = 0$ und $f_2(0) = 0$. Für die Funktionen gibt es unterschiedliche Umsetzungsmöglichkeiten, die in Abbildung 5.12 graphisch dargestellt werden. Diese sind im Einzelnen die *Grenzwert-Funktion* zu einem gegebenen Schwellwert ϑ

$$sim_{a_i}(dist(a_i(x), a_i(y))) = \begin{cases} 1, & falls\, dist(a_i(x), a_i(y)) < \vartheta \\ 0, & falls\, dist(a_i(x), a_i(y)) \geq \vartheta \end{cases} \tag{5.8}$$

die *lineare Funktion* zu einem gegebenen Minimalwert min und Maximalwert max

$$sim_{a_i}(dist(a_i(x), a_i(y))) = \begin{cases} 1, & falls\, dist(a_i(x), a_i(y)) < min \\ \frac{max - dist(a_i(x), a_i(y))}{max - min}, & falls\, min \leq dist(a_i(x), a_i(y)) \leq max \\ 0, & falls\, dist(a_i(x), a_i(y)) > max \end{cases} \tag{5.9}$$

die *Exponential-Funktion* zu einem gegebenen Parameter α

$$sim_{a_i}(dist(a_i(x), a_i(y))) = \exp^{dist(a_i(x), a_i(y)) \cdot \alpha} \tag{5.10}$$

die *Sigmoid-*, *S-* oder auch *Schwanenhalsfunktion* zu gegebenen Parameter α und Schwellwert ϑ

$$sim_{a_i}(dist(a_i(x), a_i(y))) = \frac{1}{\exp^{\frac{dist(a_i(x), a_i(y)) - \vartheta}{\alpha}} + 1} \tag{5.11}$$

und schließlich eine *treppenförmige Funktion* zu beliebig vielen, gegebenen Schwellwer-

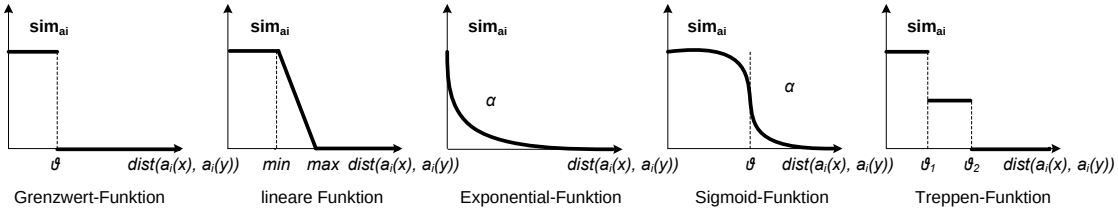

Abbildung 5.12.: Beispiele für lokale Ähnlichkeitsmaße für numerische Attribute

ten ϑ_n, z. B.

$$sim_{a_i}(dist(a_i(x), a_i(y))) = \begin{cases} 1, & falls\, dist(a_i(x), a_i(y)) < \vartheta_1 \\ 0.6, & falls\, \vartheta_1 \geq dist(a_i(x), a_i(y)) < \vartheta_2 \\ 0.3, & falls\, \vartheta_2 \geq dist(a_i(x), a_i(y)) < \vartheta_3 \\ 0, & falls\, \vartheta_3 \geq dist(a_i(x), a_i(y)) \end{cases} \tag{5.12}$$

Ähnlichkeitsfunktionen für klassifizierbare, aber nicht-hierarchisch klassifizierbare Attribute

Handelt es sich bei einem Attribut um ein klassifizierbares, aber nicht-hierarchisch klassifizierbares Attribut, kann es nur Werte aus einer Menge D_{a_i} annehmen. Man muss unterscheiden, ob zwischen diesen Werten eine Ordnung herstellbar ist oder nicht.

Bei ungeordneten, klassifizierbaren, aber nicht-hierarchisch klassifizierbaren Attributen besteht keine Ordnung auf den einzelnen Werten von D_{a_i}. Um dennoch ein Ähnlichkeitsmaß anwenden zu können, kann man eine $n \times n$-Matrix $S = [s_{ij}]$ definieren, wobei $0 \leq s_{ij} \leq 1$ ist. In diese Matrix werden die Ähnlichkeitswerte zwischen den unterschiedlichen Werten von D_{a_i} abgelegt. Die Bestimmung der Ähnlichkeit zweier Werte $a_i(x)$ und $a_i(y)$ des Attributtyps a_i erfolgt dann mittels des tabellarischen Ähnlichkeitsmaßes:

$$sim_{a_i}(a_i(x), a_i(y)) = s_{a_i(x)a_i(y)} \tag{5.13}$$

Diese Methode macht natürlich nur Sinn, wenn D_{a_i} eine überschaubare Menge an Werten enthält, da die Größe von S und die Zahl der zu verwaltenden Einträge quadratisch mit der Anzahl der Elemente in D_{a_i} wächst.

Handelt es sich um einen geordneten, klassifizierbaren, aber nicht-hierarchisch klassifizierbaren Attributtyp besteht eine totale Ordnung auf den Elementen, also $D_{a_i} = a_i(1), ..., a_i(n)$ mit $a_i(j) < a_i(j+1)$ für $j = 1, ..., n-1$. Damit kann man die Ähnlichkeitsfunktionen für numerische Attribute auf den Indizes der Werte, die durch die Ordnung entstehen, verwenden, um eine Ähnlichkeitsfunktion zu erhalten.

Ähnlichkeitsfunktionen für hierarchisch klassifizierbare Attribute

Ein Ähnlichkeitsmaß für hierarchisch klassifizierbare Attributtypen a_i muss die Anforderung erfüllen, dass wenn für drei Werte $a_i(x), a_i(y), a_i(z)$ die Bedingung $sim_{a_i}(a_i(x), a_i(z)) \leq sim_{a_i}(a_i(y), a_i(z))$ gilt, auch $\langle a_i(x), a_i(z)\rangle > \langle a_i(y), a_i(z)\rangle$ gelten muss. $\langle k_1, k_2\rangle$ steht dabei für den nächsten gemeinsamen Vorgängerknoten zweier beliebiger Knoten k_1 und k_2 in der Hierarchie und $<$ gibt eine Ordnung auf den Knoten vor, so dass bei $k_1 > k_2$ k_2 ein Nachfolger von k_1 ist.

Ein Ansatz zur Bestimmung der Ähnlichkeit zweier Knoten ist es, jedem Knoten des Baumes einen Ähnlichkeitswert $s_i \in [0, 1]$ zuzuordnen, so dass die Ähnlichkeitswerte Richtung Wurzel abnehmen. Für die Ähnlichkeitsbestimmung zweier Knoten $a_i(x)$ und $a_i(y)$ wird der nächste gemeinsame Vorgängerknoten bestimmt und dessen Ähnlichkeitswert übernommen. Je näher dieser Knoten an der Wurzel liegt desto weniger ähnlich sind die beiden Knoten. Damit ergibt sich die folgende Formel:

$$sim_{a_i}(a_i(x), a_i(y)) = \begin{cases} 1, & falls\, a_i(x) = a_i(y) \\ s_{\langle a_i(x), a_i(y)\rangle}, & falls\, a_i(x) \neq a_i(y) \end{cases} \tag{5.14}$$

Ein anderer Ansatz für den Ähnlichkeitsvergleich von zwei Knoten $a_i(x)$ und $a_i(y)$ ist es, die Länge des gemeinsamen Pfades in Verhältnis zu setzen mit der Länge der einzelnen Pfade von $a_i(x)$ und $a_i(y)$. Die Länge des gemeinsamen Pfades wird durch das Level des gemeinsamen Vorgängers $\langle a_i(x), a_i(y)\rangle$ bestimmt. Formel 5.15 veranschaulicht diese Überlegungen, $Level(k)$ steht für die Anzahl der Knoten, die sich zwischen Wurzel und einem Knoten k befinden, inklusive der Wurzel und k.

$$sim_{a_i}(a_i(x), a_i(y)) = \frac{Level(\langle a_i(x), a_i(y)\rangle)}{Level(a_i(x)) + Level(a_i(y))} \tag{5.15}$$

5.5.4. Spezielle Ähnlichkeitsfunktionen

Während im letzten Abschnitt allgemeine Funktionen vorgestellt wurden, werden im Folgenden Ansätze vorgestellt, die für spezielle Attributtypen geeignet sind.

Kollaboratives Filtern mit MASL

Kollaboratives Filtern lässt sich mit MASL unter Verwendung der in der `UsageActionHistory` gespeicherten Informationen umsetzen. Diese Informationen geben Aufschluss darüber, ob ein Inhalt vollständig betrachtet wurde oder ob er frühzeitig abgebrochen wurde.

Nach Herlocker et al. [78], besteht das *nearest-neighbor collaborative Filtering*, das für die Verwendung im Rahmen dieser Arbeit ausgewählt wurde, aus folgenden drei Schritten:

1. Vergleich aller Nutzer der Gesamtmenge U im Bezug auf ihre Ähnlichkeit mit dem aktiven Nutzer u.

2. Auswahl einer Untermenge U_u aus U, die als Menge zur Vorhersage für ein gegebenes Objekt j verwendet wird. Diese Menge wird als *nächste Nachbarn* des aktiven Nutzers u bezeichnet.

3. Berechnung des prognostizierten Empfehlungswerts $P_u(c)$ für ein gegebenes Objekt c basierend auf den Bewertungen der nächsten Nachbarn von Nutzer u, gegeben durch das jeweilige Nutzerprofil.

Nutzer und ihre Bewertungen der Inhalte bilden eine $n \times m$ Matrix M, wobei n der Anzahl aller Nutzer aus U und m der Anzahl der Inhalte aus der Gesamtmenge aller Inhalte C entspricht. Für jeden der drei Schritte steht eine Vielzahl an Algorithmen zur Verfügung, die verwendet werden können. Die im Rahmen dieser Arbeit verwendeten Algorithmen werden im Folgenden erläutert.

Bestimmung der Nutzerähnlichkeit Um die Ähnlichkeit zwischen Nutzern zu bestimmen, gibt es verschiedene Ansätze. Ein oft verwendeter ist die *Pearson Korrelation*, die die Nutzerähnlichkeit paarweise berechnet [140]. Andere Ansätze teilen mittels Clusteralgorithmen Nutzer so in Gruppen ein, dass Nutzer einer Gruppe ähnlicher zueinander sind als Nutzer von verschiedenen Gruppen. Diese Algorithmen kann man grob einteilen in hierarchische (Bestimmung von Clustern basierend auf zuvor bestimmten Clustern), partitionierende (zeitgleiche Bestimmung aller Cluster) und Dichte-basierte (Definition von Clustern als Bereich in dem die Dichte der Datenobjekte einen gewissen Schwellwert übersteigen) Verfahren einteilen.

Im Rahmen dieser Arbeit wurde der partitionierende *k-means* Algorithmus verwendet [112], da es sich um einen sehr effizienten Algorithmus mit linearer Komplexität handelt der einfach implementiert werden kann. Ziel des Algorithmus ist es, die n Nutzer in k Untermengen einzuteilen, wobei $k < n$. Dies geschieht wie folgt: zunächst werden zufällig k Nutzer als Clusterzentren ausgewählt. Für alle verbleibenden Nutzer wird der Abstand zu den gewählten Zentren berechnet, jeder Nutzer dem nächsten Zentrum zugeordnet und zu dem zugehörigen Cluster hinzugefügt. Als Distanzfunktion wird die Euklidische Distanz verwendet. Für alle k Cluster wird das Zentrum erneut berechnet und die Nutzer neu zugeteilt, bis keine Änderungen mehr ausgeführt werden müssen.

Um auf diese Weise Nutzer effizient clustern zu können, wird zuvor die Dimension der Nutzer-Objekt-Matrix reduziert. Dafür wird die *Hauptkomponentenanalyse* verwendet [69]. Es wird angenommen, dass eine Menge von l Objekten, das so genannte Anmeldeset, von jedem Nutzer bewertet wurde und die Bewertung im Profil vorliegt. Die Matrix M kann dann normalisiert werden zu einer $n \times l$ Submatrix G des Anmeldesets. Jede Bewertung $R_u(c)$ von Inhalt c durch Nutzer u kann, wie in Formel 5.16 zu sehen

ist, zu Bewertung $\tilde{R}_u(c)$ normalisiert werden. μ_c bezieht sich auf die mittlere Bewertung von Inhalt c über alle Nutzer und σ_c ist die Standardabweichung.

$$\tilde{R}_u(c) = \frac{R_u(c) - \mu_c}{\sigma_c} \tag{5.16}$$

Die Durchschnittsbewertung μ_c von Inhalt c ist abhängig von den Bewertungen $R_u(c)$ eines jeden Nutzers u aus der Menge U_c, die alle Nutzer enthält, die Inhalt c bewertet haben (siehe Formel 5.17).

$$\mu_c = \frac{1}{n} \sum_{u \in U_c} R_u(c) \tag{5.17}$$

Unter Verwendung dieser Durchschnittsbewertung kann die Standardabweichung σ_c von Inhalt c wie in Formel 5.18 berechnet werden.

$$\sigma_c = \sqrt{\frac{1}{n-1} \sum_{u \in U_c} (R_u(c) - \mu_c)^2} \tag{5.18}$$

Auswahl der nächsten Nachbarn Die Berechnung von prognostizierten Empfehlungswerten mittels kollaborativem Filtern basiert auf den Bewertungen der so genannten nächsten Nachbarn des aktiven Nutzers u. In dieser Menge befinden sich alle Nachbarn, die ähnlich zu Nutzer u sind *und* den entsprechenden Inhalt c bewertet haben. Um die Menge der nächsten Nachbarn zu bestimmen, können unterschiedliche Verfahren verwendet werden. So kann man z. B. basierend auf einem Ähnlichkeitsschwellwert alle Nutzer wählen, deren Ähnlichkeit diesen Wert übersteigt [153]. Ein weit verbreiteter Ansatz verwendet die k nächsten Nachbarn, unabhängig von ihrem Ähnlichkeitswert [140]. Eine weitere Möglichkeit sind Bayes'sche Netze, die verwendet werden, um das Nutzerverhalten zu modellieren und um Abhängigkeiten zwischen bewerteten Inhalten zu erkennen [145].

Um die nächsten Nachbarn zu bestimmen, wird im Folgenden der k nächste Nachbar-Ansatz mit einem Schwellwert kombiniert: die k (oder weniger) nächsten Nachbarn, die einen gegebenen Schwellwert übersteigen, werden ausgewählt. Dadurch verhindert man, dass Nutzer mit niedriger Ähnlichkeit zu Nutzer u im Empfehlungsprozess berücksichtigt werden. Ist die Menge leer, muss auf andere Verfahren zurückgegriffen werden.

Berechnung des prognostizierten Empfehlungswerts Nachdem dem Nutzer u ähnliche Nutzer identifiziert und daraus die nächsten Nachbarn gewählt wurden, müssen deren Bewertungen zu einem prognostizierten Empfehlungswert $P_u(c)$ für Inhalt c aggregiert werden. Eine weit verbreitete Methode hierfür ist in Formel 5.19 zu sehen [140]. $\bar{R}_u$ steht für die Durchschnittsbewertung des Nutzers u über alle Inhalte und U_u für die Menge der nächsten Nachbarn von u, die wie oben beschrieben bestimmt wurden. Die Ähnlichkeit $sim(u, v)$ zwischen zwei Nutzern u und v kann mittels Euklidischer Distanz oder mittels

Pearson Korrelation wie erläutert bestimmt werden. Dieser Faktor gewichtet die normalisierte Bewertung $R_v(c) - \bar{R}_v$ des nächsten Nachbarn v so, dass zu u ähnlichere Nutzer einen stärkeren Einfluss auf die Berechnung des prognostizierten Empfehlungswerts haben.

$$P_u(c) = \bar{R}_u + \frac{\sum_{v \in N_u} (R_v(c) - \bar{R}_v) sim(u, v)}{\sum_{v \in N_u} |sim(u, v)|} \tag{5.19}$$

Präferenz-basiertes Filtern mit MASL

Im Gegensatz zum kollaborativen Filtern bilden beim Präferenz-basierten Filtern nicht die Nutzer, sondern die Beschreibungen der Inhalte die Grundlage. Es handelt sich also um eine Sonderform des inhaltsbasierten Filterns. Im Grunde werden bei diesem Verfahren die Inhalte kategorisiert und basierend auf den Nutzerinteressen selektiert. MASL bietet zu diesem Zweck DSs zur Kategorisierung der Nutzer und der Inhalte an.

Da die Präferenzen sehr breit gefächert sein können, bietet sich eine Abbildung aller Attribute in einen Featureraum nicht an. Weiterhin wird bei MASL die Unabhängigkeit der Attribute vorausgesetzt. Von daher kann die Ähnlichkeit nicht über den Abstand oder den Winkel der Vektoren bestimmt werden. Im Folgenden werden die Ähnlichkeiten basierend auf einem bool'schen Modell bestimmt.

Allgemein wird bei diesem Modell eine (reaktive oder proaktive) Anfrage durch einen Ausdruck der bool'schen Algebra gebildet, die Ressourcen oder Inhalte werden durch eine Menge von Termen beschrieben. Um zu entscheiden, ob ein Inhalt einem Profil entspricht, werden diese Ausdrücke ausgewertet. Das klassische bool'sche Modell verfolgt einen *Exact Match*-Ansatz, d.h. nur bei Gleichheit der Attribute von Profil und Inhalt, wird der Inhalt als passend ausgewählt (Ganz-Oder-Gar-Nicht-Prinzip). Im hier vorgestellten Präferenz-basierten Verfahren soll aber keine reine binäre Auswertung verwendet werden.

Für das Präferenz-basierte Filtern sind die Sub-Elemente der `FilteringAndSearchPreferences` in einem durch MASL gegebenen Profil $pd(u)$ des Nutzers u von Relevanz. Diese können auf Grund der hierarchischen Struktur in einen bool'schen Ausdruck umgewandelt werden, indem man die verschiedenen Arten von `Preferences`, also `Classification-`, `Creation-`, `Semantic-`, `RelatedInformation-`, `Source-` und geschachtelte `FilteringAndSearchPreferences`, die sich auf derselben hierarchischen Ebene befinden, und ihre enthaltenen Elemente mit $\wedge$ (AND) verknüpft, wohingegen gleiche Arten von Preferences auf derselben Ebene mit $\vee$ (OR) verknüpft werden. Dadurch ergeben sich p Bereiche der Profilbeschreibung, wobei jeder Bereich $pd_i(u), i = 1, ..., p$ wiederum aus einer Menge von (geschachtelten oder ungeschachtelten) Ausdrücken $pd_{i_j}(u)$ zusammengesetzt wird.

Bei der Auswertung werden die Werte des bool'schen Ausdruckes bei Vorhandensein des entsprechenden Attributs in der Beschreibung $md(c)$ des Inhalts c durch 1 bzw. bei Nicht-Vorhandensein durch 0 ersetzt. Um zusätzlich noch die im Profil vorhandenen

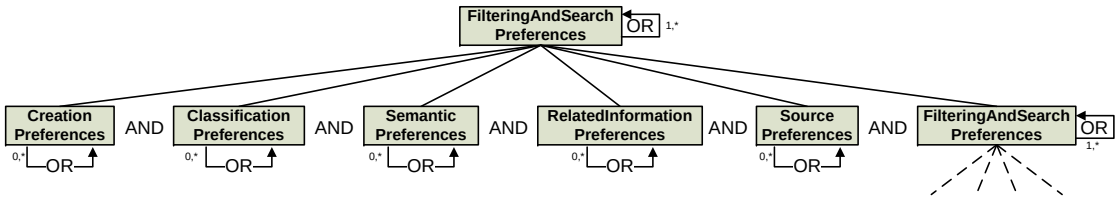

Abbildung 5.13.: Auswertung der Präferenzen-Hierarchie

$\wedge$	0	1
0	0	0
1	0	1

Abbildung 5.14.: Konjunktion

$\vee$	0	1
0	0	1
1	1	1

Abbildung 5.15.: Disjunktion

Präferenzen zu berücksichtigen, wird dieser Wahrheitswert mit dessen `preferenceValue` multipliziert und es ergibt sich das Ergebnis $E_{pd_{i_j}(u)}$ für den j-ten bool'schen Ausdruck des i-ten Bereichs (siehe Formel 5.20). Dann kann der Ausdruck ausgewertet werden, wobei $\wedge$ bzw. die Konjunktionen mittels Multiplikation (vgl. Abbildung 5.14), $\vee$ bzw. die Disjunktionen mittels Addition (vgl. Abbildung 5.15) berechnet wird. Ist das Ergebnis $E_{pd_i(u)}$ für den i-ten Bereich $pd_i(u)$ gleich 0 wird das Maximum der einzelnen Operanden als Ähnlichkeitswert verwendet, ansonsten das arithmetische Mittel, damit bei einem einzelnen nicht in der Beschreibung vorkommenden Operanden wenigstens die anderen vorkommenden dieses Bereiches berücksichtigt werden. Dieser Zusammenhang ist in Formel 5.21 dargestellt.

$$E_{pd_{i_j}(u)} = \begin{cases} 1 \cdot preferenceValue_{pd_{i_j}(u)}, & falls\, pd_{i_j}(u) \in md(c) \\ 0 \cdot preferenceValue_{pd_{i_j}(u)}, & falls\, pd_{i_j}(u) \notin md(c) \end{cases} \tag{5.20}$$

$$sim(pd_i(u), md(c)) = \begin{cases} \max pd_{i_j}(u), & falls\, E_{pd_i(u)} > 0 \\ \frac{1}{n} \cdot \sum_{j=1}^{n} pd_{i_j}(u), & falls\, E_{pd_i(u)} = 0 \end{cases}, i = 1, ..., n \tag{5.21}$$

Der Ähnlichkeitswert des Profils $pd(u)$ eines Nutzers u mit p Bereichen $pd_i(u), i = 1, ..., p$ und der Beschreibung $md(c)$ eines Inhalts c ergibt sich dann wie in Formel 5.22 zu sehen.

$$sim(pd(u), md(c)) = \max sim(pd_i(u), md(c))\, mit\, i = 1, ..., p \tag{5.22}$$

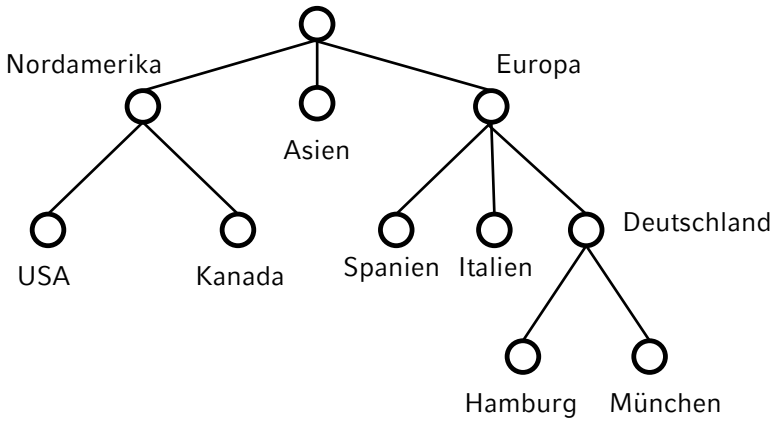

Abbildung 5.16.: Baumdarstellung symbolischer Ortsinformationen

Ort

Der aktuelle Ort eines Nutzers oder ein Ort, dem ein Inhalt zugeordnet ist, kann innerhalb der MASL-Beschreibungen auf verschiedene Arten gegeben sein. So sind z. B. symbolische Bezeichner oder Koordinaten gängige Ansätze, Ortsinformationen zu beschreiben.

Makkonen et. al beschreiben in [113] einen Ontologie-basierten Ansatz zur Bestimmung der Ähnlichkeit. Daran angelehnt kann man symbolische Bezeichner auf eine Baumstruktur abbilden (vgl. Abbildung 5.16). Mit Hilfe dieser Bäume kann man Ortsinformationen auf hierarchisch klassifizierbare Attributtypen abbilden und die in 5.5.3 eingeführte Ähnlichkeitsfunktionen für diese Attributtypen anwenden.

Sind die Ortsinformationen als Koordinaten (z. B. GPS) gegeben, kann man die Distanz zwischen ihnen über die in Formel 5.23 gegebene Distanzfunktion berechnen. Dabei steht φ_i für den Längengrad und τ_i für den Breitengrad von Koordinate i, Variable r entspricht dem Erdradius mit 6371km (mittlerer Radius).

$$dist_{location}(x, y) = \frac{2\pi \cdot r}{360°} \cdot \arccos(\sin(\varphi_x)\sin(\varphi_y) + \cos(\varphi_x)\cos(\varphi_y)\cos(|\tau_y - \tau_x|)) \quad (5.23)$$

Den Distanzen können dann in Tabellen feste Ähnlichkeitswerte zugewiesen werden. Diese kann man je nach Anwendungsfall oder Art der Fortbewegung (z. B. zu Fuß, per Rad oder Auto) definieren. Ein Beispiel ist in Tabelle 5.3 zu sehen. Andere Ansätze können eine auf diesem Distanzwert basierende Ähnlichkeitsfunktion analog zu den Ansätzen für numerische Attribute definieren.

Distanz (in Metern)	$sim_{Location}$
$> 5000 \cdot\delta$	0,2
$> 2500 \cdot\delta$	0,5
$> 750 \cdot\delta$	0,7
$> 100 \cdot\delta$	0,85
$> 20 \cdot\delta$	0,95
$> 0 \cdot\delta$	1

Tabelle 5.3.: Distanz-basierter Ähnlichkeitswert für den Ort. Der Wert von δ kann je nach Fortbewegungsart angepasst werden: Fußgänger ($\delta = 1$), Fahrradfahrer ($\delta = 5$), Autofahrer ($\delta = 10$)

Daten und Zeittypen

Für Attribute, die einen Datums- oder Zeitwert repräsentieren, können in der Regel die Ähnlichkeitsfunktionen der numerischen Attribute verwendet werden. Dafür wird der Abstand zwischen zwei Daten bzw. Zeiten berechnet und daraus eine Ähnlichkeit bestimmt.

Ein besonderer Fall tritt bei Daten oder Zeiten ein, die sich nur auf einen festen Tag oder ein Zeitintervall beziehen. Dies ist z. B. bei Feiertagen wie Weihnachten oder Ostern oder bei Öffnungszeiten von Museen und Restaurants der Fall.

Im ersten Fall nimmt ein Inhalt kontinuierlich an Bedeutung zu, je näher das aktuelle Datum dem Feiertag rückt. Nach dem Erreichen des Termins ist die Bedeutung jedoch sofort gleich 0. Genau anders herum verhält es sich im zweiten Fall: vor der Öffnungszeit ist der Inhalt nicht relevant, nähert sich die aktuelle Zeit dem Ende der Öffnungszeit, nimmt die Bedeutung kontinuierlich ab. Dieses Verhalten ist in den Abbildungen 5.17 bzw. 5.18 dargestellt. Mathematisch kann man das wie in Formel 5.24 bzw. 5.25 dargestellt ausdrücken. H bzw. OT steht für das Datum des Feiertages bzw. für die Öffnungszeit, I für das davor oder danach liegende Intervall, in dem der Inhalt relevant ist und $\vartheta \in [0, 1]$ bestimmt den Schwellwert, bis zu welchem Anteil an I der Inhalt relevant ist.

$$sim_{time}(a_i(x), H) = \begin{cases} 0, & falls\, a_i(x) < H - I\, oder\, a_i(x) > H \\ 1, & falls\, H - \vartheta \cdot I \leq a_i(x) \leq H \\ \frac{H - I + a_i(x)}{(I \cdot (1 - \vartheta)}, & falls\, H - I \leq a_i(x) < H - \vartheta \cdot I \end{cases} \tag{5.24}$$

$$sim_{time}(x, OT) = \begin{cases} 0, & falls\, x < OT oder x > OT + I \\ 1, & falls\, OT \leq x \leq OT + \vartheta \cdot I \\ \frac{OT + I - a_i(x)}{I \cdot (1 - \vartheta)} & falls\, OT + \vartheta \cdot I < x \leq OT + I \end{cases} \tag{5.25}$$

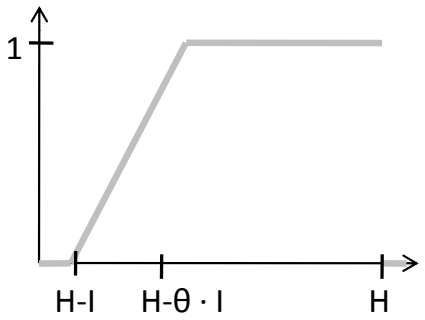

Abbildung 5.17.: Ähnlichkeit der Zeit bezüglich des Feiertages H

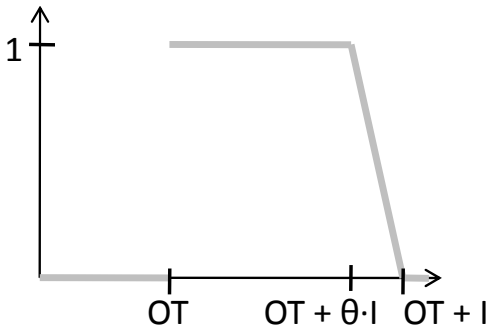

Abbildung 5.18.: Ähnlichkeit der Zeit bezüglich der Öffnungszeiten OT

Soziale Umgebung

Ein interessanter Aspekt ist, die Empfehlung von Inhalten von der sozialen Umgebung abhängig zu machen, also welche weiteren Personen sich in der Umgebung des Nutzers befinden, welchen Stellenwert diese Personen einnehmen (z. B. Freunde, Kollegen, Chef) oder um wie viele Personen es sich handelt. Diese Daten sensorisch zu erfassen, ist keine triviale Aufgabe. Eine Möglichkeit ist, über Bluetooth zu prüfen, welche anderen Geräte in der Nähe sind und eine Zuordnung zu konkreten Personen über das Adressbuch zu erreichen. Die einfachste, aber auch unkomfortabelste Lösung ist das explizite Eintragen dieser Informationen durch den Nutzer. Im Folgenden werden jedoch keine Ansätze diskutiert, wie die Informationen in das Profil gelangen, sondern es wird angenommen, dass diese Daten im Profil vorliegen.

Die Information über die soziale Umgebung kann verwendet werden, um z. B. Inhalte, die von Personen in der unmittelbaren Umgebung erzeugt wurden, zu bevorzugen. Dies kann über das oben beschriebene Präferenz-basierte Verfahren geschehen.

Die Anzahl der Personen, die in einem engen Verhältnis zum Nutzer stehen und sich in der unmittelbaren Umgebung befinden, kann man verwenden um z. B. die Anzahl der freien Plätze $a_{freiePlätze,c}$ angegeben über die informelle Beschreibung eines Inhalts c mit der Anzahl der Personen $a_{anzahlPersonen,u}$ in der Umgebung des Nutzers u zu vergleichen. Dadurch erhält man den in Formel 5.26 dargestellten Zusammenhang.

$$sim_{socialEnvironment}(md(c), pd(u)) = \begin{cases} 1, a_{freiePlätze,c} \leq a_{anzahlPersonen,u} \\ 0, a_{freiePlätze,c} > a_{anzahlPersonen,u} \end{cases} \tag{5.26}$$

Ein weiterer Einsatz der Informationen über das soziale Umfeld ist die bevorzugte

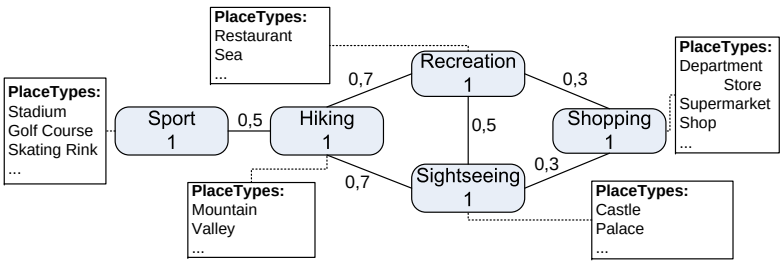

Abbildung 5.19.: Aktivitäten als Graph mit gewichteten Kanten. Jedem Knoten ist eine Liste von `PlaceType`s zugeordnet.

Auswahl von Inhalten von Personen, die in einer Beziehung zu dem Nutzer stehen, unabhängig von ihrem Aufenthaltsort, also z. B. Inhalte, die von Freunden oder Kollegen erzeugt wurden. Auch in diesem Fall verwendet man Präferenz- bzw. inhaltsbasiertes Filtern.

Aktivität

Der Abgleich einer Inhaltsbeschreibung mit der aktuellen Aktivität bietet eine Vielzahl an Möglichkeiten für interessante Anwendungen. Eine große Schwierigkeit liefert hier wieder die Bereitstellung eines Verfahrens, wie diese Information effizient und vor allem zuverlässig erfasst werden kann. Hierfür können Daten über das Bewegungsmuster des Nutzers, Kalenderinformationen oder der aktuelle soziale Kontext des Nutzers wie der aktuelle Aufenthaltsort und die ihn umgebenden Personen verwendet werden.

Große Schwierigkeiten bereitet es, wie man die Information über die aktuelle Aktivität des Nutzers sinnvoll auf die Inhaltsbeschreibungen abbilden kann. Eine Idee ist es, das im MASL-Standard enthaltene `PlaceType CS` zu verwenden, das für eine Einteilung von Orten in 14 unterschiedliche Klassen gedacht ist. Beispiele hierfür sind Sportstätten wie Stadien, Tennisplätze oder Schwimmbäder, Erholungs- bzw. Freizeitgebiete wie Gartenanlagen oder Freizeitparks, Einkaufsstätten wie Supermärkte, Einkaufszentren oder Boutiquen, historische Anlagen wie Schlösser und Kirchen oder natürliche Umgebungen wie Flüsse, Berge und Seen. Diese Ortsbeschreibungen kann man unterschiedlichen Aktivitäten wie Sport, Erholung, Einkaufen, Sightseeing oder Wandern zuordnen. So ist ein Nutzer, der sich in der Aktivität *Einkaufen* befindet eher an Informationen über Einkaufsstätten interessiert als an Inhalten über Sportstätten. Die Zusammenhänge zwischen den einzelnen Aktivitäten und den zugeordneten `PlaceType`s kann man in einem Graph mit gewichteten Kanten darstellen (vgl. Abbildung 5.19). Die Gewichte und die vorhandenen Kanten können individuell auf einen Nutzer angepasst werden. Vorstellbar ist hier z. B. der Einsatz von selbstlernenden Algorithmen, die den Graphen an das Nutzerverhalten anpassen. Dies wird im Rahmen dieser Arbeit jedoch nicht genauer untersucht.

Der Aktivitätsgraph kann verwendet werden, um geeignete Inhalte auszuwählen. Dabei eignen sich die Inhalte am besten, die einem `PlaceType` zugeordnet sind, der in der Liste der Aktivität vorhanden ist, in der sich der Nutzer momentan befindet. Dies kann mittels inhaltsbasiertem Filtern bestimmt werden. Zusätzlich sollen aber auch Inhalte mit `PlaceType`s ausgewählt werden können, die einer benachbarten Aktivität zugeordnet sind. Dafür wird der Wert mit dem entsprechenden Kantengewicht modifiziert.

Um also die Ähnlichkeit zwischen einem Inhalt c und einem Nutzerprofil u zu bestimmen, wird Formel 5.27 verwendet. $\|u_{Act}(j)\|$ bezieht sich auf die der j-ten Aktivität Act im Nutzerprofil $pd(u)$ des Nutzers u zugeordneten Liste der `PlaceType`s.

$$sim_{Activity}(a_{Place}(i), u_{Act}(j)) = \begin{cases} 1, & falls\ a_{PlaceType}(i) \in \|u_{Act}(j)\| \\ \kappa_{u_{Act}(j),u_{Act}(k)}, & falls\ a_{PlaceType}(i) \in \|u_{Act}(k)\| \\ & mit\ u_{Act}(k)\ direkt\ erreichbar \\ & von\ u_{Act}(j)\ mit\ einem\ Kanten- \\ & gewicht\ von\ \kappa_{u_{Act}(j),u_{Act}(k)} \\ \kappa_{u_{Act}(l),u_{Act}(m)} \cdot \ldots & \\ \cdot\kappa_{u_{Act}(j),u_{Act}(k)} & falls\ a_{PlaceType}(i) \in \|u_{Act}(l)\| \\ & mit\ u_{Act}(l)\ erreichbar\ von \\ & u_{Act}(j)\ über\ Kanten\ a\ und\ b\ mit \\ & einem\ Kantengewicht\ von \\ & \kappa_{u_{Act}(a),u_{Act}(b)} \\ 0, & sonst \end{cases} \tag{5.27}$$

5.5.5. Aggregation der lokalen Ähnlichkeiten

Um die einzelnen Teilergebnisse der lokalen Ähnlichkeitsfunktionen zu einem Einzelergebnis zusammenfassen zu können, werden Aggregator-Funktionen verwendet. Diese überführen die Teilergebnisse bzw. die lokalen Ähnlichkeiten der betrachteten Attributtypen mit Hilfe einer Funktion in ein Gesamtergebnis bzw. eine globale Ähnlichkeit.

Basierend auf den Arbeiten von Stahl [156] und Linnhoff [110] werden im Folgenden Ansätze aus dem Bereich der Ähnlichkeitssuche in Objekt-relationalen Datenbanken bzw. aus dem Bereich der Bestimmung von Ähnlichkeitsmetriken von Diensten vorgestellt, die auch in der vorliegenden Arbeit zur Aggregation von Kontextinformationen verwendet werden.

Bei der Aggregation mit Hilfe des Maximums bestimmt der höchste lokale Ähnlichkeitswert die globale Ähnlichkeit (siehe Formel 5.28). Es wird eine disjunktive, globale Ähnlichkeit gebildet, wobei die globale Ähnlichkeit von der höchsten lokalen Ähnlich-

keit mit entsprechender Gewichtung bestimmt wird. Diese Aggregation hat den Vorteil, dass unabhängig von den lokalen Ähnlichkeitswerten lediglich der höchste Wert einen Einfluss auf die globale Ähnlichkeit hat. Dies macht vor allem dann Sinn, wenn bei einer Anwendung nur eine der lokalen Ähnlichkeiten hoch ist, während die anderen eine niedrige Ähnlichkeit aufweisen. Allerdings ist bei der Maximum-Aggregation nur ein einziger Ähnlichkeitswert entscheidend, was für Anwendungen, die von mehreren lokalen Ähnlichkeiten abhängig sind, ungeeignet ist.

$$\pi(sim_1, ..., sim_n, \vec{\omega}) = \max_{i=1}^{n}\{\omega_i \cdot sim_i\} \tag{5.28}$$

Eine Verallgemeinerung der Maximum-Aggregation ($k = 1$) ist die Aggregation mittels des k-Maximums. Die globale Ähnlichkeit wird wie in Formel 5.29 zu sehen, von der k-größten lokalen Ähnlichkeit bestimmt. Ein Vorteil dieses Ansatzes ist, dass nicht wie bei der allgemeinen Maximum-Aggregation nur der höchste Wert berücksichtigt wird, sondern der k-höchste. Damit kann man einzelne Ausreißer der lokalen Ähnlichkeit nach oben ausgleichen.

$$\begin{aligned} \pi(sim_1, ..., sim_n, \vec{\omega}) = kmax_{i=1}^{n}\{\vec{\omega} \cdot sim_i\} = \omega_{i_k} \cdot sim_{i_k} \ , \\ mit: \ \omega_{i_k} \cdot sim_{i_k} \geq \omega_{i_{k+1}} \cdot sim_{i_{k+1}} \end{aligned} \tag{5.29}$$

Analog zur Maximum-Aggregation ist die Aggregation mit Hilfe des Minimums definiert, bei der die globale Ähnlichkeit durch die kleinste lokale Ähnlichkeit bestimmt wird (siehe Formel 5.30). Es wird eine konjunktive globale Ähnlichkeit gebildet, wobei die globale Ähnlichkeit von der höchsten lokalen Ähnlichkeit mit entsprechender Gewichtung bestimmt wird. Diese Aggregation hat den Vorteil, dass unabhängig von den lokalen Ähnlichkeitswerten lediglich der niedrigste Wert einen Einfluss auf die globale Ähnlichkeit hat.

$$\pi(sim_1, ..., sim_n, \vec{\omega}) = \min_{i=1}^{n}\{\omega_i \cdot sim_i\} \tag{5.30}$$

Auch die Minimum-Aggregation kann zur Aggregation mittels des k-Minimums verallgemeinert werden ($k = 1$). Die globale Ähnlichkeit wird von der k-kleinsten lokalen Ähnlichkeit bestimmt (siehe Formel 5.31). Die k-Minimum Aggregation erlaubt analog zur k-Maximum-Aggregation den Ausgleich von einzelnen Ausreißern der lokalen Ähnlichkeit nach unten.

$$\begin{aligned} \pi(sim_1, ..., sim_n, \vec{\omega}) = kmin_{i=1}^{n}\{\vec{\omega} \cdot sim_i\} = \omega_{i_k} \cdot sim_{i_k} \ , \\ mit: \ \omega_{i_k} \cdot sim_{i_k} \leq \omega_{i_{k+1}} \cdot sim_{i_{k+1}} \end{aligned} \tag{5.31}$$

Eine oft verwendete Aggregation ist die mit Hilfe der gewichteten Summe wie in Formel 5.32 zu sehen. Der Anteil, den eine lokale Ähnlichkeit an der globalen Ähnlichkeit

hat, wird durch das zugewiesene Gewicht bestimmt. Diese Form der Aggregation hat den Vorteil, dass sie sehr flexibel einsetzbar ist. Im Gegensatz zur Maximum- oder Minimum-Aggregation können alle lokalen Ähnlichkeiten Einfluss auf die globale Ähnlichkeit nehmen. Durch Setzen des Gewichts einer Ähnlichkeit auf 0 können jedoch auch explizit einige lokale Ähnlichkeiten ausgeschlossen werden. Andererseits werden u. U. niedrige Ähnlichkeitswerte mit berücksichtigt, was sich negativ auf die globale Ähnlichkeit auswirken kann. Dies kann man durch variable Gewichte verhindern, die beim Unterschreiten eines Grenzwerts automatisch auf 0 gesetzt werden.

$$\pi(sim_1, ..., sim_n, \vec{\omega}) = \sum_{i=1}^{n} \omega_i \cdot sim_i \tag{5.32}$$

Eine Verallgemeinerung der Aggregation mit Hilfe der gewichteten Summe (für $p = 1$) ist die Minkowski Aggregation (siehe Formel 5.33). Je größer bei dieser p gewählt wird, desto größer ist der Einfluss der größten Ähnlichkeit auf die globale Ähnlichkeit. Für $p \to \infty$ verhält sich die Minkowski Aggregation wie die Aggregation mit Hilfe des Maximums. Der Vorteil hier liegt darin, dass in Abhängigkeit von dem Parameter p hohe lokale Ähnlichkeitswerte eine stärkere Gewichtung erhalten. Daher ist ein Einsatz dann sinnvoll, wenn nur wenige hohe Werte als lokale Ähnlichkeiten vorliegen, da diese stärker berücksichtigt werden als mehrere niedrige Ähnlichkeiten. Ein weiterer Vorteil ist, dass diese Aggregation eine Verallgemeinerung der gewichteten Summe und der Maximum-Aggregation ist.

$$\pi(sim_1, ..., sim_n, \vec{\omega}) = (\sum_{i=1}^{n} \omega_i \cdot sim_i^p)^{\frac{1}{p}} \tag{5.33}$$

5.5.6. Systemfeedback

Eine wichtige Anforderung an die Kontext-abhängige Empfehlung war, dass das Empfehlungssystem dem Nutzer mitteilt, warum ein bestimmter Inhalt ausgewählt wurde. Die Berechnung der lokalen Teilergebnisse für die Inhalte kann als Grundlage dafür verwendet werden. Ein Ansatz besteht darin, alle errechneten Teilergebnisse dem Nutzer detailliert darzustellen. Da aber im Normalfall eine umfangreiche Menge an Attributtypen in die Berechnung einfließt, dürfte eine solche Ausgabe den Nutzer eher verwirren oder überfordern als ihm helfen. Eine bessere Methode ist es, dem Nutzer nur diejenigen Attributtypen und Attributwerte als Gründe für die Empfehlung anzugeben, die die Entscheidung am stärksten beeinflusst haben.

Da für die Bewertung von Inhalten ein Wertebereich zwischen $[0, 10]$ gewählt wurde (vgl. Abschnitt 4.1.3), soll der für einen Nutzer auf Basis des aggregierten Gesamtergebnisses errechnete prognostizierte Empfehlungswert denselben Wertebereich besitzen. Dies erleichtert die Vergleichbarkeit von tatsächlichen Bewertungen und berechneten

Empfehlungswerten. Da sowohl lokale Ähnlichkeitsfunktionen als auch die Aggregationsfunktionen einen Wert im Intervall $[0, 1]$ zurückliefern, erreicht man einen passenden prognostizierten Empfehlungswert durch Multiplikation des Gesamtergebnisses mit 10. Dem Nutzer werden die n Attributwerte zurückgeliefert, die den stärksten Einfluss auf die Berechnung der Gesamtähnlichkeit haben. Dies kann z. B. durch Betrachtung der Gewichte und der Ergebnisse der lokalen Ähnlichkeiten bestimmt werden. Der genaue Wert von n ist anwendungsabhängig.

5.5.7. Empfehlungen für mehrere Nutzer

Oft genug ist der Konsum von multimedialen Inhalten kein Einzel-, sondern ein Gruppenerlebnis. Von daher sollen die hier vorgestellten Konzepte auf mehr als einem Nutzerprofil arbeiten können. Das System muss also erkennen, welche Inhalte sich in der Schnittmenge aller Nutzerinteressen befinden. Das erreicht man, indem man erst den Ähnlichkeitswert eines Inhalts basierend auf jedem Nutzerprofil einzeln berechnet und anschließend den Durchschnitt der Einzelbewertungen ermittelt. Der oder die Inhalte mit dem größten Durchschnitt werden den Nutzern angezeigt.

5.6. Nutzerprofil-basierte Empfehlungen von multimedialen Inhalten

Im folgenden Abschnitt wird ein Ansatz für die Nutzerprofil-basierte Empfehlung von multimedialen Inhalten näher erläutert, der auf dem klassischen inhaltsbasierten Filterverfahren (vgl. Abschnitt 5.1.2) beruht [186]. Dieser Ansatz kann sowohl unabhängig von den zuvor diskutierten Konzepten verwendet werden, oder wie diese ein Teilergebnis berechnen, das dann mittels Aggregationsfunktion in ein Gesamtergebnis einfließt. Zunächst wird die grundsätzliche Idee dieses Ansatzes erläutert. Dann wird der generelle Ablauf der Empfehlung gezeigt, bevor die genauen Details näher untersucht werden. Abschließend wird erklärt, wie der prognostizierte Empfehlungswert zu einem Inhalt errechnet werden kann.

5.6.1. Grundidee

Klassische inhaltsbasierte Verfahren arbeiten auf einem *Objekt-zu-Objekt*-Vergleich. Um eine Vorhersage über einen prognostizierten Empfehlungswert für einen Inhalt treffen zu können, der das abgeschätzte Gefallen des Nutzers an einem unbekannten Objekt repräsentiert, muss dieses Objekt mit ähnlichen, bekannten Objekten im Profil und deren Bewertungen verglichen werden. Ein großes Problem bei diesem Ansatz ist der *Curse of Dimensionality*. Dieses Problem ergibt sich aus dem exponentiellen Anstieg des Volumens von mathematischen Räumen beim Hinzufügen von weiteren Attributen. Objekte in hochdimensionalen Räumen streuen in der Regel stark und sind somit nicht mehr

sinnvoll zu clustern [10]. Fügt man also bei der Beschreibung von multimedialen Inhalten weitere Attribute hinzu, ist es deutlich schwieriger einem unbekannten Objekt ein ähnliches zu zuordnen.

Ein weiteres Problem ist, dass die Qualität der Beschreibungen der Inhalte variieren kann, was vor Allem auf die Quantität der beschreibenden Attribute zurückzuführen ist (vgl. [117]). So erlaubt MASL zwar umfangreiche Beschreibungen multimedialer Inhalte, es kann aber nicht davon ausgegangen werden, dass diese tatsächlich in der Beschreibung eines konkreten Inhalts vorhanden sind. Dies führt dazu, dass die Objekte nicht gut miteinander verglichen werden können, da bei sehr spärlichen Beschreibungen kaum Überschneidungen der vorhandenen Attributwerte vorliegen. Außerdem können bei der klassischen inhaltsbasierten Filterung einzelne, u. U. sogar falsche oder irreführende Bewertungen einen zu großen Einfluss auf den gesamten Empfehlungsprozess haben, wenn sie zufällig dem unbekannten Inhalt ähnlich sind.

In dem hier vorgestellten Verfahren soll statt dem klassischen Objekt-zu-Objekt-Vergleich ein unbekannter Inhalt mit einem ganzen Nutzerprofil abgeglichen werden. Es wird also innerhalb des Profils statt nach einem einzelnen Objekt, das in allen oder zumindest möglichst vielen Attributen mit dem unbekannten Objekt identisch ist, nach *allen* Objekten gesucht, die zumindest *einen* Attributwert vom Attributtyp (z. B. `Action` beim Attributtyp `Genre`) mit dem unbekannten Objekt gemeinsam haben. Bei der Empfehlungsgenerierung werden die einzelnen Objekteigenschaften für einen Ähnlichkeitsvergleich verwendet, nicht ganze Objekte. Man kann davon ausgehen, dass im Laufe der Nutzungszeit des Systems eine große Anzahl von Inhalten bewertet wird und damit auch alle möglichen Attributtypen und ihre Werte nach einer gewissen Zeit geeignet abgedeckt sind, indem sie bewertet im Profil vorliegen. Diese Bewertungen kommen dadurch zustande, dass bei der expliziten und impliziten Bewertung eines Inhalts allen Attributen der zugehörigen Metadatenbeschreibung diese Bewertung zugeordnet wird (vgl. Abschnitt 4.2). Für jeden Attributwert eines Attributtyps, der in der Beschreibung eines unbekannten Objekts vorkommt, wird der aus allen Bewertungen dieses Werts im Profil gebildete Mittelwert betrachtet, der im Profil zusätzlich gespeichert wird (`AverageRating`). Darauf basierend wird der prognostizierte Empfehlungswert eines Inhalts berechnet.

Das Verfahren erlaubt es, sich ändernde Nutzerpräferenzen schnell zu erkennen und bei der Empfehlung zu berücksichtigen. Bei der Entwicklung des Ansatzes war ein weiteres Kriterium, dass nicht nur Einzelattribute berücksichtigt werden, sondern auch Kreuzkorrelationen zwischen Attributen. So kann bei der Empfehlungsgenerierung z. B. berücksichtigt werden, wenn ein Nutzer `Harrison Ford` in Filmen des `Genre` `Action` mag, nicht jedoch in `Comedies`. Das hier vorgestellte Verfahren wird im Folgenden als *Objekt-zu-Profil*-Vergleich oder als *Nutzerprofil-basierte Empfehlung* bezeichnet.

5.6.2. Grundsätzlicher Ablauf einer Empfehlungsgenerierung

Die Verarbeitung eines Inhalts c im vorgestellten Nutzerprofil-basierten Empfehlungsverfahren gliedert sich grob in die zwei sequentiellen Schritte:

1. Berechnung eines prognostizierten Empfehlungswerts $P(c)$ für den Inhalt c

2. Filterung des Inhalts anhand des prognostizierten Empfehlungswerts $P(c)$

Im ersten Schritt dieses Ansatzes wird zunächst für einen zu bewertenden Inhalt c ein numerischer prognostizierter Empfehlungswert $P(c)$ errechnet, der abschätzt, wie gut dieser Inhalt dem Nutzer gefallen wird. Da für die Bewertung von Inhalten ein Wertebereich zwischen $[0, 10]$ gewählt wurde (vgl. Abschnitt 4.1.3), soll dieser auch für den prognostizierten Empfehlungswert verwendet werden. Dies erleichtert die Vergleichbarkeit von tatsächlichen Bewertungen und berechneten Empfehlungswerten, was insbesondere für das Systemfeedback (vgl. unten) wichtig ist. Soll das System in Kombination mit den Ansätzen aus Abschnitt 5.5 nur ein Teilergebnis liefern, muss der hier berechnete Wert entsprechend durch 10 dividiert und dann an die Aggregationsfunktion übergeben werden.

Für die Berechnung des Empfehlungswerts $P(c)$ werden alle Attributwerte $a_i(j)$ aller relevanten Attributtypen a_i aus der Metadatenbeschreibung $md(c)$ von c einzeln betrachtet und überprüft, ob zu dem Wert eine Bewertung im Nutzerprofil $pd(u)$ des Nutzers u zur Verfügung steht. Ist dies der Fall, wird dieser Bewertungswert verwendet, um einen prognostizierten Empfehlungswert $P(a_i)$ für den Attributtyp a_i zu berechnen. Die Empfehlungswerte aller vorhandenen Attributtypen $P(a_i)$ ... $P(a_n)$ werden anschließend bei der Berechnung des prognostizierten Empfehlungswerts $P(c)$ für Inhalt c kombiniert. Dabei können sie je nach Inhaltsklasse unterschiedlich gewichtet in die Berechnung einfließen.

Basierend auf dem für den Inhalt errechneten prognostizierten Empfehlungswert $P(c)$ kann das Empfehlungssystem entscheiden, ob bzw. wie sehr ein Inhalt c für den Nutzer relevant ist. Je nach Anwendungsfall kann unterschiedlich mit dem berechneten Ergebnis umgegangen werden. Das Ziel ist keine binäre Klassifizierung der Inhalte in *empfehlenswert* und *nicht empfehlenswert*, sondern soll durch die einzelnen prognostizierten Empfehlungswerte eine feinere Einteilung erreicht werden. So kann man z. B. mehrere Inhalte anhand der prognostizierten Empfehlungswerte für die einzelnen Attributtypen a_i oder insgesamt vergleichen und dem Nutzer nur die mit den höchsten Werten, unabhängig wie hoch diese sind, empfehlen. Oder man gibt dem Nutzer eine nach ihren prognostizierten Empfehlungswerten sortierte Auflistung aller Inhalte aus oder alle Inhalte, deren prognostizierter Empfehlungswert einen gewissen Schwellwert übersteigt.

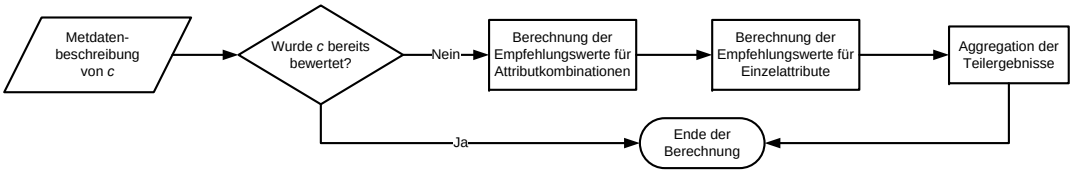

Abbildung 5.20.: Ablauf der Nutzerprofil-basierten Empfehlung

5.6.3. Berechnung des Empfehlungswerts für Inhalt c

Der prognostizierte Empfehlungswert $P(c)$ mit Metadatenbeschreibung $md(c)$ von c wird in folgenden Schritten berechnet (siehe auch Abbildung 5.20):

1. **Überprüfung, ob der Inhalt c schon bewertet wurde.** Für alle Inhalte, auf die der Nutzer schon einmal zugegriffen hat, besitzt das Nutzerprofil $pd(u)$ des Nutzers u bereits eine explizite oder implizite Bewertung. Von daher muss dieser Wert nicht extra berechnet werden.

2. **Berechnung des prognostizierten Empfehlungswerts für alle in $md(c)$ enthaltenen Attributkombinationen.** Eine wichtige Eigenschaft des Systems ist es, Kreuzkorrelationen bei der Berechnung von $P(c)$ zu berücksichtigen. Welche man dabei berücksichtigt, kann je nach Inhaltsklasse und Anwendungsfall durch die Konfigurationsschnittstelle spezifiziert werden.

3. **Berechnung des prognostizierten Empfehlungswerts für alle in $md(c)$ enthaltenen Einzel-Attribute.** Für jedes Attribut, das noch nicht in irgendeiner Kombination berücksichtigt wurde, wird ein einzelner prognostizierter Empfehlungswert berechnet. Wie diese Berechnung durchgeführt wird, ist abhängig davon wie die Struktur des Attributs ist, also ob es flach oder hierarchisch aufgebaut ist.

4. **Aggregation der Teilergebnisse**. Die zuvor berechneten Teilergebnisse werden zu einer Gesamtbewertung von c aggregiert.

Die einzelnen Schritte, die für diese Teilaufgaben benötigt werden, werden im Folgenden genauer erläutert.

Berechnung des prognostizierten Empfehlungswerts für Attributkombinationen

Um Kreuzkorrelationen bei der Berechnung eines Empfehlungswerts zu berücksichtigen, müssen geeignete Attributkombinationen berücksichtigt werden. Im Gegensatz zum klassischen Objekt-zu-Objekt-Abgleich ist es das Ziel, auch Querbeziehungen zwischen Attributwerten abzuleiten, wie z. B. dass ein Nutzer `Harrison Ford` als Schauspieler zwar in

Filmen des Genre `Action` mag, nicht jedoch in `Comedies`. Dadurch werden Empfehlungen deutlich passender, was die Nutzerzufriedenheit erhöht. Welche Attributkombinationen berücksichtigt werden, ist abhängig vom Anwendungsfall und kann durch die Konfigurationsschnittstelle angepasst werden. Zu den zuvor genannten relevanten Attributen (vgl. Abschnitt 5.4.1) sind z. B. folgende Kombinationen interessant:

- Kombination mehrerer Genre-Werte
- Kombination mehrerer Mitwirkender in unterschiedlichen Rollen
- Kombination von Genre und Mitwirkenden
- Kombination von Inhalteanbieter und Sendezeit
- Kombination von Genre und Sendezeit
- Kombination von Genre und Inhalteanbieter
- Kombination von Genre und Ort des Empfängers
- Kombination von Genre und Aktivität des Empfängers

Der prognostizierte Empfehlungswert $P(a_{k/i})$ einer Attributkombination der Attributtypen a_k und a_i kann mittels Formel 5.34 berechnet werden. $R(a_{k/i}(j))$ steht für die Bewertung dieser Kombination, wie sie im MASL-Profil im Element `TimeDependentAttributeCombinations` gespeichert ist und $cnt(a_{i/j}(m))$ für die Anzahl aller Inhalte, die diese Kombination enthalten. Berücksichtigt werden nur die Kombinationen, die schon bewertet wurden, also wo $cnt(a_{i/j}(m)) > 0$. Durch die Berücksichtigung der Anzahl der Inhalte mit dieser Kombination, bekommen oft bewertete Attributkombinationen einen stärkeren Einfluss.

$$P(a_{k/i}) = \frac{\sum_{j=1}^{n} R(a_{k/i}(j)) * cnt(a_{k/i}(j))}{\sum_{j=1}^{n} cnt(a_{k/i}(j))} \tag{5.34}$$

Für den Fall, dass eines oder beide der kombinierten Attributtypen in `main`, `secondary` und `other` unterteilt wird (z. B. `Keyword`), werden die entsprechenden Werte mit unterschiedlichem Gewicht $class_{a_k}$ in Formel 5.34 berücksichtigt und diese zu Formel 5.35 modifiziert. Die Gewichte sind wie in Formel 5.36 definiert und können in Abhängigkeit des Anwendungsfalls modifiziert werden.

$$P(a_{k/i}) = \frac{\sum_{j=1}^{n} R(a_{k/i}(j)) * cnt(a_{k/i}(j)) * class_{a_k} * class_{a_i}}{\sum_{j=1}^{n} cnt(a_{k/i}(j)) * class_{a_k} * class_{a_i}} \tag{5.35}$$

$$class_{a_i} = \begin{cases} 1, & falls\, a_i\, main \\ 0.5, & falls\, a_i\, secondary; \\ 0.3, & falls\, a_i\, other \end{cases} \tag{5.36}$$

Berechnung des prognostizierten Empfehlungswerts für Einzelattribute ohne hierarchische Struktur

Einige Attribute sind flach strukturiert, und die Werte können nicht hierarchisch klassifiziert werden (vgl. Abschnitt 5.4.1). Beispiele hierfür sind die Attributtypen `Keyword` oder der Anbieter (`Provider`) eines Inhalts. Der prognostizierte Empfehlungswert $P(a_i)$ dieser Attributtypen wird mittels Formel 5.37 berechnet. `totalCount` bezieht sich auf die Gesamtanzahl aller bewerteten Inhalte im Nutzerprofil. Das Verhältnis zwischen der Anzahl der Vorkommen $cnt(a_i(m))$ von Attribut $a_i(m)$ wird durch α modifiziert, eine konfigurierbare Konstante nahe 0. Dadurch bleibt der Faktor etwas unter 1, was dessen Einfluss auf den Term einschränkt. Mittels durchgeführter Nutzertests wurde ein Wert um die 0.01 als geeignet identifiziert. Wiederum ist bei dieser Formel der Einfluss eines Attributwerts umso größer, je öfter er bewertet wurde.

$$P(a_i) = \frac{\sum_{j=1}^{n} R(a_i(j)) * cnt(a_i(j)) * \left(\frac{cnt(a_i(j))}{totalCount}\right)^{\alpha}}{\sum_{j=1}^{n} cnt(a_i(j))} \tag{5.37}$$

Auch hier muss man wieder unterscheiden, ob der Attributtyp a_i in `main`, `secondary` und `other` unterteilt werden kann. Dadurch verändert sich Formel 5.37 wie folgt ($class_{a_i}$ definiert wie in Formel 5.36):

$$P(a_i) = \frac{\sum_{j=1}^{n} R(a_i(j)) * cnt(a_i(j)) * class_{a_i} * \left(\frac{cnt(a_i(j))}{totalCount}\right)^{\alpha}}{\sum_{j=1}^{n} cnt(a_i(j)) * class_{a_i}} \tag{5.38}$$

Einen Sonderfall nimmt das Attribut `Creator` ein, da dieselbe Person in unterschiedlichen Rollen im Profil vorliegen kann. So ist z. B. `George Lucas` sowohl `Director` als auch `Scriptwriter` des Films aus Listing B.1 im Anhang B. Dadurch kann es vorkommen, dass eine Person zwar schon bewertet wurde, jedoch noch nicht in einer speziellen Rolle. Dies soll im Empfehlungsprozess berücksichtigt werden. In MASL werden Präferenzen für Personen in einer bestimmten Rolle als Attributkombination gespeichert. Es wird ein Gewicht *role* eingeführt, das die Relevanz reduziert, wenn eine Person zwar bewertet

wurde, aber noch nicht in der entsprechenden Rolle. Der Wert ist wie in Formel 5.39 definiert und kann je nach Anwendungsfall angepasst werden.

$$role = \begin{cases} 1, & falls\, Wertung\, der\, Person\, in\, passender\, Rolle\, vorliegt \\ 0.4, & andernfalls \end{cases} \tag{5.39}$$

Damit wird Formel 5.37 wie folgt modifiziert:

$$P(a_{Creator}) = \frac{\sum_{j=1}^{n} R(a_{Creator}(j)) * cnt(a_{Creator}(j)) * role * \left(\frac{cnt(a_{Creator}(j))}{totalCount}\right)^{\alpha}}{\sum_{j=1}^{n} cnt(a_{Creator}(j)) * role} \tag{5.40}$$

Berechnung des prognostizierten Empfehlungswerts für Einzelattribute mit hierarchischer Struktur

Einige Attributtypen sind hierarchisch strukturiert, d.h. ihre Werte können in unterschiedlichem Abstraktionsniveau gegeben sein. Ein Beispiel ist der Attributtyp `Genre`, der abstrakte Werte wie `Sports` erlaubt und deutlich konkretere wie `Football`, `Golf` oder `Tennis`, die alle Subtypen des Werts `Sports` sind. Um dieser Tatsache Rechnung zu tragen, wurde die Formel 5.37 zu Formel 5.41 modifiziert. Dazu wurde die Leveldifferenz l eingeführt, definiert wie in Formel 5.42. Sie gibt die Differenz zwischen dem Abstraktionslevel $lev(a_i(j))$ des in der Beschreibung vorkommenden Attributwerts $a_i(j)$ und dem Abstraktionslevel $lev(x)$ des ersten bewerteten Vorfahren x von $a_i(j)$ an, wenn $a_i(j)$ selbst nicht bewertet wurde. Je größer der Unterschied dieser Abstraktionslevel, desto geringer ist der Einfluss von l auf die Gesamtberechnung.

$$P(a_i) = \frac{\sum_{j=1}^{n} R(a_i(j)) * cnt(a_i(j)) * l(a_i(j)) * \left(\frac{cnt(a_i(j))}{totalCount}\right)^{\alpha}}{\sum_{j=1}^{n} cnt(a_i(j)) * l(a_i(j))} \tag{5.41}$$

$$l(a_i(j)) = \begin{cases} 1, & falls\ Level(a_i(j)) = Level(x); \\ \frac{0.5}{|(Level(a_i(j)) - Level(x))|}, & falls\ Level(a_i(j)) \neq Level(x). \end{cases} \tag{5.42}$$

Handelt es sich um einen Attributtyp a_i, der in `main`, `secondary` und `other` unterteilt werden kann, muss die Formel 5.41 wie folgt angepasst werden:

$$P(a_i) = \frac{\sum_{j=1}^{n} R(a_i(j)) * cnt(a_i(j)) * l(a_i(j)) * class_{a_i} * \left(\frac{cnt(a_i(j))}{totalCount}\right)^{\alpha}}{\sum_{j=1}^{n} cnt(a_i(j)) * l(a_i(j)) * class_{a_i}} \tag{5.43}$$

Berechnung des prognostizierten Empfehlungswerts für den Inhalt

Wurden die oben genannten Schritte durchgeführt, also alle Empfehlungswerte für die Attributtypen und Attributkombinationen bestimmt, kann daraus mittels Formel 5.44 der Empfehlungswert $P(c)$ von Inhalt c errechnet werden. $w(a_i)$ bezieht sich auf das Gewicht, das einem Attributtyp bzw. einer Attributkombination zugeordnet ist. Dieses wird für jede Inhaltsklasse und je nach Anwendungsfall festgelegt (siehe Abschnitt 5.4.2). Wichtig ist, dass das Ergebnis im Wertebereich zwischen $[0, 10]$ liegt. Das erreicht man, indem man durch die Summe aller Gewichte statt nur durch die Anzahl teilt.

$$P(c) = \frac{\sum_{i=1}^{n} P(a_i) * w(a_i)}{\sum_{i=1}^{n} w(a_i)} \tag{5.44}$$

Caching von Zwischenergebnissen

Werden in einem System die prognostizierten Empfehlungswerte für mehrere Inhalte hintereinander errechnet, ist es sinnvoll, Zwischenergebnisse im System geeignet abzuspeichern und wiederzuverwenden. Ähnlich wird mit den Bewertungen von Attributwerten bzw. von Attributkombinationen verfahren, die im Profil dauerhaft abgespeichert werden. Diese werden dann neu berechnet, wenn ein neuer Inhalt mit den entsprechenden Attributwerten in das System eingebracht wird. Dadurch, dass diese Berechnung asynchron durchgeführt wird, kann die eigentliche Berechnung des prognostizierten Empfehlungswerts deutlich schneller erfolgen.

Systemfeedback

Um dem Nutzer mitteilen zu können, warum ein bestimmter Inhalt gewählt wurde, wird während des Berechnungsprozesses des Empfehlungswerts für jeden Attributtyp bestimmt, welcher Wert den stärksten Einfluss hatte. Dies ergibt sich für Attributtypen ohne Hierarchie unter Berücksichtigung von Formel 5.37, wie in Formel 5.45 zu sehen ist (analog die anderen betrachteten Fälle). Der Einfluss eines Attributtyps wird somit durch

Abbildung 5.21.: Beispiel für ein dem Nutzer angezeigtes Systemfeedback

die Wertung und die Anzahl der Einzelwertungen jedes Attributwerts festgelegt. Nach der Berechnung aller Empfehlungswerte werden die n Attributwerte mit dem größten Einfluss ermittelt, wobei n sich aus dem Anwendungsfall ergibt. Diese werden dem Nutzer dann geeignet präsentiert. Zusätzlich kann es je nach Anwendungsfall noch sinnvoll sein, dem Nutzer ergänzende Informationen zu einem Inhalt anzuzeigen. Eine mögliche Ausgabe ist in Abbildung 5.21 zu sehen.

$$eff(a_i) = R(a_i(j)) * cnt(a_i(j)) * \left(\frac{cnt(a_i(j))}{totalCount}\right)^{\alpha} \tag{5.45}$$

5.6.4. Filterung der Ergebnisse

Basierend auf dem für den Inhalt errechneten prognostizierten Empfehlungswert $P(c)$ kann das Empfehlungssystem entscheiden, ob bzw. wie sehr ein Inhalt c für den Nutzer relevant ist. Je nach Anwendungsfall kann unterschiedlich mit dem berechneten Ergebnis umgegangen werden.

Eine Möglichkeit ist es z. B., dem Nutzer nur den am höchsten bewerteten Inhalt zur Verfügung zu stellen. Dies bietet sich für einen Video-Empfehlungsdienst an. In Anwendungen wie einem Touristenführer oder einem Nachrichtendienst ist es sinnvoller, dem Nutzer eine Auswahl an relevanten Inhalten anzuzeigen, um ihn dann selbstständig aus dieser aussuchen zu lassen. Die angezeigten Inhalte sollen zumindest einen vorher festgelegten Schwellwert übersteigen.

5.7. Empfehlung strukturierter Inhalte

Eine besondere Herausforderung ist die personalisierte Auswahl strukturierter Daten, also von Inhalten und ihren zugehörigen Annotationen. Einen Kommentar unabhängig von dem kommentierten Inhalt zu empfehlen macht keinen Sinn, da ein Kommentar meist nur im Kontext dieses Inhalts gültig ist. Deshalb soll die Struktur zwischen Inhalt und Kommentar, sowie zwischen den Kommentaren untereinander, bei der Berechnung

der Empfehlungen berücksichtigt werden. Dieser Ansatz kann dabei autark oder auch in Kombination mit den zuvor vorgestellten Konzepten eingesetzt werden.

Da es sich bei Kommentaren meist um Nutzer-generierte Inhalte handelt, ist nicht anzunehmen, dass diese auch umfangreich beschrieben sind. In Abschnitt 4.1 wurden zwar Ansätze beschrieben, wie man den Nutzer bei der Vergabe von Schlüsselwörtern unterstützen bzw. diese automatisch ableiten kann, dennoch sind Kommentare meist schnell erzeugt und werden nicht umfangreich verschlagwortet. Die Empfehlung von Kommentaren basiert auf kollaborativem Filtern (siehe Abschnitt 5.1.3), genauer gesagt auf dem *nearest-neighbor collaborative Filtering* wie von Herlocker et al. vorgestellt [79], da es trotz seiner Einfachheit gute Empfehlungen liefert und auf einem Ähnlichkeitsvergleich der Nutzer basiert, was insbesondere für Community-Dienste interessant ist. Prinzipiell kann der hier vorgestellte Algorithmus auch mit inhaltsbasierten Verfahren verwendet werden.

Im Folgenden wird der Ansatz für kollaboratives Filtern in strukturierten Daten näher erläutert. Außerdem werden einige Ansätze diskutiert, die diese Art des Filterns noch verbessern können.

5.7.1. Kollaboratives Filtern in strukturierten Daten

Wie bereits beschrieben (vgl. Abschnitt 5.5.4) besteht das *nearest-neighbor collaboraive Filtering* aus folgenden drei Schritten:

1. Vergleich aller Nutzer U im Bezug auf ihre Ähnlichkeit mit dem aktiven Nutzer u
2. Auswahl einer Untermenge U_u aus U, die als Menge zur Vorhersage für ein gegebenes Objekt j verwendet wird. U_u wird auch als Menge der nächsten Nachbarn des aktiven Nutzers u bezeichnet.
3. Berechnung des prognostizierten Empfehlungswerts $P_u(c)$ für ein gegebenes Objekt c basierend auf den Bewertungen der nächsten Nachbarn von Nutzer u.

Nutzer und ihre Bewertungen werden durch eine $n \times m$ Matrix M repräsentiert. Für jeden dieser drei Schritte steht eine Vielzahl an Algorithmen zur Verfügung, die verwendet werden können (siehe Abschnitt 5.5.4). Bei dem hier vorgestellten Ansatz wird die Strukturierung der Daten nicht in diesen drei Schritten berücksichtigt, sondern ein zusätzlicher Algorithmus entwickelt, der entscheidet, für welche Objekte Empfehlungswerte berechnet werden sollen.

5.7.2. Berücksichtigung der Struktur der Daten

Um strukturerhaltende Empfehlungen auf Daten zu ermöglichen, wurde der Algorithmus entwickelt, der schematisch in Abbildung 5.22 zu sehen ist. Alle zuvor vorgestellten Ansätze betrachten Inhalte als einzelne Einträge in der Nutzer-Inhalts-Matrix, mögliche zugehörige Kommentare werden nicht berücksichtigt. Im hier betrachteten Fall sind Inhalte

und zugehörige Kommentare zusätzlich miteinander verlinkt. In einer Art Offline-Phase werden die Abstände zwischen den Nutzern berechnet. Dabei werden Inhalte und Kommentare berücksichtigt, um zu spärliche Bewertungen zu vermeiden. Darauf basierend werden wie beschrieben die nächsten Nachbarn bestimmt. Diese Schritte werden periodisch wiederholt um evtl. Änderungen in den Profilbeschreibungen berücksichtigen zu können. Darauf basierend wird der prognostizierte Empfehlungswert wie oben beschrieben berechnet. Wenn der Wert für einen Inhalt den Schwellwert übersteigt und dieser dem Nutzer empfohlen werden soll, werden in einem nächsten Schritt alle Kommentare auf erster Ebene betrachtet (1). Übersteigen deren Empfehlungswerte ebenfalls den Schwellwert, werden auch deren Kommentare weiter berücksichtigt (2). Übersteigt hingegen ein Empfehlungswert nicht den Schwellwert, kann es dennoch in einem tieferen Level in der Kommentarhierarchie einen für den Nutzer interessanten Kommentar geben (3). Deswegen werden diese Inhalte und deren Nachfolger in der Kommentarhierarchie trotzdem berücksichtigt. Der zugehörige Ast wird allerdings markiert. Übersteigt ein weiterer Inhalt in einem markierten Ast nicht den Schwellwert, wird der zugehörige Ast abgeschnitten (4). Auf diese Weise werden Empfehlungen von zusammenhängenden Inhalten und Kommentaren ermöglicht, die nicht durch zu viele für den Nutzer uninteressante Inhalte unterbrochen werden. Wenn alle Knoten in der Hierarchie überprüft wurden oder sich in abgeschnittenen Zweigen befinden, können die entsprechenden Knoten dem Nutzer empfohlen werden. Alle Knoten, die selber nicht den Schwellwert überstiegen, aber Nachfolger von Interesse haben, werden dem Nutzer als Hinweis präsentiert.

5.7.3. Verbesserung der Empfehlungsqualität

Die Qualität von Empfehlungen steht in starkem Zusammenhang zur Nutzerzufriedenheit. Wenn die Empfehlung von guter Qualität ist, und das auch noch ohne großen Nutzereinsatz geschieht, steigt automatisch die Nutzerzufriedenheit. Gerade für ein Communitybasiertes System ist diese wichtig, um sich gegenüber den Mitbewerbern abzugrenzen. Außerdem ist bei der großen Menge an Kommentaren eine hohe Empfehlungsqualität entscheidend. Deswegen werden im Folgenden einige Ansätze diskutiert, die sich im hier vorgestellten Szenario für strukturierte Daten einsetzen lassen.

Recursive Prediction

Wenn die Nutzer-Inhaltsmatrix M nur spärlich besetzt ist, hat das einen sehr kritischen Einfluss auf die Empfehlungsqualität, da die Wahrscheinlichkeit, einen ähnlichen Nutzer zu finden, der auch noch den entsprechenden Inhalt bewertet hat, mit einer steigenden Anzahl an Inhalten sinkt. Deswegen kann es passieren, dass die Mehrheit der ähnlichen Nutzer nicht als nächste Nachbarn gewählt werden können, bzw. dass die ausgewählten Nachbarn dem Nutzer nicht sehr ähnlich sind. Dieses Problem verstärkt sich extrem, wenn man strukturierte Daten betrachtet, da mit der wachsenden Tiefe der Kommentar-

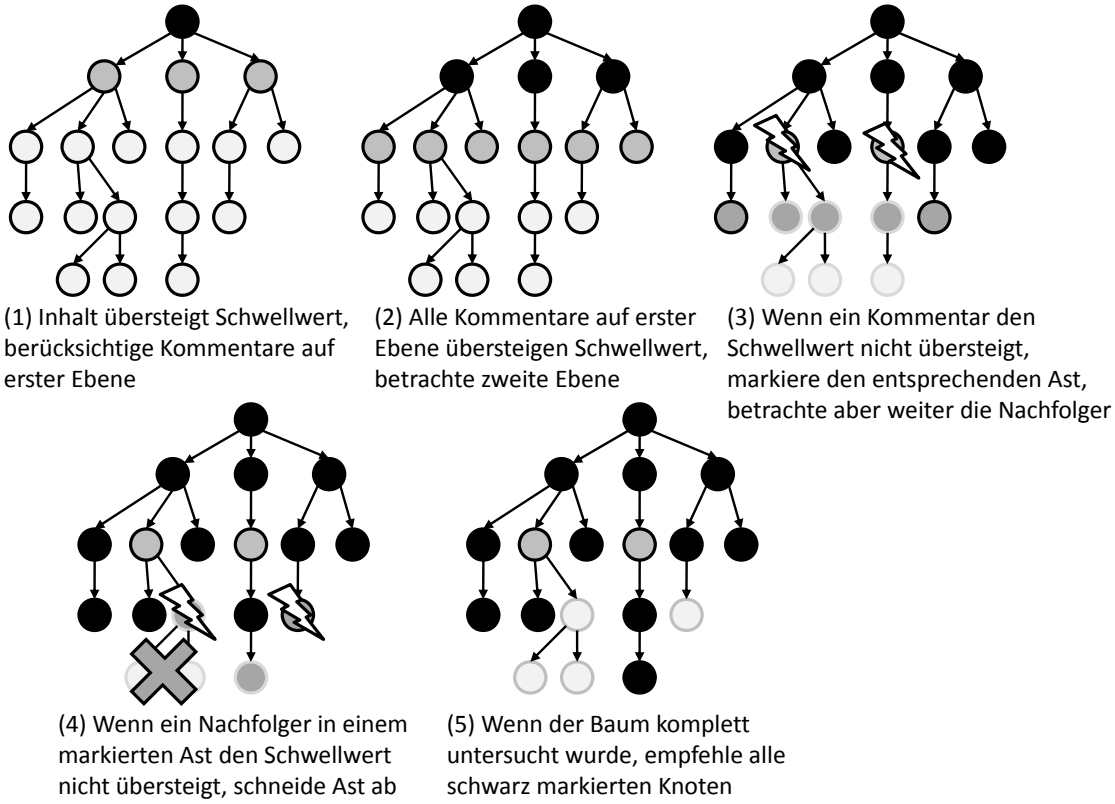

Abbildung 5.22.: Struktur-erhaltender Empfehlungsalgorithmus. Knoten, die empfohlen werden, sind schwarz markiert, zu überprüfende dunkelgrau, alle anderen hellgrau. Knoten von markierten Ästen haben eine hellere Umrandung.

hierarchie Kommentare zunehmend weniger bewertet sind.

Ein Ansatz, der versucht, dieses Problem zu lösen, ist der so genannte *Recursive Prediction Algorithm* [191]. Dieser erlaubt, ähnliche Nutzer in der Berechnung zu berücksichtigen, auch wenn sie nicht den betrachteten Inhalt bewertet haben. Dies erreicht man dadurch, dass rekursiv für alle fehlenden Bewertungen der ähnlichen Nutzer prognostizierte Empfehlungswerte errechnet werden und diese bei der Berechnung des Empfehlungswerts für den aktiven Nutzer berücksichtigt werden. Die Grundidee dieses Ansatzes ist in Abbildung 5.23 zu sehen. Wenn der dem aktiven Nutzer Si ähnliche Nutzer Jo den betrachteten Inhalt noch nicht bewertet hat, wird diese Bewertung rekursiv durch Jo's nächsten Nachbarn Ma berechnet. Dadurch kann Jo als nächster Nachbar bei der Empfehlungsgenerierung für Nutzer Si betrachtet werden.

Erste Experimente zeigen, dass der *Recursive Prediction Algorithm* die Genauigkeit der

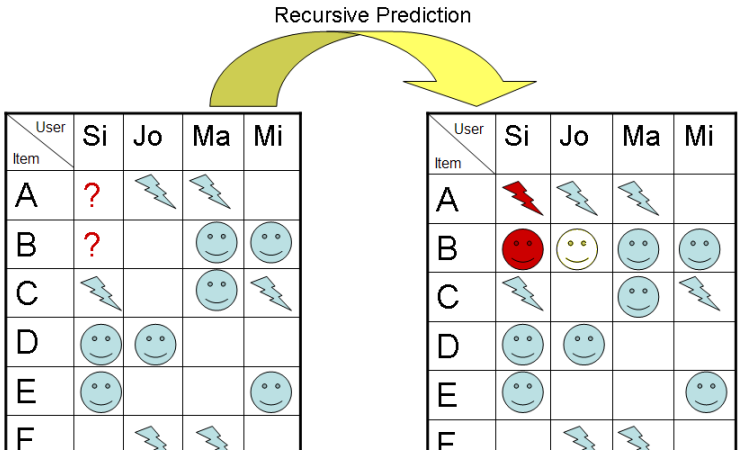

Abbildung 5.23.: Die linke Nutzer-Inhalt Matrix zeigt alle verfügbaren Bewertungen. Der prognostizierte Empfehlungswert von Inhalt A und B soll für Nutzer Si berechnet werden. Ein ähnlicher Nutzer von Si ist Jo. Da Jo Inhalt B noch nicht bewertet hat, wird diese Bewertung mit dem Recursive Prediction Algorithm prognostiziert. Basierend auf Jo's vorhandenen (A) und berechneten (B) Bewertungen, können die Empfehlungen für Si erzeugt werden (nach [191])

Empfehlung entscheidend verbessern kann [191]. Der Algorithmus benötigt jedoch einen hohen Berechnungsaufwand. Dennoch ist er eine vernünftige Erweiterung des Empfehlungsprozesses für strukturierte Daten.

Example Critiquing

Eine wichtige Anforderung für Empfehlungssysteme besteht darin, Inhalte zur Verfügung zu stellen, die für den Nutzer von Interesse sind. Dennoch wurde durch Studien gezeigt, dass die Nutzer viele ihrer Präferenzen für einzelne Attribute der Inhalte erst während des Suchprozesses finden. So genannte *Critiquing*-Techniken versuchen dieser Tatsache Rechnung zu tragen. Ein Beispiel hierfür ist der so genannte *Example Critiquing*-Ansatz, der zusätzlich zu den Empfehlungen dem Nutzer Beispiele zur Verfügung stellt. Das Ziel ist, dem Nutzer ein breiteres Bild von den im System verfügbaren Inhalten zu geben. Dadurch können ungeeignete und ältere Bewertungen ausgeglichen und die Nutzer-Inhalte-Matrix schneller gefüllt werden. Example Critiquing erzielt die besten Ergebnisse, wenn es in Kombination mit dem *lookahead principle* verwendet wird [172]. Dieses besagt, dass die dem Nutzer gezeigten Beispiele nicht gemäß dem aktuellen Nutzerprofil optimal sein sollen, sondern es soll eine möglichst hohe Optimalität aufweisen für den Fall, dass weitere Informationen dem Nutzerprofil hinzugefügt werden. Durch

das lookahead principle wird der so genannte *anchoring effect* vermieden, der besagt, dass der Nutzer in einem lokalen Optimum stecken bleibt und keine Möglichkeit mehr zur Verbesserung sieht. Dieser Ansatz eignet sich ausgezeichnet für die Empfehlung von strukturierten Daten und wird deswegen in diesen Ansatz integriert.

Um Beispielempfehlungen unter Verwendung des *lookahead principle* zu erzeugen, verwendet man am besten die *Pareto Optimality*. Diese vermeidet numerische Ungenauigkeiten, da sie unabhängig von numerischen Werten berechnet werden kann. Man bezeichnet eine Option als Pareto optimal, wenn sie nicht durch eine andere Option (Pareto) dominiert wird. Eine Option O ist (Pareto) dominiert durch eine andere Option o, wenn

- unter Berücksichtigung aller bekannter Präferenzen des Nutzermodells $\mathbf{P}$ o nicht schlechter ist als O: $\forall c_i \in \mathbf{P} : c_i(o) \leq c_i(O)$
  wobei $c_i(o)$ die Kostenfunktion von Attribut i für Option o ist und
  ihr Wert niedriger ist, je mehr der i präferiert

- o strikt besser ist als O für mindestens eine Präferenz
  $\exists c_i \in \mathbf{P} : c_i(o) < c_i(O)$

Nutzerfeedback

Die Verwendung von zusätzlichen Feedback-Mechanismen führt zu einem weiteren Anstieg in der Qualität der Empfehlungen, da das System und das Nutzerprofil zusätzliche Informationen erhalten. Eine Methode, dieses Feedback zu integrieren, ist das so genannte *Relevance Feedback*, das es dem Nutzer erlaubt, erzeugte Empfehlungen und die Beispiele gegeben durch das Example Critiquing in Bezug auf ihre Relevanz zu bewerten. Dadurch können Empfehlungen und Profile Schritt für Schritt verfeinert werden [122]. Durch Relevance Feedback kann der Nutzer einen aktiven Anteil am Empfehlungsprozess nehmen, indem er interaktiv seine Präferenzen erneuert und verfeinert. Ein Ziel ist es, die optimale Empfehlung für einen Nutzer in möglichst wenigen Iterationen zu finden. Relevance Feedback ist eine wichtige Methode im Bereich der Bildabfrage. Die meisten Ansätze verwenden eine Query-Point-Methode, die die vom Nutzer als relevant eingestuften Empfehlungen verwendet, um optimale Empfehlungen zu erzeugen [111]. Andere Ansätze verwenden eine Anpassung der Gewichtung und passen die Gewichte von einzelnen Attributen während des Suchprozesses an, um die besten Empfehlungen zu finden [178].

Für die Empfehlung von strukturierten Daten wird Relevance Feedback basierend auf dem Ansatz von Middleton et al. [122] verwendet. Der Nutzer kann dabei erzeugte Empfehlungen mittels der Optionen *optimal passend*, *passend*, *nicht passend* und – wenn die Empfehlung als Antwort auf eine explizite Schlüsselwortsuche erzeugt wurde – *passend zu <keyword>* bewerten.

5.8. Framework zur Kontext-abhängigen Empfehlung von Inhalten

Im Rahmen dieser Arbeit wird ein Framework zur Kontext-abhängigen Personalisierung entwickelt, das im Folgenden näher vorgestellt wird [183]. Dieses integriert alle hier vorgestellten Ansätze und erlaubt ebenso die kombinierte Nutzung verschiedener Verfahren. Ziel der Entwicklung des Frameworks ist es, ein möglichst allgemeines System zu schaffen, das je nach Anwendungsfall konfiguriert und angepasst werden kann. Dies beinhaltet die Festlegung, wo die Profildaten des Nutzers liegen, welche Kontextinformationen bei der Berechnung der Empfehlung berücksichtigt werden, wie diese miteinander verglichen werden und wie die Einzelergebnisse aggregiert werden.

Zunächst wird die Grundidee des Frameworks näher erläutert und der Aufbau des Frameworks erklärt. Ein besonderer Schwerpunkt wird dabei auf den verteilten Abgleich der Kontextinformationen gelegt. Anschließend wird gezeigt, wie das Framework konfiguriert werden kann.

5.8.1. Überblick

Damit Empfehlungen für einen Nutzer generiert werden, müssen die Beschreibungen der Inhalte und Nutzer im System hinterlegt sein. Beschreibungen der Inhalte sind insbesondere bei inhaltsbasierten Verfahren notwendig, da diese Algorithmen direkt auf den Attributen der Beschreibungen arbeiten. Umfangreiche Beschreibungen werden bei kollaborativen Ansätzen nicht benötigt, jedoch müssen die Inhalte zumindest einen eindeutigen Bezeichner besitzen, der auch vom System zu verwalten ist. Unter Berücksichtigung der immensen Menge an Inhalten und der zugehörigen Metadaten, die sehr umfangreich sein können, macht eine Speicherung dieser Informationen auf dem Client wenig Sinn. So nehmen bereits die Metainformationen bei Flickr[1] einen oberen dreistelligen Gigabyte-Bereich ein, was die Speicherkapazität mobiler Endgeräte bei Weitem übersteigt.

Das Nutzerprofil kann hingegen sowohl auf dem Server als auch auf dem Client gespeichert werden. Für eine Speicherung auf dem Server spricht, dass hier eine deutlich umfangreichere Rechenleistung vorhanden ist. Außerdem können so kollaborative Filteransätze eingesetzt werden, die auf den zentral gespeicherten Nutzerprofilen mehrerer Nutzer basieren. Weiterhin erlaubt die Server-seitige Speicherung die einfachere Unterstützung von proaktiven Diensten. Dafür müssen aber permanent die Kontextinformationen überwacht und Änderungen an den Server kommuniziert werden, was insbesondere bei hoch dynamischen Daten wie dem aktuellen Aufenthaltsort oder die den Nutzer umgebenden Personen, die meist auf dem Client erfasst werden, zu einer hohen Belastung der Luftschnittstelle führt. Dieses Problem kann man mit der Speicherung der Profildaten auf dem Client umgehen, da diese Daten auf dem sie erfassenden Gerät verarbeitet

[1] www.flickr.com

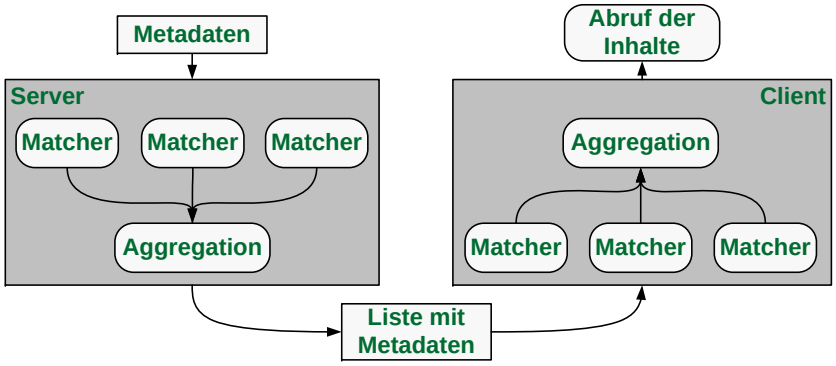

Abbildung 5.24.: Schematische Darstellung des Frameworks zur Kontext-abhängigen Empfehlung von Inhalten.

werden. Weiterhin handelt es sich gerade bei dynamischen Kontextinformationen um sehr sensible Nutzerdaten, die der Nutzer u. U. nicht weitergeben will. Durch die Speicherung des Profils auf dem Client bleibt die Privatsphäre des Nutzers maximal gewahrt.

Um einen möglichst guten Kompromiss zwischen den jeweiligen Vor- und Nachteilen zu finden, wird für das hier vorgestellte Framework eine verteilte Speicherung des Profils vorgeschlagen: die statischen Kontextinformationen sollen sich eher auf Seite des Servers, die dynamischen eher auf Seite des Clients befinden. Je nach Anwendungsfall kann die genaue Aufteilung der Informationen konfiguriert werden. Wichtig ist nur, dass eine eindeutige Identifikation des Nutzers möglich bleibt.

Die verteilte Speicherung der Profilinformationen führt dazu, dass auch die Auswertung der Daten bei der Erzeugung einer Empfehlung zwischen Server und Client aufgeteilt werden muss (vgl. Abbildung 5.24). Der Server greift auf eine Menge an verfügbaren Metadatenbeschreibungen für die Inhalte zu. Mit Hilfe von Server-seitigen Matcher-Komponenten werden diese mit dem auf dem Server gespeicherten Teil des Profils abgeglichen und pro Matcher ein Teilergebnis erzeugt. Beispiel für einen solchen Matcher ist z. B. der Nutzerprofil-basierte Ansatz oder ein Orts-Matcher (siehe Abschnitt 5.6 bzw. Abschnitte 5.5.3 und 5.5.4). Eine Aggregator-Komponente (siehe Abschnitt 5.5.5) fasst die Teilergebnisse zu einem Einzelergebnis zusammen und wählt darauf basierend eine Menge an Metadaten von potenziell für den Nutzer interessanten Inhalten aus. Diese wird an den Client übertragen, wo weitere Matcher diese Beschreibungen mit dem Client-seitigen Profil abgleichen. Die einzelnen Ergebnisse werden wiederum aggregiert und der Client kann den oder die relevanten Inhalte direkt vom Inhalteanbieter beziehen ohne nochmals die Server-Komponente involvieren zu müssen.

Die Auswahl der zuständigen Matcher, die zu verwendende Aggregatfunktion und die Anzahl der zu beziehenden Inhalte kann je nach Anwendungsfall beliebig konfiguriert werden (siehe Abschnitt 5.8.3).

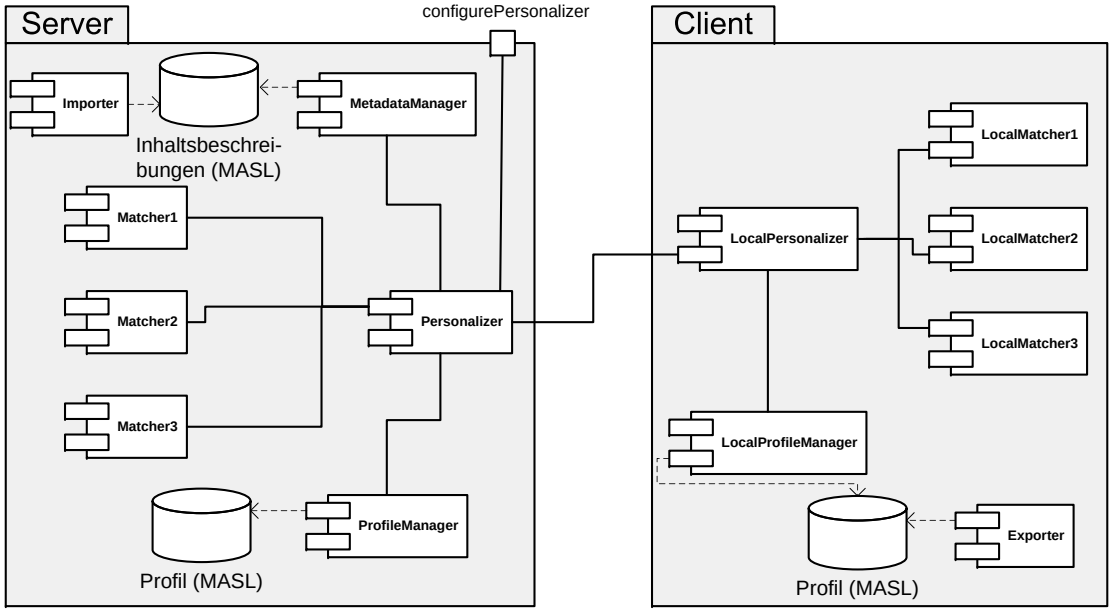

Abbildung 5.25.: Komponenten des Frameworks zur Kontext-abhängigen Empfehlung von Inhalten.

5.8.2. Aufbau des Frameworks

Wie beschrieben setzt sich das Framework aus einem Server–seitigen und einem Client-seitigen Teil zusammen, die beide einen Teil des Nutzerprofils verwalten und die Berechnung der Empfehlungen für einen Nutzer verteilt durchführen. Ein Überblick über alle beteiligten Komponenten ist in Abbildung 5.25 zu sehen.

Das Profil wird Server-seitig von der `ProfileManager`-Komponente verwaltet. Eine analoge `LocalProfileManager`-Komponente befindet sich auf dem Client. Mit Hilfe dieser Komponenten kann der Nutzer seine Präferenzen und Einstellungen in das System eintragen und zu einem späteren Zeitpunkt anpassen oder aktualisieren. Die darin enthaltenen Informationen können von den weiteren Komponenten angefragt und bereitgestellt werden. Client-seitig müssen von der `LocalProfileManager`-Komponente die Sensorinformationen erfasst und in ein für das System geeignetes Format (MASL) konvertiert werden. Diese Sensorerfassung ist nicht Teil der vorliegenden Arbeit, es wird angenommen, dass diese Informationen im Profil vorhanden sind. Eine `Exporter`-Komponente erlaubt es dem Nutzer, sein Profil auch auf andere Endgeräte zu übertragen.

Der Zugriff auf die Metadaten der Inhalte erfolgt über den `MetadatenManager`, der alle Metadaten verwaltet. Hier können alle Metadaten vom Inhalteanbieter aktualisiert oder gelöscht werden. Über einen `Importer` können Metadatenbeschreibungen, die in anderen Formaten vorliegen, in MASL konvertiert und damit in das System eingebracht werden.

Der Berechnungsprozess der Empfehlungen wird Server-seitig von der `Personalizer`- und Client-seitig von der `LocalPersonalizer`-Komponente gesteuert. Dafür greift der `Personalizer` auf Metadaten und Profilinformationen zu, wählt die passenden `Matcher` aus, übergibt ihnen die Profilinformationen und Metadatenbeschreibungen und aggregiert die Teilergebnisse zu einem Gesamtergebnis. Daraus entsteht eine Liste mit potenziell für den Nutzer interessanten Inhalten, die an den `LocalPersonalizer` übertragen werden. Dieser greift auf die Profilinformation zu, die auf dem Client verwaltet werden, wählt die `Matcher` auf dem Client aus, übergibt ihnen die für die Berechnung benötigten Informationen und aggregiert deren Teilergebnisse. Dann wählt er die Inhalte aus, die tatsächlich an das Endgerät übertragen werden. Diese können dem Nutzer oder einer entsprechenden Anwendung zusammen mit den Gründen für die Wahl des Inhalts übergeben werden.

Die `Personalizer`-Komponente besitzt die Konfigurationsschnittstelle `configurePersonalizer`, die es erlaubt, das Framework gemäß den Anforderungen der Anwendung zu konfigurieren. Diese Konfiguration wird dann vom Server an alle `LocalPersonalizer` der relevanten Clients übertragen. Der `LocalPersonalizer` ist für die Übertragung der Profildaten an den Server zuständig, die dort verwaltet werden.

Auf Client und Server sind dieselben `Matcher`-Komponenten vorhanden, um bei der Konfiguration größtmögliche Freiheit zu haben. Sie erhalten vom `Personalizer` bzw. `LocalPersonalizer` alle notwendigen Profilinformationen und Metadatenbeschreibungen, um diese entsprechend abgleichen zu können. Es gibt generische `Matcher`, die für beliebige Kontextinformationen verwendet werden, und auf spezielle Kontextinformationen spezialisierte `Matcher` (vgl. Abschnitt 5.5.3 und 5.5.4). Alle `Matcher` implementieren eine einheitliche Schnittstelle, über die immer die kompletten Beschreibungen übergeben werden und nicht nur der benötigte Teil. Dieser wird erst von der Komponente selbst extrahiert. Dadurch können leicht neue `Matcher` hinzugefügt werden, weil keine Kenntnisse über deren interne Arbeitsweise benötigt werden. Die `Matcher` liefern einen Wert zwischen 0 und 1 zurück, der anschließend zu einem Gesamtergebnis aggregiert wird.

5.8.3. Konfigurationsschnittstelle

Um das Framework einer speziellen Anwendung anzupassen, können über eine Konfigurationsschnittstelle alle relevanten Informationen an die `Personalizer`-Komponente übertragen werden. Dafür wurde ein XML-Schema definiert, das es erlaubt, diese Konfigurationen als XML in das System einzutragen (siehe Anhang A.3 und Abbildung 5.26).

Die Konfigurationsdaten werden mit dem `Config`-Element umschlossen. Die unterschiedlichen Anwendungen und deren Inhaltsklassen werden über einen einheitlichen Bezeichner (`applID` bzw. `classID`) unterschieden. Die Daten werden eingeteilt in `Server`-Konfiguration und `Client`-Konfiguration. Der `Personalizer` überträgt die für den Client bestimmten Daten an alle für eine Anwendung registrierten Clients.

Innerhalb des `Server`-Tags werden alle Daten des `Profile` gespeichert, die Server-seitig verwaltet werden. Ein einzelner Profileintrag wird über einen eindeutigen Bezeichner ge-

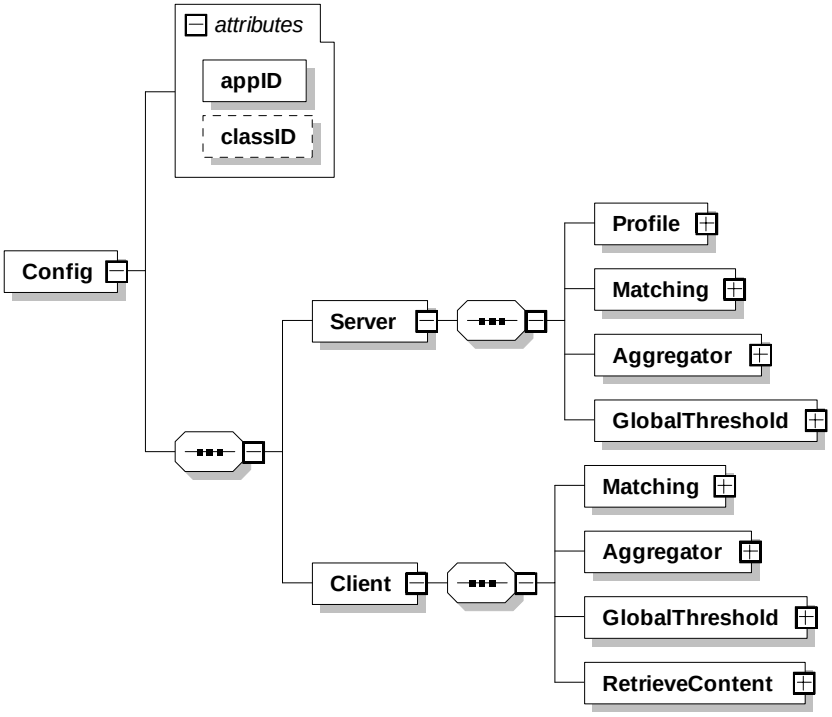

Abbildung 5.26.: Aufbau der Konfigurationsschnittstelle.

mäß der MASL-Beschreibungen definiert. Das Element, das mit seinen Subelementen als Profil auf den Server übertragen wird, wird eingetragen. Client-seitig wird das komplette Profil verwaltet, über die Attribute `updateMethod` und `updateRate` wird bestimmt, wann bzw. wie oft das Server-seitige Profil aktualisiert werden soll.

Über das `Matching`-Tag kann spezifiziert werden, welcher `Matcher` für welche Profilinformation verwendet werden soll. Die `Matcher` werden eindeutig über ein `matcher`-Attribut bestimmt. Das Gewicht, mit dem das von ihnen berechnete Teilergebnis in das Gesamtergebnis einfließen soll, wird mit dem `weight`-Attribut angegeben. Ist dieser Wert nicht vorhanden, wird das Gewicht gleichmäßig auf alle `Matcher` verteilt. Über das Attribut `threshold` kann spezifiziert werden, ab welchem Wert ein Teilergebnis bei der Berechnung des Gesamtergebnisses überhaupt berücksichtigt wird.

Die Aggregatorfunktion zur Berechnung des Gesamtergebnisses wird mit dem `Aggregator`-Tag festgelegt, das die Funktion über einen eindeutigen Bezeichner identifiziert. Manche der Funktionen wie die Maximum- oder Minimumaggregation benötigen keine weiteren Eingabewerte. Um aber beim k-Minimum den Parameter k oder bei der Minkowski-Aggregation den Parameter p festzulegen, muss über das `value`-Attribut ein weiterer Wert übergeben werden. Weiterhin kann man mittels `rangeMin` und `rangeMax`

festlegen in welchem Wertebereich die Teilergbnisse liegen, damit sie für die weitere Berechnung berücksichtigt werden. Der Schwellwert, ab wann ein Inhalt in der näheren Auswahl berücksichtigt wird, wird mittels `GlobalThreshold` festgelegt.

Der `Client`-Teil der Konfigurationsschnittstelle umfasst ebenfalls die `Matching`, `Aggregator` und `GlobalThreshold`-Elemente wie oben beschrieben. Das `Profile`-Element hingegen fehlt, da das Profil auf dem Client komplett vorgehalten wird. Zusätzlich muss hier noch der Parameter `RetrieveContent` angegeben werden, der genau festlegt, wie viele Inhalte letztendlich übertragen werden sollen.

Das Erstellen einer solchen Konfigurationsdatei erfordert in der Regel umfangreiches Wissen über die internen Abläufe des Frameworks und über MASL. Vorstellbar ist, die Erzeugung über eine graphische Nutzerschnittstelle zu unterstützen, die dann automatisch die konforme Datei erzeugt. Da dieser Vorgang für die Erläuterung der eigentlichen Grundkonzepte des Frameworks nicht notwendig ist, liegt er nicht im Fokus der vorliegenden Arbeit und wird daher nicht näher erläutert.

5.9. Bewertung der eigenen Ansätze

Um die in diesem Kapitel vorgestellten Ansätze zur Kontext-abhängigen Empfehlung multimedialer Inhalte zu bewerten, werden die Anforderungen verwendet, die in Abschnitt 5.2 identifiziert wurden.

1. *Generierung passender Empfehlungen*
   Im Rahmen dieses Kapitels wurden zahlreiche Ansätze vorgestellt, die für die Generierung passender Empfehlungen verwendbar sind. Diese basieren auf bereits existierenden und gut untersuchten Konzepten und erweitern diese, kombinieren sie neuartig oder passen sie geeignet an. Dadurch können dem Nutzer gute Empfehlungen präsentiert werden, was dessen Bereitschaft steigert, das System auch zu nutzen.

2. *Empfehlung bekannter und unbekannter Objekte*
   Das System wählt, wie es für ein Empfehlungssystem wichtig ist, gezielt für den Nutzer interessante Inhalte aus der Menge der bisher unbekannten Inhalte aus. Zusätzlich bietet es dem Nutzer auch bekannte Objekte an, mit denen er in der Vergangenheit bereits positive Erfahrungen gemacht hat. Dies steht in engem Zusammenhang mit der Nutzerzufriedenheit und ist für manche Anwendungen wie einen mobilen Touristenführer unabdingbar.

3. *Empfehlung von Kommentaren*
   Wie in Abschnitt 2.2.1 beschrieben, soll das System nicht nur Inhalte empfehlen, sondern auch die zugehörigen Kommentare geeignet filtern. Dafür wurde in Abschnitt 5.7 ein Algorithmus entwickelt, der auswählt, für welche Inhalte bzw.

Kommentare ein prognostizierter Empfehlungswert berechnet werden soll. Dadurch werden strukturerhaltende Empfehlungen auf strukturierten Daten ermöglicht.

4. *Berücksichtigung von Kontextinformationen*
   Eine neue Idee, die im Rahmen dieses Kapitels eingehender untersucht wurde, war es, Kontextinformationen bei der Empfehlung multimedialer Inhalte geeignet zu berücksichtigen. Dazu wurden Verfahren entwickelt, die bestimmen können, wann zwei Kontextinformationen, die z. B. in einer Inhaltsbeschreibung und im Nutzerprofil vorliegen, identisch oder ähnlich sind. Diese umfassen allgemein gehaltene Ähnlichkeitsmaße, die für eine Vielzahl von Kontextinformationen einsetzbar sind, aber es werden auch Besonderheiten einiger Kontextinformationen durch speziell entwickelte Ähnlichkeitsmaße berücksichtigt.

5. *Berücksichtigung von Kreuzkorrelationen der Attribute*
   Die Attribute der Inhaltsbeschreibungen nicht unabhängig voneinander zu betrachten, sondern auch ihre jeweiligen Kreuzkorrelationen, führt zu einer Erhöhung der Nutzerzufriedenheit. Dies wurde explizit in das Nutzerprofil-basierte Empfehlungsverfahren integriert, das in diesem Kapitel vorgestellt wurde. Dadurch werden in dem Gesamtsystem nicht nur die Nutzerpräferenzen von Einzelattributen, sondern auch von Attributkombinationen berücksichtigt.

6. *Reaktive und proaktive Empfehlungen*
   Das Framework wurde nicht speziell für eine rein reaktive oder proaktive Empfehlungsgenerierung entwickelt, sondern unterstützt beide Fälle. In der Regel wird der Nutzer selber nicht auf das Framework zugreifen, sondern mit ihm über eine zusätzliche Applikation interagieren, die auf dem Framework aufsetzt. Für die proaktive Erzeugung von Empfehlungen wird eine Überwachung von Kontextveränderungen benötigt, die mit verschiedenen Strategien z. B. Zeit-, Distanz- oder Zonen-basiert arbeitet. Diese stößt bei Veränderungen einen neuen Berechnungsprozess an. Die Konzeptionierung der Strategien ist nicht Teil dieser Arbeit.

7. *Systemfeedback*
   Um dem Nutzer das Gefühl zu geben, dem Empfehlungssystem nicht ausgeliefert zu sein, und ihm begreiflich zu machen, warum ein Inhalt ausgewählt wurde, soll das System in geeigneter Form Nutzerfeedback geben können. Die Berechnung des prognostizierten Empfehlungswerts für die Inhalte und Teilergebnisse wie z. B. die Ähnlichkeiten der Kontextinformationen kann als Grundlage dafür verwendet werden. Da aber eine sehr umfangreiche Menge an Attributtypen in die Berechnung einfließen kann, werden dem Nutzer nur diejenigen Attributtypen und Attributwerte als Gründe für die Empfehlung angegeben, die den prognostizierten Empfehlungswert am stärksten beeinflusst haben. Dies ist u. a. abhängig von den in die Aggregatsfunktion einfließenden Gewichten für die lokalen Ähnlichkeiten.

8. *Anwendungsunabhängigkeit*
   Alle in diesem Kapitel vorgestellten Ansätze wurden in ein generisches Framework eingebettet. Dieses kann je nach gewünschtem Anwendungsfall angepasst werden, indem u. a. die für einen bestimmten Anwendungsfall benötigten Ähnlichkeitsfunktionen, die geeigneten Schwellwerte und eine passende Aggregationsfunktion konfiguriert werden.

9. *Unterstützung unterschiedlicher Empfehlungsverfahren*
   Die unterschiedlichen Empfehlungstechniken sind je nach Anwendungsfall unterschiedlich geeignet für die Erzeugung einer passenden Empfehlung. Deswegen integriert das hier vorgestellte Framework klassisches inhaltsbasiertes und kollaboratives Filtern und den neu entwickelten Ansatz des Nutzerprofil-basierten Filterns und eine Struktur erhaltende Anpassung des kollaborativen Filterns. Dadurch können verschiedene Verfahren eingesetzt und geeignet kombiniert werden.

10. *Unterstützung unterschiedlicher Endgeräte*
    Im Prinzip kann das vorgestellte Framework für unterschiedliche Endgeräte wie PCs, Set-Top-Boxen oder mobile Geräte wie Handys oder Smart Phones eingesetzt werden, da es auf keiner konkreten Technologie aufsetzt. Abhängig vom verwendeten Gerät ist, wie die Kontextinformationen mittels Sensoren gemessen und geeignet in MASL-konforme Beschreibungen umgewandelt werden. Dies ist jedoch nicht Teil der Aufgabenstellung dieser Arbeit.

    Um dem Nutzer über einen einzelnen Anwendungsfall hinweg maximale Endgeräteunabhängigkeit zu gewährleisten, wurde eine Exporter-Komponente in das Framework integriert, die es erlaubt, das Profil auf andere Geräte zu übertragen. Dadurch erhält der Nutzer dieselben Empfehlungen in derselben Empfehlungsqualität, auch wenn er nicht sein übliches Endgerät verwendet.

11. *Unterstützung unterschiedlicher Metadatenformate*
    Das hier vorgestellte Framework ist speziell für die Anwendung mit der in Kapitel 3 definierten Sprache MASL gedacht. Dennoch können über eine Importer-Komponente andere Metadatenformate in MASL konvertiert und damit in MASL eingebracht werden. XML-basierte Standards wie TV-Anytime, MPEG-7/21 können einfach mittels XSLT transformiert werden, andere Metadatenformate müssen mittels einer eigens dafür entwickelten Importer-Komponente integriert werden. Zu beachten ist, dass ohne Unterstützung von Kontextinformationen in den Metadaten die speziell für die Kontext-abhängige Auswahl entwickelten Mechanismen nicht verwendbar sind, sondern nur die Standard-Techniken wie z. B. das inhaltsbasierte Filtern bzw. deren hier vorgestellte Anpassung.

12. *Konfigurationsschnittstelle*
    In Abschnitt 5.8.3 wurde die Konfigurationsschnittstelle des Frameworks näher

erläutert, die eine Anpassung an einen expliziten Anwendungsfall erlaubt.

13. *Schutz der Privatsphäre des Nutzers*
Gerade bei Kontext-abhängigen Systemen ist der Schutz der Privatsphäre ein kritischer Punkt. Um Empfehlungen basierend auf Kontextinformationen erzeugen zu können, müssen sie aber innerhalb des Systems gesammelt, gespeichert und verarbeitet werden. Deshalb muss zwischen dem Recht des Nutzers auf Privatsphäre und der effizienten und passenden Filterung und Empfehlung von Inhalten basierend auf Kontextinformationen ein geeignetes Mittelmaß gefunden werden. In diesem Kapitel wurde eine verteilte Speicherung der Profile vorgeschlagen, was zwar keinen vollständigen Schutz der Privatsphäre bietet, aber zumindest die Problematik etwas lindert. Je nach Anwendungsfall und Konfiguration kann das Framework auch alle Profildaten alleine auf dem Client speichern. Dies kann aber zu einer starken Belastung der Luftschnittstelle führen, da die Inhaltsbeschreibungen ungefiltert an einen mobilen Client übertragen werden. Außerdem kann in diesem Fall kein kollaboratives Filtern durchgeführt werden.

14. *Geringe Belastung der Luftschnittstelle*
Kontext-abhängige Empfehlung ist gerade bei mobilen Endgeräten interessant, da hier die dynamischen Kontextinformationen größeren Veränderungen unterliegen. Bei allen kabellosen Systemen ist die Luftschnittstelle ein limitierender Faktor, weswegen die Kommunikation zwischen Client und Server minimiert werden soll. Das vorgestellte Framework erreicht dies durch einen verteilten Abgleich: so werden nicht alle Metadaten auf den Client übertragen und dort erst gefiltert, bzw. das komplette Profil auf dem Server vorgehalten, was ja ein permanentes Update der dynamischen Kontextinformationen nach sich zieht. Durch das Server-seitig verwaltete Profil kann der Server die Metadaten vorfiltern und dennoch muss nicht das ganze Profil auf dem Server aktuell gehalten werden.

15. *Skalierbarkeit*
Der verteilte Abgleich zwischen Profil und Inhaltsbeschreibungen des Frameworks wirkt sich positiv auf die Skalierbarkeit des Gesamtsystems aus. Da der Abgleich nur teilweise auf einer zentralen Komponente durchgeführt wird, können hier deutlich höhere Nutzerzahlen bedient werden. Eine geeignete Strategie zur Replikation der Server kann dies noch zusätzlich unterstützen.

16. *Effiziente Berechnung der Empfehlungen*
Innerhalb des Frameworks wurde darauf geachtet, die Empfehlungen möglichst effizient berechnen zu können. Dies wird ermöglicht durch ein Parallelisieren der einzelnen Matching-Verfahren und der anschließenden Aggregation zu einem Gesamtergebnis, durch Vorberechnungen von Teilergebnissen in einer Art Offline-Phase, also

wenn das System nicht beschäftigt ist, Cachingstrategien und durch die Wiederverwendung von errechneten Teilergebnissen z. B. zur Erzeugung von Systemfeedback.

5.10. Zusammenfassung

Nach der Vorstellung der grundlegenden Techniken von Empfehlungssystemen wurden Anforderungen an ein Kontext-abhängiges Empfehlungssystem aufgestellt und verwandten Arbeiten mit diesen verglichen. Es wurde diskutiert, wie man Kontextinformationen im Empfehlungsprozess berücksichtigen kann. Dafür wurde das Lokal-Global-Prinzip eingeführt, das besagt, dass die Ähnlichkeit zwischen einem Inhalt und einem Nutzerprofil oder zwei Inhalten nicht in einer zusammenhängenden Berechnung bestimmt wird, sondern dass jeder Attributtyp einzeln betrachtet wird und die Teilergebnisse im Anschluss zu einem Gesamtergebnis aggregiert werden. Es wurden verschiedene Verfahren zur Ähnlichkeitsmessung von Kontext vorgestellt, die im folgenden Kapitel für die Anpassung der Darstellung verwendet werden. Für die Berechnung der Teilergebnisse und des Gesamtergebnisses wurden verschieden Möglichkeiten diskutiert.

Um das bei herkömmlichen inhaltsbasierten Verfahren auftretende Problem des *Curse of Dimensionality* zu beheben oder zumindest zu mildern, wurde klassisches inhaltsbasiertes Filtern zum Nutzerprofil-basierten Filtern erweitert.

Um Empfehlungen auf strukturierten Daten wie Inhalten und Kommentaren zu ermöglichen, wurde ein Algorithmus vorgestellt, der bestimmt, für welche Inhalte bzw. Kommentare ein prognostizierter Empfehlungswert berechnet wird.

Schließlich wurde gezeigt, wie alle in diesem Kapitel entwickelten Ansätze in ein einheitliches Framework für die Kontext-abhängige Empfehlung multimedialer Inhalte integriert werden.

Kapitel 6.

Kontext-abhängige Anpassung der Darstellung multimedialer Inhalte

Nachdem in Kapitel 5 die Kontext-abhängige Auswahl der Inhalte genauer erörtert wurde, wird in diesem Kapitel die zweite, in Abschnitt 2.1.3 vorgestellte Art der Kontext-abhängigen Personalisierung vorgestellt, die Kontext-abhängige Anpassung der Darstellung multimedialer Inhalte. Dafür werden in Abschnitt 6.1 Anforderungen an ein System für eine derartige Anpassung aufgestellt und in Abschnitt 6.2 verwandte Arbeiten untersucht. In Abschnitt 6.3 wird eine Middleware vorgestellt, die die Anpassung der Inhalte unterstützt.

6.1. Anforderungen an die Kontext-abhängige Anpassung der Darstellung

Im folgenden Abschnitt werden Anforderungen an die Kontext-abhängige Anpassung der Darstellung multimedialer Inhalte vorgestellt. Ein Teil der Anforderungen ergibt sich direkt aus den allgemeinen Anforderungen aus Abschnitt 2.2.4.

1. *Unterstützung unterschiedlicher Medientypen*
   Die hier erarbeiteten Konzepte sollen nicht nur für einen Medientyp anwendbar sein, sondern möglichst generisch für Inhalte von unterschiedlichem Medientyp wie Audio, Bild, Video, Text, Webseiten, Animationen etc. verwendbar sein. Dadurch erlaubt man dem Nutzer das größtmögliche multimediale Erlebnis und schränkt den Inhalteanbieter nicht bei der Bereitstellung der Inhalte auf einen oder eine nur geringe Menge von Medientypen ein.

2. *Anpassung gemäß Gerätecharakteristika*
   Mobile Endgeräte kann man in unterschiedliche Arten einteilen wie Smartphones, PDAs, Laptops oder Subnotebooks. Jede Art hat spezifische Charakteristika wie z. B. die vorhandenen Eingabemöglichkeiten. Aber auch innerhalb derselben Art

gibt es gravierende Unterschiede bei den Charakteristika z. B. bzgl. der Displaygröße des Endgeräts. Diese Gerätecharakteristika umfassen überwiegend statische Kontextinformationen, die das Gerät beschreiben, wie die Displaygröße, die Speichergröße, die Rechenleistung, die vom Gerät unterstützten Netze, die vorhandenen Eingabe- bzw. Ausgabemöglichkeiten, die vorhandenen Codecs zur Anzeige von Medieninhalten, installierte Anwendungen, sowie weitere gerätespezifische Ausstattungsmerkmale. Aus der großen Menge an Charakteristika ergibt sich eine Vielzahl an Bedingungen, die bei der Anpassung der Darstellung multimedialer Inhalte beachtet werden müssen.

3. *Anpassung gemäß Nutzerpräferenzen*
   Die Anpassung der Darstellung soll davon abhängig sein, wie ein Nutzer präferiert die Inhalte angezeigt haben will. So kann ein Nutzer z. B. die Präferenz haben, dass ihm Texte vorgelesen werden, da er sich viel mit dem Auto bewegt, ein anderer Nutzer hat hingegen eine Hörschwäche und möchte gesprochene Passagen als Untertitel zur Verfügung gestellt haben.

4. *Anpassung gemäß dynamischer Kontextinformationen*
   Während es sich bei den Gerätecharakteristika und den Nutzerpräferenzen um überwiegend statische Kontextinformationen handelt, soll ebenso der dynamische Kontext berücksichtigt werden. Man kann unterscheiden zwischen Nutzer- und Geräteabhängigen dynamischen Kontextinformationen. Die Nutzer-abhängigen werden durch den Nutzer bestimmt, wie der Ort, an dem sich dieser befindet. Wie darauf reagiert wird, ist dabei einerseits abhängig von allgemeinen Regeln (z. B. ersetze Audio durch Text, sobald der Geräuschpegel der Umgebung zu hoch wird) oder durch individuelle Nutzerpräferenzen. Geräte-abhängige Kontextinformationen ändern sich in Abhängigkeit des Geräts, wie der aktuelle Ladestand des Akkus oder das Vorhandensein von externen Geräten wie Kopfhörern. Auch hier wird die Reaktion auf die Kontextinformationen in Abhängigkeit von allgemeinen Regeln oder von Nutzerpräferenzen bestimmt.

5. *Erfassen und Erkennen von Anpassungsparametern*
   Die Anpassung der Darstellung soll abhängig sein von Parametern, die durch die Gerätecharakteristika, die Nutzerpräferenzen und die aktuellen Werte der dynamischen Kontextinformationen bestimmt werden. Dynamische Parameter sollen automatisch erfasst, Änderungen erkannt und geeignet darauf reagiert werden. Ebenso sollen die statischen Gerätecharakteristika erkannt werden, sowie der Nutzer die Möglichkeit haben, seine Präferenzen zu erstellen und zu verändern.

6. *Unterstützung geeigneter Methoden zur Anpassung*
   Die Anpassung der Darstellung kann auf unterschiedliche Weise durchgeführt werden. Die einfachste ist es, einen den Anpassungsparametern entsprechenden Inhalt

vom Server zu selektieren. Dies bedeutet für einen Inhalteanbieter, dass er denselben Medieninhalt in unterschiedlichen Formaten vorhalten muss, um die ganze Breite möglicher Anpassungsparameter unterstützen zu können. Dies führt jedoch zu einem erhöhten Ressourcenbedarf und soll vermieden werden.

Von daher soll keine Selektion der Inhalte gemäß den Anpassungsparametern, sondern eine explizite Anpassung der Inhalte an diese durchgeführt werden. Dies kann durch unterschiedliche Methoden realisiert werden, die im Folgenden aufgelistet sind.

a) *Skalieren*: Diese Methode beschreibt die Anpassung einer Größe. Bezogen auf den Inhalt können unterschiedliche Größen angepasst werden. Gerade bei mobilen Endgeräten ist die Anpassung an die Displaygröße relevant, da diese beträchtlich variieren kann. Deshalb muss ein visueller Inhalt wie ein Bild oder ein Video in der Größe der Darstellung den Gegebenheiten entsprechend verändert werden. Es soll darauf geachtet werden, dass der Inhalt nicht abgeschnitten wird oder dass Texte nicht zu klein dargestellt werden und noch lesbar bleiben. Weitere Skalierungen beziehen sich auf die Dateigröße, die z. B. bei geringer Speicherkapazität des Endgeräts oder schlechter Konnektivität reduziert werden soll, auf die Qualität des Inhalts oder auf die Datenrate.

b) *Konvertieren*: Dieser Vorgang beschreibt die Veränderung des Formats eines Inhalts. Mobile Endgeräte sind in der Regel für unterschiedliche Plattformen konzipiert und unterstützen nicht alle dieselben Formate. Um zu vermeiden, dass der Inhalteanbieter seine Medieninhalte in unterschiedlichen Formaten vorhalten muss, soll die Konvertierung in andere Formate unterstützt werden.

c) *Entfernen*: Darunter versteht man das Entfernen einzelner Komponenten des Medieninhalts. Prinzipiell soll dies vermieden werden, da dadurch ggf. für den Nutzer relevante Informationen verloren gehen. Manchmal ist dies jedoch nötig oder sogar gewünscht, um z. B. umfangreiches Bildmaterial einer Webseite nicht über eine Verbindung mit niedriger Bandbreite übertragen zu müssen. Außerdem können Medieninhalte entfernt werden, wenn sie nicht durch eine der anderen Methoden gemäß den Anpassungsparametern veränderbar sind.

d) *Hinzufügen*: Dies beschreibt den Vorgang, bestehende Medieninhalte geeignet zu erweitern. So können zum Beispiel zu einem Video zusätzlich zur Audiospur noch Untertitel angeboten werden, auch wenn der Medieninhalt diese nicht explizit vorsieht.

e) *Ersetzen*: Dieser Vorgang setzt sich aus den beiden Methoden *Entfernen* und *Hinzufügen* von anderen Inhalten zusammen. So kann man Medieninhalte, die für ein Endgerät ungeeignet sind, entfernen und andere, besser geeignete hinzufügen.

In der Regel werden die hier aufgeführten Methoden in Kombination parallel oder sequentiell ausgeführt.

7. *Anwendungsunabhängigkeit*
   Wie schon in Abschnitt 2.2.4 beschrieben, soll der in dieser Arbeit vorgestellte Ansatz zur Kontext-abhängigen Personalisierung anwendungsunabhängig sein. Dafür muss diese Anforderung auch für den Teilaspekt der Kontext-abhängigen Anpassung der Darstellung gelten. Das hier entwickelte System soll für unterschiedliche Einsatzszenarien funktionieren und mit den verschiedensten Applikationen verwendbar sein.

8. *Geringe Belastung der Luftschnittstelle*
   Bei allen kabellosen Systemen ist die Luftschnittstelle ein limitierender Faktor. So können nicht wie bei kabelgebundenen Systemen beliebig viele Daten zwischen einem Client und einem Server übertragen werden, da dies mit einem hohem Zeitaufwand und, sofern keine Datenflatrate vorhanden ist, mit hohen monetären Kosten verbunden ist. Deswegen sollen die Ressourcen der Luftschnittstelle effizient genutzt und die Client-Server-Kommunikation auf ein Minimum beschränkt werden. Insbesondere soll vermieden werden, speicherintensive Inhalte zu übertragen, die sowieso entfernt oder bedeutend kleiner skaliert werden.

9. *Zeitnahe und effiziente Anpassung der Darstellung*
   Da die Übertragung von Inhalten auf mobile Endgeräte Zeit in Anspruch nimmt, soll die Anpassung der Darstellung möglichst zeitnah erfolgen. In der Regel ist der Nutzer nicht bereit auf die Anpassung der Darstellung zu warten, von daher senkt eine gute, allerdings zeitintensive Anpassung die Nutzerzufriedenheit.

10. *Skalierbarkeit*
    Das System zur Anpassung der Darstellung soll von einer großen Anzahl Nutzer verwendbar sein. Daher muss es bezüglich der Anzahl der Nutzer für die die Anpassung vorgenommen werden soll, skalierbar sein.

11. *Schutz der Privatsphäre des Nutzers*
    Bei jedem Kontext-abhängigen System müssen immer Daten über den Nutzer gesammelt, gespeichert und verarbeitet werden, die als Eingabewerte dienen. Das ist besonders bei dynamischen Kontextinformationen kritisch, da diese Daten sehr sensibel sind. Von daher muss ein geeignetes Mittelmaß gefunden werden zwischen dem Recht des Nutzers auf Privatsphäre und der effizienten und passenden Anpassung der Darstellung von Inhalten basierend auf diesen Kontextinformationen.

12. *Transparente Eingliederung in bestehende Anwendungen*
    Um die Anwendungsunabhängigkeit noch zu erhöhen, soll sich das hier entwickelte System transparent in bestehende Dienste und Anwendungen eingliedern lassen.

Das System soll also derart verwendet werden, dass an den Anwendungen keine Anpassungen benötig werden und dass diese die Anpassung der Darstellung nicht bemerken.

6.2. Verwandte Ansätze

Im folgenden Abschnitt werden einige verwandte Arbeiten aus dem Bereich der Kontextabhängigen Anpassung multimedialer Inhalte näher vorgestellt und mit den aufgestellten Anforderungen verglichen.

6.2.1. Synchronized Multimedia Integration Language (SMIL)

Die vom W3C entwickelte *Synchronized Multimedia Integration Language* (SMIL, ausgesprochen wie engl. *smile*) [26] dient zur zeitsynchronisierten Anzeige von multimedialen Inhalten. Sie wurde basierend auf dem XML-Standard entwickelt. Üblicherweise wird sie für *rich media*-Präsentationen im Internet verwendet, die Web-Seiten mit Animationen, Videos oder anderen Medieninhalten anreichern.

Mit SMIL beschreibt man eine komplette multimediale Präsentation, die unterschiedliche multimediale Inhalte verknüpft und festlegt, ob diese parallel oder sequentiell abgespielt werden sollen. Das gesamte Layout der Präsentationsfläche wird genau spezifiziert. SMIL unterstützt die Auswahl von Versionen passend zur aktuellen Übertragungsbandbreite, Bildschirmauflösung und Farbtiefe des Endgeräts oder von verschiedenen Sprachversionen derselben Präsentation. Ursprünglich für Internetseiten entwickelt, wird SMIL seit 2005 auf mobilen Endgeräten eingesetzt und als Standard zur Anzeige von über den Multimedia Message Service (MMS) verschickten Nachrichten verwendet.

SMIL besteht in Version 3.0 aus 12 funktionalen Bereichen, die unterschiedliche Aufgaben bei der Erstellung von Präsentationen abdecken (siehe Abbildung 6.1). Die grundlegende Struktur eines SMIL-Dokuments wird mittels dem `Structure` Modul spezifiziert. Dies umfasst zum einen sehr rudimentäre Metadatenbeschreibungen über z. B. den Titel der Präsentation mittels `Metainformation` und das genaue Erscheinungsbild der Präsentationsfläche mittels `Layout`. Auf dieser befinden sich Elemente der Module `Media Objects` für beliebige Medientypen und `SmilText` für Inline Texte. Da es sich um einen für das Web gedachten Standard handelt, soll es möglich sein, diese Elemente zu verlinken. Dies wird mit Werkzeugen aus dem `Linking` Modul erreicht. Durch das `Animation` Modul können über die Veränderung von bestimmten Eigenschaften von Objekten über die Zeit Animationen erzeugt werden. Die Organisation des zeitlichen Ablaufs einer Präsentation wird über `Timing` festgelegt. Manipulationsmöglichkeiten für diesen zeitlichen Ablauf und Übergänge wie Ein- oder Ausblendungen werden in den Modulen `Time Manipulations` bzw. `Transitions` spezifiziert. Der aktuelle Status der Präsentation der z. B. beschreibt, welche Teile gerade aktiv sind, wird über das `State`-Element beschrieben. Das `ContentControl` Modul fasst Mechanismen zusammen, die

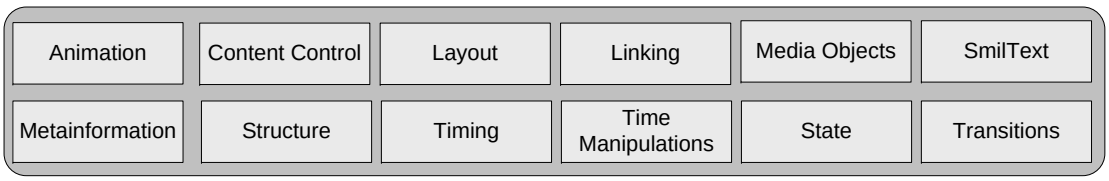

Abbildung 6.1.: Module des SMIL-Standards

eine passende Auswahl der Inhalte zur Laufzeit der Präsentation und eine Optimierung der Übertragung auf das Endgerät erlauben. Dadurch können Inhalte basierend z. B. auf der Displaygröße oder der Bandbreite ausgewählt werden.

SMIL bietet umfangreiche Möglichkeiten, multimediale Präsentationen mit einer großen Anzahl an unterstützten Medientypen anzuzeigen, und unterstützt unterschiedliche Anwendungsbereiche. Es muss jedoch ein entsprechender SMIL-Player vorhanden sein, andere Anwendungen werden nicht unterstützt. In geringem Maß werden Unterschiede der Endgeräte und deren aktuelle Konnektivität berücksichtigt. Beispiele hierfür sind die Displaygröße und dynamische Kontextinformationen wie die verfügbare Übertragungsbandbreite. Da es sich um einen für das Web entwickelten Standard handelt, wo es nicht so starke Unterschiede der Endgeräte wie im mobilen Bereich gibt, sind diese Aspekte nur rudimentär berücksichtigt. Der Nutzer hat die Möglichkeit, bei textuellen und auditiven Medien die bevorzugt zu verwendende Sprache anzugeben, weitere Präferenzen werden nicht berücksichtigt. Mittels SMIL findet keine Anpassung der Darstellung statt, sondern für jede Art der Darstellung muss eine eigene Datei vorliegen. Mögliche Anpassungsparameter müssen vom Player geeignet erfasst und erkannt werden. Durch die Bereitstellung eigener Dateien können die für ein Endgerät geeigneten zeitnah und effizient ausgewählt und übertragen werden, was eine gute Skalierbarkeit bewirkt. Diese Auswahl kann bezüglich einer geringen Belastung der Luftschnittstelle optimiert werden. Da es nicht nötig ist, Kontextinformationen zu versenden, ist der Schutz der Privatsphäre gewährleistet.

6.2.2. Wissensbasiertes Framework zur Adaption von Medieninhalten

Von Jannach et al. wurde ein leicht erweiterbares Framework zur Anpassung multimedialer Inhalte entwickelt [91, 92, 93]. Dieses arbeitet mit einem wissensbasierten Ansatz und nutzt MPEG-7/21-Beschreibungen für die Anpassung. Gemäß den in den Beschreibungen vorkommenden Charakteristika der Endgeräte, der Nutzerpräferenzen und den dynamischen Kontextinformationen werden geeignete Anpassungs-Operationen ausgewählt und die Anpassung der Inhalte vorgenommen.

Die grobe Architektur des Frameworks ist in Abbildung 6.2 zu sehen. Eine Client-

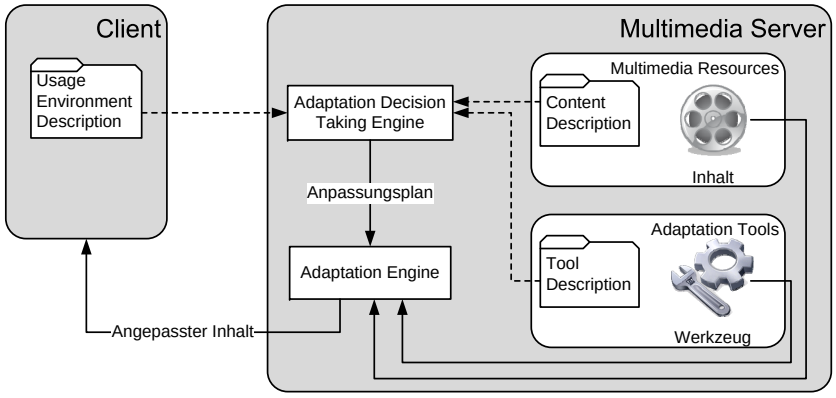

Abbildung 6.2.: Architektur des Wissensbasierten Frameworks nach [93]

Komponente sendet eine Anfrage nach einem multimedialen Inhalt an den *Multimedia Server*, der die Inhalte zur Verfügung stellt. Diese Anfrage beinhaltet u. a. eine Beschreibung der Gerätecharakteristika, der dynamischen Kontextinformationen, die auf dem Endgerät zum Zeitpunkt der Anfrage vorherrschen wie die Umgebungslautstärke oder die verfügbare Bandbreite, und der Nutzerpräferenzen bezüglich der Anpassung. Diese Informationen werden mittels Metadaten des *Digital Item Adaptation* (DIA)-Werkzeuges aus dem MPEG-21-Standard erfasst. Die *Adaptation Decision Taking Engine* vergleicht die DIA-Beschreibung mit der vorliegenden MPEG-7-Beschreibung des angefragten Medieninhalts.

Alle im System verfügbaren Anpassungswerkzeuge werden durch einen Namen, die benötigten Parameter, die für die Ausführung benötigten Vorbedingungen sowie die Effekte nach Ausführung der Operation deklarativ beschrieben. Mit diesen Informationen wird ein Anpassungsplan erstellt, der eine Reihe von Werkzeugen auswählt, die hintereinander geschaltet den Inhalt von der Ursprungsdatei in einer der DIA-Beschreibung entsprechenden Art ändern. Bei der Erstellung des Planes werden Methoden der künstlichen Intelligenz eingesetzt. Das Vorgehen wird als klassische Zustandsraum-Planung betrachtet, die einen Zustandsraum von einem Anfangszustand mittels verschiedener Aktionen in einen Zielzustand überführt. Jede Aktion hat eine Vorbedingung, die gelten muss, und einen Effekt, der durch die Ausführung entsteht. Durch die Planung wird ein geeigneter Pfad vom Anfangszustand zum Zielzustand durch Anwendung der vorhandenen Aktionen gesucht. Der erstellte Anpassungsplan wird an die *Adaptation Engine* übergeben, die die eigentliche Anpassung durchführt und den angepassten Inhalt an den Client zurückliefert.

Das vorgestellte Framework liefert eine einfache und dynamisch erweiterbare Lösung zur Anpassung multimedialer Inhalte. Es berücksichtigt sowohl statische Kontextinforma-

tionen wie Gerätecharakteristika und Nutzerpräferenzen, und dynamische Daten wie die Umgebungslautstärke, die Helligkeit oder die verfügbare Bandbreite. Die Anpassungsparameter werden nicht erkannt, sondern liegen in entsprechenden Beschreibungen vor. Die unterstützten Medientypen sind abhängig von den verfügbaren Anpassungswerkzeugen, prinzipiell sind aber alle Medientypen erlaubt, da ausschließlich mit den Beschreibungen der Inhalte und Werkzeuge gearbeitet wird. Alle gewünschten Arten der Anpassung, also Skalierung, Konvertierung und Hinzufügen, Entfernen bzw. Ersetzen von Teilen, können prinzipiell unterstützt werden.

Das Framework kann in vielen Einsatzszenarien verwendet werden. Eine transparente Integration in bestehende Anwendungen ist nicht möglich. Dafür wird eine zusätzliche Komponente benötigt, die im Client DIA-Beschreibungen erzeugt. Da das gesamte Nutzerprofil an den Server gesendet wird, ist kein ausreichender Schutz der Privatsphäre möglich, und die Luftschnittstelle wird dadurch belastet. Das aufwändige Erstellen des Anpassungsplans ist zeitintensiv, wodurch auch die Skalierbarkeit des Gesamtsystems leidet.

6.2.3. mobileMM4U

Von Boll und Scherp wurde das generische, Desktop-orientierte Framework zur dynamischen Erstellung von personalisierten Multimedia-Präsentationen MM4U (Akronym für *Multimedia for you*) entwickelt [20, 147]. Um den speziellen Anforderungen mobiler Endgeräte Rechnung zu tragen, wurde es um für diese Geräte benötigte Komponenten erweitert und damit das mobileMM4U-Framework geschaffen [148, 149]. Beide Frameworks werden für unterschiedliche Anwendungsbereiche eingesetzt und unterstützen eine breite Auswahl an Endgeräten. Durch vereinheitlichte Schnittstellen werden die im Framework vorhandenen Funktionalitäten um anwendungsspezifische erweitert.

Das Framework ist schichtweise aufgebaut (siehe Abbildung 6.3). Die Anpassung der Darstellung basiert auf dem Nutzerprofil und den Metadaten der Inhalte. Diese Informationen werden über *User Profile* bzw. *Media Data* Konnektoren in das Framework eingebunden. Sie stellen vereinheitlichte Schnittstellen zur Verfügung, die den Import der Daten aus unterschiedlichen Datenbanken oder Speicherstrukturen mit unterschiedlichen Formaten ermöglichen. Bei Bedarf können weitere Konnektoren hinzugefügt werden, um zusätzliche Datenquellen zu berücksichtigen. Die importierten Nutzerprofile und Metadaten werden gemäß eines internen, abstrakten Datenmodells umgewandelt, das von den Komponenten *User Profile Accessor* bzw. *Media Pool Accessor* zur Verfügung gestellt wird. Das Datenmodell der Nutzerprofile beinhaltet Informationen über Wissen, Vorlieben, Fähigkeiten und Einschränkungen der Nutzer, Gerätecharakteristika und dynamische Kontextinformationen während der Nutzung, die bei der Anpassung berücksichtigt werden sollen.

Die im internen Datenmodell hinterlegten Informationen werden von der darüber liegenden Schicht verwendet, um personalisierte Multimedia-Präsentationen zusammenzu-

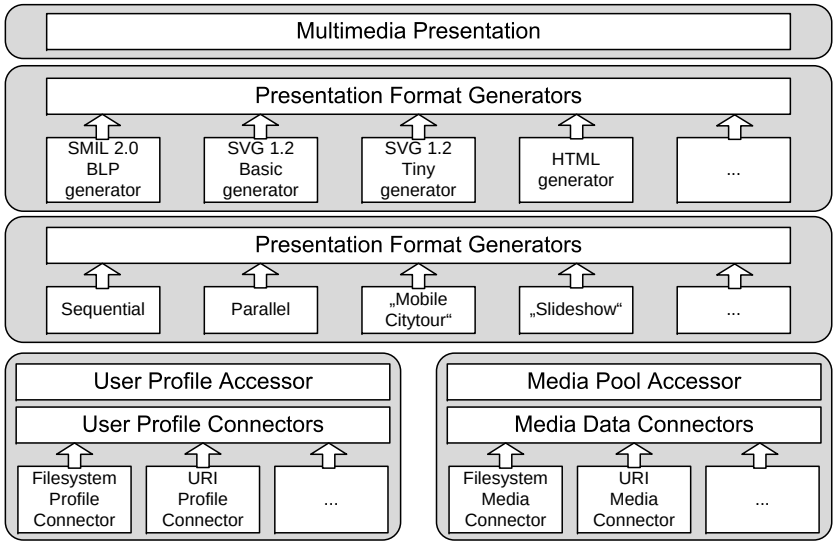

Abbildung 6.3.: Architektur des mobileMM4U-Frameworks nach [147]

stellen. Der Schicht stehen dafür Operatoren zum sequentiellen oder parallelen Abspielen von Inhalten zur Verfügung und komplexere Werkzeuge z. B. zur Erstellung von multimedialen Stadtführungen. Auf diese Weise entsteht eine interne Darstellung der personalisierten Multimedia-Präsentation, die über ein Darstellungsmodell repräsentiert wird. Dieses Modell wird durch die *Presentation Format Generators* in eine Multimedia-Präsentation konvertiert, die in verschiedenen Formaten wie SVG, SMIL oder Flash vorliegen kann. Die Präsentation kann dann durch die *Multimedia Presentation* Schicht über auf dem mobilen Endgerät zur Verfügung stehenden Playern abgespielt werden.

Für eine konkrete Anwendung werden die jeweils benötigten Teile des Frameworks ausgewählt und geeignet kombiniert. Als Beispiel-Anwendung wurde Sightseeing4U, ein personalisierter Stadtführer, entwickelt, der Sehenswürdigkeiten den Interessen und Vorlieben des Nutzers entsprechend auswählt und anzeigt. Der Prototyp wurde als Stadtführer in Oldenburg mit ca. 25 Sehenswürdigkeiten eingesetzt.

Das mobileMM4U-Framework ist ein anwendungsunabhängiger Ansatz für die personalisierte Anpassung der Darstellung multimedialer Inhalte. Es unterstützt eine umfangreiche Menge an Medientypen. Bei der Anpassung werden als Parameter die Gerätecharakteristika, die Nutzerpräferenzen und dynamische Informationen verwendet, wobei aus der Literatur nicht ersichtlich ist, ob das Framework an sich diese Informationen erfassen und erkennen kann, oder ob dies nur durch eine externe Komponente oder eine Erweiterung möglich ist. Für jeden Nutzer und jede Anfrage wird eine eigene Präsentation

erstellt, was sich negativ auf die Bereitstellungszeit der Inhalte auswirkt. Dies relativiert sich jedoch, da das Framework die Inhalte lediglich gemäß der Parameter aus der Menge der vorhandenen auswählt und diese nicht wirklich angepasst werden. Die Dienste und Anwendungen, die die Anpassung durch mobileMM4U nutzen wollen, müssen explizit auf dem Framework aufsetzen, eine transparente Eingliederung ist nicht möglich.

Die Schichten des Frameworks können als Thick- oder Thin-Client auf dem mobilen Endgerät umgesetzt werden. Während sich beim Thick-Client alle Schichten auf dem Endgerät befinden, besitzt der Thin-Client nur die *Multimedia Presentation* Schicht. Der Thick-Client bietet im Gegensatz zum Thin-Client einen Schutz der Privatsphäre und bessere Skalierbarkeit, da die Profildaten nur auf dem Endgerät gespeichert werden. Um beim Thin-Client das Server-seitig gespeicherte Profil aktuell zu halten, müssen Änderungen permanent übertragen werden. Dies führt zu einer hohen Belastung der Luftschnittstelle und schlechterer Skalierbarkeit, insbesondere wenn es sich um dynamische Profilinformationen handelt.

6.2.4. Niccimon

Am Niedersächsischen Kompetenzzentrum Informationssysteme für die mobile Nutzung (Niccimon) wurde ein modulares System entwickelt, das die schnelle und flexible Umsetzung von mobilen und ortsbezogenen Anwendungen erlaubt. Diese so genannte Niccimon-Plattform [7, 64] unterstützt für ortsbezogene Dienste relevante Techniken wie die Lokalisierung, die Visualisierung des aktuellen Standorts oder die Anzeige von Points of Interest (PoIs) und von multimedialen Inhalten.

Die Plattform besteht aus den zwei Kern-Komponenten *Niccimon Mediator* und *Niccimon Component Interface* (vgl. Abbildung 6.4). Über den Niccimon Mediator wird die Integration von unterschiedlichen Modulen in die Plattform unterstützt und deren Interaktion koordiniert. Dazu gehört die Steuerung von Audioausgaben und Präsentationen, um Überlagerungen von Medieninhalten zu vermeiden, die durch unterschiedliche Module erzeugt wurden. Alle in die Plattform integrierten Module müssen das *Niccimon Component Interface* implementieren. Dieses beinhaltet die *Event Communication* zur Plattform-übergreifenden Kommunikation über Ereignisse und das *Presentation Interface* zur Steuerung von Präsentationen und Audioausgaben. Über das *Network Interface* kann das Modul mit der Server-Anwendung kommunizieren, wobei es einer Bandbreiten- und QoS-Kontrolle unterliegt. Für effizientes Ressourcenmanagement kann ein Modul vom Mediator über die *Live-Cycle Control* aktiviert oder deaktiviert werden.

Die Niccimon-Plattform setzt einige Basis-Module für grundlegende Funktionalitäten um. Dazu gehört das *Location Module* zur Übermittlung der aktuellen Ortsinformationen, das *Mobile GIS Module* zur Visualisierung der Position des Nutzers auf einer 2D-Karte, ein *Points of Interest Module* zur Verwaltung von POIs und ein *Multimedia Module* zur Darstellung multimedialer Inhalte.

Die Plattform wurde für unterschiedliche Anwendungsfälle eingesetzt, um die leichte

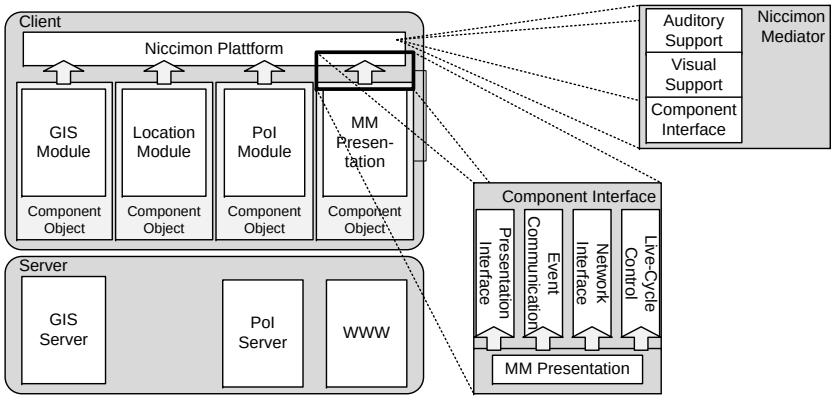

Abbildung 6.4.: Aufbau der Niccimon-Plattform nach [7]

Erweiterbarkeit zu demonstrieren. Ein Beispiel ist das mobile, touristische Informationssystem MobiDENK [101], das als persönlicher, mobiler Assistent dient und 2003 für die Besucher des *Großen Gartens von Herrenhausen* in Hannover zur Verfügung gestellt wurde. Die Applikation erlaubte den Besuchern eine Navigation und Orientierung innerhalb des Gebiets und stellte Texte und historisches Bildmaterial zur Verfügung. Mit AccessSight [97] wurde ein multimodales mobiles touristisches Informationssystem umgesetzt, das ähnlich wie MobiDENK arbeitet, jedoch von sehbehinderten Nutzern genutzt werden kann. Ein- und Ausgaben werden je nach Nutzer angepasst, wobei bei sehbehinderten Nutzern Texte vorgelesen werden und Bilder oder Kartenmaterial sowie die Navigation über auditive Informationen vermittelt werden.

Die Niccimon-Plattform ist ein modularer, leicht erweiterbarer und damit anwendungsunabhängiger Ansatz zur Anpassung der Darstellung multimedialer Inhalte. Eine transparente Integration in bestehende Anwendungen ist nicht möglich, da diese auf die Plattform angepasst werden müssten. Es werden Texte, Bilder und Audio-Dateien unterstützt, nicht jedoch Videos. Bei der Anpassung wird ausschließlich die Displaygröße als Gerätecharakteristikum, aber keinerlei dynamische Kontextinformationen berücksichtigt. Die AccessSight-Anwendung zeigt, dass die Anpassung an Nutzerpräferenzen möglich ist. Diese Präferenzen und die Displaygröße können vom System als Anpassungsparameter erkannt werden. Als Methoden zur Anpassung wird ausschließlich die Skalierung von Bildern und Texten unterstützt.

Die Anpassung mittels der Plattform erfolgt auf Seiten des Clients, was zu einer erhöhten Belastung der Luftschnittstelle führt, da die Daten vor der Anpassung übertragen werden müssen, aber auch zu einer guten Skalierbarkeit. Da auf dem Client eine geringere Rechenleistung vorhanden ist, ist die Anpassung zeitintensiver als bei Server-seitigen

Lösungen. Der Vorteil aber ist, dass keine privaten Daten an einen zentralen Server versendet werden müssen und die Privatsphäre des Nutzers gut geschützt wird.

6.2.5. Opera Extensible Rendering Architecture

Die meisten im Netz verfügbaren Webseiten sind für einen bestimmten Bildschirmauflösungsbereich entwickelt worden und nicht für die Anzeige auf Endgeräten mit niedriger Auflösung wie mobilen Geräten gedacht. Dadurch wird der Inhalt über den darstellbaren Bereich des Geräts hinaus angezeigt, und der Nutzer kann diesen nur mit Hilfe des horizontalen Scroll-Balkens gänzlich erfassen. Von der Firma Opera Software ASA wurde deswegen ein Framework namens *Extensible Rendering Architecture* (ERA) entwickelt [187], das es den Opera Webbrowsern erlaubt, dynamisch die Webseiten an eine beliebige Diaplaygröße anzupassen, und dadurch horizontales Scrollen vermeidet, ohne dass speziell angepasste Seiten oder Stylesheets benötigt werden.

Innerhalb des Frameworks werden unterschiedliche Werkzeuge zur Anpassung, Umformatierung und Skalierung von Webseiten für die unterschiedlichen Endgeräte zur Verfügung gestellt. Diese umfassen u. a. das Skalieren von Texten und Bildern, die Umstrukturierung von Tabellen, die Anpassung des Kontrasts und das Umpositionieren und Ausblenden von Inhalten. Die Werkzeuge können je nach Anwendungsfall und zu unterstützender Endgeräteklasse zu verschiedenen Konfigurationen kombiniert werden.

Der Web-Browser *Opera for Mobile* wurde für Smart Phones und PDAs und *Opera Mini* für Mobiltelefone entwickelt. Beide beinhalten die Technologien der ERA. Während Opera for Mobile wie ein herkömmlicher Browser funktioniert, bezieht der Opera Mini alle Webinhalte über einen speziellen Proxy-Server, der von Opera ASA direkt betrieben wird. Dieser passt die Inhalte gemäß der Konfiguration an, um die kleineren Displays zu unterstützen und um das Datenvolumen zu reduzieren und die Übertragungszeit zu verbessern. Dazu werden Informationen wie die Displaygröße des Endgeräts übertragen.

ERA erlaubt durch die Vielzahl an unterstützten Konfigurationen und Werkzeugen eine Anpassung von Web-Seiten an unterschiedliche Endgeräte und Displaygrößen. Es ist stark auf die Verwendung mit Web-Inhalten ausgelegt und unterstützt keine anderen Medientypen. Ebenso ist es auf Web-Anwendungen beschränkt und kann nicht in bestehende Anwendungen integriert werden, ist aber gut skalierbar. Opera Mini berücksichtigt bei der Anpassung der Darstellung Text und Bilder, aber keine Videos oder Audiodateien. Die Displaygröße wird als einzige Gerätecharakteristik berücksichtigt. Nutzerpräferenzen werden nur sehr vereinzelt beachtet, bei Opera Mini z. B. ob Bilder angezeigt werden sollen oder welche Qualität und welche Schriftgröße verwendet werden soll. Dynamische Kontextinformationen werden hingegen nicht betrachtet, weswegen kaum Anpassungsparameter wie Displaygröße und nur rudimentäre Nutzerpräferenzen von Opera ERA erfasst werden.

Als Verfahren zur Anpassung werden Skalierung und das Entfernen ausschließlich von Bildern unterstützt. ERA achtet per se nicht auf eine effiziente Nutzung der Luftschnittstelle oder eine zeitnahe Anpassung, in der Umsetzung als Opera Mini wird das zu übertragende Datenvolumen jedoch explizit reduziert. Durch das Senden von Daten an den zentralen Proxy-Server kann die Anpassung Zeit-effizient durchgeführt werden, es ist aber kein Schutz der Privatsphäre gegeben.

6.2.6. MPEG-7/-21-basierte Video-Personalisierung

Das von Tseng et al. entwickelte Framework zur Zusammenfassung und Personalisierung von Videos erlaubt neben der Kontext-abhängigen Empfehlung eine Kontext-abhängige Anpassung der Darstellung [167]. Eine genaue Beschreibung des Ansatzes ist in Abschnitt 5.3.3 zu finden.

Die Anpassung der Darstellung basiert auf dem MPEG-7/21-Standard. Das Framework verwendet Beschreibungen der aktuellen Nutzungsumgebung und Anpassungsdeklarationen, um zu bestimmen, wie der Inhalt angepasst werden soll. Der angepasste Inhalt wird dann auf den Client übertragen.

Wie in Abschnitt 5.3.3 erläutert, ist das System anwendungsunabhängig für Videos nutzbar und erlaubt keine anderen Medientypen. Eine Nutzung auf unterschiedlichen Endgeräten wird unterstützt und bei der Übertragung die Einschränkungen bezüglich der Luftschnittstelle berücksichtigt. Nutzerprofile werden nicht permanent an einer zentralen Stelle verwaltet, sondern nur bei einer Anfrage übertragen, was zumindest den Schutz der Privatsphäre im Vergleich zu rein zentralen Ansätzen verbessert, sich aber negativ auf die Skalierbarkeit auswirkt.

Kontextinformationen, die bei der Anfrage an das System erfasst werden, dienen ausschließlich der Anpassung der Inhalte, nicht dem Empfehlungsprozess. Durch die Verwendung des DIA aus dem MPEG-21-Standard können Nutzerpräferenzen, Gerätecharakteristika und dynamische Kontextinformationen bei der Anpassung berücksichtigt werden. Bei der Beschreibung des Systems wird nicht darauf eingegangen, ob diese Möglichkeiten vollständig genutzt werden. Lediglich die Gerätecharakteristika, die physischen Eigenschaften des Netzes und die Charakteristika der Übertragungsschicht werden erwähnt, weswegen davon ausgegangen werden kann, dass keine Nutzerpräferenzen berücksichtigt werden. Die Anpassungskomponente bestimmt die optimale Variante des Inhalts, die entweder als MPEG-1- oder MPEG-4-Datei vorliegen. Eine wirkliche Anpassung findet nicht statt, sondern nur eine Auswahl passender Versionen der Inhalte. Ebenso wenig werden die Anpassungsparameter geeignet erfasst. Der Ansatz kann nicht transparent in bestehende Anwendungen eingegliedert werden.

6.2.7. Fazit

Wenn man die Anforderungen an die Anpassung der Darstellung multimedialer Inhalte aus Abschnitt 6.1 mit den hier vorgestellten Ansätzen vergleicht, erkennt man, dass sie bei keinem Ansatz ausreichend erfüllt werden. Tabelle 6.1 fasst die Beurteilungen zusammen, wobei ein + anzeigt, dass eine Anforderung erfüllt wird, ein –, dass sie nicht erfüllt wird, und ein o bedeutet, dass eine Anforderung nur teilweise oder nur mit umfangreichen Anpassungen des jeweiligen Ansatzes erfüllt wird.

Die meisten vorgestellten Systeme können breitflächig unterschiedliche Medientypen unterstützen. Opera ERA ist stark auf Web-Inhalte ausgelegt, bei denen reine Texte und Bilder dominieren, Niccimon fehlt lediglich die Video-Unterstützung. Der Ansatz aus Abschnitt 6.2.6 unterstützt ausschließlich Videos. Bei der Anpassung werden als Parameter meist Gerätecharakteristika und größtenteils Nutzerpräferenzen berücksichtigt. Obwohl der verwendete MPEG-21-Standard diese Präferenzen unterstützt, wurde diese Möglichkeit in der Dokumentation von dem in Abschnitt 6.2.6 beschriebenen Ansatz nicht diskutiert. Deswegen wird davon ausgegangen, dass sie nicht verwendet werden. Die dynamischen Kontextinformationen sind schwerer zu erfassen als statische und finden deswegen nur bei einigen Ansätzen wie mobileMM4U oder dem wissensbasierten Framework Beachtung. Die in den Anforderungen geforderte Erkennung der Parameter wird außer bei Niccimon nicht betrachtet, die Daten werden meistens als gegeben angenommen. Deswegen findet auch keine Erfassung dieser Daten statt. Die Anpassung der Inhalte ist in den Ansätzen, die lediglich aus einer Menge von vorhandenen Inhalten auswählen, wie SMIL oder dem Ansatz aus Abschnitt 6.2.6, mit keinem großen zeitlichen Aufwand verbunden und effizient. Dies ist auch bei mobileMMU4 der Fall, auf Grund der aufwändigen Erstellung der Präsentationen für jeden einzelnen Nutzer wurde dieser Punkt aber nur mit o bewertet.

Das System, das dem hier angedachten Konzept am ähnlichsten ist, ist das Wissensbasierte Framework. Als einziger Ansatz unterstützt es alle geforderten Methoden zur Anpassung. Niccimon und Opera ERA erlauben lediglich das Skalieren, bei mobileMM4U, dem in Abschnitt 6.2.6 beschriebenen Ansatz und SMIL müssen unterschiedliche Dateien für alle gewünschten Anpassungen vorgehalten werden, was die Ansätze sehr unflexibel macht. Bis auf Opera ERA, das als reiner Webbrowser dient, können alle anderen Systeme breitflächig für unterschiedliche Anwendungsfälle eingesetzt werden. Es ist jedoch immer eine explizite Anpassung der Anwendungen und/oder Dienste nötig, eine transparente Eingliederung ist nicht möglich.

Bis auf SMIL und die Thick Client-Lösung von mobileMM4U arbeiten alle Systeme mit zentral gespeicherten Nutzerdaten. Der Ansatz in Abschnitt 6.2.6 speichert diese Daten nicht permanent, sondern überträgt sie nur bei einer Anfrage. Dies wirkt sich negativ auf die Bewertung bezüglich des Schutzes der Privatsphäre und der Skalierbarkeit aus. In der Regel müssen dadurch umfangreiche Daten über die Luftschnittstelle übertragen werden, lediglich Opera ERA und der Ansatz aus Abschnitt 6.2.6 versuchen aktiv, die

Anforderung	SMIL	Wissensb. Framework	mobileMM4U	Niccimon	Opera ERA	6.2.6
1. Unterstützung unterschiedlicher Medientypen	+	+	+	o	o	−
2. Anpassung gemäß Gerätecharakteristika	o	+	+	o	o	+
3. Anpassung gemäß Nutzerpräferenzen	o	+	+	+	o	−
4. Anpassung gemäß dynamischer Kontextinformationen	o	+	+	−	−	+
5. Erfassen und Erkennen von Anpassungsparametern	−	−	n/a	+	o	−
6. Unterstützung geeigneter Methoden zur Anpassung:						
6a) Skalieren	−	+	−	+	+	−
6b) Konvertieren	−	+	−	−	−	−
6c) Entfernen	−	+	−	−	o	−
6d) Hinzufügen	−	+	−	−	−	−
6e) Ersetzen	−	+	−	−	−	−
7. Anwendungsunabhängigkeit:	+	+	+	+	−	+
8. Geringe Belastung der Luftschnittstelle:	+	−	+/−*	−	o	+
9. Zeitnahe und effiziente Anpassung der Darstellung:	+	o	−	o	o	+
10. Skalierbarkeit:	+	−	+/−*	+	+	−
11. Schutz der Privatsphäre des Nutzers:	+	−	+/−*	−	−	o
12. Transparente Eingliederung in bestehende Anwendungen:	−	−	−	−	−	−

*: Thick/Thin Client

Tabelle 6.1.: Vergleich verwandter Arbeiten mit Anforderungen

zu übertragende Datenmenge zu reduzieren.

6.3. Middleware zur Anpassung der Darstellung multimedialer Inhalte

Für die Kontext-abhängige Anpassung der Darstellung multimedialer Inhalte wurde im Rahmen dieser Arbeit eine Middleware entwickelt [184]. Diese berücksichtigt im Anpassungsprozess statische Kontextinformationen, wie die aktuellen Gerätecharakteristika oder die Nutzerpräferenzen bezüglich der Anpassung und dynamische Kontextinformationen, wie die aktuell verfügbare Bandbreite des genutzten Netzes oder die Umgebungslautstärke. Ziel des im folgenden erläuterten Ansatzes ist nicht die Entwicklung der eigentlichen Werkzeuge, die für die einzelnen Anpassungsmethoden benötigt werden, sondern vielmehr das Bereitstellen eines Rahmens, der eine umfangreiche Anpassung gemäß verschiedener Kontextinformationen und die Erstellung eines Plans für diese Anpassung ermöglicht. Die Werkzeuge können aus der Menge verfügbarer in die Middleware integriert werden. Im Folgenden wird davon ausgegangen, dass eine Menge an geeigneten Werkzeugen zur Anpassung vorhanden ist. Die zur Anpassung benötigten Informationen sowie die Beschreibungen der Medieninhalte sind jeweils in MASL gegeben (siehe Kapitel 3).

6.3.1. Überblick über die Kontext-abhängige Anpassung der Darstellung

Die Anpassung der Darstellung kann Client-seitig oder Server-seitig durchgeführt werden. Für eine Anpassung durch den Server spricht dessen deutlich höhere Rechenleistung, die eine effizientere Verarbeitung der Anpassung erlaubt. Außerdem kann dadurch vermieden werden, dass unnötigerweise umfangreiche Dateien an den Client übertragen werden, die im Rahmen des Anpassungsprozesses in kleinere Dateien konvertiert werden müssen. Für eine Anpassung auf dem Client sprechen hingegen ein besserer Schutz der Privatsphäre des Nutzers, da keinerlei Informationen über ihn oder sein Gerät an eine zentrale Instanz übertragen und verarbeitet werden müssen, und eine bessere Skalierbarkeit, da die Anpassung nicht an einer zentralen Stelle durchgeführt wird. Um die Vorteile beider Ansätze zu kombinieren, wird eine verteilte Verarbeitung der Anpassung vorgeschlagen: der Server nimmt eine Voranpassung des Inhalts vor, der Client beendet den gesamten Anpassungsprozess.

Für diese Verteilung der Anpassung der Darstellung wird von der Middleware für jeden abzurufenden multimedialen Inhalt vom Client ein Anpassungsplan basierend auf Nutzerpräferenzen, Gerätecharakteristika und den dynamischen Kontextinformationen erstellt. Diese Informationen werden in MASL beschrieben. Für die Berücksichtigung insbesondere der dynamischen Informationen muss von der Middleware Client-seitig die aktuelle Situation basierend auf allen verfügbaren statischen und dynamischen Kontext-

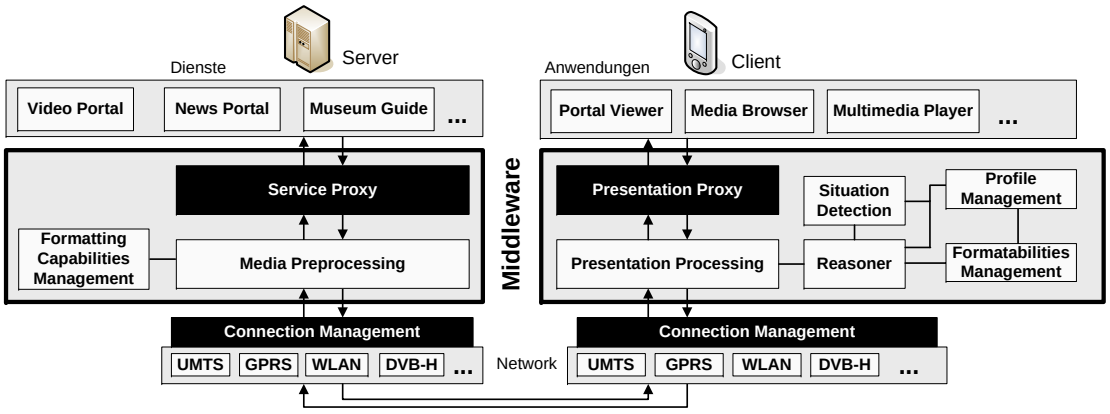

Abbildung 6.5.: Übersicht über die Middleware zur Kontext-abhängigen Anpassung. Die schwarz markierten Komponenten wurden nicht vollständig spezifiziert, da sie nicht im Kernbereich der Kontext-abhängigen Anpassung der Darstellung liegen.

informationen ermittelt und mögliche Änderungen überwacht werden. Diese Situationen werden von der Middleware Regel-basiert verarbeitet: *wenn* eine Situation eintritt, *dann* wird eine bestimmte Anpassung vorgenommen. Regeln können Default-Regeln sein oder vom Nutzer selbst spezifiziert werden.

Der für den Server relevante Teil des erstellten Anpassungsplans wird vom Client übertragen. Der Server nutzt diese Informationen für die Bestimmung der relevanten Schritte der Vorverarbeitung des angefragten Inhalts und schickt den Inhalt bereits teilweise angepasst an den Client. Hier wird der gesamte Anpassungsprozess vollendet und der Inhalt an die Anwendung übergeben, die ihn angefragt hat.

Die Anpassung soll, wie in den Anforderungen gefordert, den Anwendungen auf dem Client bzw. den Diensten auf Seiten des Servers gegenüber transparent gehalten werden. Deswegen wird die Middleware Client-seitig zwischen den Anwendungen und den Komponenten zur Verwaltung der Netzverbindungen positioniert, bzw. analog auf dem Server zwischen dem Verbindungsmanagement und den multimedialen Diensten (siehe Abbildung 6.5). Indem die Middleware geeignete Schnittstellen für Dienste, Anwendungen und das Connection Management anbietet, kann sie leicht in bestehende Architekturen integriert werden.

6.3.2. Komponentensicht Client

Die Client-seitigen Komponenten der Middleware fügen sich zwischen dem *Connection Management* und der Anwendungsschicht ein. Die Komponenten und ihre Funktionen werden im Folgenden näher erläutert.

Connection Management

Die Kommunikation zwischen der Client- und der Server-seitigen Komponente der Middleware erfolgt über das *Connection Management*. Dieses verschattet alle mit der Kommunikation in Zusammenhang stehenden Funktionalitäten wie die Gewährleistung einer korrekten Datenübermittlung, die Adressierung von Client und Server, die Auswahl eines geeigneten Übertragungsmediums sowie die Erstellung von Hin- und Rückkanal für den Datenaustausch. Um eine Blockierung des Clients zu vermeiden, erfolgt die Kommunikation asynchron.

Das *Connection Management* ist im Rahmen dieser Arbeit nicht näher spezifiziert worden, da es für eine Übertragung der Daten, nicht jedoch für die eigentliche Anpassung der Darstellung relevant ist. Es kann beliebig umgesetzt werden, muss jedoch die in Listing 6.1 dargestellte Schnittstelle implementieren, um es den Client-seitigen Komponenten der Middleware zu ermöglichen, Daten und Anfragen an den Server-seitigen Teil zu übermitteln und dessen Antworten zu erhalten.

```
interface IClientConnectionManagement {
 GetContent(in request, in requestID,
      in callbackFunc : Sig(ResponseCallback)) : bool;
 RetrieveList(in callbackFunc) : Sig(ListCallback)) : bool;

 ResponseCallback(in responseData, in requestID);
 ListCallback(in list);
}
```

Listing 6.1: Schnittstelle des Client-seitigen *Connection Managements*

Ein von einer Anwendung angefragter Inhalt wird vom Client-seitigen Teil der Middleware über die `GetContent`-Funktion an das *Connection Management* übergeben. Bei erfolgreicher Übertragung der Anfrage an den Server liefert diese Methode den Wert `true` zurück. Die Anfragen werden durch eindeutige Identifikatoren (`requestID`) unterschieden, um einem später empfangenen Inhalt die Anfrage und damit die anfragende Anwendung zuordnen zu können. Wird vom Server eine Antwort empfangen, wird sie an die Funktion `callbackFunc` übergeben. Diese Funktion muss dabei die Signatur von `ResponseCallback` besitzen, also den Identifikator der Anfrage und die eigentlich zurückgelieferten Daten `responseData` übernehmen können. Zur Anfrage der `Formatabilities`-Liste vom Server wird eine weitere Methode `RetrieveList` benötigt, die ebenfalls bei erfolgreicher Übertragung `true` zurückgibt. Wird die Liste vom Client empfangen, wird sie an eine Rückgabefunktion mit der Signatur `ListCallback` übergeben. Die genaue Funktion dieser Liste wird bei der Erklärung der Komponente *Formatabilities Management* beschrieben.

Presentation Proxy

Der *Presentation Proxy* ist die Client-seitige Kommunikationsschnittstelle der Middleware mit den Anwendungen und hält den Anpassungsprozess für diese transparent. Er sorgt dafür, dass jede Kommunikation der Anwendungen mit einem multimedialen Dienst exklusiv über die Middleware durchgeführt wird. Für die Anwendungen ist die Kommunikation mit dem Proxy identisch mit der direkten Kommunikation mit dem Dienst, weswegen keine Anpassungen an den Anwendungen durchgeführt werden müssen. Falls entsprechende Schnittstellen vorhanden sind, kann der Proxy Statusinformationen der Anwendungen abfragen, die u. U. für den Anpassungsprozess relevant sind, z. B. ob ein Videoplayer gerade ein Video abspielt oder pausiert. Eine Änderung dieses Status kann eine erneute Anpassung nach sich ziehen. Von den Anwendungen kommende Anfragen werden vom *Presentation Proxy* an das *Presentation Processing* weitergereicht.

Da diese Komponente nur am Rande mit dem eigentlichen Anpassungsprozess zu tun hat, wird sie nicht näher betrachtet. Für die unterschiedlichen Arten, wie Anwendungen mit den Diensten kommunizieren, muss der *Presentation Proxy* geeignet spezifiziert werden. Ein Beispiel hierfür ist die Verwendung eines http-Daemons, der auf http `get`-Anfragen auf einem konfigurierten Port des Endgeräts horcht und diese umleitet. Die Schnittstelle des Proxys zu den weiteren Komponenten der Middleware ist in Listing 6.2 zu sehen.

```
interface IPresentationProxy{
 RegisterRequestHandler(in handler : IRequestHandler) : bool;
 UnregisterRequestHandler(in handler : IRequestHandler);

 ReturnContent(in resonseData, in requestID)
}

interface IRequestHandler {
 ProcessRequest(in uri, in requestID) : bool;
}
```

Listing 6.2: Schnittstelle des *Presentation Proxy*

Die *Presentation Processing*-Komponente, die die eintreffende Anfrage innerhalb der Middleware weiterverarbeiten soll, registriert sich zu Beginn als `IRequestHandler` über die `RegisterRequestHandler`-Methode beim *Presentation Proxy*, die bei erfolgreicher Anmeldung `true` zurückliefert. Die Abmeldung erfolgt analog über `UnregisterRequestHandler`.

Der von der Anwendung angefragte Inhalt wird über eine eindeutige `uri` identifiziert. Um die Anfrage einer Anwendung zuordnen zu können, wird eine `requestID` vergeben. Bei einer eingehenden Anfrage wird die `uri` des Inhalts und die `requestID` an die `ProcessRequest`-Funktion des `IRequestHandlers`, also des *Presentation Processing* übergeben. Wurde ein angefragter Inhalt durch die Middleware abgerufen und entspre-

chend angepasst, wird er zusammen mit der `requestID` der zugehörigen Anfrage über die `ReturnContent`-Methode an den *Presentation Proxy* zurückgegeben.

Profile Management

Im *Profile Management* werden die Kontextinformationen des Nutzers als MASL-Beschreibungen verwaltet, die insbesondere die Nutzerpräferenzen, die Gerätecharakteristika, die dynamischen Kontextinformationen zum Zeitpunkt der Dienstnutzung und die Anpassungsregeln enthalten. Der Nutzer kann mittels Präferenzen angeben, ob er z. B. Untertitel auditiven Informationen vorzieht, oder Präferenzen bezüglich des Schutzes der Privatsphäre spezifizieren. Diese bestimmen, welche Daten des Profils in welchen Situationen nicht an den Server übertragen werden dürfen. Bei den Gerätecharakteristika handelt es sich um überwiegend statische Informationen wie die Displaygröße. Diese Informationen ändern sich in der Regel nicht während der Dienstnutzung, müssen aber z. B. bei der Installation von neuen Codecs oder Anwendungen angepasst werden. Dynamische Kontextinformationen beinhalten die aktuelle Nutzungsumgebung und dynamische Informationen über das Endgerät wie den aktuellen Akkustand, angeschlossene externe Geräte wie Kopfhörer oder die verfügbare Bandbreite des Netzes.

Im MASL-Profil werden die Anpassungsregeln der Middleware spezifiziert. Diese können über eine Nutzerschnittstelle durch den Nutzer eingetragen, bearbeitet oder gelöscht werden. Um die Nutzerfreundlichkeit zu erhöhen, können vordefinierte Standard-Regeln importiert werden. Diese spezifizieren, wie sich das System in welcher Situation verhalten soll. Der Aufbau der Anpassungsregeln wurde in Abschnitt 3.4.2 erläutert.

Die im *Profile Management* verwalteten Informationen werden von der *Situation Detection* zur Erkennung der aktuellen Situation weiterverarbeitet. Die `Reasoner`-Komponente greift auf die Anpassungsregeln zu, um Anpassungspläne zu erstellen.

Formatabilities Management

Bei der hier vorgeschlagenen verteilten Anpassung kann es zu Problemen kommen, wenn dem Client nicht bekannt ist, was für Möglichkeiten der Skalierung und Konvertierung auf dem Server gegeben sind. Ebenso kann der Server nicht nach Alternativen zum Anpassungsplan suchen, wenn er die Anpassungsmöglichkeiten und unterstützten Formate des Clients nicht kennt. Um dieses Problem zu lösen, muss bei der initialen Anmeldung des Clients am Server-seitigen Teil der Middleware zunächst für alle Medientypen und –formate die Schnittmenge der Client-seitig darstellbaren und Server-seitig von den unterschiedlichen Anpassungsmöglichkeiten unterstützten Formate bestimmt werden. Zu diesen Server-seitigen Formaten gehören jene Formate, die durch Konvertierung oder Skalierung entstehen können. Dazu verwaltet der Server eine in XML spezifizierte `FormattingCapabilities`-Liste, die alle verfügbaren Anpassungsoperationen enthält. Im Client selektiert der *Reasoner* basierend auf den von ihm selbst unterstütz-

ten Formaten aus dieser Liste alle passenden Anpassungsoperationen und erstellt eine `Formatabilities`-Liste, die vom *Formatabilities Management* verwaltet wird. Hierbei werden auch Formate berücksichtigt, die durch das Hintereinanderschalten von mehreren Anpassungsoperationen entstehen. Ist diese Liste leer, kann keine Anpassung erfolgen und die Inhalte werden unverändert übertragen. Der Client kann aber den Server darüber informieren, damit geeignete Operationen nachinstalliert werden. In der Regel tritt dieser Fall jedoch eher selten auf, da man davon ausgehen kann, dass ein System zur Anpassung alle gängigen Formate geeignet unterstützt.

Ein Beispiel für eine `FormattingCapabilities`-Liste ist in Listing 6.3 zu sehen. Innerhalb des `FormattingCapabilities`-Tag können mehrere `Conversion` und `Scaling`-Tags enthalten sein, die jeweils eine Möglichkeit zur Konvertierung bzw. Skalierung definieren. Das `Conversion`-Tag beinhaltet die Attribute `name` zur Unterscheidung der einzelnen Operationen, `from` zur Angabe des Ursprungsformats und `to` zur Angabe des Zielformats der Konvertierung. `Scaling` besitz ebenfalls ein `name` Attribut und ein `format`-Attribut zur Angabe des skalierbaren Formats. Über das `scale`-Attribut wird die Größe bestimmt, bezüglich der skaliert werden soll, also z. B. `display` für die Skalierung bezüglich der Displaygröße. Die `Formatabilities`-Liste enthält eine Teilmenge dieser Liste und ersetzt die umschließenden `FormattingCapabilities`-Tags durch `Formatabilities`-Tags. Die Schema-Definitionen der `FormattingCapabilities`- und `Formatabilities`-Liste sind in Listing A.43 Anhang A.4 zu finden.

```
<FormattingCapabilities>
  <Conversion name="WmvToMpeg2" from="video/wmv" to="video/mpeg2"
     />
  <Conversion name="Mpeg2ToWmv" from="video/mpeg2" to="video/wmv"
     />
  <Conversion name="Mpeg2ToMpeg4" from="video/mpeg2" to="video/
     mpeg4" />
  <Conversion name="Mpeg4ToMpeg2" from="video/mpeg4" to="video/
     mpeg2" />
  <Conversion name="WmvToMpeg4" from="video/wmv" to="video/mpeg4"
     />
  <Conversion name="Mpeg4ToWmv" from="video/mpeg4" to="video/wmv"
     />
  <Scaling name="Mpeg2" format="video/mpeg2" scale="display"/>
  <Scaling name="Mpeg4" format="video/mpeg4" scale="display"/>
  <Scaling name="Wmv" format="video/wmv" scale="display"/>
</FormattingCapabilities>
```

Listing 6.3: `Formatabilities`

Situation Detection

Die *Situation Detection* verarbeitet die vom *Profile Management* verwalteten Informationen, um die aktuelle Situation zu bestimmen, in der sich der Nutzer bzw. sein verwendetes Endgerät befindet. Die Komponente kann, wenn nötig, auf die in Abschnitt 5.5.3 und 5.5.4 vorgestellten Ansätze zur Bestimmung der Ähnlichkeit von Kontextinformationen zurückgreifen. Die bestimmte Situation wird an den *Reasoner* übergeben, um sie mit den Anpassungsregeln abzugleichen und zu bestimmen, welche Inhalte wie angepasst werden sollen. Die Komponente erlaubt die Registrierung des *Reasoners* für Benachrichtigungen bei Situationsänderungen.

Reasoner

Der *Reasoner* erhält über die Komponente *Situation Detection* Informationen über die aktuelle Situation und führt einen Abgleich zwischen den durch das *Profile Management* verwalteten Anpassungsregeln und der erkannten Situation durch. Basierend auf diesem Abgleich und der `Formatabilities`-Liste werden passende Regeln zu einem Leitfaden für die Anpassung der Inhalte zusammengefasst. Dieser gibt allgemein an, welche Inhalte wie angepasst werden, unabhängig davon, ob ein solcher Inhalt tatsächlich abgerufen wird. Er besteht aus einem Auszug der Anpassungsregeln spezifiziert in MASL.

Wird von einer Anwendung ein Inhalt angefragt, leitet die *Presentation Processing* Komponente die URI des Medieninhalts an die *Reasoner* Komponente weiter. Aus der Dateiendung kann der Typ des Inhalts und daraus die passende Regel des Leitfadens bestimmt werden. Diese wird benötigt, um für die Anpassung eines abzurufenden Medieninhalts basierend auf dem Leitfaden den tatsächlichen Anpassungsplan zu erzeugen. Dieser beinhaltet zusätzlich zu den relevanten Informationen des Leitfadens genaue Anweisungen, welche Anpassung auf dem Server und welche auf dem Client in welcher Reihenfolge durchgeführt werden sollen. Das wird maßgeblich durch die `PrivacyPreferences` und die `FormattingCapabilities`-Liste des Servers bestimmt. Aus Effizienzgründen werden maximal viele Anpassungsprozesse auf den Server ausgelagert. Dieser Plan wird an die *Presentation Processing* Komponente weitergeleitet.

Der *Reasoner* kann sich bei den Komponenten *Profile Management* und *Situation Detection* für Änderungen an Anpassungsregeln bzw. der aktuellen Situation registrieren um zeitnah auf Änderungen reagieren zu können.

Presentation Processing

Die eigentliche Anpassung der Inhalte basiert auf dem vom *Reasoner* erstellten Anpassungsplan und wird Client-seitig vom *Presentation Processing* durchgeführt. Diese Komponente ruft den Anpassungsplan ab, sobald der *Presentation Proxy* eine Anfrage nach einem Inhalt stellt. Der vom Server-seitigen Teil der Middleware benötigte Auschnitt des Anpassungsplans wird extrahiert und mit der Anfrage an diesen gesendet. Wird ein

Inhalt vom Server an den Client übertragen, vollendet das *Presenataion Processing* die Anpassung gemäß Anpassungsplan.

6.3.3. Komponentensicht Server

Die Server-seitigen Komponenten der Middleware fügen sich zwischen dem *Connection Management* und den multimedialen Diensten ein. Die Komponenten und ihre Funktionen werden im Folgenden näher erläutert.

Connection Management

Das *Connection Management* dient zur Kommunikation mit dem Client-seitigen Teil der Middleware. Es arbeitet reaktiv, empfängt also Anfragen über das Client-seitige *Connection Management* und versendet entsprechende Antworten. Alle mit der Kommunikation in Zusammenhang stehenden Funktionalitäten wie die Gewährleistung einer korrekten Datenübermittlung, die Auswahl eines Übertragungsmediums sowie die Erstellung von Hin- und Rückkanal für den Datenaustausch werden hier verschattet. Für jeden Client existiert eine eigene Instanz der Komponenten *Connection Management*, *Media Preprocessing* und *Service Proxy* auf dem Server, um parallele Anfragen mehrerer Clients zu unterstützen.

Das Server-seitige *Connection Management* wird im Rahmen dieser Arbeit nicht näher spezifiziert. Es muss lediglich die in Listing 6.4 spezifizierte Schnittstelle implementieren, um es den Komponenten der Middleware zu ermöglichen, Client-Anfragen zu empfangen und entsprechende Antworten zu generieren.

```
interface IServerConnectionManagement {
 RegisterRequestHandler(in handler : IRequestHandler) : bool;
 UnregisterRequestHandler(in handler : IRequestHandler);

 RegisterListHandler(in handler : IListRequestHandler) : bool;
 UnregisterListHandler(in handler : IListRequestHandler);
}

interface IRequestHandler {
 ProcessRequest(in request) : Response;
}

interface IListRequestHandler {
 RetrieveList();
}
```

Listing 6.4: Schnittstelle des Server-seitigen *Connection Managements*

Die zugehörige Instanz der *Media Preprocessing* Komponente, die auf der Seite des Servers eine eingehende Anfrage für einen Client weiterverarbeitet, muss sich zunächst beim

Connection Management über die Funktion `RegisterRequestHandler` als `IRequestHandler` registrieren. War diese Anmeldung erfolgreich, liefert die Methode `true` zurück. Die Abmeldung einer Komponente erfolgt über die Methode `UnregisterRequestHandler`. Empfängt das *Connection Management* eine Anfrage, wird diese mittels der Methode `ProcessRequest` synchron an den registrierten `IRequestHandler`, also die zuständige Instanz der *Media Preprocessing* Komponente, weitergeleitet. Diese liefert als Antwort (`Response`) an den Client den vorverarbeiteten multimedialen Inhalt zurück.

Analog kann sich das *Media Preprocessing* für Anfragen nach der `FormattingCapabilities`-Liste als `IListRequestHandler` registrieren bzw. deregistrieren, diese werden an die `RetrieveList`-Methode übergeben und anschließend an das *Formatting Capabilities Management* weitergeleitet.

Service Proxy

Ähnlich zum *Presentation Proxy* dient der *Service Proxy* als Server-seitige Kommunikationsschnittstelle. Er bezieht die Medieninhalte von den Anbietern der multimedialen Inhalte, die von den Anwendungen angefragt wurden. Er kommuniziert auf dieselbe Weise wie die Anwendung an sich, wodurch keine Änderungen bei den Dienstanbietern benötigt werden. Da diese Komponente nicht in den eigentlichen Anpassungsprozess involviert ist, wird sie nicht näher betrachtet. Ihre Schnittstelle ist in Listing 6.5 spezifiziert. Der eindeutige Identifikator des Inhalts `uri` wird einer `GetContent`-Methode übergeben, die den entsprechenden Inhalt zurückliefert.

```
interface IServiceProxy{
 GetContent(in uri): Content;
}
```

Listing 6.5: Schnittstelle des *Presentation Proxy*

Media Preprocessing

Innerhalb dieser Komponente findet die Server-seitige Anpassung der Darstellung basierend auf den Informationen des Anpassungsplans statt, der mit der Anfrage übertragen wird. Da auf dem Server immer nur der aktuelle Anpassungsplan und keine Anpassungshistorie gespeichert wird, kann ein gewisses Maß an Schutz der Privatsphäre gewährleistet werden.

Formatting Capabilities Management

Die für die Bestimmung der `Formatabilities`-Liste benötigte `FormattingCapabilities`-Liste des Servers wird in dieser Komponente verwaltet. Bei der initialen Verbindungsaufnahme zwischen Client und Server wird sie an den Client übertragen.

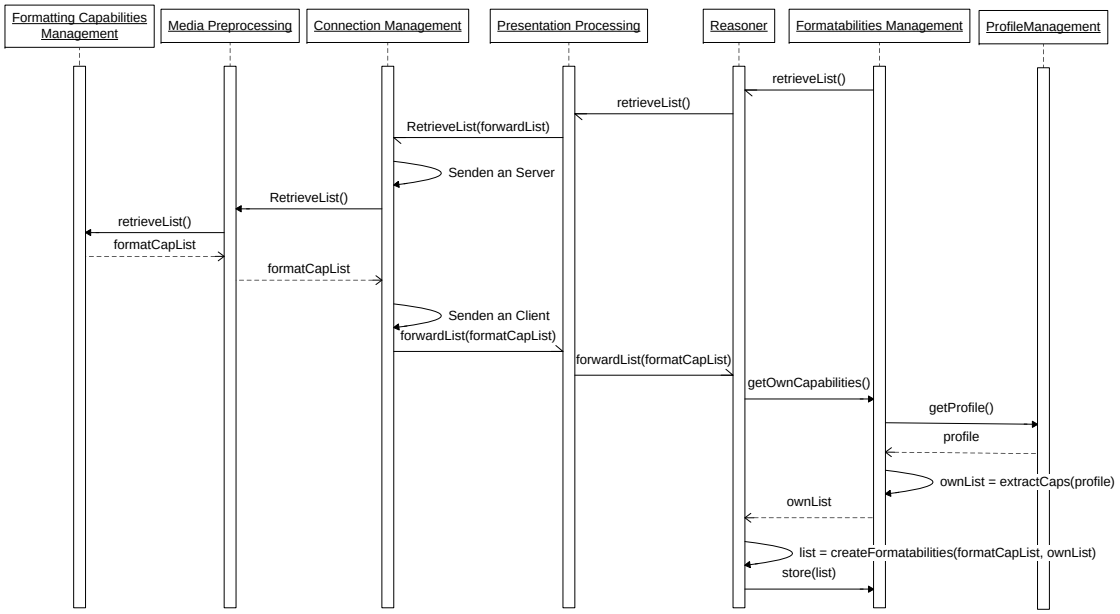

Abbildung 6.6.: Erzeugung der `Formatabilities`-Liste

6.3.4. Dynamische Sicht

Nachfolgend werden die wichtigsten Abläufe innerhalb der Middleware aufgezeigt und die Zusammenarbeit der Komponenten näher erläutert.

Erzeugung der Formatabilities-Liste

Meldet sich ein Client erstmalig bei einem Server an, ist die vom *Formatabilities Management* verwaltete `Formatabilities`-Liste leer. Deswegen wird die Anfrage nach der `Formatting Capabilities`-Liste von dieser Komponente asynchron über die Komponenten *Reasoner* und *Presentation Processing* an das *Connection Management* übergeben, an den Server übertragen und dort (synchron) über die zugehörige Instanz der *Media Preprocessing* Komponente an das *Formatting Capabilities Management* übergeben. Der Server sendet die dort verwaltete Liste an den Client zurück. Über die zuvor angegebene Rückgabefunktion `forwardList` wird die vom *Connection Management* empfangene Liste an das *Presentation Processing* übergeben, das es an den *Reasoner* weiterleitet. Dieser erfragt über das *Formatabilities Management* die Liste der auf dem Endgerät verfügbaren Formate, die aus dem MASL-Profil ausgelesen werden kann, gleicht diese mit der vom Server erhaltenen Liste ab und erzeugt die `Formatabilities`-Liste. Diese wird zur Speicherung an das *Formatabilities Management* übergeben. Abbildung 6.6 stellt diesen Ablauf als Sequenzdiagramm dar.

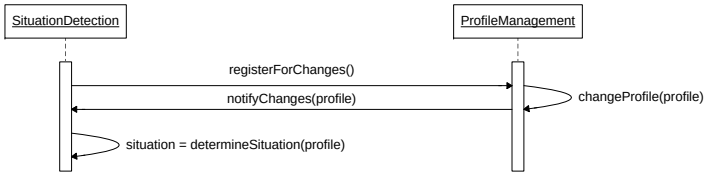

Abbildung 6.7.: Ablauf der Situationserkennung

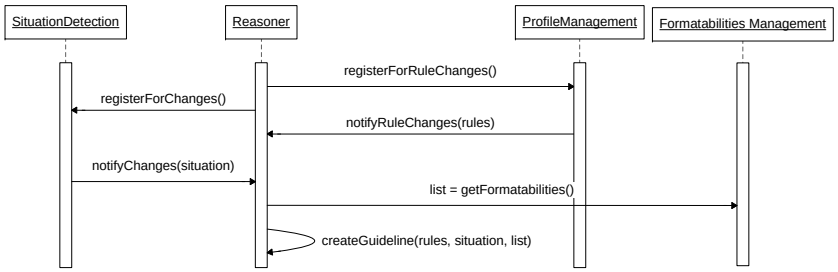

Abbildung 6.8.: Ablauf der Erstellung des Leitfadens zur Anpassung

Situationserkennung

Für die Erkennung der aktuellen Situation benötigt die *Situation Detection*-Komponente die vom *Profile Manager* verwalteten Informationen, also dynamische Kontextinformationen, Gerätecharakteristika und Nutzerpräferenzen. Um Änderungen an diesen Daten zeitnah zu erfassen, kann sie sich registrieren, so dass sie bei jeder Änderung des Profils eine Benachrichtigung erhält. Erhält sie vom *Profile Management* die aktuellen Profilinformationen, wird die aktuelle Situation neu berechnet. Der gesamte Ablauf ist in Abbildung 6.7 zu sehen.

Erstellung des Leitfadens zur Anpassung

Unabhängig von der Anfrage eines konkreten Inhalts, hält der *Reasoner* permanent einen Leitfaden zur Anpassung parat. Dieser umfasst eine Teilmenge der Anpassungsregeln gegeben in MASL, die der aktuellen Situation, bestimmt durch die *Situation Detection*, entsprechen. Der *Reasoner* registriert sich bei beiden Komponenten, um über Änderungen an der aktuellen Situation oder an den Anpassungsregeln benachrichtigt zu werden. Erhält er eine solche Benachrichtigung, bezieht der die `Formatabilities`-Liste vom `Formatabilities Management` und berechnet den aktuell gültigen Leitfaden (siehe Abbildung 6.8).

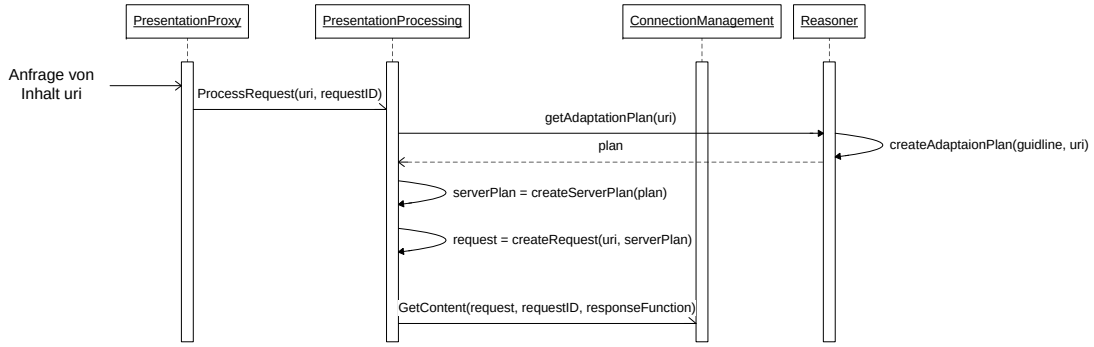

Abbildung 6.9.: Ablauf der Anfrage eines Inhalts

Anfrage eines Inhalts

Versucht eine Anwendung einen Inhalt abzufragen, wird diese Anfrage vom *Presentation Proxy* abgefangen und ihr eine eindeutige `requestID` zugewiesen. Diese Anfrage wird an das zuvor beim *Presentation Proxy* registrierte *Presentation Processing* über die registrierte Methode `ProcessRequest` weitergeleitet. Das *Presentation Processing* übergibt die URI des angefragten Inhalts an den *Reasoner*, der die Dateiendung nutzt, um aus dem Leitfaden die relevanten Informationen für den Anpassungsplan zu extrahieren und diesen zu erstellen. Das *Presentation Processing* bestimmt den Server-seitigen Teil des Planes, verpackt ihn mit der URI in eine Anfrage an den Server und übergibt die Anfrage zum Versenden an das *Connection Management* (siehe Abbildung 6.9).

Server-seitige Anpassung des Inhalts

Nachdem der Server die Anfrage erhalten hat, wird diese vom *Connection Management* über die registrierte `ProcessRequest`-Methode an die Instanz der *Media Preprocessing*-Komponente übergeben. Diese extrahiert den Server-seitigen Anpassungsplan sowie die URI des abzurufenden Inhalts. Über die `GetContent`-Methode der *Server Proxy*-Schnittstelle wird die URI weitergereicht, der zugehörige Inhalt beim Dienstanbieter angefragt und an das *Media Preprocessing* übergeben. Hier wird gemäß des Anpassungsplans der Inhalt geeignet vorformatiert und schließlich über das *Connection Management* zurück an den Client gesendet. Der gesamte Ablauf ist in Abbildung 6.10 dargestellt.

Client-seitige Anpassung des Inhalts

Wird der vom Server vorformatierte Inhalt vom *Connection Management* des Client empfangen, wird er an die bei der Anfrage spezifizierte Rückgabemethode der *Presentation Processing*-Komponente übergeben. Diese nutzt die `requestID` um den zugehörigen Plan zu bestimmen und vollendet den Anpassungsprozess. Der angepasste Inhalt wird an den

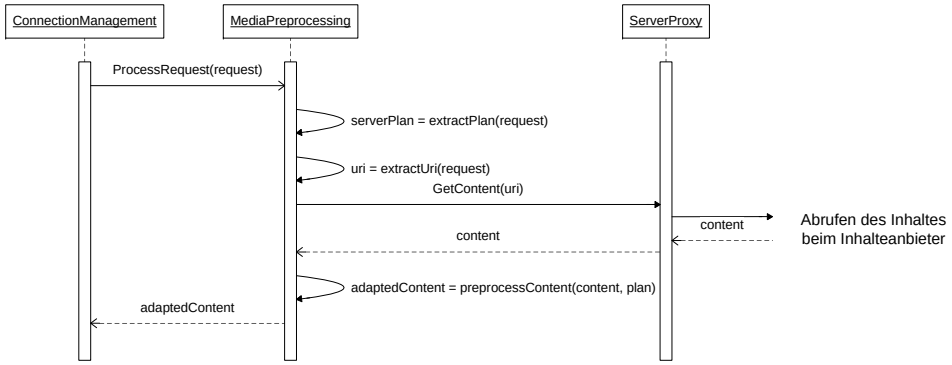

Abbildung 6.10.: Ablauf der Server-seitigen Anpassung des Inhalts

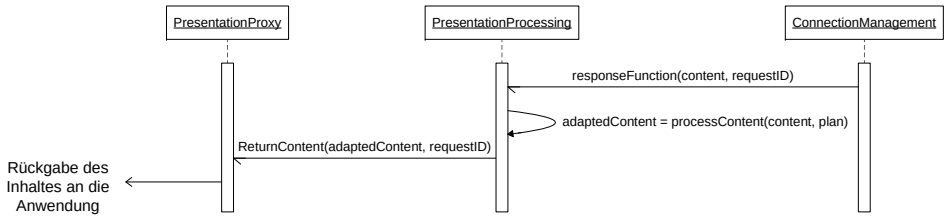

Abbildung 6.11.: Ablauf der Client-seitigen Anpassung des Inhalts

Presentation Proxy übergeben, der die `requestID` nutzt um die zugehörige Anwendung zu bestimmen. An diese wird der Inhalt schließlich ausgeliefert (siehe Abbildung 6.11).

6.4. Bewertung des eigenen Ansatzes

Um die in diesem Kapitel vorgestellten Ansätze zur Kontext-abhängigen Anpassung der Darstellung multimedialer Inhalte zu bewerten, werden die Anforderungen verwendet, die in Abschnitt 6.1 identifiziert wurden.

1. *Unterstützung unterschiedlicher Medientypen*
   Die in diesem Kapitel vorgestellte Middleware ist speziell für die Anwendung mit der in Kapitel 3 definierten Sprache MASL gedacht, die die Beschreibung unterschiedlicher Medientypen unterstützt. Für jeden Typ müssen geeignete Methoden zur eigentlichen Anpassung bereitgestellt werden, dies war jedoch nicht Ziel der Entwicklung der Middleware und wurde vorausgesetzt. Aus der Architektur der Middleware, den internen Abläufen und dem Aufbau der Anpassungsregeln für die Inhalte ergeben sich keine Einschränkungen bezüglich der unterstützen Medientypen.

2. *Anpassung gemäß Gerätecharakteristika*
   MASL erlaubt die Beschreibung der Charakteristika unterschiedlicher mobiler Endgeräte. Diese Charakteristika umfassen überwiegend statische Kontextinformationen, die das Gerät beschreiben, wie die Display-Größe, die Speichergröße, die Rechenleistung, die vom Gerät unterstützten Netze, die vorhandenen Eingabe- bzw. Ausgabemöglichkeiten, die vorhandenen Codecs zur Anzeige von Medieninhalten, die installierten Anwendungen und weitere gerätespezifische Ausstattungsmerkmale. Aus der großen Menge an Charakteristika ergibt sich eine Vielzahl von Bedingungen, die bei der Anpassung der Darstellung multimedialer Inhalte beachtet werden müssen. Da die Middleware mittels der MASL-Beschreibungen entscheidet, wann und wie ein Inhalt angepasst werden soll, kann sie diese Informationen bei der Erstellung des Anpassungsplanes für einen Inhalt nutzen.

3. *Anpassung gemäß Nutzerpräferenzen*
   Mittels MASL können durch den Nutzer Präferenzen bezüglich der Anpassung der Darstellung spezifiziert werden. So kann ein Nutzer z. B. die Präferenz haben, dass ihm Texte vorgelesen werden sollen, ein anderer Nutzer möchte gesprochene Passagen als Untertitel zur Verfügung gestellt haben. Da die Middleware mittels der MASL-Beschreibungen entscheidet, wann und wie ein Inhalt angepasst werden soll, kann sie diese Informationen auch bei der Erstellung des Anpassungsplans für einen Inhalt nutzen.

4. *Anpassung gemäß dynamischer Kontextinformationen*
   Neben den statischen Kontextinformationen Gerätecharakteristika und Nutzerpräferenzen, soll der dynamische Kontext berücksichtigt werden. Auch dieser findet sich in den in MASL formulierten Beschreibungen. Sowohl Nutzer- als auch Geräteabhängige Kontextinformationen werden unterstützt. Da die Middleware mittels der MASL-Beschreibungen entscheidet, wann und wie ein Inhalt angepasst werden soll, kann sie diese Informationen bei der Erstellung des Anpassungsplans für einen Inhalt nutzen.

5. *Erfassen und Erkennen von Anpassungsparametern*
   Die Anpassung der Darstellung soll abhängig sein von Parametern, die durch die Gerätecharakteristika, die Nutzerpräferenzen und die aktuellen Werte der dynamischen Kontextinformationen bestimmt werden. Alle diese Informationen werden als Beschreibungen in MASL gegeben, die Middleware muss nicht sich ändernde Anpassungsparameter selbst erkennen können.

   Die Gerätecharakteristika ändern sich in der Regel nicht, sie müssen einmalig bei der Initialisierung des Systems aus der Geräteverwaltung des Betriebssystems ausgelesen werden. Dennoch kann das *Profile Management* dieses regelmäßig nach geänderten Charakteristika überprüfen.

Durch die explizite Eingabe neuer Regeln oder dem Import von neuen Default-Regeln durch den Nutzer mittels eines User Interfaces können sich die Nutzerpräferenzen ändern. Die mit diesen Daten arbeitenden Komponenten werden darüber informiert und ihnen die neuen Daten übergeben. Basierend darauf werden die Anpassungspläne neu errechnet.

Die dynamischen Kontextinformationen werden von Sensoren erfasst und in MASL geeignet beschrieben. Kommt es zu umfangreichen Änderungen, müssen die darauf arbeitenden Komponenten davon benachrichtigt werden.

6. *Unterstützung geeigneter Methoden zur Anpassung*
   Die Anpassung der Darstellung kann auf unterschiedliche Weise durchgeführt werden. Vermieden werden sollen die Bereitstellung eines identischen Medieninhalts in unterschiedlichen Formaten und die Auswahl einer den Anpassungsparametern entsprechenden Variante. Wenn man die durch MASL unterstützen Parameter betrachtet, erkennt man, dass dieser Ansatz zu einem stark erhöhten Ressourcenbedarf führt. Die vorgestellte Middleware geht von einer expliziten Anpassung der Inhalte aus.

   Ziel der Entwicklung der Middleware war es nicht, die eigentlichen Werkzeuge zur Anpassung bereitzustellen oder zu entwickeln, sondern einen Rahmen zu bieten, der eine umfangreiche Anpassung gemäß verschiedener Kontextinformationen ermöglicht und einen Anpassungsplan geeignet erzeugt. Bei der Erstellung des Planes werden Verfahren zur Konvertierung und Skalierung sowie zum Entfernen, Hinzufügen oder Ersetzen von einzelnen Komponenten der Inhalte explizit berücksichtigt. Die Methoden können sequentiell hintereinander geschaltet werden.

7. *Anwendungsunabhängigkeit*
   Die vorgeschlagene Middleware ist nicht für den Einsatz in einem speziellen Szenario entwickelt worden und kann generisch für unterschiedliche Anwendungen eingesetzt werden. Dies wird zusätzlich durch den Einsatz von MASL innerhalb der Middleware unterstützt, da diese keine Einschränkungen bezüglich Medientyp oder Anwendung macht.

8. *Geringe Belastung der Luftschnittstelle*
   Bei multimedialen Diensten wird die Belastung der Luftschnittstelle durch die Größe der zu übertragenden Datei bestimmt. Durch die Verteilung des Anpassungsprozesses auf einen Server- und einen Client-seitigen Teil wird diese Belastung reduziert. So kann vermieden werden, große Dateien, die für die Anwendung kleiner skaliert werden sollen, erst auf den Client zu übertragen und dann zu reduzieren, wie es bei Client-seitiger Anpassung der Fall wäre. Dafür muss zusätzlich der Anpassungsplan an den Server übertragen werden. Dieser verbraucht jedoch bei der Übertragung bedeutend weniger Ressourcen als ein multimedialer Inhalt.

9. *Zeitnahe und effiziente Anpassung der Darstellung*
   In der Regel ist der Nutzer nicht bereit, auf die Anpassung der Darstellung zu warten. Von daher senkt eine gute, allerdings zeitintensive Anpassung die Nutzerzufriedenheit. Basierend auf der aktuellen Situation wird der Anpassungsleitfaden permanent aktuell gehalten. Wird ein konkreter Inhalt angefragt, müssen nur die relevanten Teile des Leitfadens extrahiert werden. Die Bestimmung der Situation und der darauf basierenden Regeln bei jedem angefragten Inhalt wird dadurch vermieden und der Anpassungsprozess effizient und schnell gestaltet.

10. *Skalierbarkeit*
    Die Verteilung des Anpassungsprozesses auf Client und Server wirkt sich positiv auf die Skalierbarkeit des Gesamtsystems aus. Die zeitintensiven Vorgänge zur Erstellung des Leitfadens finden auf den Clients statt. Da die eigentliche Anpassung nur teilweise auf einer zentralen Komponente durchgeführt wird, können hier deutlich höhere Nutzerzahlen bedient werden. Eine geeignete Strategie zur Replikation der Server kann dies noch zusätzlich unterstützen.

11. *Schutz der Privatsphäre des Nutzers*
    Die in diesem Kapitel vorgestellte Middleware passt die Inhalte verteilt an. Die Nutzerprofile werden exklusiv auf Seiten des Clients gehalten und nicht an den Server übertragen. Wird ein Inhalt angefragt, muss der für den Server relevante Teil des Anpassungsplans übertragen werden. Dieser kann zumindest teilweise Hinweise auf die verwendeten Anpassungsparameter geben. Würde der Server über einen längeren Zeitraum die Anpassungspläne speichern, könnte man aus der Summe der Pläne kritische Informationen über den Nutzer ableiten. Die gesendeten Pläne werden jedoch vom Server nicht gespeichert. Dadurch wird ein akzeptabler Schutz der Privatsphäre erreicht.

12. *Transparente Eingliederung in bestehende Anwendungen*
    Um die Anwendungsunabhängigkeit noch zu erhöhen, lässt sich die vorgestellte Middleware transparent in bestehende Dienste und Anwendungen eingliedern. Das bedeutet, dass sie verwendet werden kann, ohne dass an diesen Anpassungen nötig werden oder sie überhaupt die Anpassung der Darstellung bemerken. Dies wird durch die Komponenten *Presentation Proxy* und *Service Proxy* ermöglicht. Der *Presentation Proxy* fängt Anfragen nach Inhalten von Anwendungen ab und leitet diese an die Middleware weiter, der *Service Proxy* stellt die Anfrage an die Dienste. Anwendungen und Dienstanbietern erscheint die Kommunikation über die Middleware wie eine direkte Kommunikation, weswegen keine Änderungen erforderlich sind. Im Rahmen dieser Arbeit wurden die Komponenten nicht näher konzipiert, sondern nur die benötigten Schnittstellen spezifiziert, da sie nicht im Kernbereich der Kontext-abhängigen Personalisierung liegen.

6.5. Zusammenfassung

In diesem Kapitel wurden die wichtigsten Anforderungen an ein System zur Kontext-abhängigen Anpassung der Darstellung multimedialer Inhalte aufgestellt. Es wurden verwandte Arbeiten aus diesem Gebiet vorgestellt und gemäß der angeführten Anforderungen bewertet.

Kern dieses Kapitels bildete die Vorstellung einer Middleware, die die Anpassung der Inhalte gemäß Gerätecharakteristika, Nutzerpräferenzen und dynamischer Kontextinformationen unterstützt. Diese arbeitet nach einem verteilten Ansatz: während der Serverseitige Teil eine Voranpassung der Inhalte durchführt, beendet der Client den gesamten Anpassungsprozess. Bei dieser Verteilung kann das Problem auftreten, dass der Client einen Anpassungsplan erzeugt, der jedoch vom Server nicht durch geeignete Methoden unterstützt wird. Deswegen werden bei der initialen Anmeldung die vom Server unterstützten Methoden übertragen und eine `Formabilities`-Liste erzeugt, die bei der Planerstellung berücksichtigt wird. Die Middleware erzeugt umfangreiche Anpassungspläne für die multimedialen Inhalte. Dadurch dass eine tatsächliche Anpassung der Inhalte durch Methoden wie Skalierung oder Konvertierung durchgeführt wird, müssen nicht dieselben Inhalte in unterschiedlichen Formaten vorgehalten werden.

Kapitel 7.

Implementierung und Bewertung der vorgestellten Konzepte

Im diesem Kapitel wird die Implementierung der vorgestellten Konzepte erläutert und eine abschließende Bewertung des Gesamtkonzepts durchgeführt.

7.1. Evaluierung mittels prototypischer Implementierung

Die in den letzten Kapiteln vorgestellten Konzepte wurden prototypisch implementiert. Es entstand kein Gesamtsystem, sondern eine Menge einzelner Prototypen, die jeweils einen anderen Aspekt des Gesamtkonzepts evaluieren. Diese werden im Folgenden vorgestellt. Im Anschluss wird diskutiert, wie die Einzelsysteme geeignet kombinierbar sind.

7.1.1. Erzeugung von Schlüsselwörtern

Das Konzept des automatischen und nutzergestützten Taggings aus Abschnitt 4.1 wurde mit Hilfe des *Flex-Frameworks*[1] und *Actionscript 3.0*[2] implementiert, da diese eine schnelle und unkomplizierte Umsetzung ermöglichen. Der Prototyp unterstützt als Ressourcen Texte und Bilder, kann aber leicht um andere Medientypen erweitert werden, falls ein geeignetes externes Werkzeug zur Inhaltsanalyse zur Verfügung steht.

Über eine graphische Oberfläche kann der Nutzer Inhalte einfügen, mit Schlüsselwörtern versehen, den Prozess zum automatischen Taggen anstoßen oder Tags bewerten. Sollen Schlüsselwörter automatisch erzeugt werden, kommuniziert der Prototyp mit einem passenden Drittanbieter für die Inhaltsanalyse. Texte werden mit *OpenCalais*[3] analysiert, das einen Datenaustausch über das SOAP-Protokoll erlaubt. Der Dienst schickt als Rückantwort eine XML-Datei, die neben den vorgeschlagenen Schlüsselwörtern auch zugehörige Relevanzwerte enthält. Diese geben an, als wie wichtig ein Schlüsselwort für den

[1] www.adobe.com/de/products/flex/
[2] livedocs.adobe.com/flash/9.0_de/ActionScriptLangRefV3/
[3] www.opencalais.com

untersuchten Text eingestuft wird. Außerdem wird gezählt, wie oft das Schlüsselwort im Text vorkommt. Für die Analyse von Bildern wird *Alipr*[4] verwendet. Der Datenaustausch erfolgt über http-Requests. Als Antwort wird ebenfalls eine XML-Datei zurückgeliefert, die neben den Schlüsselwörtern die errechneten Relevanzwerte enthält. Die Rückgabewerte der externen Dienste werden für die Phase der semantischen Analyse weiter verwendet.

Auf das von *WordNet*[5] zur Verfügung gestellte Wörterbuch kann über das *Java WordNet Interface* (JWI)[6] zugegriffen werden. Es erlaubt Anfragen nach Wortbeziehungen wie Synonymen, Homonymen oder Meronymen in Java-Syntax. Im Prototyp wird ein lokal gehaltenes Abbild des aktuellen *WordNet* Thesaurus genutzt. Der zweite Schritt der semantischen Analyse kann wie in Abschnitt 4.1 beschrieben unter Verwendung von *YAGO* oder *DBpedia* umgesetzt werden. Er wurde jedoch im Prototyp nicht implementiert, da er keine weiteren Erkenntnisse für die Evaluierung geliefert hätte. Dem Nutzer, der den automatischen Prozess gestartet hat, wird eine Liste mit den extrahierten Schlüsselwörtern und ihren errechneten Bewertungen zurückgeliefert.

7.1.2. Profilgenerator und Nutzerprofil-basiertes Empfehlen

Die in Abschnitt 4.2 bzw. Abschnitt 5.6 vorgestellten Konzepte des Profiling bzw. der Nutzerprofil-basierten Empfehlung benötigen eine umfangreiche Interaktion mit dem Nutzer, um ausreichend evaluiert werden zu können. Da das Framework für die Kontextabhängige Empfehlung vorsieht, dass nicht der Nutzer selbst, sondern eine passende Anwendung mit ihm interagiert, wurden diese Konzepte in einem eigenen Prototyp implementiert. Dafür eignete sich die vom *Institut für Rundfunktechnik* (IRT)[7] entwickelte MHP-Referenzimplementierung *Middleware for interactive TV* (m4iTV) [48], die im Rahmen einer Kooperation genutzt werden konnte. Sie kann als Mediacenter verwendet werden und ist auf einem PC ausführbar. Die Plattform unterstützt die DVB-Standards, IPTV und TV-Anytime. Durch die Verwendung von m4iTV existierten allerdings einige Einschränkungen. So konnte MASL nicht als Beschreibungssprache für Profil und Inhalte verwendet werden. Allerdings wurden die benötigten Teile der Sprache passend als Tabellen einer relationalen *Apache Derby*-Datenbank[8] nachgebildet. m4iTV ist ausschließlich auf Fernsehinhalte ausgelegt, weswegen nur diese berücksichtigt werden konnten. Außerdem wurde die Referenzimplementierung in der Programmiersprache Java umgesetzt, weswegen diese für den hier vorgestellten Prototyp genutzt werden musste.

In Abschnitt 4.2 wurden verschiedene Defaultwerte verwendet, mit denen Variablen bei der impliziten Abschätzung der Nutzerbewertung belegt werden können. Beim Start des Prototyps können diese über eine Konfigurationsdatei angepasst werden. Zu den Varia-

[4] www.alipr.com
[5] wordnet.princeton.edu
[6] projects.csail.mit.edu/jwi/
[7] www.irt.de/
[8] db.apache.org/derby/

blen gehören u. a. die maximale Dauer einer Werbeunterbrechung (`maximumCommercialBreak`) oder die Verfallsdauer einer Bewertung (`ratingExpiryTime`). Die von einem Nutzer konsumierten Fernsehinhalte werden über eine `WatchList` als `WatchListItem` verwaltet, in denen charakteristische Eigenschaften der Sendung gespeichert werden. Diese verwaltet auch die Variablen, die festhalten, wie oft und wie lange ein Zuschauer auf den Inhalt umgeschaltet hat. Bis zum Ende der Sendung hat der Nutzer Zeit, diese zu bewerten. Tut er dies nicht, wird der implizite Bewertungsprozess basierend auf den im `WatchListItem` gespeicherten Daten wie beschrieben angestoßen. Die Bewertung wird anschließend in der Datenbank hinterlegt.

Die Empfehlung wird vom Nutzer initiiert, indem dieser den entsprechenden Menüeintrag in der graphischen Oberfläche des m4iTV-Systems wählt. Die Klassen, die das Nutzerprofil-basierte Empfehlen umsetzen, lesen die im Profil gespeicherten Daten aus und verarbeiten sie wie in Abschnitt 5.6 beschrieben. Den Attributen sind je nach Inhaltsklasse variable Gewichte zugeteilt. Diese werden in einer `ParameterHelper`-Klasse als `Hashtable` hinterlegt. Durch die Berechnung der prognostizierten Empfehlungswerte wird ein `Vector` aus `RecommendationImpl`-Objekten erzeugt, die jeweils einen Inhalt und seinen Prognosewert (`finalPredictedValue`) repräsentieren. Dieser wird an die `filterRecommendations`-Methode übergeben und zusätzlich über eine `integer`-Zahl die gewünschte Filtermethode angezeigt. Die folgenden Methoden stehen u. a. zur Verfügung: Rückgabe aller Inhalte sortiert nach dem Prognosewert (Parameter 0), Rückgabe des aktuell ausgestrahlten Inhalts mit dem höchsten Prognosewert (1) oder Rückgabe des Inhalts mit höchsten Prognosewert, der zur Primetime (20:15 Uhr) läuft (3).

Das beschriebene System wurde Nutzertests unterzogen, die bei der Festlegung der Parameter für den Berechnungsprozess hilfreich waren. Probleme bereitete hier, dass Metadaten nicht in der ganzen Bandbreite des TV-Anytime-Standards zur Verfügung standen.

7.1.3. Metadaten-Importer

Die entwickelten Konzepte zur Kontext-abhängigen Personalisierung multimedialer Inhalte arbeiten mit der in Kapitel 3 entwickelten Metadatensprache MASL. Um eine breitere Menge an Metadaten unterstützen zu können, wurde für XML-basierte Sprachen ein Importer-Tool entwickelt. Die Konvertierung von MPEG-7/21 in MASL ist einfach umsetzbar, da MASL einige Werkzeuge übernimmt. Um zusätzlich TV-Anytime unterstüten zu können, wurden die Elemente dieses Standards auf die MPEG-7 Beschreibungswerkzeuge abgebildet und dann in MASL konvertiert. Dies erforderte eine umfangreiche Analyse des TV-Anytime-Standards, um die Elemente geeignet zuordnen zu können. So sind z. B. die Informationen, die in TV-Anytime in den Tabellen `ProgramInformationTable` und `ProgramLocationTable` enthalten sind, im *MediaInformation DS* von MPEG-7 enthalten. Zur Konvertierung wurde XSLT eingesetzt, das zur Transformation von XML verwendet wird.

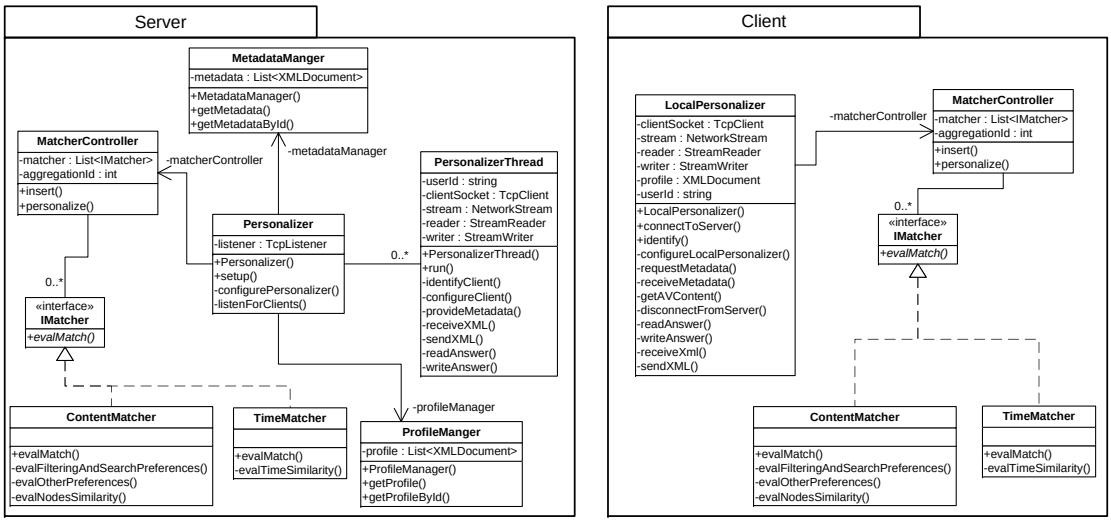

Abbildung 7.1.: Vereinfachtes Klassendiagramm des Frameworks für die Kontext-abhängige Empfehlung

7.1.4. Framework für die Kontext-abhängige Empfehlung

Im Folgenden soll die Implementierung des Frameworks zur Kontext-abhängigen Empfehlung, das in Abschnitt 5.8 entwickelt wurde, erläutert werden. Die Implementierung setzt sich aus den zwei Paketen `Server` und `Client` zusammen. Die implementierten Klassen und Schnittstellen sind in Abbildung 7.1 zu sehen. Aus Platzgründen wurde auf die vollständigen Methodensignaturen verzichtet. Der interne Ablauf des Frameworks wurde bereits beschrieben. Insbesondere werden im Folgenden die Unterschiede der entwickelten Architektur zur Implementierung erläutert. Für die Kommunikation zwischen Server und Client wurde ein Kommunikationsprotokoll entwickelt, dass anschließend näher erklärt wird.

Die Implementierung wurde Server-seitig mit Microsoft *.NET Framework*[9] bzw. Client-seitig mit Microsoft *.NET Compact Framework*[10] umgesetzt. Der Prototyp wurde für Windows-PCs bzw. für Pocket PCs mit Microsoft Windows Mobile 6.0 entwickelt.

Paket `Server`

Bei der Implementierung des Server-seitigen Teils des Frameworks wurden die Komponenten der Architektur, die in Abbildung 5.25 aus Kapitel 5 dargestellt sind, jeweils durch eine gleichnamige Klasse umgesetzt. Zusätzlich mussten noch einige Hilfsklassen

[9] msdn.microsoft.com/en-us/netframework
[10] msdn2.microsoft.com/en-us/netframework/aa497273.aspx

hinzugefügt werden, deren Funktionalität im Folgenden erläutert wird.

Das System wird über die Klasse `Server` gestartet, die die `Main`-Methode enthält. Die im Dateisystem hinterlegten Profildaten der Nutzer und verfügbare Metadaten der Inhalte im MASL-Format werden von den jeweiligen Manager-Komponenten eingelesen (`setup` in der Klasse `Personalizer`). Über die Konfigurationsdatei (vgl. Abschnitt 5.8.3) wird das Framework den gewünschten Einstellungen und Parametern angepasst (`configurePersonalizer`).

Der `Personalizer` öffnet einen TCP-Socket, auf dem er permanent nach Verbindungsaufbauwünschen von Clients horcht. Empfängt er einen solchen, startet er einen `PersonalizerThread`. Diese Klasse wurde dem Prototyp hinzugefügt, um mehrere Clients zeitgleich unterstützen zu können. Jeder Client erhält für die Abarbeitung seiner Anfrage einen eigenen, verwaltenden Thread.

Nach der Authentifizierung und Konfiguration des Clients (siehe unten), kann dieser über die `provideMetadata`-Methode Anfragen nach passenden Metadaten an den Server stellen. Welche passend sind, wird über die einzelnen `Matcher` bestimmt. Für deren zentrale Verwaltung wurde die Klasse `MatcherController` hinzugefügt. Sie erlaubt die Registrierung der in der Konfigurationsdatei eingetragenen `Matcher` und verwaltet sie in einer Liste. Soll eine Liste mit Metadaten für einen Nutzer erzeugt werden, ruft der `PersonalizerThread` die Profildaten des Nutzers und Metadaten bei den Managern ab und übergibt diese dem `MatcherController` über die Methode `personalize`. Dieser leitet die Anfrage über die Methode `evalMatch` an die registrierten `Matcher` weiter. Der `Matcher` berechnet die jeweiligen Ähnlichkeitswerte der Metadaten mit dem Profil gemäß seines Algorithmus (vgl. Abschnitt 5.5) und liefert sie als Liste zurück. Implementiert wurden Server und Client-seitig dieselben Matcher-Komponenten, nämlich ein `Matcher` für das Präferenz-basierte Filtern, ein `TimeMatcher` für Öffnungszeiten, ein `LocationMatcher` sowie einige der allgemeinen Ähnlichkeitsfunktionen. Innerhalb des Frameworks können beliebig viele `Matcher`-Komponenten umgesetzt werden. Sie müssen hierfür nur die Schnittstelle `IMatcher` implementieren.

Alle Rückgabewerte der `Matcher` werden vom `MatcherController` gesammelt und zu einem Gesamtergebnis gemäß der eingestellten Aggregationsfunktion zusammengefasst. Die Liste der Inhalte, deren Gesamtähnlichkeiten mit dem Profil über einem globalen Schwellwert liegen, wird anschließend an den zuständigen `PersonalizerThread` zurückgegeben und an den Client versendet, wo ein weiterer Abgleich stattfindet. Für die Anfrage des eigentlichen Inhalts, der durch das Framework ausgewählt wurde, ist die Methode `getAVContent` des `LocalPersonalizer` im Client zuständig. Da dies jedoch nicht mehr Teil des Konzepts ist, gibt sie lediglich einen kurzen Hinweis an den Nutzer aus.

Paket `Client`

Auch beim Client-seitigen Teil wurden die in Abschnitt 5.8 beschriebenen Komponenten durch eine entsprechende Klasse umgesetzt und eine zusätzliche `MatcherController`-

Klasse hinzugefügt. Deren Funktionalität ist identisch zum Server-seitigen `MatcherController`.

Der Client wird über die Klasse `Client` gestartet. Im Gegensatz zum Server wird hier nur ein Nutzerprofil eingelesen, das im Dateisystem des Clients hinterlegt ist. Hierfür wurde keine eigene Manager-Komponente implementiert, die `LocalPersonalizer` Klasse übernimmt diese Funktionalität. Anschließend versucht der Client, sich mit dem Server zu verbinden.

Kommunikationsprotokoll Client und Server

Die Client/Server-Kommunikation findet im Prototyp über TCP/IP-Sockets statt. Um die Kommunikation zu steuern, wurde ein Kommunikationsprotokoll entwickelt. Es basiert auf folgenden Konventionen: zwischen dem Client und dem Server können Nachrichten und XML-Dateien ausgetauscht werden. Jede Nachricht ist eine Zeichenkette mit fester Länge, kürzere Nachrichten werden mit Leerzeichen aufgefüllt. Eine Nachricht kann einen Befehl oder auch eine Quittung an die Gegenstelle enthalten. Abgeschlossen wird der Identifikator der Nachricht über das Zeichen |, an dieses kann sich eine Zahl als Parameter des Befehls anschließen. Eine Liste aller erlaubten Befehle und ihrer Parameter ist in Anhang D zu finden.

Wird ein Befehl an die Gegenstelle gesendet, liest die entsprechende Komponente die ersten 20 Zeichen aus, extrahiert den Identifikator des Befehls, der durch das Zeichen | abgeschlossen wird, und ggf. den Parameter. Die restlichen der 20 Zeichen werden verworfen. Eine XML-Datei wird versendet, indem sie in einen String konvertiert und dessen Länge bestimmt wird. Dieser String wird übertragen, wobei dessen Länge als Befehlsidentifikator genutzt wird (vgl. Anhang D).

Client und Server kommunizieren aus unterschiedlichen Gründen miteinander, wie dem Austausch von Dateien, der Authentifizierung oder dem Verbindungsabbau. Je nach Methode wird ein bestimmter Identifikator versendet, der wiederum den Aufruf einer bestimmten Methode bei der Gegenstelle bewirkt. Das Zusammenspiel dieser Methoden der Klassen `PersonalizerThread` und `LocalPersonalizer` ist in der folgenden Liste erklärt. Die Methoden `writeAnswer`, `readAnswer`, `sendXML` und `receiveXML` dieser Klassen sind nur Hilfsmethoden zum korrekten Lesen und Schreiben von Daten auf dem TCP/IP Socket.

- `connectToServer` (Client)
  Die Methode dient dem Client zum Verbindungsaufbau mit dem Server. Ist dies erfolgreich, wird ein neues Objekt der Klasse `PersonalizerThread` erzeugt.

- `identify` (Client) ⇒ `identifyClient` (Server)
  Der Client versucht sich beim Server zu identifizieren, indem er den im Profil gespeicherten Namen des Nutzers verschickt.

- `configureClient` (Server) ⇒ `configureLocalPersonalizer` (Client)
  Der Server teilt dem Client mit, welche Konfiguration er verwenden soll. Dazu gehören die zu verwendenden Matcher, die Aggregationsfunktion und die Schwellwerte.

- `requestMetadata` (Client) ⇒ `provideMetadata` (Server) ⇒ `receiveMetadata` (Client)
  Der Client fragt beim Server eine Liste von Metadaten an, die auf das Profil passen und wartet solange, bis er eine Rückantwort erhält.

- `disconnectFromServer` (Client)
  Der Client baut die Verbindung zum Server ab.

7.1.5. Middleware zur Anpassung der Darstellung multimedialer Inhalte

Im Folgenden soll die Implementierung der Middleware zur Anpassung der Darstellung erläutert werden, die in Abschnitt 6.3 vorgestellt wurde. Die Implementierung setzt sich aus den zwei Paketen `Server` und `Client` zusammen. Die implementierten Klassen sind in Abbildung 7.2 zu sehen. Aus Platzgründen wurde das Modell stark vereinfacht und auf vollständige Methodensignaturen verzichtet. Da der interne Ablauf des Frameworks und die Schnittstellen bereits erläutert wurden, werden im Folgenden in erster Linie die Unterschiede der entwickelten Architektur zur Implementierung und einige Besonderheiten gezeigt. Für die Implementierung wurde ebenfalls das Microsoft *.NET Framework* bzw. *.NET Compact Framework* verwendet (s. o.). Zunächst werden einige Einschränkungen erläutert, die sich auf die Implementierung ausgewirkt haben. Dann werden die beiden Pakete näher vorgestellt.

Einschränkungen bei der Prototyp-Entwicklung

Um auf den Nutzer angepasste Multimedia-Präsentationen zu erstellen, wurde bei der Prototyp-Entwicklung auf SMIL (vgl. Abschnitt 6.2.1) zurückgegriffen. Diese Sprache liefert die ideale Basis, um die entwickelten Konzepte schnell prototypisch umzusetzen. Die aktuelle Version des Standards ist 3.0, die Implementierung stützt sich jedoch auf die Versionen 2.0 bzw. 2.1, da für diese eine größere Anzahl an SMIL-Playern für mobile Endgeräte zur Verfügung stand. Dieser wird zur Anzeige der Inhalte benötigt und repräsentiert die eigentliche Anwendung. Auf Grund einer umfangreichen Analyse der vorhandenen Player wurden der Player PocketSMIL 2.0 [82] und der Ambulant Player 1.6.1 [39, 27] in die nähere Auswahl genommen. Leider hatten beide starke Einschränkungen, die sich auch auf die Implementierung auswirkten. Verwendet wurde letztendlich der Ambulant Player. Dieser verfügt zwar über keine geeignete Videounterstützung, läuft aber auf einem Pocket PC mit Windows Mobile 6.0 deutlich stabiler als PocketSMIL. Außerdem verwendet er zu SMIL konforme Beschreibungen.

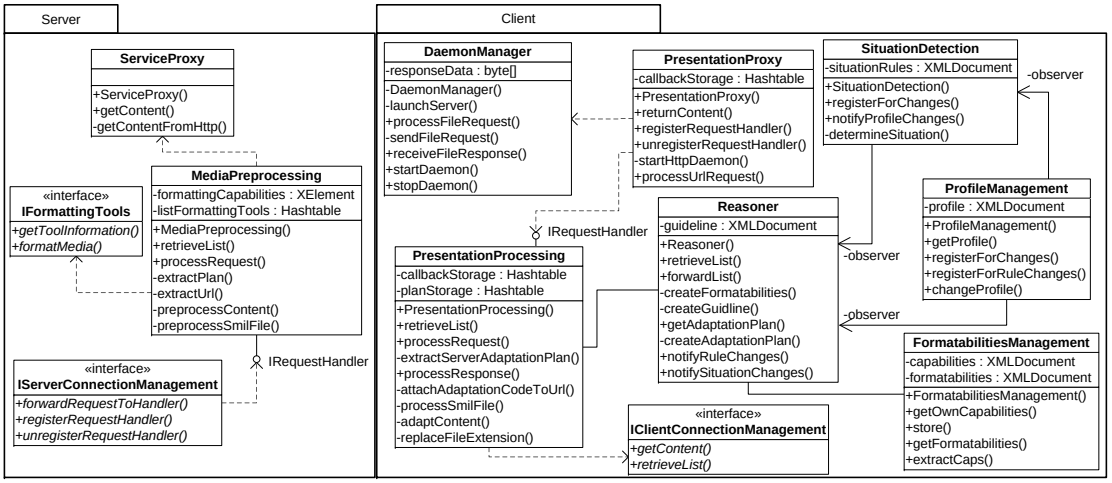

Abbildung 7.2.: Vereinfachtes Klassendiagramm der Middleware zur Anpassung der Darstellung.

Bei der Implementierung musste berücksichtigt werden, dass der Ambulant Player keine Videos unterstützt und Inhalte nicht über ein Netzwerk laden kann. Um dennoch dieses Verhalten zu simulieren, wurde ein `DownloadHelper` entwickelt. Dieser übernimmt über eine graphische Schnittstelle auf dem Client die URL des anzuzeigenden Inhalts, übergibt diese an die Middleware, die den Inhalt bezieht, anpasst und im Dateisystem hinterlegt. Der Player kann anschließend die Datei öffnen und anzeigen. Im Prototyp wurde der `Reasoner` so umgesetzt, dass er keine automatische Anpassung der Darstellung unterstützt, falls sich die Anpassungsparameter während der Darstellung eines Medieninhalts verändern. Eine Anpassung der Darstellung an die aktuellen Anpassungsparameter ist jedoch möglich. Diese Einschränkung ist rein implementierungstechnisch bedingt und hat keine Auswirkung auf die Evaluierung des Konzepts.

Paket `Client`

Die Schnittstelle zwischen der Middleware und den Client-seitigen Anwendungen wurde als http-Daemon realisiert, der von der `PresentationProxy`-Klasse gestartet wird. Für den Daemon wurde eine bereits bestehende Implementierung für das .NET Framework verwendet[11]. Der Daemon läuft im Hintergrund und wartet auf einkommende http `get`-Anfragen, die an einen konfigurierbaren Port des mobilen Endegeräts gestellt werden. Die Anfrage wird über die Klasse `DaemonManager`, die für die Kommunikation zwischen Daemon und `PresentationProxy` zuständig ist, zur weiteren Verarbeitung an die Middleware übergeben (`processUrlRequest`).

[11] www.codeproject.com/KB/cs/single_threaded_nhttpd.aspx

Beim Prototyp werden zwei Arten von Anfragen unterschieden: die Anfrage nach der SMIL-Datei und die Anfrage nach den in der SMIL-Datei eingebundenen Medieninhalten. Zunächst wird vom Daemon die SMIL-Datei angefragt. Die `PresentationProcessing` (oder Server-seitig die `MediaPreprocessing`)-Komponente kann gemäß den vorliegenden Anpassungsplänen einige Anpassungen an dieser Datei sofort vornehmen. Dazu gehört das Hinzufügen, Entfernen oder Ersetzen von Medieninhalten. Dies wird realisiert, indem die dazu benötigten SMIL-Tags angepasst werden (`processSmilFile`). Skalierung und Konvertierung können erst an den eigentlichen Medieninhalten vorgenommen werden (`adaptContent`). Die Middleware kodiert diese später durchzuführenden Operationen in der URL der Verknüpfung zu dem Medieninhalt in der SMIL-Datei (`attachAdaptationCodeToUrl`). So gibt z. B. die URL *http://www.example.com/mediafile,op=sc(160x120).jpg* an, dass die angefragte JPEG-Datei mit der URL *http://www.example.com/mediafile.jpg* auf das Format 160x120 skaliert werden soll. Ob diese Skalierung Client oder Server-seitig geschehen soll, wird durch den Anpassungsplan bestimmt.

Innerhalb des Clients ist es an verschiedenen Stellen nötig, dass sich einzelne Klassen für eine Benachrichtigung über ein Ereignis registrieren können. Dieses Konzept wurde mit dem *Observer*-Pattern [66] implementiert. Dafür registriert sich die zu benachrichtigende Klasse beim `Observable` (z. B. `registerForRuleChanges`). Tritt das Ereignis ein, wird der `Observer` benachrichtigt (z. B. `notifyRuleChanges`).

Der Client kann über die Schnittstelle `IClientConnectionManagement` dem Server-seitigen Teil der Middleware Nachrichten übermitteln. Dafür stellt diese die Methoden `getContent` zum Abrufen von Inhalten und `retrieveList` zum Abrufen der Liste der `FormattingCapabilities` zur Verfügung. Da der Aufruf asynchron abläuft, übergibt die Klasse `PresentationProcessing` die Methoden, die für die Rückgabe verwendet werden sollen. `processResponse` übernimmt die Inhalte, `retrieveList` die `FormattingCapabilities`. Auf dieselbe Weise wird die Antwort an den `PresentationProxy` (`returnContent`) und den `DaemonManager` (`receiveFileResponse`) weitergeleitet.

Paket `Server`

Das Server-seitige `ConnectionManagement` wurde in dieser Arbeit abstrakt spezifiziert. Um es im Prototyp geeignet umsetzen zu können, wurde es als einfacher Web Service implementiert. Es bietet eine Webschnittstelle an, auf die das Client-seitige *Connection-Management* zugreifen kann, um Daten zur Verarbeitung zu übermitteln. Dafür muss auf dem Server der *Microsoft Internet Information Service*[12] in der Version 5.1 oder höher vorhanden sein. Die empfangenen Daten werden mit `forwardRequestToHandler` an die `processRequest`-Methode der als `IRequestHandler` registrierten `MediaPreprocessing`-Instanz übergeben und weiter verarbeitet.

Wie schon erwähnt, werden beim Prototyp Anfragen nach der SMIL-Datei und Anfragen nach den in der SMIL-Datei eingebundenen Medieninhalten unterschieden. Beide

[12] www.iis.net

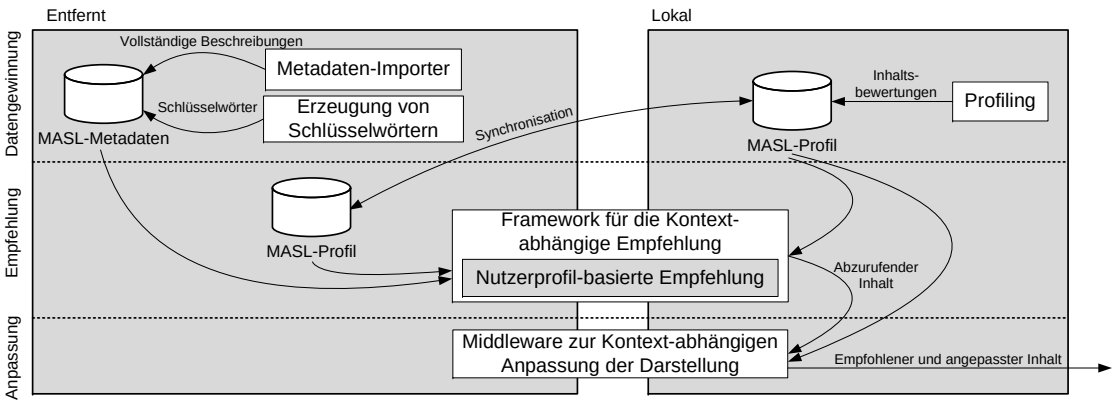

Abbildung 7.3.: Mögliche Kombination der Prototypen.

Anfragen werden zunächst an den `ServiceProxy` weitergereicht (`getContent`), der den Inhalt anfragt. Die erhaltene Datei wird dann je nach Art der Datei mit der `preprocessSmilFile` oder der `preprocessContent`-Methode angepasst. Die dem Server zur Verfügung stehenden Anpassungswerkzeuge müssen die Schnittstelle `IFormattingTools` implementieren. Diese beinhaltet die Methode `getToolInformation`, die Informationen über das Anpassungswerkzeug zurückliefert, und `formatMedia` für die eigentliche Anpassung eines Inhalts.

Die Funktionalität der Komponente *Formatting Capabilities Management* aus Abbildung 6.5 wurde bei der Implementierung in die Klasse `MediaPreprocessing` integriert.

7.1.6. Zusammenspiel der Prototypen

Die verschiedenen Prototypen können geeignet kombiniert werden, um das in dieser Arbeit vorgestellte Gesamtkonzept umzusetzen. Abbildung 7.3 zeigt diesen Zusammenhang. Dabei unterscheidet man die drei Phasen der *Datengewinnung*, der *Empfehlung* und der *Anpassung* eines empfohlenen Inhalts.

Die *MASL-Metadaten* können z. B. auf einem zentralen Server gespeichert sein, der diese dem Empfehlungssystem und dem Anpassungssystem zur Verfügung stellt. Mit Hilfe des *Metadaten-Importer* können vollständige, in MPEG-7/21 oder TV-Anytime gegebene Beschreibungen in die Sammlung aufgenommen werden. Zum automatischen *Erzeugen von Schlüsselwörtern* steht der entsprechende Prototyp bereit. Die Komponente für das explizite und implizite *Profiling* befindet sich lokal auf dem Endgerät des Nutzers und erzeugt Inhaltsbewertungen, die in das lokale Profil übernommen werden. Für den Empfehlungsprozess wird eine verteilte Speicherung des Profils angenommen, weswegen die lokal und entfernt gespeicherten Profile geeignet synchronisiert werden müssen.

Über eine geeignete Anwendung wie einem mobilen Touristenführer wird der Empfeh-

lungsprozess im *Framework für die Kontext-abhängige Empfehlung* angestoßen. Dieser bestimmt basierend auf den Metadaten und dem Profil einen oder mehrere Inhalte, die für den Nutzer im gegebenen Kontext passend sind. Als einer der `Matcher` kann hier die Nutzerprofil-basierte Empfehlung eingesetzt werden. Die Id des oder der abzurufenden Inhalte wird an die *Middleware zur Kontext-abhängigen Anpassung der Darstellung* übergeben. Diese bezieht die Inhalte und passt sie geeignet an. Die Middleware hat im Gegensatz zum Framework Server-seitig keinen Zugriff auf die Nutzerprofile, was einen besseren Schutz der Privatsphäre gewährleistet.

7.2. Evaluierung gemäß Anforderungskatalog

Im Abschnitt 2.2.4 wurde ein Anforderungskatalog entwickelt, der die Anforderungen an die Kontext-abhängige Personalisierung widerspiegelt. Im Folgenden wird gezeigt, wie die in dieser Arbeit vorgestellten Konzepte diese Anforderungen erfüllen.

1. *Unterstützung unterschiedlicher Medientypen*
   Im Rahmen dieser Arbeit wurde MASL als Metadatensprache für die Kontext-abhängige Personalisierung entwickelt (siehe Kapitel 3). Diese unterstützt Beschreibungen von Inhalten unabhängig vom Medientyp, in dem diese vorliegen. Auf den Medieninhalt an sich muss lediglich in der Analyse-Phase für die automatische Gewinnung von Schlüsselwörtern (siehe Abschnitt 4.1) und bei der eigentlichen Anpassung der Inhalte (siehe Abschnitt 6) zugegriffen werden. Alle anderen Konzepte arbeiten ausschließlich mit den Metadaten. Der Ansatz zum automatischen Taggen schlägt vor, für die Inhaltsanalyse spezialisierte externe Dienste zu integrieren. Sind solche vorhanden, können diese einfach zum Taggingprozess hinzugefügt werden. Die Middleware dient in erster Linie dazu, festzulegen, welche Anpassung beim Inhalt durchgeführt werden soll. Für die tatsächliche Anpassung verwendet sie ebenfalls externe Werkzeuge. Von daher kann dieser Ansatz unterschiedliche Medientypen unterstützen.

2. *Formale Beschreibung der Inhalte*
   Mit MASL können umfangreiche Beschreibungen des Inhalts festgelegt werden. Welche Beschreibungen für die Kontext-abhängige Personalisierung benötigt werden, wurde genauer in Kapitel 3 untersucht. Dass MASL diese in vollem Umfang unterstützt, wurde in Abschnitt 3.5 ausführlich diskutiert.

3. *Formale Beschreibung des Nutzers*
   Neben den Beschreibungen der Inhalte umfasst der Sprachschatz von MASL Beschreibungen der Nutzer des Systems. Wie in Abschnitt 3.5 angeführt, unterstützt es alle benötigten Informationen wie Nutzerdaten, Nutzungsdaten, sowie dynamische Kontextinformationen während der Dienstnutzung.

4. *Formale Beschreibung von situativen Anpassungsregeln*
   Neben der Beschreibung der Inhalte und der Nutzer kann MASL situative Anpassungsregeln spezifizieren. Diese erlauben es dem Nutzer festzulegen, wann ein Inhalt wie angepasst werden soll.

5. *Leichte Erweiterbarkeit der Anpassungsregeln, Nutzer- und Inhaltsbeschreibungen*
   Anpassungsregeln, Nutzer- oder Inhaltsbeschreibung liegen in MASL als XML-Datei vor. Da XML als generischer Standard entwickelt wurde und leicht erweiterbar ist, kann auch MASL ohne Probleme an erweiterte Anforderungen und Anwendungsszenarien angepasst werden.

6. *Unterstützung unterschiedlicher Personalisierungsverfahren*
   Das Framework zur Kontext-abhängigen Auswahl von Inhalten erlaubt die Integration einer großen Anzahl von so genannten Matcher-Komponenten, die je ein unterschiedliches Empfehlungsverfahren umsetzen können. Welche tatsächlich genutzt werden, wird durch die Konfiguration des Frameworks für eine bestimmte Anwendung bestimmt. Wichtig ist lediglich, dass die einzelnen Matcher die `evalMatch`-Schnittstelle implementieren. Da dieser Schnittstelle die kompletten Daten des Nutzerprofils und der Inhaltsbeschreibungen übergeben werden und nicht nur ein relevanter Teilausschnitt, können die Matcher alle benötigten Daten selbstständig extrahieren. Somit können unterschiedliche Empfehlungsverfahren einzeln oder in Kombination verwendet werden.
   Ziel der Entwicklung der Middleware war in erster Linie einen Anpassungsplan zu erstellen, wann ein Inhalt wie angepasst werden soll. Für die eigentliche Anpassung wurden keine Werkzeuge konzipiert, sondern externe integriert. Bei der Planerzeugung werden als Anpassungsverfahren Skalierung, Konvertierung sowie das Hinzufügen, Entfernen und Ersetzen berücksichtigt. Damit unterstützt die Middleware den gleichzeitigen Einsatz von unterschiedlichen Anpassungsverfahren.

7. *Unterstützung von Nutzer-generierten Inhalten*
   In Abschnitt 4.1 wurde ein Ansatz diskutiert, wie Inhalte ohne Beschreibung mit Schlüsselwörtern versehen werden können. Dies kann entweder durch den Nutzer geschehen oder automatisch berechnet werden. Dafür werden externe Tools zur Inhaltsanalyse angewendet und geeignet weiterverarbeitet. Dieser Punkt ist besonders wichtig, um Nutzer-generierte Inhalte im Personalisierungsprozess unterstützen zu können.
   Eine besondere Form der Nutzer-generierten Inhalte sind Kommentare. Diese stehen in engem Zusammenhang mit dem Inhalt, auf den sie sich beziehen. Kommentare können wiederum selbst mit Kommentaren versehen sein. Daraus ergibt sich eine komplexe Kommentarstruktur. In Abschnitt 5.7 wurde ein Ansatz entwickelt, wie man Kommentare strukturerhaltend empfehlen kann.

8. *Implizites Profiling*
   In Abschnitt 4.2 wurde ein Mechanismus zum expliziten und impliziten Profiling vorgestellt, der auf Nutzerbewertungen basiert. Der Nutzer kann entweder zu einem konsumierten Inhalt eine Bewertung abgeben oder das System versucht diese Bewertung an Hand von Beobachtungen des Nutzerverhaltens abzuleiten. Die auf diese Weise erhaltenen Bewertungen können geeignet in einem MASL-Profil gespeichert und weiter verarbeitet werden.

9. *Verfahren zur Ähnlichkeitsbestimmung von Kontextinformationen*
   Für die Kontext-abhängige Personalisierung müssen Verfahren bereitgestellt werden, die die Ähnlichkeit zwischen Kontextinformationen bestimmen können. In Abschnitt 5.5.3 und 5.5.4 wurden Ansätze vorgestellt, die dies ermöglichen. Diese können entweder als Matcher-Komponente in das Framework integriert werden oder in der Middleware von der *Situation Detection* ausgewertet werden.

10. *Anwendungsunabhängigkeit*
    Das entwickelte Gesamtkonzept der Kontext-abhängigen Personalisierung soll möglichst generisch sein, um unterschiedliche Anwendungen realisieren zu können. Dafür wurde es stark modularisiert. Bei der Entwicklung der einzelnen Module, also Datengewinnung, die Entwicklung der Metadatensprache MASL und die Ansätze zur Auswahl und Anpassung der Inhalte, wurden diese ebenfalls anwendungsunabhängig gehalten. Die Bewertungen der Anwendungsunabhängigkeit der Teilkonzepte befinden sich im Anschluss des jeweiligen Konzeptkapitels. Zusätzlich sind Konfigurationsschnittstellen vorhanden, die eine Anpassung an das gewünschte Anwendungsszenario erlauben. In der Summe kann so das Gesamtkonzept anwendungsunabhängig eingesetzt werden.

11. *Schutz der Privatsphäre des Nutzers*
    Bei den entwickelten Verfahren wurde soweit wie möglich auf einen guten Schutz der Privatsphäre geachtet. Die Anpassung der Darstellung sieht keinerlei Speicherung von Nutzerprofilen bei einer zentralen Instanz vor. Lediglich die Empfehlung benötigt zentral abgelegte Profile. Dadurch wird kollaboratives Filtern ermöglicht. Andererseits kann das Framework theoretisch mit rein Client-seitig gespeicherten Informationen arbeiten. Dies widerspricht jedoch der Grundidee und zieht eine erhöhte Belastung der Luftschnittstelle nach sich.

12. *Geringe Belastung der Luftschnittstelle*
    Die im Rahmen dieser Arbeit entwickelten Konzepte zur Personalisierung basieren auf einer verteilten Verarbeitung der Daten. Bei multimedialen Diensten wird die Belastung der Luftschnittstelle durch die Größe der zu übertragenden Datei bestimmt. Durch die Verteilung wird vermieden, dass umfangreiche Metadatenbeschreibungen bzw. unnötig große Dateien übertragen werden. Ebenso wird die

Kommunikation der verteilten Komponenten so gering wie möglich gehalten.

13. *Skalierbarkeit*
    Die Verteilung wirkt sich ebenfalls positiv auf die Skalierbarkeit des Gesamtkonzepts bezüglich der Anzahl der Nutzer aus. Dadurch, dass ein Teil der Verarbeitung auf den Client ausgelagert wird, wird die zentrale Komponente entlastet, und es können deutlich höhere Nutzerzahlen bedient werden. Eine geeignete Strategie zur Replikation der Server kann dies noch zusätzlich unterstützen.

14. *Effiziente Durchführung der Personalisierung*
    Bei der Entwicklung der Konzepte wurde auf eine effiziente und zeitnahe Durchführung der Personalisierung geachtet. Die Matcher-Komponenten des Frameworks können parallel die lokalen Ähnlichkeiten bestimmen. Außerdem können Teilergebnisse wie die Bestimmung ähnlicher Nutzer vorberechnet, geeignete Cachingstrategien eingesetzt sowie Teilergebnisse wiederverwendet werden.
    Bei der Anpassung der Darstellung wird der Anpassungsleitfaden permanent aktuell gehalten, wodurch bei der Anfrage eines Inhalts nur noch der relevante Teil extrahiert werden muss. Dies beschleunigt die Anpassung maßgeblich.

7.3. Zusammenfassung

In diesem Kapitel wurden die einzelnen Prototypen vorgestellt, die für die Umsetzung der Arbeiten, die in den letzten Kapiteln erläutert wurden, entwickelt wurden. Die Implementierungen zeigen die Tragfähigkeit der vorgestellten Ansätze. Es konnten alle wichtigen Einzelkonzepte realisiert und ihre Funktion überprüft werden. Daran anschließend wurde diskutiert, wie sich diese zu einer Gesamtlösung intergieren lassen. Neben der Implementierung wurde eine umfassende Evaluierung des Gesamtkonzepts vorgenommen. Diese erfolgte durch eine Bewertung gemäß dem Anforderungskatalog aus Abschnitt 2.2.4. Es konnte gezeigt werden, dass dieser umfassend erfüllt wird.

Kapitel 8.

Zusammenfassung und Ausblick

Dieses Kapitel fasst die wichtigsten Ergebnisse zusammen, die im Rahmen der vorliegenden Arbeit vorgestellt wurden. Daran anschließend werden mögliche Fragestellungen für weitere Forschungsarbeiten diskutiert.

8.1. Zusammenfassung

In der vorliegenden Arbeit wurden die folgenden grundsätzlichen Arten der Kontext-abhängigen Personalisierung multimedialer Inhalte für mobile Endgeräte näher betrachtet:

- die personalisierte Auswahl der Inhalte und
- die personalisierte Anpassung der Darstellung der Inhalte

Zu diesem Zweck wurden zunächst Szenarien untersucht, in denen diese Formen der Kontext-abhängigen Personalisierung eine Rolle spielen. Darauf basierend wurde ein Anforderungskatalog erstellt und die grundlegenden Problemstellungen, die in dieser Arbeit zu betrachten waren, wurden abgeleitet.

Die vorgestellten Ansätze basieren auf vorhandenen Inhaltsbeschreibungen und Nutzerprofilen. Um diese zu erfassen, wurde die *Multimedia Adaptation and Selection Language* (MASL) entwickelt. Diese erlaubt umfangreiche Beschreibungen von unterschiedlichen Kontextinformationen, Nutzerbewertungen, Kommentaren, erweiterte Nutzerpräferenzen bezüglich der personalisierten Auswahl und der Anpassung der Darstellung, sowie informelle Zusatzinformationen zu einem Inhalt. Damit können alle wichtigen Informationen für eine Kontext-abhängige Personalisierung festgehalten werden.

Eine Annahme dieser Arbeit ist es, dass die Beschreibungen der Inhalte und Nutzer in MASL vorliegen. Exemplarisch wurden zwei Ansätze vorgestellt, wie man einen Teil dieser Daten gewinnen kann. Dazu gehört die automatische und nutzergestützte Gewinnung von Schlüsselwörtern. Diese basiert auf Ansätzen aus dem Bereich der Informationsanalyse und des Wissenserwerbs. Sie nutzt vorhandene Technologien wie Thesauri und

Ontologien, um die Semantik der Inhalte zu analysieren. Als zweites wurde erläutert, wie man basierend auf Beobachtungen des Nutzerverhaltens dessen Präferenzen oder Abneigungen für bestimmte Inhalte ableiten kann. Dazu errechnet das System automatisch erzeugte Nutzerbewertungen. Die beiden vorgestellten Ansätze zeigen umfangreiche Möglichkeiten, explizit und implizit Daten über Inhalte und Nutzer zu gewinnen und die MASL-Beschreibungen zu vervollständigen.

Die Kontext-abhängige Auswahl der Inhalte wurde im Anschluss diskutiert. Das vorgestellte Konzept beruht auf dem Lokal-Global-Prinzip. Dieses führt die Gesamtähnlichkeit zweier Objekte auf lokale Ähnlichkeiten zurück und aggregiert die Teilergebnisse zu einem Gesamtergebnis. Es wurden allgemeine und spezielle Verfahren zur Ähnlichkeitsmessung von Kontextinformationen entwickelt und passende Aggregationsfunktionen aufgezeigt. Auf diese Weise können Kontextinformationen im Empfehlungsprozess berücksichtigt werden. Ein häufiges Problem bei klassischen Empfehlungssystemen ist der *Curse of Dimensionality*, also der exponentielle Anstieg des Volumens von mathematischen Räumen beim Hinzufügen von weiteren Attributen. Um dieses zu lösen, wurde der Ansatz der Nutzerprofil-basierten Empfehlung entwickelt. Dieser führt einen Abgleich von unbekannten Inhalten mit dem kompletten Profil durch und nicht nur mit anderen Inhalten, die dem unbekannten Inhalt in allen Attributen ähnlich sind, wie bei klassischen Verfahren üblich. Außerdem wurde ein Algorithmus zum strukturerhaltenden Empfehlen vorgestellt, der es erlaubt, Inhalte mit ihren zugehörigen Nutzerkommentaren zu filtern. Alle entwickelten Ansätze wurden in ein umfassendes Framework zur Kontext-abhängigen Empfehlung integriert. Dieses unterstützt u. a. einen verteilten Abgleich von Kontextinformationen, die Kombination unterschiedlicher Empfehlungsverfahren und die geeignete Konfiguration für unterschiedliche Anwendungsfälle.

Für die Kontext-abhängige Anpassung der Darstellung wurde eine Middleware vorgestellt, die in einem verteilten Prozess die Inhalte geeignet verändern kann. Ziel der Entwicklung der Middleware war u. a. die effiziente Erstellung von passenden Anpassungsplänen durch den Client und die Situationsüberwachung bei größtmöglichem Schutz der Privatsphäre. Der Server beginnt die Anpassung, übergibt den vorverarbeiteten Inhalt an den Client und dieser beendet den Anpassungsprozess. Im Gegensatz zu verwandten Arbeiten, die für jede mögliche Darstellung eine eigene Datei vorhalten, ist dieser Ansatz besser skalierbar in Bezug auf die Anzahl unterschiedlicher Endgeräte und Inhalte. Bei der Verteilung kann jedoch das Problem auftreten, dass der Client einen Anpassungsplan erzeugt, der vom Server nicht durch geeignete Methoden unterstützt wird. Deswegen werden bei der initialen Anmeldung des Clients die vom Server unterstützten Methoden übertragen und bei der Planerstellung berücksichtigt.

Durch die Implementierung der dargestellten Konzepte und eine Evaluierung gemäß Anforderungskatalog wurde die Tragfähigkeit des Gesamtkonzepts bewiesen.

8.2. Ausblick

Die in dieser Arbeit entwickelten Konzepte bieten unterschiedliche Anknüpfungspunkte für Erweiterungen oder Ergänzungen. Diese sollen im Folgenden diskutiert werden.

Eine grundlegende Annahme dieser Arbeit ist, dass alle benötigten Informationen in den Beschreibungen von Inhalten und Nutzern vorhanden sind. Wie diese erfasst werden, wurde in Kapitel 4 exemplarisch anhand der Schlüsselwörter und der Nutzerbewertungen erläutert. Die Gewinnung von Kontextinformationen z. B. aus Sensorrohdaten und deren geeignete Kombination zu high-level Kontextinformationen wurden bewusst außen vor gelassen und bieten Raum für weiterführende Arbeiten. Dazu gehören auch Konzepte, die geeignet auf sich widersprechende oder fehlende Sensorinformationen reagieren.

Die beiden vorgestellten Ansätze zur Datengewinnung bieten Platz für Erweiterungen. Für die Inhaltsanalyse zur Gewinnung von Schlüsselwörtern wurden existierende Algorithmen verwendet, die dem hier vorgestellten Verfahren Daten für die weitere Verarbeitung liefern. Diese Algorithmen sind jedoch noch nicht ausgereift: sie eignen sich meist nur für spezielle Anwendungsfällen und sind nur für einen einzelnen Medientyp geeignet. Die Entwicklung generischer Algorithmen zur Analyse ist eine spannende Problemstellung in diesem Umfeld. Zusätzlich können für die Erzeugung von Schlüsselwörtern noch umfangreiche Ontologien und Thesauri entwickelt werden.

Das implizite Profiling kann für visuelle Medientypen um Verfahren aus dem Bereich des *Eye-Tracking* erweitert werden. Diese erfassen die Blickrichtung des Nutzers und können bestimmen, ob der Nutzer wirklich den Inhalt betrachtet. Ansätze für weitere Arbeiten liefert auch das *Emotion-Tracking*. Dieser Forschungsbereich versucht aus der Beobachtung der Mimik des Nutzers Emotionen wie Lachen, Weinen oder Angst zu erkennen. Aus beiden Verfahren können implizite Nutzerbewertung für die Profilbeschreibungen abgeleitet werden.

Im Zusammenhang mit den Kontext-abhängigen Empfehlungssystemen wurden allgemeine und spezielle Methoden zur Ähnlichkeitsmessung von Kontextinformationen aufgezeigt. Diese erheben keinesfalls Anspruch auf Vollständigkeit. In weiteren Arbeiten können die Kontextinformationen noch eingehender untersucht, weitere Ähnlichkeitsfunktionen aufgestellt und in das entwickelte Framework integriert werden. Dabei spielen auch Verfahren zur Erkennung von Homonymen und Synonymen eine wichtige Rolle. Mit deren Hilfe kann man semantisch gleiche Kontextinformationen bestimmen, die unterschiedlich in den Beschreibungen vorliegen. Dies beinhaltet auch den Fall, dass für die Beschreibungen unterschiedliche Sprachen verwendet werden.

Bei der Kontext-abhängigen Anpassung der Darstellung bedarf die Schnittstelle zu den Anwendungen noch näherer Untersuchung. Außerdem können die durch die Middleware erstellten Anpassungspläne widersprüchliche Regeln beinhalten. Dadurch können Konflikte bei der Anpassung entstehen, die geeignet gelöst werden müssen. Ein grundlegender Mechanismus mittels Prioritäten wurde in dieser Arbeit bereits erläutert. Näherer Untersuchung bedarf jedoch, ob sich durch eine feinere Einstufung der Prioritäten oder durch

ein gänzlich anderes Konzept eine Verbesserung der Konfliktbehandlung erzielen lässt.

Bei der Entwicklung der Konzepte wurde auf einen grundlegenden Schutz der Privatsphäre geachtet. Dennoch liefert dieser Bereich umfangreiche Möglichkeiten für weitere Betrachtungen. Dazu gehören zum einen die Untersuchung von Fragestellungen aus dem Bereich des Vertrauens, also wem vertraut man seinen Kontext in welchem Umfang an. Dies wiederum liefert neue Anforderungen für die umfassende Entwicklung von Methoden zum Schutz der Privatsphäre.

Im Rahmen dieser Arbeit wurde nicht untersucht, wie man die einzelnen Konzepte vor Manipulation schützen kann. Durch geeignete Veränderungen der Inhaltsbeschreibungen oder Nutzerprofile können insbesondere die Methoden der Kontext-abhängigen Empfehlung beeinflusst werden. So können z. B. zu einem Inhalt unpassende Beschreibungen hinzugefügt werden. Arbeiten Filteransätze basierend auf der Anzahl der Nutzerzugriffe auf einen Inhalt, kann sich die Beeinflussung dieses Werts in der Beschreibung auf die Empfehlung auswirken. Das Profil kann z. B. durch das Hinzufügen falscher Präferenzen manipuliert werden.

Für die vorgestellten Ansätze zur Kontext-abhängigen Personalisierung müssen noch geeignete Abrechnungsverfahren entwickelt werden. Das in dieser Arbeit vorgestellte Rollenmodell umfasst im Gegensatz zu klassischen multimedialen oder Kontext-abhängigen Diensten zusätzliche Rollen. Wie sich eine Abrechnung mit diesen neuen Rollen gestalten lässt und welche Interaktionen dafür benötigt werden, sind weitere offene Forschungspunkte.

Anhang A.

Schemadefinitionen

Im folgenden Abschnitt befinden sich die wichtigsten XML-Schemadefinitionen, die im Rahmen dieser Arbeit entwickelt wurden. Dazu gehört die Schemadefiniton der *Multimedia Adaptation and Selection Language (MASL)* (Anhang A.1), die für MASL definierten Classification Schemes (CS) (Anhang A.2), sowie die Schemadefinitonen der Konfigurationsschnittstelle (Anhang A.3), der `FormattingCapabilities` und `Formatabilities` (Anhang A.4).

A.1. Schemadefinition von MASL

Im Folgenden werden die Schemadefinitionen der *Multimedia Adaptation and Selection Language (MASL)*, die im Rahmen dieser Arbeit entwickelt wurde, gezeigt. Die beiden Wurzelelemente sehen wie folgt aus:

```
<element name="Metadata">
 <complexType>
  <sequence>
   <!-- Eindeutiger Bezeichner des Inhalts -->
   <element name="ProgramIdentifier" type="mpeg7:UniqueIDType"/>
   <!-- Metadaten der Beschreibung -->
   <element name="DescriptionMetadata" type="mpeg7:
      DescriptionMetadataType" minOccurs="0"/>
   <!-- Beschreibung des Medientyps der Inhalte (Video, Audio etc
      .);erlaubt auch die Beschreibung von einzelnen Teilen der
      Inhalte-->
   <element name="DescribedEntity" type="mpeg7:
      MultimediaContentType"/>
   <!-- Semantische Beschreibungen -->
   <element name="SemanticDescription" type="mpeg7:SemanticType"/>
   <!-- Beschreibung der Dateiinformationen-->
   <element name="MediaDescription" type="mpeg7:
      MediaInformationType"/>
   <!-- Beschreibung der Erzeugungsinformationen-->
```

```
   <element name="CreationDescription" type="masl:
      CreationInformationExtType"/>
   <!-- Verwendungshinweise -->
   <element name="UsageDescription" type="masl:
      UsageInformationExtType"/>
  </sequence>
 </complexType>
</element>
```

Listing A.1: MASL: Inhaltsbeschreibung

```
<element name="Profile">
 <complexType>
  <sequence>
   <!-- Beschreibung demographischer Daten -->
   <element name="DemographicInformation" type="masl:PersonExtType
      "/>
   <!-- Beschreibung der Nutzungsumgebung -->
   <element name="UsageEnvironmentDescription">
    <complexType>
     <sequence>
      <element name="TerminalCapabilities" type="dia:
         TerminalCapabilityBaseType" minOccurs="0" maxOccurs="
         unbounded"/>
      <element name="NetworkCharacteristics" type="dia:
         NetworkCharacteristicBaseType" minOccurs="0" maxOccurs="
         unbounded"/>
      <element name="NaturalEnvironmentCharacteristics" type="dia:
         NaturalEnvironmentCharacteristicBaseType" minOccurs="0"
         maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
   <!--Beschreibung der Nutzungshistorie-->
   <element name="UsageHistory" type="masl:UsageHistoryType"/>
   <!--Enthält die inmhaltlichen Präferenzen, benötigt für die
      Auswahl der Inhalte-->
   <element name="UserPreferences">
    <complexType>
     <sequence minOccurs="0" maxOccurs="unbounded">
      <element name="FilteringAndSearchPreferences" type="masl:
         FilteringAndSearchPreferencesExtType" minOccurs="0"
         maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
```

```
   <!-- Zeitabhängige Bewertungen von Attributen und
      Attributkombinationen-->
   <element name="TimeDependentUserPreferences">
    <complexType>
     <sequence>
      <element name="TimeDependentAttributes">
       <complexType>
        <sequence minOccurs="0" maxOccurs="unbounded">
         <element name="TimeDependentAttribute" type="masl:
            TimeDependentAttributeType" maxOccurs="unbounded"/>
        </sequence>
       </complexType>
      </element>
      <element name="TimeDependentAttributeCombinations">
       <complexType>
        <sequence minOccurs="0" maxOccurs="unbounded">
         <element name="TimeDependentAttributeCombination" type="
            masl:TimeDependentCombinationType" maxOccurs="
            unbounded"/>
        </sequence>
       </complexType>
      </element>
     </sequence>
    </complexType>
   </element>
   <!-- Beschreibung von Präferenzen bzgl. der Anpassung-->
   <element name="PresentationPreferences" type="dia:
      PresentationPreferencesType"/>
   <!-- Beschreibung der Situations-abhängigen Anpassungsregeln-->
   <element name="AdaptationRules">
    <complexType>
     <sequence>
      <element name="Situation" type="masl:SituationType"
         minOccurs="0" maxOccurs="unbounded"/>
      <element name="Behaviour" type="masl:BehaviorType" minOccurs
         ="0" maxOccurs="unbounded"/>
      <element name="Rule" type="masl:RuleType" minOccurs="0"
         maxOccurs="unbounded"/>
     </sequence>
    </complexType>
   </element>
  </sequence>
 </complexType>
</element>
```

Listing A.2: MASL: Profilbeschreibung

In der folgenden Auflistung werden die wichtigsten für MASL definierten Typen in alphabetischer Reihenfolge gezeigt. Das in manchen Typbezeichnungen enthaltene *Ext* steht für *Extension* und weißt auf die Erweiterung eines aus MEPG-7 oder MPEG-21 übernommenen Typs hin. Aus Platzgründen wurden `ClassificationPreferencesExtType` und `SourcePreferencesExtType` nicht aufgeführt, diese passen lediglich den Präferenzwert auf das in MASL gegebene Intervall an. Dies passiert wie bei den Typen `FilteringAndSearchPreferencesExtType` und `CreationPreferencesExtType` gezeigt.

```
<complexType name="AudioOutputCapabilitiesExtType">
 <complexContent>
  <extension base="dia:AudioOutputCapabilitiesType">
   <attribute name="outputDevice" type="string" use="optional"/>
  </extension>
 </complexContent>
</complexType>
```

Listing A.3: MASL: `AudioOutputCapabilitiesExtType`

```
<complexType name="BehaviorType">
 <sequence>
  <element name="Function" maxOccurs="unbounded">
   <complexType>
    <sequence>
     <element name="Object" maxOccurs="unbounded">
      <complexType>
       <attribute name="name" type="string" use="required"/>
      </complexType>
     </element>
     <element name="Parameter" minOccurs="0" maxOccurs="unbounded
        ">
      <complexType>
       <attribute name="name" type="string" use="required"/>
      </complexType>
     </element>
    </sequence>
    <attribute name="name" type="string" use="required"/>
   </complexType>
  </element>
 </sequence>
 <attribute name="id" type="ID" use="required"/>
</complexType>
```

Listing A.4: MASL: `BehaviorType`

```
<complexType name="ContentAnnotationType">
 <sequence>
```

```
  <element name=" MediaUri" type="anyURI"/>
  <element name=" DescriptionUri" type="anyURI"/>
 </sequence>
 <attribute name="type">
  <simpleType>
   <restriction base="NMTOKEN">
    <enumeration value="audio"/>
    <enumeration value="video"/>
    <enumeration value="audiovisual"/>
    <enumeration value="image"/>
    <enumeration value="text"/>
    <enumeration value="multimedia"/>
   </restriction>
  </simpleType>
 </attribute>
</complexType>
```

Listing A.5: MASL: `ContentAnnotationType`

```
<complexType name="ContextInformationRestrictionType">
 <complexContent>
  <extension base="masl:ContextInformationType">
   <attribute name="constraint" use="required">
    <simpleType>
     <restriction base="string">
      <enumeration value="max"/>
      <enumeration value="maxEqual"/>
      <enumeration value="min"/>
      <enumeration value="minEqual"/>
      <enumeration value="equal"/>
     </restriction>
    </simpleType>
   </attribute>
  </extension>
 </complexContent>
</complexType>
```

Listing A.6: MASL: `ContextInformationRestrictionType`

```
<complexType name="ContextInformationType">
 <attribute name="type" type="string" use="required"/>
 <attribute name="scale" type="string" use="required"/>
 <attribute name="value" type="string" use="required"/>
</complexType>
```

Listing A.7: MASL: `ContextInformationType`

```
<complexType name="ContextType">
 <complexContent>
  <extension base="dia:NaturalEnvironmentCharacteristicBaseType">
   <sequence>
    <element name="Context" type="masl:ContextInformationType"
       maxOccurs="unbounded"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.8: MASL: `ContextType`

```
<complexType name="CreationExtType">
 <complexContent>
  <extension base="mpeg7:DSType">
   <sequence>
    <element name="Title" type="mpeg7:TitleType" maxOccurs="
       unbounded"/>
    <element name="TitleMedia" type="mpeg7:TitleMediaType"
       minOccurs="0"/>
    <element name="Abstract" type="masl:TextAnnotationExtType"
       minOccurs="0" maxOccurs="unbounded"/>
    <element name="Creator" type="mpeg7:CreatorType" minOccurs="0"
        maxOccurs="unbounded"/>
    <element name="CreationCoordinates" minOccurs="0" maxOccurs="
       unbounded">
     <complexType>
      <sequence>
       <element name="Location" type="mpeg7:PlaceType" minOccurs
          ="0"/>
       <element name="Date" type="mpeg7:TimeType" minOccurs="0"/>
       <element name="ContextInformation" type="masl:
          ContextInformationType" minOccurs="0"/>
      </sequence>
     </complexType>
    </element>
    <element name="CreationTool" type="mpeg7:CreationToolType"
       minOccurs="0" maxOccurs="unbounded"/>
    <element name="CopyrightString" type="mpeg7:TextualType"
       minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.9: MASL: `CreationExtType`

```
<complexType name="CreationInformationExtType">
 <complexContent>
  <extension base="mpeg7:DSType">
   <sequence>
    <element name="Creation" type="masl:CreationExtType"/>
    <element name="Classification" type="mpeg7:ClassificationType"
        minOccurs="0"/>
    <element name="RelatedMaterial" type="mpeg7:
       RelatedMaterialType" minOccurs="0" maxOccurs="unbounded"/>
    <element name="RelatedInformation" type="masl:
       RelatedInformationType" minOccurs="0" maxOccurs="unbounded
       "/>
    <element name="RelatedAnnotation" type="masl:
       ContentAnnotationType" minOccurs="0" maxOccurs="unbounded
       "/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.10: MASL: `CreationInformationExtType`

```
<complexType name="CreationPreferencesExtType">
 <complexContent>
  <extension base="mpeg7:DSType">
   <sequence>
    <element name="Title" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <simpleContent>
       <extension base="mpeg7:TitleType">
        <attribute name="preferenceValue" type="masl:
           preferenceValueExtType" use="optional" default="5"/>
       </extension>
      </simpleContent>
     </complexType>
    </element>
    <element name="Creator" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <complexContent>
       <extension base="mpeg7:CreatorType">
        <attribute name="preferenceValue" type="masl:
           preferenceValueExtType" use="optional" default="5"/>
       </extension>
```

```
  </complexContent>
 </complexType>
</element>
<element name="Keyword" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <simpleContent>
   <extension base="mpeg7:TextualType">
    <attribute name="preferenceValue" type="masl:
       preferenceValueExtType" use="optional" default="5"/>
   </extension>
  </simpleContent>
 </complexType>
</element>
<element name="Location" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <complexContent>
   <extension base="mpeg7:PlaceType">
    <attribute name="preferenceValue" type="masl:
       preferenceValueExtType" use="optional" default="5"/>
   </extension>
  </complexContent>
 </complexType>
</element>
<element name="DatePeriod" minOccurs="0" maxOccurs="unbounded
   ">
 <complexType>
  <complexContent>
   <extension base="mpeg7:TimeType">
    <attribute name="preferenceValue" type="masl:
       preferenceValueExtType" use="optional" default="5"/>
   </extension>
  </complexContent>
 </complexType>
</element>
<element name="Context" minOccurs="0" maxOccurs="unbounded">
 <complexType>
  <complexContent>
   <extension base="masl:ContextInformationType">
    <attribute name="preferenceValue" type="masl:
       preferenceValueExtType" use="optional" default="5"/>
   </extension>
  </complexContent>
 </complexType>
</element>
<element name="Tool" minOccurs="0" maxOccurs="unbounded">
 <complexType>
```

```
      <complexContent>
       <extension base="mpeg7:TermUseType">
        <attribute name="preferenceValue" type="masl:
           preferenceValueExtType" use="optional" default="5"/>
       </extension>
      </complexContent>
     </complexType>
    </element>
   </sequence>
   <attribute name="preferenceValue" type="masl:
      preferenceValueExtType" use="optional" default="5"/>
  </extension>
 </complexContent>
</complexType>
```

Listing A.11: MASL: `CreationPreferencesExtType`

```
<complexType name="DisplayCapabilityExtType">
 <complexContent>
  <extension base="dia:DisplayCapabilityType">
   <attribute name="outputDevice" type="string" use="optional"/>
  </extension>
 </complexContent>
</complexType>
```

Listing A.12: MASL: `DisplayCapabilityExtType`

```
<complexType name="FilteringAndSearchPreferencesExtType">
 <complexContent>
  <extension base="mpeg7:DSType">
   <sequence>
    <element name="CreationPreferences" type="masl:
       CreationPreferencesExtType" minOccurs="0" maxOccurs="
       unbounded"/>
    <element name="ClassificationPreferences" type="masl:
       ClassificationPreferencesExtType" minOccurs="0" maxOccurs="
       unbounded"/>
    <element name="SemanticPreferences" type="masl:
       SemanticPreferencesType" minOccurs="0" maxOccurs="unbounded
       "/>
    <element name="RelatedInformationPreferences" type="masl:
       RelatedInformationPreferencesType" minOccurs="0" maxOccurs
       ="unbounded"/>
    <element name="SourcePreferences" type="masl:
       SourcePreferencesExtType" minOccurs="0" maxOccurs="
       unbounded"/>
```

```
    <element name="PreferenceCondition" type="masl:
       PreferenceConditionExtType" minOccurs="0" maxOccurs="
       unbounded"/>
    <element name="FilteringAndSearchPreferences" type="masl:
       FilteringAndSearchPreferencesExtType" minOccurs="0"
       maxOccurs="unbounded"/>
   </sequence>
   <attribute name="preferenceValue" type="masl:
      preferenceValueExtType" use="optional" default="5"/>
  </extension>
 </complexContent>
</complexType>
```

Listing A.13: MASL: `FilteringAndSearchPreferencesExtType`

```
<complexType name="KeywordAnnotationExtType">
 <sequence>
  <element name="Keyword" maxOccurs="unbounded">
   <complexType>
    <simpleContent>
     <extension base="mpeg7:TextualType">
      <attribute name="rating" use="required">
       <simpleType>
        <restriction base="double">
         <minInclusive value="0"/>
         <maxInclusive value="10"/>
        </restriction>
       </simpleType>
      </attribute>
      <attribute name="ratingCount" type="integer" use="required
         "/>
      <attribute name="categoryRating" type="boolean" use="
         required"/>
     </extension>
    </simpleContent>
   </complexType>
  </element>
 </sequence>
</complexType>
```

Listing A.14: MASL: `KeywordAnnotationExtType`

```
<complexType name="MediaIdentificationExtType">
 <complexContent>
  <extension base="mpeg7:MediaIdentificationType">
   <sequence>
```

```
    <element name="TextDomain" type="mpeg7:ControlledTermUseType"
       minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.15: MASL: `MediaIdentificationExtType`

```
<complexType name="PersonExtType">
 <complexContent>
  <extension base="mpeg7:PersonType">
   <sequence>
    <element name="DateOfBirth" type="mpeg7:timePointType"
       minOccurs="0"/>
    <element name="Gender" minOccurs="0">
     <simpleType>
      <restriction base="string">
       <enumeration value="female"/>
       <enumeration value="male"/>
      </restriction>
     </simpleType>
    </element>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.16: MASL: `PersonExtType`

```
<complexType name="PowerCharacteristicsExtType">
 <complexContent>
  <extension base="dia:PowerCharacteristicsType">
   <attribute name="batteryBeingCharged" type="boolean" use="
      optional"/>
  </extension>
 </complexContent>
</complexType>
```

Listing A.17: MASL: `PowerCharacteristicsExtType`

```
<complexType name="PreferenceConditionExtType">
 <complexContent>
  <extension base="mpeg7:PreferenceConditionType">
   <sequence>
    <element name="Context" type="masl:ContextInformationType"
       minOccurs="0" maxOccurs="unbounded"/>
```

```
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.18: MASL: PreferenceConditionExtType

```
<simpleType name="preferenceValueExtType">
 <restriction base="integer">
  <minInclusive value="0"/>
  <maxInclusive value="10"/>
 </restriction>
</simpleType>
```

Listing A.19: MASL: preferenceValueExtType

```
<complexType name="RatingExtType">
 <complexContent>
  <extension base="mpeg7:RatingType">
   <sequence>
    <element name="RatingScheme">
     <complexType>
      <complexContent>
       <extension base="mpeg7:TermUseType">
        <attribute name="best" type="float" fixed="10"/>
        <attribute name="worst" type="float" fixed="0"/>
        <attribute name="style" use="required" fixed="higherBetter
           "/>
       </extension>
      </complexContent>
     </complexType>
    </element>
   </sequence>
   <attribute name="fixed" type="boolean"/>
   <attribute name="explicit" type="boolean" use="required"/>
  </extension>
 </complexContent>
</complexType>
```

Listing A.20: MASL: RatingExtType

```
<complexType name="RelatedInformationPreferencesType">
 <complexContent>
  <extension base="mpeg7:DSType">
   <sequence>
    <element name="InformationType" type="mpeg7:
       ControlledTermUseType"/>
```

```
    <element name="Information" type="mpeg7:TextualType"/>
   </sequence>
   <attribute name="preferenceValue" type="masl:
      preferenceValueExtType" use="optional" default="5"/>
  </extension>
 </complexContent>
</complexType>
```

Listing A.21: MASL: `RelatedInformationPreferencesType`

```
<complexType name="RelatedInformationType">
 <complexContent>
  <extension base=" mpeg7:DSType">
   <sequence>
    <element name="InformationType" type="mpeg7:
       ControlledTermUseType"/>
    <element name="Information" type="mpeg7:TextualType"/>
    <choice minOccurs="0">
     <element name="CreationInformation" type="masl:
        CreationInformationExtType"/>
     <element name=" CreationInformationRef " type=" mpeg7:
        ReferenceType "/>
    </choice>
    <choice minOccurs="0">
     <element name=" UsageInformation " type=" masl:
        UsageInformationExtType "/>
     <element name=" UsageInformationRef " type=" mpeg7:
        ReferenceType "/>
    </choice>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.22: MASL: `RelatedInformationType`

```
<complexType name="RuleType">
 <attribute name="bref" type="IDREF" use="required"/>
 <attribute name="sref" type="IDREF" use="required"/>
 <attribute name="priority">
  <simpleType>
   <restriction base="integer">
    <minInclusive value="0"/>
    <maxInclusive value="5"/>
   </restriction>
  </simpleType>
```

```
 </attribute>
</complexType>
```

Listing A.23: MASL: `RuleType`

```
<complexType name="SemanticPreferencesType">
 <complexContent>
  <extension base="mpeg7:DSType">
   <sequence>
    <element name="Object" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <complexContent>
       <extension base="mpeg7:ObjectType">
        <attribute name="preferenceValue" type="masl:
           preferenceValueExtType" use="optional" default="5"/>
       </extension>
      </complexContent>
     </complexType>
    </element>
    <element name="Event" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <complexContent>
       <extension base="mpeg7:EventType">
        <attribute name="preferenceValue" type="masl:
           preferenceValueExtType" use="optional" default="5"/>
       </extension>
      </complexContent>
     </complexType>
    </element>
    <element name="Concept" minOccurs="0" maxOccurs="unbounded">
     <complexType>
      <complexContent>
       <extension base="mpeg7:ConceptType">
        <attribute name="preferenceValue" type="masl:
           preferenceValueExtType" use="optional" default="5"/>
       </extension>
      </complexContent>
     </complexType>
    </element>
    <element name="SemanticState" minOccurs="0" maxOccurs="
       unbounded">
     <complexType>
      <complexContent>
       <extension base="mpeg7:SemanticStateType">
        <attribute name="preferenceValue" type="masl:
           preferenceValueExtType" use="optional" default="5"/>
```

```
      </extension>
     </complexContent>
    </complexType>
   </element>
   <element name="SemanticPlace" minOccurs="0" maxOccurs="
      unbounded">
    <complexType>
     <complexContent>
      <extension base="mpeg7:SemanticPlaceType">
       <attribute name="preferenceValue" type="masl:
          preferenceValueExtType" use="optional" default="5"/>
      </extension>
     </complexContent>
    </complexType>
   </element>
   <element name="SemanticTime" minOccurs="0" maxOccurs="
      unbounded">
    <complexType>
     <complexContent>
      <extension base="mpeg7:SemanticTimeType">
       <attribute name="preferenceValue" type="masl:
          preferenceValueExtType" use="optional" default="5"/>
      </extension>
     </complexContent>
    </complexType>
   </element>
  </sequence>
  <attribute name="preferenceValue" type="masl:
     preferenceValueExtType" use="optional" default="5"/>
 </extension>
</complexContent>
</complexType>
```

Listing A.24: MASL: `SemanticPreferencesType`

```
<complexType name="SituationType">
 <sequence>
  <element name="SituationInformation" maxOccurs="unbounded">
   <complexType>
    <choice>
     <element name="Context">
      <complexType>
       <choice>
        <element name="NetworkCharacteristic" type="dia:
           NetworkCharacteristicBaseType"/>
```

```
        <element name="NaturalEnvironmentCharacteristic" type="dia
            :NaturalEnvironmentCharacteristicBaseType"/>
       </choice>
      </complexType>
     </element>
     <element name="DeviceParameter" type="dia:
         TerminalCapabilityBaseType"/>
     <element name="UserPreference">
      <complexType>
       <choice>
        <element name="AudioPreferences" type="dia:
            AudioPresentationPreferencesType"/>
        <element name="DisplayPresentationPreferences" type="dia:
            DisplayPresentationPreferencesType"/>
        <element name="GraphicsPresentationPreferences" type="dia:
            GraphicsPresentationPreferencesType"/>
       </choice>
      </complexType>
     </element>
     <element name="Situation" type="masl:SituationType"/>
    </choice>
    <attribute name="constraint">
     <simpleType>
      <restriction base="string">
       <enumeration value="max"/>
       <enumeration value="maxEqual"/>
       <enumeration value="min"/>
       <enumeration value="minEqual"/>
       <enumeration value="equal"/>
      </restriction>
     </simpleType>
    </attribute>
   </complexType>
  </element>
  <element name="PrivacyPreferences" minOccurs="0">
   <complexType>
    <sequence>
     <element name="lockedData" maxOccurs="unbounded">
      <complexType>
       <attribute name="dataType" type="string" use="required"/>
      </complexType>
     </element>
    </sequence>
   </complexType>
  </element>
 </sequence>
```

```
 <attribute name="id" type="ID" use="required"/>
 <attribute name="connective">
  <simpleType>
   <restriction base="string">
    <enumeration value="or"/>
    <enumeration value="and"/>
    <enumeration value="non-or"/>
    <enumeration value="non-and"/>
   </restriction>
  </simpleType>
 </attribute>
</complexType>
```

Listing A.25: MASL: `SituationType`

```
<complexType name="TextAnnotationExtType">
 <choice maxOccurs="unbounded">
  <element name="FreeTextAnnotation" type="mpeg7:TextualType"/>
  <element name="StructuredAnnotation" type="mpeg7:
     StructuredAnnotationType"/>
  <element name="DependencyStructure" type="mpeg7:
     DependencyStructureType"/>
  <element name="KeywordAnnotation" type="mpeg7:
     KeywordAnnotationType"/>
  <element name="RatedKeywordAnnotation" type="masl:
     KeywordAnnotationExtType"/>
 </choice>
 <attribute name="relevance" type="mpeg7:zeroToOneType" use="
    optional"/>
 <attribute name="confidence" type="mpeg7:zeroToOneType" use="
    optional"/>
 <attribute ref="xml:lang"/>
</complexType>
```

Listing A.26: MASL: `TextAnnotationExtType`

```
<complexType name="TextSegmentType">
 <complexContent>
  <extension base="mpeg7:SegmentType">
   <sequence>
    <choice minOccurs="0">
     <element name="SpatialLocator" type="mpeg7:RegionLocatorType
        "/>
     <element name="SpatialMask" type="mpeg7:SpatialMaskType"/>
    </choice>
    <choice minOccurs="0">
```

```
      <element name="MediaTimePoint" type="mpeg7:mediaTimePointType
          "/>
      <element name="MediaRelTimePoint" type="mpeg7:
          MediaRelTimePointType"/>
      <element name="MediaRelIncrTimePoint" type="mpeg7:
          MediaRelIncrTimePointType"/>
     </choice>
     <element name="FontColor" type="mpeg7:TermUseType" minOccurs
        ="0"/>
     <element name="FontSize" type="positiveInteger" minOccurs
        ="0"/>
     <element name="FontType" type="string" minOccurs="0"/>
     <element name="SpatialDecomposition" type="mpeg7:
        StillRegionSpatialDecompositionType" minOccurs="0"
        maxOccurs="unbounded"/>
    </sequence>
   </extension>
  </complexContent>
</complexType>
```

Listing A.27: MASL: `TextSegmentType`

```
<complexType name="TextType">
 <complexContent>
  <extension base="mpeg7:MultimediaContentType">
   <sequence>
    <element name="Text" type="masl:TextSegmentType"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.28: MASL: `TextType`

```
<complexType name="TimeDependentAttributeType">
 <sequence>
  <element name="TimeDependentAttributeValue" maxOccurs="unbounded
      ">
   <complexType>
    <sequence>
     <element name="ProgramIdentifier" type="mpeg7:UniqueIDType"
         maxOccurs="unbounded"/>
     <element name="AverageRating" type="masl:RatingExtType"
         minOccurs="0"/>
     <element name="Validity" type="masl:ValidityType"/>
    </sequence>
```

```
    <attribute name="value" type="string" use="required"/>
   </complexType>
  </element>
 </sequence>
 <attribute name="typeName" type="string" use="required"/>
</complexType>
```

Listing A.29: MASL: TimeDependentAttributeType

```
<complexType name="TimeDependentCombinationType">
 <sequence>
  <element name="TimeDependentCombinationValue" maxOccurs="
     unbounded">
   <complexType>
    <sequence>
     <element name="ProgramIdentifier" type="mpeg7:UniqueIDType"
        maxOccurs="unbounded"/>
     <element name="AverageRating" type="masl:RatingExtType"
        minOccurs="0"/>
     <element name="Validity" type="masl:ValidityType"/>
    </sequence>
    <attribute name="firstValue" type="string" use="required"/>
    <attribute name="secondValue" type="string" use="required"/>
   </complexType>
  </element>
 </sequence>
 <attribute name="firstType" type="string" use="required"/>
 <attribute name="secondType" type="string" use="required"/>
</complexType>
```

Listing A.30: MASL: TimeDependentCombinationType

```
<complexType name="UsageHistorySummaryType">
 <sequence minOccurs="0" maxOccurs="unbounded">
  <element name="ConsumedContent">
   <complexType>
    <sequence>
     <element name="ProgramIdentifier" type="mpeg7:UniqueIDType"/>
     <element name="Title" type="mpeg7:TitleType"/>
     <element name="Rating" type="masl:RatingExtType" minOccurs
        ="0"/>
     <element name="TimeStamp" type="mpeg7:timePointType"/>
     <element name="Validity" type="masl:ValidityType"/>
    </sequence>
   </complexType>
  </element>
```

```
 </sequence>
</complexType>
```

Listing A.31: MASL: `UsageHistorySummaryType`

```
<complexType name="UsageHistoryType">
 <sequence>
  <element name="UsageHistorySummary" type="masl:
     UsageHistorySummaryType" minOccurs="0"/>
  <element name="UserActionHistory" type="mpeg7:
     UserActionHistoryType" minOccurs="0" maxOccurs="unbounded"/>
 </sequence>
</complexType>
```

Listing A.32: MASL: `UsageHistoryType`

```
<complexType name="UsageInformationExtType">
 <complexContent>
  <extension base="mpeg7:UsageInformationType">
   <sequence>
    <element name="UsageContext" type="masl:ContextInformationType
       " minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Listing A.33: MASL: `UsageInformationExtType`

```
<simpleType name="ValidityType">
 <restriction base="decimal">
  <minInclusive value="0"/>
  <maxInclusive value="1"/>
 </restriction>
</simpleType>
```

Listing A.34: MASL: `ValidityType`

A.2. MASL Classification Schemes

Die folgenden Classification Schemes wurden für MASL neu definiert bzw. erweitert (in alphabetischer Reihenfolge).

```
<ClassificationScheme uri="urn:MASL:schema:2009:ContentCS:2009"
   domain="//MediaInformation/MediaProfile/MediaFormat/Content">
```

```
 <Term termID="1">
  <Name xml:lang="en">Audio</Name>
   <Term termID="1.1">
    <Name xml:lang="en">Natural</Name>
   </Term>
   <Term termID="1.2">
    <Name xml:lang="en">Synthetic</Name>
   </Term>
 </Term>
 <Term termID="2">
  <Name xml:lang="en">Audiovisual</Name>
 </Term>
 <Term termID="3">
  <Name xml:lang="en">Scene</Name>
 </Term>
 <Term termID="4">
  <Name xml:lang="en">Visual</Name>
  <Term termID="4.1">
   <Name xml:lang="en">Image</Name>
  </Term>
  <Term termID="4.2">
   <Name xml:lang="en">Video</Name>
  </Term>
  <Term termID="4.3">
   <Name xml:lang="en">Graphics</Name>
  </Term>
 </Term>
 <Term termID="5">
  <Name xml:lang="en">Textual</Name>
 </Term>
</ClassificationScheme>
```

Listing A.35: Content CS

```
<ClassificationScheme uri="urn:MASL:schema:2009:DisseminationExtCS
   :2009"
 domain="//CreationInformation/RelatedMaterial/DisseminationFormat
 //UsageInformation/Availability/Dissemination/DisseminationFormat
 //UserPreferences/FilteringAndSearchPreferences/SourcePreferences
    /DisseminationFormat">
 <Term termID="6.4">
  <Name xml:lang="en">BluRay</Name>
  <Definition xml:lang="en">BluRay</Definition>
 </Term>
 <Term termID="11">
  <Name xml:lang="en">AirInterface</Name>
```

```
   <Definition xml:lang="en">Publication available through the air
       interface</Definition>
 </Term>
</ClassificationScheme>
```

Listing A.36: Erweiterung des Dissemination CS

```
<ClassificationScheme uri="urn:MASL:schema:2009:FormatExtCS:2009"
   domain="//CreationInformation/Classification/Form">
 <Term termID="1">
  <Name xml:lang="en">Proprietary</Name>
  <Definition xml:lang="en">
   For use where proprietary extensions are required, or the use
      of keywords that do not fit in any classification below
  </Definition>
 </Term>
 <Term termID="2">
  <Name xml:lang="en">NON-FICTION/INFORMATION</Name>
  <Definition xml:lang="en">Time-sensitive information</Definition
     >
 </Term>
 <Term termID="3">
  <Name xml:lang="en">SPORTS</Name>
 </Term>
 <Term termID="4">
  <Name xml:lang="en">FICTION/DRAMA</Name>
  <Definition xml:lang="en">
   Programme consisting of a prose or verse composition, resp. one
       telling a story, written for or as if for performance by
      actors, puppets or animated
  </Definition>
 </Term>
 <Term termID="5">
  <Name xml:lang="en">AMUSEMENT/ENTERTAINMENT</Name>
 </Term>
 <Term termID="6">
  <Name xml:lang="en">MUSIC</Name>
 </Term>
 <Term termID="7">
  <Name xml:lang="en">INTERACTIVE GAMES</Name>
 </Term>
 <Term termID="8">
  <Name xml:lang="en">LEISURE/HOBBY/LIFESTYLE</Name>
 </Term>
 <Term termID="9">
  <Name xml:lang="en">ADULT</Name>
```

```
 </Term>
</ClassificationScheme>
```

Listing A.37: Format CS

```
<ClassificationScheme uri="urn:MASL:schema:2009:GenreExtCS:2009"
   domain="//CreationInformation/Classification/Genre">
 <Term termID="1.2.6">
  <Name xml:lang="en">Gourmet eating/cooking</Name>
  <Term termID="1.2.6.1">
   <Name xml:lang="en">European cooking</Name>
   <Term termID="1.2.6.1.1">
    <Name xml:lang="en">Italian cooking</Name>
   </Term>
   <Term termID="1.2.6.1.2">
    <Name xml:lang="en">German cooking</Name>
   </Term>
   <Term termID="1.2.6.1.3">
    <Name xml:lang="en">Greek cooking</Name>
   </Term>
   <Term termID="1.2.6.1.4">
    <Name xml:lang="en">Spanish cooking</Name>
   </Term>
   ...
  </Term>
  <Term termID="1.2.6.2">
   <Name xml:lang="en">Asian cooking</Name>
   <Term termID="1.2.6.2.1">
    <Name xml:lang="en">Chinese cooking</Name>
   </Term>
   <Term termID="1.2.6.2.2">
    <Name xml:lang="en">Thai cooking</Name>
   </Term>
   ...
  </Term>
  <Term termID="1.2.6.3">
   <Name xml:lang="en">African cooking</Name>
   <Term termID="1.2.6.3.1">
    <Name xml:lang="en">Egypt cooking</Name>
   </Term>
   <Term termID="1.2.6.3.2">
    <Name xml:lang="en">Southafrican cooking</Name>
   </Term>
   ...
  </Term>
  <Term termID="1.2.6.4">
```

```
   <Name xml:lang="en">American cooking</Name>
   <Term termID="1.2.6.4.1">
    <Name xml:lang="en">US American cooking</Name>
   </Term>
   <Term termID="1.2.6.4.2">
    <Name xml:lang="en">Canadian cooking</Name>
   </Term>
   ...
  </Term>
  <Term termID="1.2.6.5">
   <Name xml:lang="en">Australian cooking</Name>
  </Term>
  <Term termID="1.2.6.6">
   <Name xml:lang="en">Other</Name>
   <Term termID="1.2.6.6.1">
    <Name xml:lang="en">Fish restaurant</Name>
   </Term>
   <Term termID="1.2.6.6.2">
    <Name xml:lang="en">Vegetarian restaurant</Name>
   </Term>
   ...
  </Term>
 </Term>
 ...
 <Term termID="1.10.2">
  <Name xml:lang="en">Art</Name>
  <Term termID="1.10.2.1">
      <Name xml:lang="en">Gallery</Name>
  </Term>
  <Term termID="1.10.2.2">
      <Name xml:lang="en">Ancient Cultures</Name>
  </Term>
  <Term termID="1.10.2.3">
      <Name xml:lang="en">Prehistoric Museum</Name>
  </Term>
  ...
 </Term>
</ClassificationScheme>
```

Listing A.38: Erweiterung des GenreCS

```
<ClassificationScheme uri="urn:MASL:schema:2009:InformationCS
   :2009" domain="//CreationInformation/RelatedInformation/
   Information">
 <Term termID="1">
  <Name xml:lang="en">Static Information</Name>
```

```
  <Definition xml:lang="en">Static Informal Descriptions</
     Definition>
  <Term termID="1.1">
   <Name xml:lang="en">Opening Hours</Name>
  </Term>
  <Term termID="1.2">
   <Name xml:lang="en">Maximum Capacity</Name>
  </Term>
  <Term termID="1.3">
   <Name xml:lang="en">Dress Code</Name>
  </Term>
  ...
 </Term>
 <Term termID="2">
  <Name xml:lang="en">Dynamic Information</Name>
  <Definition xml:lang="en">Dynamic Informal Descriptions</
     Definition>
  <Term termID="2.1">
   <Name xml:lang="en">Current Capacity</Name>
  </Term>
  <Term termID="2.2">
   <Name xml:lang="en">Todays Special</Name>
  </Term>
  ...
 </Term>
</ClassificationScheme>
```

Listing A.39: Information CS

```
<ClassificationScheme uri="urn:MASL:schema:2009:MediumExtCS:2009"
    domain="//MediaInformation/MediaProfile/MediaFormat/Medium">
 <Term termID="1.4">
  <Name xml:lang="en">BluRay</Name>
  <Definition xml:lang="en">BluRay</Definition>
 </Term>
</ClassificationScheme>
```

Listing A.40: Medium CS

```
<ClassificationScheme uri="urn:MASL:schema:2009:FormatExtCS:2009"
    domain="//MediaInformation/MediaIdentification/TextDomain">
 <Term termID="1">
 <Name xml:lang="en">Source</Name>
 <Definition xml:lang="en">Type of text source</Definition>
  <Term termID="1.1">
   <Name xml:lang="en">Ink</Name>
```

```
  </Term>
  <Term termID="1.2">
   <Name xml:lang="en">Keyboard</Name>
  </Term>
  <Term termID="1.3">
   <Name xml:lang="en">Scanned Document</Name>
  </Term>
 </Term>
 <Term termID="2">
  <Name xml:lang="en">Acquisition</Name>
  <Definition xml:lang="en">Type of Content</Definition>
  <Term termID="2.1">
   <Name xml:lang="en">Book</Name>
  </Term>
  <Term termID="2.2">
   <Name xml:lang="en">Article</Name>
  </Term>
  <Term termID="2.3">
   <Name xml:lang="en">InProceedings</Name>
  </Term>
  <Term termID="2.4">
   <Name xml:lang="en">Manual</Name>
  </Term>
  <Term termID="2.5">
   <Name xml:lang="en">Thesis</Name>
  </Term>
  <Term termID="2.6">
   <Name xml:lang="en">Misc</Name>
  </Term>
 </Term>
 <Term termID="3">
  <Name xml:lang="en">Use</Name>
  <Definition xml:lang="en">Original usage domain</Definition>
  <Term termID="3.1">
   <Name xml:lang="en">Scientific</Name>
  </Term>
  <Term termID="3.2">
   <Name xml:lang="en">Medical</Name>
  </Term>
  <Term termID="3.3">
   <Name xml:lang="en">Security</Name>
  </Term>
  <Term termID="3.4">
   <Name xml:lang="en">Broadcast</Name>
  </Term>
  <Term termID="3.4">
```

```
   <Name xml:lang="en">Remote-sensing</Name>
  </Term>
 </Term>
</ClassificationScheme>
```

Listing A.41: TextDomainCS

A.3. Schemadefinition der Konfigurationsschnittstelle

Das folgende Schema zeigt den Aufbau der Konfigurationsdatei, die in das Framework zur Kontext-abhängigen Auswahl von multimedialen Inhalten eingelesen wird.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema">
 <element name="Config">
  <complexType>
   <sequence>
    <element name="Server">
     <complexType>
      <sequence>
       <element name="Profile">
        <complexType>
         <sequence>
          <element name="ProfileInfo" type="ProfileInfoType"
             minOccurs="0" maxOccurs="unbounded"/>
         </sequence>
         <attribute name="updateRate" type="time"/>
         <attribute name="updateMethod"/>
        </complexType>
       </element>
       <element name="Matching" type="MatcherType"/>
       <element name="Aggregator" type="AggregatorType"/>
       <element name="GlobalThreshold" type="ThresholdType"/>
      </sequence>
     </complexType>
    </element>
    <element name="Client">
     <complexType>
      <sequence>
       <element name="Matching" type="MatcherType"/>
       <element name="Aggregator" type="AggregatorType"/>
       <element name="GlobalThreshold" type="ThresholdType"/>
       <element name="RetrieveContent">
        <complexType>
         <attribute name="number" type="integer" use="required"/>
```

```
        </complexType>
       </element>
      </sequence>
     </complexType>
    </element>
   </sequence>
   <attribute name="appID" type="IDREF" use="required"/>
   <attribute name="classID" type="IDREF"/>
  </complexType>
 </element>
 <complexType name="ProfileInfoType">
  <attribute name="name" type="string" use="required"/>
 </complexType>
 <complexType name="MatcherType">
  <sequence>
   <element name="ProfileMatcherInfo" minOccurs="0" maxOccurs="
      unbounded"/>
  </sequence>
 </complexType>
 <complexType name="ProfileMatcherType">
  <complexContent>
   <extension base="ProfileInfoType">
    <attribute name="matcher" type="string" use="required"/>
    <attribute name="threshold" type="double"/>
    <attribute name="weight" type="double"/>
   </extension>
  </complexContent>
 </complexType>
 <complexType name="AggregatorType">
  <attribute name="id" type="ID" use="required"/>
  <attribute name="value" type="double"/>
  <attribute name="rangeMin" type="double"/>
  <attribute name="rangeMax" type="double"/>
 </complexType>
 <complexType name="ThresholdType">
  <attribute name="value" type="double" use="required"/>
 </complexType>
</schema>
```

Listing A.42: Besipiel für eine Konfigurationsschnittstelle

A.4. Schemadefinition der FormattingCapabilities und Formatabilities

Die folgenden Schemadefinition legt den Aufbau der Listen der FormattingCapabilities und Formatabilities fest.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema">
 <element name="FormattingCapabilities">
  <complexType>
   <sequence>
    <element name="Conversion" type="form:ConversionType"
       minOccurs="0" maxOccurs="unbounded"/>
    <element name="Scaling" type="form:ScalingType" minOccurs="0"
       maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
 <element name="Formatabilities">
  <complexType>
   <sequence>
    <element name="Conversion" type="form:ConversionType"
       minOccurs="0" maxOccurs="unbounded"/>
    <element name="Scaling" type="form:ScalingType" minOccurs="0"
       maxOccurs="unbounded"/>
   </sequence>
  </complexType>
 </element>
 <complexType name="ConversionType">
  <attribute name="name" type="string" use="required"/>
  <attribute name="to" type="string" use="required"/>
  <attribute name="from" type="string" use="required"/>
 </complexType>
 <complexType name="ScalingType">
  <attribute name="name" type="string" use="required"/>
  <attribute name="format" type="string" use="required"/>
  <attribute name="scale" type="string" use="required"/>
 </complexType>
</schema>
```

Listing A.43: FormattingCapabilities und Formatabilities

Anhang B.

Beispiele

Um die in dieser Arbeit erläuterten Konzepte besser verdeutlichen zu können, werden im Folgenden Beispielbeschreibungen in MASL gegeben. Es handelt sich nicht um eine vollständige Beschreibungen, sondern lediglich um Ausschnitte. Die entsprechenden Werte des Inhalts wurden bei Bedarf aus der *Internet Movie Database* (IMDB)[1] entnommen.

B.1. Inhaltsbeschreibungen

```
<?xml version="1.0" encoding="UTF-8"?>
<masl:Metadata xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:masl="urn:MASL:schema:2009" xmlns:mpeg7="urn:
   mpeg:mpeg7:schema:2004">
 <masl:ProgramIdentifier authority="IMDB">tt0076759</masl:
    ProgramIdentifier>
 <!-- Metadaten der Beschreibung -->
 <masl:DescriptionMetadata>
  <mpeg7:Confidence>1.0</mpeg7:Confidence>
  <mpeg7:Version>1.0</mpeg7:Version>
  <mpeg7:LastUpdate>2009-04-03</mpeg7:LastUpdate>
  <mpeg7:Creator>
   <mpeg7:Role href="mpeg7:creatorCS">
    <mpeg7:Name>Creator</mpeg7:Name></mpeg7:Role>
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>Diana</mpeg7:GivenName>
     <mpeg7:FamilyName>Weiß</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </mpeg7:Creator>
  <mpeg7:CreationLocation>
   <mpeg7:Region>de</mpeg7:Region>
   <mpeg7:PostalAddress>
```

[1] http://www.imdb.com/

```
    <mpeg7:AddressLine>Oettingenstr. 67</mpeg7:AddressLine>
    <mpeg7:AddressLine>München</mpeg7:AddressLine>
    <mpeg7:PostingIdentifier>80538</mpeg7:PostingIdentifier>
   </mpeg7:PostalAddress>
  </mpeg7:CreationLocation>
  <mpeg7:CreationTime>2009-04-03</mpeg7:CreationTime>
  <mpeg7:Instrument>
   <mpeg7:Tool>
    <mpeg7:Name>XML-Spy</mpeg7:Name>
   </mpeg7:Tool>
  </mpeg7:Instrument>
 </masl:DescriptionMetadata>
 <!-- Beschreibung des Medientyps -->
 <masl:DescribedEntity xsi:type="mpeg7:AudioVisualType">
  <mpeg7:AudioVisual/></masl:DescribedEntity>
 <!-- Semantische Beschreibung -->
 <masl:SemanticDescription>
  ...
  <mpeg7:SemanticBase xsi:type="mpeg7:AgentObjectType">
   ...
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>Luke</mpeg7:GivenName>
     <mpeg7:FamilyName>Skywalker</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </mpeg7:SemanticBase>
  <mpeg7:SemanticBase xsi:type="mpeg7:AgentObjectType">
   ...
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>Han</mpeg7:GivenName>
     <mpeg7:FamilyName>Solo</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </mpeg7:SemanticBase>
 </masl:SemanticDescription>
 <!-- Beschreibung der Dateiinformationen -->
 <masl:MediaDescription>
  <mpeg7:MediaProfile>
   <mpeg7:MediaFormat>
    <mpeg7:Content href="MPEG7ContentCS">
     <mpeg7:Name>audiovisual</mpeg7:Name>
    </mpeg7:Content>
    <mpeg7:Medium href="urn:MASL:schema:2009:MediumExtCS
       :2009:1.4">
     <mpeg7:Name>BluRay</mpeg7:Name></mpeg7:Medium>
   </mpeg7:MediaFormat>
  </mpeg7:MediaProfile>
 </masl:MediaDescription>
```

```
<!-- Beschreibung der Erzeugung -->
<masl:CreationDescription>
 <masl:Creation>
  <masl:Title type="original">Star Wars</masl:Title>
  <masl:Abstract xsi:type="masl:TextAnnotationExtType">
   <masl:RatedKeywordAnnotation>
    <masl:Keyword rating="7.4" ratingCount="235" categoryRating="
       false">Jedi Knight</masl:Keyword>
    <masl:Keyword rating="5.3" ratingCount="865" categoryRating="
       false">Space</masl:Keyword>
    <masl:Keyword rating="8.1" ratingCount="475" categoryRating="
       false">Rebellion</masl:Keyword>
   </masl:RatedKeywordAnnotation>
  </masl:Abstract>
  <masl:Creator>
   <mpeg7:Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:DIRECTOR"/>
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>George</mpeg7:GivenName>
     <mpeg7:FamilyName>Lucas</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </masl:Creator>
  <masl:Creator>
   <mpeg7:Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:SCRIPTWRITER
      "/>
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>George</mpeg7:GivenName>
     <mpeg7:FamilyName>Lucas</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </masl:Creator>
  <masl:Creator>
   <mpeg7:Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:ACTOR"/>
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>Marc</mpeg7:GivenName>
     <mpeg7:FamilyName>Hamill</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </masl:Creator>
  <masl:Creator>
   <mpeg7:Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:ACTOR"/>
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>Harrison</mpeg7:GivenName>
     <mpeg7:FamilyName>Ford</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </masl:Creator>
  <masl:CreationCoordinates>
   <masl:Location>
    <mpeg7:Name>Death Valley National Park</mpeg7:Name>
```

```
    <mpeg7:GeographicPosition>
     <mpeg7:Point latitude="36.64" longitude="-117.50"></mpeg7:
        Point>
    </mpeg7:GeographicPosition>
   </masl:Location>
  </masl:CreationCoordinates>
 </masl:Creation>
 <masl:Classification>
  <mpeg7:Form href="urn:MASL:schema:2009:GenreExtCS:4">
   <mpeg7:Name>Fiction/Drama</mpeg7:Name></mpeg7:Form>
  <mpeg7:Genre href="urn:MASL:schema:2009:GenreExtCS:6.4">
   <mpeg7:Name>Science fiction</mpeg7:Name></mpeg7:Genre>
  <mpeg7:Genre href="urn:MASL:schema:2009:GenreExtCS:6.13">
   <mpeg7:Name>Adventure</mpeg7:Name></mpeg7:Genre>
  <mpeg7:Genre href="urn:MASL:schema:2009:GenreExtCS:6.14">
   <mpeg7:Name>Fantasy</mpeg7:Name></mpeg7:Genre>
  <mpeg7:Purpose href="urn:mpeg:mpeg7:cs:IntentionCS:2001:1.1">
   <mpeg7:Name xml:lang="en">Pure entertainment</mpeg7:Name>
  </mpeg7:Purpose>
  <mpeg7:Language type="original">en</mpeg7:Language>
  <mpeg7:Release date="1977-05-25">
   <mpeg7:Region>US</mpeg7:Region>
  </mpeg7:Release>
  <mpeg7:ParentalGuidance>
   <mpeg7:ParentalRating href="urn:mpeg:mpeg7:cs:
      MPAAParentalRatingCS:2001:PG"/>
    <mpeg7:Region>US</mpeg7:Region>
  </mpeg7:ParentalGuidance>
  <mpeg7:MediaReview>
   <mpeg7:Rating>
    <mpeg7:RatingValue>8.8</mpeg7:RatingValue>
    <mpeg7:RatingScheme best="10" worst="1" style="higherBetter">
    <mpeg7:Name>Overall</mpeg7:Name>
    </mpeg7:RatingScheme>
   </mpeg7:Rating>
   <mpeg7:ReviewReference>
    <mpeg7:DisseminationFormat href="urn:mpeg:mpeg7:cs:
       DisseminationFormatCS:2001:4">
     <mpeg7:Name xml:lang="en">Internet</mpeg7:Name>
    </mpeg7:DisseminationFormat>
    <mpeg7:MediaLocator>
     <mpeg7:MediaUri>www.imdb.de</mpeg7:MediaUri>
    </mpeg7:MediaLocator>
   </mpeg7:ReviewReference>
  </mpeg7:MediaReview>
 </masl:Classification>
```

```
 </masl:CreationDescription>
 <!-- Beschreibung der Verwendung -->
 <masl:UsageDescription>...</masl:UsageDescription>
</masl:Metadata>
```

Listing B.1: Beispiel-Beschreibung des Filmes *Star Wars* in MASL

```
<?xml version="1.0" encoding="UTF-8"?>
<masl:Metadata xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:masl="urn:MASL:schema:2009" xmlns:mpeg7="urn:
   mpeg:mpeg7:schema:2004">
 <masl:ProgramIdentifier authority="DW">dm1</masl:
    ProgramIdentifier>
 <!-- Metadaten der Beschreibung -->
 <masl:DescriptionMetadata>...</masl:DescriptionMetadata>
 <!-- Beschreibung des Medientyps -->
 <masl:DescribedEntity xsi:type="masl:TextType">
  <masl:Text>
   <masl:FontColor>
    <mpeg7:Name>blue</mpeg7:Name>
   </masl:FontColor>
  </masl:Text>
 </masl:DescribedEntity>
 <!-- Semantische Beschreibung -->
 <masl:SemanticDescription>
  ...
  <mpeg7:SemanticBase xsi:type="mpeg7:SemanticPlaceType">
   <mpeg7:Label>
    <mpeg7:Name>Deutsches Museum</mpeg7:Name>
   </mpeg7:Label>
   <mpeg7:Place>
    <mpeg7:GeographicPosition>
     <mpeg7:Point latitude="48.13" longitude="11.58"/>
    </mpeg7:GeographicPosition>
   </mpeg7:Place>
  </mpeg7:SemanticBase>
  <mpeg7:SemanticBase xsi:type="mpeg7:SemanticTimeType">
   <mpeg7:Label>
    <mpeg7:Name>Beschriebene Zeit: 1870 - 1900</mpeg7:Name>
   </mpeg7:Label>
   <mpeg7:SemanticTimeInterval>
    <mpeg7:TimePoint origin="January 1, 1870">
     <mpeg7:Displacement unit="year" value="30"
      measurementType="length"/>
     <mpeg7:Direction unit="direction" value="after"
        measurementType="during"/>
```

```
    </mpeg7:TimePoint>
   </mpeg7:SemanticTimeInterval>
  </mpeg7:SemanticBase>
 </masl:SemanticDescription>
 <!-- Beschreibung der Dateiinformationen -->
 <masl:MediaDescription>
  <mpeg7:MediaIdentification xsi:type="masl:
     MediaIdentificationExtType">
   <mpeg7:EntityIdentifier authority="DW">dm1</mpeg7:
      EntityIdentifier>
   <masl:TextDomain href="urn:MASL:schema:2009:TextDomainCS
      :2009:2.2">
    <mpeg7:Name>Article</mpeg7:Name>
   </masl:TextDomain>
  </mpeg7:MediaIdentification>
  <mpeg7:MediaProfile>...</mpeg7:MediaProfile>
 </masl:MediaDescription>
 <!-- Beschreibung der Erzeugung -->
 <masl:CreationDescription>
  <masl:Creation>
   <masl:Title>Deutsches Museum um die Jahrhunderwende</masl:Title
      >
   <masl:Abstract>
    <masl:KeywordAnnotation>
     <mpeg7:Keyword>Museum</mpeg7:Keyword>
     <mpeg7:Keyword>Kaiserzeit</mpeg7:Keyword>
     <mpeg7:Keyword>Geschichte</mpeg7:Keyword>
     <mpeg7:Keyword>Technik</mpeg7:Keyword>
    </masl:KeywordAnnotation>
   </masl:Abstract>
   <masl:Creator>
    <mpeg7:Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:PUBLISHER"/>
    <mpeg7:Agent xsi:type="mpeg7:PersonType">
     <mpeg7:Name><mpeg7:GivenName>Diana</mpeg7:GivenName>
      <mpeg7:FamilyName>Weiß</mpeg7:FamilyName></mpeg7:Name>
    </mpeg7:Agent>
   </masl:Creator>
   <masl:CreationCoordinates>
    <masl:Location><mpeg7:PostalAddress>
      <mpeg7:AddressLine>Oettingenstr. 67</mpeg7:AddressLine>
      <mpeg7:AddressLine>München</mpeg7:AddressLine>
      <mpeg7:PostingIdentifier>81549</mpeg7:PostingIdentifier>
     </mpeg7:PostalAddress></masl:Location>
    <masl:Date><mpeg7:TimePoint>2009-04-03</mpeg7:TimePoint></masl
       :Date>
```

```
    <masl:ContextInformation scale="degree" type="temperature"
       value="20"/>
   </masl:CreationCoordinates>
  </masl:Creation>
  <masl:Classification>
   ...
   <mpeg7:Purpose href="urn:mpeg:mpeg7:cs:IntentionCS:2001:3.1">
    <mpeg7:Name>General enrichment</mpeg7:Name>
   </mpeg7:Purpose>
   <mpeg7:Target><mpeg7:Age min="12" max="100"/></mpeg7:Target>
  </masl:Classification>
  <masl:RelatedInformation>
   <masl:InformationType href="urn:MASL:schema:2009:InformationCS
      ">
    <mpeg7:Name>Opening Hours</mpeg7:Name>
   </masl:InformationType>
   <masl:Information>9 - 17 Uhr</masl:Information>
  </masl:RelatedInformation>
  <masl:RelatedAnnotation type="image" >
   <masl:MediaUri>Annotation.jpeg</masl:MediaUri>
   <masl:DescriptionUri>Description.xml</masl:DescriptionUri>
  </masl:RelatedAnnotation>
 </masl:CreationDescription>
 <!-- Beschreibung der Verwendung -->
 <masl:UsageDescription>
  ...
  <masl:UsageContext scale="moodScale" type="mood" value="
     expectant"/>
  <masl:UsageContext scale="activityScale" type="activity" value="
     sightseeing"/>
  <masl:UsageContext xsi:type="masl:
     ContextInformationRestrictionType" scale="degree" type="
     temperature" value="25" constraint="maxEqual"></masl:
     UsageContext>
 </masl:UsageDescription>
</masl:Metadata>
```

Listing B.2: Beispiel-Beschreibung eines Texts über das *Deutsche Museum* in MASL

B.2. Nutzerprofil

```
<?xml version="1.0" encoding="UTF-8"?>
<masl:Profile xmlns:masl="urn:MASL:schema:2009" xmlns:mpeg7="urn:
   mpeg:mpeg7:schema:2004" xmlns:xsi="http://www.w3.org/2001/
   XMLSchema-instance" xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS">
```

```
<masl:DemographicInformation>
 <mpeg7:Name xml:lang="de">
  <mpeg7:GivenName>Diana</mpeg7:GivenName>
  <mpeg7:FamilyName>Weiß</mpeg7:FamilyName></mpeg7:Name>
 <mpeg7:Affiliation><mpeg7:Organization>
   <mpeg7:Name>Ludwig-Maximilians University</mpeg7:Name>
  </mpeg7:Organization></mpeg7:Affiliation>
 <mpeg7:ElectronicAddress>
  <mpeg7:Email>diana.weiss@ifi.lmu.de</mpeg7:Email>
 </mpeg7:ElectronicAddress>
 <masl:DateOfBirth>1979-09-09</masl:DateOfBirth>
 <masl:Gender>female</masl:Gender>
</masl:DemographicInformation>
<masl:UsageEnvironmentDescription>
 <masl:TerminalCapabilities xsi:type="masl:
    PowerCharacteristicsExtType" batteryBeingCharged="false"
    batteryTimeRemaining="4200" />
 <masl:TerminalCapabilities xsi:type="dia:AudioOutputsType">
  <dia:AudioOutput><dia:AudioOutputCapability xsi:type="masl:
     AudioOutputCapabilitiesExtType" outputDevice="headset" />
  </dia:AudioOutput>
 </masl:TerminalCapabilities>
 <masl:TerminalCapabilities xsi:type="dia:DisplaysType">
  <dia:Display><dia:DisplayCapability xsi:type="masl:
     DisplayCapabilityExtType" />
  </dia:Display>
 </masl:TerminalCapabilities>
 <masl:NetworkCharacteristics
  xsi:type="dia:NetworkConditionType">
  <dia:AvailableBandwidth maximum="256000" average="80000" />
  <dia:Delay packetTwoWay="330" delayVariation="66"/>
  <dia:Error packetLossRate="0.05"/>
 </masl:NetworkCharacteristics>
 <masl:NaturalEnvironmentCharacteristics xsi:type="dia:TimeType">
  <dia:Time xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS">
   <mpeg7:TimePoint>2009-05-04T21:00+01:00</mpeg7:TimePoint>
  </dia:Time>
 </masl:NaturalEnvironmentCharacteristics>
 <masl:NaturalEnvironmentCharacteristics
  xsi:type="dia:LocationType">
  <dia:Location><mpeg7:GeographicPosition>
    <mpeg7:Point latitude="48.15" longitude="11.59" />
   </mpeg7:GeographicPosition></dia:Location>
 </masl:NaturalEnvironmentCharacteristics>
 <masl:NaturalEnvironmentCharacteristics
  xsi:type="masl:ContextType">
```

```
   <masl:Context scale="degree" type="temperature" value="25" />
  </masl:NaturalEnvironmentCharacteristics>
  <masl:NaturalEnvironmentCharacteristics
   xsi:type="masl:ContextType">
   <masl:Context scale="degree" type="temperature" value="25"/>
   <masl:Context scale="weatherScale" type="weather"
    value="sunny"/>
   <masl:Context scale="activityScale" type="activity"
    value="recreation"/>
   <masl:Context scale="moodScale" type="mood" value="excited"/>
   <masl:Context scale="moodScale" type="mood" value="expectant"/>
  </masl:NaturalEnvironmentCharacteristics>
 </masl:UsageEnvironmentDescription>
 <masl:UsageHistory>
  <masl:UsageHistorySummary>
   <masl:ConsumedContent>
    <masl:ProgramIdentifier authority="IMDB">tt0076759
     </masl:ProgramIdentifier>
    <masl:Title xml:lang="en">Star Wars</masl:Title>
    <masl:Rating explicit="true" fixed="true">
     <mpeg7:RatingValue>10</mpeg7:RatingValue>
     <mpeg7:RatingScheme style="higherBetter"/>
     <masl:RatingScheme best="10" worst="0" style="higherBetter"/>
    </masl:Rating>
    <masl:TimeStamp>2008-12-24T22:13+01:00</masl:TimeStamp>
    <masl:Validity>1</masl:Validity>
   </masl:consumedContent>
   <masl:consumedContent>
    <masl:ProgramIdentifier authority="DW">dm1
     </masl:ProgramIdentifier>
    <masl:Title>Deutsches Museum um die Jahrhunderwende</masl:
       Title>
    ...
   </masl:consumedContent>
   <masl:consumedContent>
    <masl:ProgramIdentifier authority="IMDB">tt0080684
     </masl:ProgramIdentifier>
    <masl:Title xml:lang="en">The Empire Strikes Back</masl:Title>
    ...
   </masl:consumedContent>
   <masl:consumedContent>
    <masl:ProgramIdentifier authority="IMDB">tt0086190
     </masl:ProgramIdentifier>
    <masl:Title xml:lang="en">Return of the Jedi</masl:Title>
    ...
   </masl:consumedContent>
```

```
 </masl:UsageHistorySummary>
</masl:UsageHistory>
<masl:UserPreferences>
<masl:FilteringAndSearchPreferences>
 <masl:CreationPreferences preferenceValue="10" >
  <masl:Creator>
   <mpeg7:Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:DIRECTOR"/>
   <mpeg7:Agent xsi:type="mpeg7:PersonType">
    <mpeg7:Name><mpeg7:GivenName>George</mpeg7:GivenName>
     <mpeg7:FamilyName>Lucas</mpeg7:FamilyName></mpeg7:Name>
   </mpeg7:Agent>
  </masl:Creator>
 </masl:CreationPreferences>
</masl:FilteringAndSearchPreferences>
<masl:FilteringAndSearchPreferences>
 <masl:CreationPreferences>
  <masl:Keyword>Technik</masl:Keyword>
  <masl:Keyword>Naturwissenschaft</masl:Keyword>
 </masl:CreationPreferences>
</masl:FilteringAndSearchPreferences>
<masl:FilteringAndSearchPreferences>
 <masl:SemanticPreferences preferenceValue="10" >
  <masl:SemanticTime>
   <mpeg7:Label>
    <mpeg7:Name>Präferenz für das 19te Jahrhundert</mpeg7:Name>
   </mpeg7:Label>
   <mpeg7:SemanticTimeInterval>
    <mpeg7:TimePoint origin="January 1, 1800">
     <mpeg7:Displacement unit="year" value="100" measurementType
        ="length"/>
     <mpeg7:Direction unit="direction" value="after"
        measurementType="during"></mpeg7:Direction>
    </mpeg7:TimePoint>
   </mpeg7:SemanticTimeInterval>
  </masl:SemanticTime>
 </masl:SemanticPreferences>
</masl:FilteringAndSearchPreferences>
</masl:UserPreferences>
<masl:TimeDependentUserPreferences>
 <masl:TimeDependentAttributes>
  <masl:TimeDependentAttribute typeName="Genre">
   <masl:TimeDependentAttributeValue value="Science fiction">
    <masl:ProgramIdentifier authority="IMDB">tt0076759
     </masl:ProgramIdentifier>
    <masl:ProgramIdentifier authority="IMDB">tt0080684
     </masl:ProgramIdentifier>
```

```
    <masl:ProgramIdentifier authority="IMDB">tt0086190
     </masl:ProgramIdentifier>
    <masl:AverageRating explicit="false">
     <mpeg7:RatingValue>8.7</mpeg7:RatingValue>
     <mpeg7:RatingScheme style="higherBetter"/>
    <masl:RatingScheme best="10" worst="0" style="higherBetter"/>
    </masl:AverageRating>
    <masl:Validity>0.3</masl:Validity>
   </masl:TimeDependentAttributeValue>
   <masl:TimeDependentAttributeValue value="Adventure">
   ...</masl:TimeDependentAttributeValue>
  </masl:TimeDependentAttribute>
 </masl:TimeDependentAttributes>
 <masl:TimeDependentAttributeCombinations>
  <masl:TimeDependentAttributeCombination firstType="Genre"
     secondType="Genre" >
   <masl:TimeDependentCombinationValue firstValue="Science
      fiction" secondValue="Adventure">
    <masl:ProgramIdentifier authority="IMDB">tt0076759
     </masl:ProgramIdentifier>
    <masl:ProgramIdentifier authority="IMDB">tt0080684
     </masl:ProgramIdentifier>
    <masl:ProgramIdentifier authority="IMDB">tt0086190
     </masl:ProgramIdentifier>
    <masl:AverageRating explicit="false">...</masl:AverageRating>
    <masl:Validity>0.9</masl:Validity>
   </masl:TimeDependentCombinationValue>
  </masl:TimeDependentAttributeCombination>
 </masl:TimeDependentAttributeCombinations>
</masl:TimeDependentUserPreferences>
<masl:PresentationPreferences>
 <dia:Audio>
  <dia:VolumeControl>0.75</dia:VolumeControl></dia:Audio>
 <dia:Display><dia:BrightnessPreference>
   <dia:BinNumber>255</dia:BinNumber>
   <dia:Value><dia:PreferredValue>138</dia:PreferredValue>
    <dia:ReferenceValue>103</dia:ReferenceValue></dia:Value>
   <dia:Value><dia:PreferredValue>152</dia:PreferredValue>
    <dia:ReferenceValue>150</dia:ReferenceValue></dia:Value>
  </dia:BrightnessPreference></dia:Display>
</masl:PresentationPreferences>
<masl:AdaptationRules>
 <masl:Situation id="lowBatteryPower" connective="or">
  <masl:SituationInformation>
   <masl:Situation id="lowBatteryPower1" connective="and">
    <masl:SituationInformation constraint="max">
```

```
      <masl:DeviceParameter xsi:type="dia:PowerCharacteristics
       Type" batteryTimeRemaining="4200" />
     </masl:SituationInformation>
     <masl:SituationInformation constraint="equal">
      <masl:DeviceParameter xsi:type="masl:PowerCharacteristicsExt
       Type" batteryBeingCharged="false" />
     </masl:SituationInformation>
    </masl:Situation>
   </masl:SituationInformation>
   <masl:SituationInformation>
    <masl:Situation id="lowBatteryPower2" connective="and">
     <masl:SituationInformation constraint="max">
      <masl:DeviceParameter xsi:type="dia:PowerCharacteristics
       Type" batteryTimeRemaining="1800" />
     </masl:SituationInformation>
     <masl:SituationInformation constraint="equal">
      <masl:DeviceParameter xsi:type="masl:PowerCharacteristicsExt
       Type" batteryBeingCharged="true" />
     </masl:SituationInformation>
    </masl:Situation>
   </masl:SituationInformation>
   <masl:PrivacyPreferences><masl:lockedData
    dataType="NaturalEnvironmentCharacteristics"/></masl:
       PrivacyPreferences>
  </masl:Situation>
  <masl:Behaviour id="subtitlesInsteadOfAudio">
   <masl:Function name ="remove"><masl:Object name ="audio" />
    </masl:Function >
   <masl:Function name="add"><masl:Object name="subtitles"/>
    </masl:Function>
   <masl:Function name="convert"><masl:Object name="video"/>
    <masl:Parameter name="mpeg -4"/></masl:Function>
  </masl:Behaviour>
  <masl:Rule sref="lowBatteryPower"
   bref="subtitlesInsteadOfAudio"/>
 </masl:AdaptationRules>
</masl:Profile>
```

Listing B.3: Beispiel-Profil in MASL

Anhang C.

Verwendete Bezeichnungen

Der folgende Abschnitt gibt einen Überblick über die wichtigsten in der Arbeit verwendeten Bezeichnungen:

C:	Gesamtmenge aller verfügbaren multimedialen Inhalte
c:	Ein einzelner multimedialer Inhalt, $c \in C$
$md(c)$:	Metadatenbeschreibung des Inhalts c gegeben in MASL
a_i:	i-ter Attributtyp, z. B. Genre
D_{a_i}:	Wertebereich des Attributtyps a_i
$a_i(j)$:	j-te Ausprägung des i-ten Attributtyps, auch Attributswert genannt z. B. Action bei Typ Genre
U:	Menge aller Nutzer
u:	Ein einzelner Nutzer, $u \in U$
U_u:	Menge der nächsten Nachbarn von Nutzer u
$pd(u)$:	Profilbeschreibung des Nutzer u
$R(c)$:	Bewertung eines Inhalts c, Wertebereich: $[0, 10]$
$R_u(c)$:	Bewertung des Inhalts c durch den Nutzer u lt. Profil
$R(a_i(j))$:	Bewertung der j-ten Ausprägung des Attributtyps i
$w(a_i)$:	Gewicht des i-ten Attributstyp für eine Empfehlung
$P(c)$:	Prognostizierter Empfehlungswert des Inhalts c, Wertebereich: $[0, 10]$
$P(a_i)$:	Prognostizierter Empfehlungswert des Attributtyps i

Anhang D.

Protokollbefehle des Frameworks zur Kontext-abhängigen Empfehlung

ID|....
: Initiiert den Identifizierungsvorgang des Clients beim Server. Existierende Nutzer müssen ihre UserId in der dieser Nachricht folgenden senden.
Antwort:
OK|.... wenn die Identifizierung vom Server akzeptiert wurde
ERR|.... sonst.

<UserId>|....
: Identifizierung des Nutzers mit Identifikator <UserId>.
Antwort:
CONFIG|.... wenn die Identifizierung erfolgreich war
ERR|.... sonst.

CONFIG|....
: Senden des Konfigurationsbefehl an den Client als Antwort auf eine erfolgreiche Identifzierung eines Nutzers.
Antwort:
OK|.... wenn der Client bereit ist, die Konfiguration des `LocalPersonalizer` zu starten
ERR|.... sonst.

THR|<threshold>....
: Teil der Clientkonfiguration, enthält den globalen Schwellwert, der für das Framework verwendet werden soll.
Antwort:
OK|.... wenn der Client den Schwellwert korrekt empfangen hat.
ERR|.... sonst.

AGG|<aggregatorId>....
: Teil der Clientkonfiguration, enthält den Identifikator <aggregatorId> der Aggregatorfunktion, die verwendet werden soll. 0 ist die <aggregatorId> für die

gewichtete Summe, 1 für die Maximumfunktion, -1 für die Minimumfunktion
Antwort:
OK|.... wenn der Client den Identifikator der Aggregatorfunktion korrekt empfangen hat.
ERR|.... sonst.

MATCH|<numberOfMatcher>....
Teil der Clientkonfiguration, enthält die Anzahl <numberOfMatcher> der auf dem Client zu aktivierenden Matcherkomponenten.
Antwort:
OK|.... wenn der Client die Anzahl korrekt empfangen hat.
ERR|.... sonst.

<MatcherClassName>|<weight>....
Teil der Clientkonfiguration, enthält den Klassennamen <MatcherClassName> der auf dem Client zu aktivierenden Matcherkomponenten und das zu verwendende Gewicht <weight>.
Antwort:
OK|.... wenn der Client Klassenname und Gewicht korrekt empfangen hat und den zugehörigen Matcher aktivieren konnte.
ERR|.... sonst.

REQUEST|....
Client fordert eine Liste mit passenden Metadaten von Inhalten beim Server an. Der Server erzeugt diese Liste und sendet sie an den Client.
Antwort:
RES|<number>.... gefolgt von <number> mal <Filesize>|<XMLFileAsString>.... siehe unten

RES|<number>....
Enthält die Anzahl <number> an Metadaten-Dateien, die als passend bewertet wurden. Diese werden nachfolgend mit <number> Nachrichten versendet.

<FileSize>|<XMLFileAsString>....
Enthält die Metadaten-Datei <XMLFileAsString> als String der Länge <FileSize>.
Antwort:
OK|.... wenn der Client die Datei korrekt empfangen hat.
ERR|.... sonst.

BYE|....
Enthält den Verbindungsaubbauwunsch des Clients.
Antwort:
OK|.... wenn der Server bereit ist, die Verbindung abzubauen.
ERR|.... sonst.

Abbildungsverzeichnis

Literaturverzeichnis

[1] F. Abel, M. Frank, N. Henze, D. Krause, D. Plappert, P. Siehndel. GroupMe! - Where Semantic Web meets Web 2.0. In *Proc. of 6th International Semantic Web Conference*, S. 871–878, Busan, 2007. Springer.

[2] B. Adrian, L. Sauermann, T. Roth-Berghofer. ConTag: a Semantic Tag Recommendation System. In *Proc. of the International Conference on Semantic Systems*, S. 297–304, Graz, 2007. JUCS.

[3] A. Almeida, B. Sotomayor, J. Abaitua, D. L. de Ipiña. folk2onto: Bridging the Gap Between Social Tags and Ontologies. In *Proc. of the 1st International Workshop on Knowledge Reuse and Reengineering over the Semantic Web*, Teneriffa, 2008.

[4] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In *Proc. of the 6th International Semantic Web Conference*, S. 722–735, Busan, 2007. Springer.

[5] R. Baeza-Yates, B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley, 1999.

[6] M. Balabanović, Y. Shoham. Fab: Content-based, Collaborative Recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[7] J. Baldzer, S. Boll, P. Klante, J. Krösche, J. Meyer, N. Rump, A. Scherp. Location-Aware Mobile Multimedia Applications on the Niccimon Platform. In *Proc. of the 2nd Symposium für Informationssysteme für mobile Anwendungen*, S. 318–334, Brunswick, 2004.

[8] W.-T. Balke, J. Gonzalez-Pinto, Q. Wang, W. Kießling. P-News: A Personalized Notification Service for MPEG-7 Libraries. Technical report, Institut für Informatik, Universität Augsburg, 2003.

[9] N. J. Belkin, W. B. Croft. Information Filtering and Information Retrieval: Two Sides of the Same Coin? *Communications of the ACM*, 35(12):29–38, 1992.

[10] R. E. Bellman. *Dynamic Programming.* Dover Publications, 2003.

[11] A. Berglund, S. Boag, D. Chamberlin, M. F. Fernández et al.(Hrsg.). XML Path Language (XPath) 2.0. W3C recommendation, W3C, 2007.

[12] K. Bergmann. *Personalisierung im Geschichtsunterricht - Erziehung zur Demokratie?* Klett, 1977.

[13] T. Berners-Lee. Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as Used in the World-Wide Web. RFC 1630 (Informational), 1994.

[14] T. Berners-Lee, R. Fielding, L. Masinter. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986 (Standard), 2005.

[15] P. V. Biron, A. Malhotra (Hrsg.). XML Schema Part 2: Datatypes Second Edition. W3C recommendation, W3C, 2004.

[16] Y. Blanco-Fernández, J. Arias, A. Solla, M. Cabrer, M. Nores. AVATAR: Modeling Users by Dynamic Ontologies in a TV Recommender System based on Semantic Reasoning. In *Proc. of 3rd European Conference on Interactive Television: User Centred ITV Systems, Programmes and Applications*, S. 173–182, 2005.

[17] Y. Blanco Fernández, J. J. Pazos Arias, A. Gil Solla, M. Ramos Cabrer. AVATAR: a Flexible Approach to Improve the Personalized TV by Semantic Inference. In *Proc. of the 1st Workshop on Web Personalization, Recommender Systems and Intelligent User Interfaces*, S. 76–85, Reading, 2005.

[18] Y. Blanco-Fernández, J. J. P. Arias, A. Gil-Solla, M. R. Cabrer, M. L. Nores. *A Hybrid Strategy to Personalize the Digital Television by Semantic Inference*, chapter in: Interactive Digital Television: Technologies and Applications, S. 33 – 51. Idea Group Inc., 2008.

[19] Y. Blanco-Fernández, J. J. P. Arias, A. Gil-Solla, M. R. Cabrer, M. L. Nores et al. AVATAR: Enhancing the Personalized Television by Semantic Inference. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(2):397–421, 2007.

[20] S. Boll. MM4U - A Framework for Creating Personalized Multimedia Content. In *Proc. of the 9th International Conference on Distributed Multimedia Systems*, S. 12–16, Miami, 2003.

[21] T. Bray, D. Hollander, A. Layman, R. Tobin (Hrsg.). Namespaces in XML 1.0 (Second Edition). W3C recommendation, W3C, 2006.

[22] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau (Hrsg.). Extensible Markup Language (XML) 1.0. W3C recommendation, W3C, 2006.

[23] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau, J. Cowan (Hrsg.). Extensible Markup Language (XML) 1.1. W3C recommendation, W3C, 2006.

[24] D. Brickley, R. Guha (Hrsg.). RDF Vocabulary Description Language 1.0: RDF Schema. W3C recommendation, W3C, 2004.

[25] P. J. Brown. The Stick-e Document: a Framework for Creating Context-aware Applications. In *Proc. of the Conference on Electronic Publishing and Document Manipulation*, Palo Alto, 1996.

[26] D. Bulterman, J. Jansen, P. Cesar, S. Mullender, E. Hyche, M. DeMeglio et al. (Hrsg.). Synchronized Multimedia Integration Language (SMIL 3.0). W3C recommendation, W3C, 2008.

[27] D. Bulterman, J. Jansen, K. Kleanthous, K. Blom, D. Benden. Ambulant: A Fast, Multi-Platform Open Source SMIL Player. In *Proc. of the 12th International Conference on Multimedia*, S. 492–495, New York, 2004. ACM.

[28] A. Chen. Context-Aware Collaborative Filtering System: Predicting the User's Preference in the Ubiquitous Computing Environment. In *Proc. of the International Workshop on Location- and Context-Awareness*, S. 244–253, München, 2005. Springer.

[29] H. Chen. *An Intelligent Broker Architecture for Context-Aware Systems.* Doktorarbeit, University of Maryland Baltimore County, 2003.

[30] H. Chen, T. Finin, A. Joshi. An Ontology for Context-Aware Pervasive Computing Environments. *Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review*, 18(3):197–207, 2004.

[31] K. Cheverst, N. Davies, K. Mitchell, A. Friday. Experiences of Developing and Deploying a Context-aware Tourist Guide: the GUIDE Project. In *Proc. of the 6th International Conference on Mobile Computing and Networking*, S. 20–31, New York, 2000. ACM.

[32] K. Cheverst, N. Davies, K. Mitchell, A. Friday, C. Efstratiou. Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, S. 17–24, New York, 2000. ACM.

[33] K. Cheverst, K. Mitchell, N. Davies. Design of an object model for a context sensitive tourist GUIDE. *Computers and Graphics*, 23(6):883–891, 1999.

[34] ChoiceStream. 2006 ChoiceStream Personalization Survey, 2006.

[35] ChoiceStream. 2008 ChoiceStream Personalization Survey, 2008.

[36] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, M. Sartin. Combining Content-Based and Collaborative Filters in an Online Newspaper. In *Proc. of ACM SIGIR Workshop on Recommender Systems*, Berkeley, 1999. ACM.

[37] comScore, Inc. Number of Online Videos Viewed in the U.S. Jumps 13 Percent in March to 11.5 Billion. www.comscore.com/Press_Events/Press_Releases/2008/05/US_Online_Video_Usage, Mai 2008. zuletzt zugegriffen im 15.06.2009.

[38] J. L. Crowley, J. Coutaz, G. Rey, P. Reignier. Perceptual Components for Context Aware Computing. In *Proc. of the International Conference on Ubiquitous Computing*, S. 117–134, Goteborg, 2002. Springer.

[39] CWI (Centrum Wiskunde & Informatica), Netherlands. *AMBULANT SMIL Player*, 2008. www.cwi.nl/projects/Ambulant, zuletzt zugegriffen am 15.06.2009.

[40] Department of Information Technology (ITEC), Universität Klagenfurt. Information Technology — MPEG-21 Multimedia Framework, 2005. mpeg-21.itec.uni-klu.ac.at/cocoon/mpeg21, zuletzt zugegriffen am 15.06.2009.

[41] S. DeRose, E. Maler, D. Orchard (Hrsg.). XML Linking Language (XLink) Version 1.0. W3C recommendation, W3C, 2001.

[42] A. Dey. Context-Aware Computing: the CyberDesk Project. In *Proc. of the Spring Symposium on Intelligent Environments*, S. 51–54, 1998.

[43] A. Dey. *Providing Architectural Support for Building Context-Aware Applications.* Doktorarbeit, Georgia Institute of Technology, November 2000.

[44] A. Dey. Understanding and Using Context. *Personal Ubiquitous Computing*, 5(1):4–7, 2001.

[45] A. Dey, G. Abowd. Towards a Better Understanding of Context and Context-Awareness. In *Proc. of the Workshop on the What, Who, Where, When, Why and How of Context-Awareness*, 2000.

[46] M. Duerst, M. Suignard. Internationalized Resource Identifiers (IRIs). RFC 3987 (Proposed Standard), 2005.

[47] DVB Project Office. EN 300 468: Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems, 1998.

[48] A. Erk, C. Kunert. Benutzer-freundliches Auffinden und Aufzeichnen von TV-Programmen. Jahresbericht, Institut für Rundfunktechnik, 2005.

[49] S. Pemberton, M. Altheim, D. Austin et al. (Hrsg.). XHTML^{TM} 1.0 The Extensible HyperText Markup Language (Second Edition). W3C recommendation, W3C, 2002.

[50] ETSI. TS 102 822-7: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime Phase 1"); Part 7: Bi-directional Metadata Delivery Protection, 2003.

[51] ETSI. TS 102 822-5: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime Phase 1"); Part 5: Rights Management and Protection (RMP) Information for Broadcast Applications, 2005.

[52] ETSI. TS 102 822-1: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime"); Part 1: Benchmark Features, 2006.

[53] ETSI. TS 102 822-5-2: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime"); Part 5: Rights Management and Protection (RMP) Sub-part 2: RMPI binding, 2006.

[54] ETSI. TS 102 822-2: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime"); Part 2: Phase 1 - System Sescription, 2007.

[55] ETSI. TS 102 822-4: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime"); Part 4: Phase 1 - Content Referencing, 2007.

[56] ETSI. TS 102 822-8: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime"); Part 8: Phase 2 - Interchange Data Format, 2007.

[57] ETSI. TS 102 822-9: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime"); Part 9: Phase 2 - Remote Programming, 2007.

[58] ETSI. TS 102 822-5-1: Broadcast and On-line Services: Search, Select, Rightful Use of Content on Personal Storage Systems ("TV-Anytime"); Part 5: Rights Management and Protection (RMP); Sub-part 1: Information for Broadcast Applications, 2008.

[59] European Broadcasting Union. ETR 211: Guidelines on Implementation and Usage of Service Information (SI), 1997.

[60] B. A. Farshchian, J. Zoric, L. Mehrmann, A. Cawsey et al. Developing Pervasive Services for Future Telecommunication Networks. In *Proc of the IADIS International Conference WWW/Internet*, S. 977–982, Madrid, 2004. IADIS.

[61] J. Ferraiolo, D. Jackson et al. (Hrsg.). Scalable Vector Graphics (SVG) 1.1 Specification. W3C recommendation, W3C, 2003.

[62] J. Ferraiolo, D. Jackson et al. (Hrsg.). Dublin Core Metadata Element Set, Version 1.1. Dublin Core Metadata Initiative recommendation, Dublin Core Metadata Initiative, 2008.

[63] S. Fickas, G. Kortuem, Z. Segall. Software Organization for Dynamic and Adaptable Wearable Systems. In *Proc. of the First International Symposium on Wearable Computers*, S. 13–14, 1997.

[64] H. Föllscher, J. Baldzer, S. Kopetzki, J. Krösche, J. Meyer et al. Der Mobile Persönliche Kommunikationsassistent: Ein Technologiedemonstrator des Niccimon-Projekts. In *Proc. of the Symposium für Informationssysteme für mobile Anwendungen*, S. 75–89, Brunswick, 2002.

[65] D. Freitag. Machine Learning for Information Extraction in Informal Domains. *Machine Learning*, 39(2-3):169–202, 2000.

[66] E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Design Patterns: Elements of Reusable Object-oriented Software.* Addison-Wesley, 1995.

[67] J. F. Gantz, D. Reinsel, C. Chute, W. Schlichting, J. Mcarthur, S. Minton, et al. The Expanding Digital Universe: A Forecast of Worldwide Information Growth Through 2010. IDC White Paper Report, 2007.

[68] A. Ghoshal, P. Ircing, S. Khudanpur. Hidden Markov Models for Automatic Annotation and Content-based Retrieval of Images and Video. In *Proc. of the 28th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 544–551, New York, 2005. ACM.

[69] K. Goldberg, T. Roeder, D. Gupta, C. Perkins. Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[70] E. Gomez, F. Gouyon, P. Herrera, X. Amatriain. Using and Enhancing the Current MPEG-7 Standard for a Music Content Processing Tool. In *Proc. of the 114th Audio Engineering Society Convention*, 2003.

[71] P. Grosso, E. Maler, J. Marsh, N. Walsh (Hrsg.). XPointer Framework. W3C recommendation, W3C, 2007.

[72] M. T. Hansen, N. Nohria, T. Tierney. Wie managen Sie das Wissen in Ihrem Unternehmen? *Harvard Business Manager*, 5:85–96, 1998.

[73] J. A. Harmer. Mobile Multimedia Services. *BT Technology Journal*, 21(3):169–180, 2003.

[74] M. F. Harper, X. Li, Y. Chen, J. A. Konstan. An Economic Model of User Rating in an Online Recommender System. In *Proc. of the 10th International Conference on User Modeling*, S. 307–316. Springer, 2005.

[75] T. Hauser, A. Kappler, J. Pohl. Tag-Clouds aus Design- und Usability-Sicht - Wortwolken am Webhimmel. *Internet Professionell*, 2, 2007.

[76] M. P. Haustein. Ähnlichkeitssuche in objekt-relationalen Datenbanksystemen. Masterarbeit, Technische Universität Kaiserslautern, 2002.

[77] P. A. Henning. *Taschenbuch Multimedia.* Carl Hanser Verlag, München, Wien, 2001.

[78] J. Herlocker, J. A. Konstan, J. Riedl. An Empirical Analysis of Design Choices in Neighborhood-Based Collaborative Filtering Algorithms. *Information Retrieval*, 5(4):287–310, 2002.

[79] J. L. Herlocker, J. A. Konstan, A. Borchers, J. Riedl. An Algorithmic Framework for Performing Collaborative Filtering. In *Proc. of the 22nd annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 230–237, Berkeley, 1999. ACM.

[80] A. Hotho, R. Jäschke, C. Schmitz, G. Stumme. BibSonomy: A Social Bookmark and Publication Sharing System. In *Proc. of the Conceptual Structures Tool Interoperability Workshop at the 14th International Conference on Conceptual Structures*, Aalborg, 2006.

[81] T.-H. Hwang, K.-H. Choi, I.-H. Joo, J.-H. Lee. MPEG-7 Metadata for Video-based GIS Applications. In *Proc of the Geoscience and Remote Sensing Symposium*, 2003.

[82] INRIA Rhône-Alpes: Project WAM (Web, Adaptation and Multimedia), France. *PocketSMIL 2.0*, 2008. wam.inrialpes.fr, zuletzt zugegriffen am 15.06.2009.

[83] International Organisation for Standardisation. ISO/IEC 15938-5:2003: Information Technology — Multimedia Content Description Interface — Part 5: Multimedia Description Schemes, 2003.

[84] International Organisation for Standardisation. ISO/IEC JTC 1/SC 29/WG 11/N5845: Information Technology — Multimedia Framework — Part 7: Digital Item Adaptation, 2003.

[85] International Organisation for Standardisation. ISO/IEC TR 21000-1:2004(E): Information Technology — Multimedia Framework (MPEG-21) — Part 1: Vision, Technologies and Strategy, 2004.

[86] International Organisation for Standardisation: José M. Martínez (Hrsg.). ISO/IEC JTC1/SC29/WG11 N5873: MPEG-21 Requirements v 1.5, 2003.

[87] International Organisation for Standardisation: José M. Martínez (Hrsg.). ISO/IEC JTC1/SC29/WG11N6828: MPEG-7 Overview (version 10), 2004.

[88] ISO/IEC MPEG Group. www.chiariglione.org/mpeg, zuletzt zugegriffen am 15.06.2009.

[89] H. Isozaki, H. Kazawa. Efficient Support Vector Classifiers for Named Entity Recognition. In *Proc. of the 19th International Conference on Computational Linguistics*, S. 1–7, Morristown, 2002. Association for Computational Linguistics.

[90] L. Issing, P. Klimsa. *Information und Lernen mit Multimedia und Internet.* Verlagsgruppe Beltz, Weinheim, 2002.

[91] D. Jannach, K. Leopold, H. Hellwagner. An Extensible Framework for Knowledge-Based Multimedia Adaptation. In *Proc. of the 17th International Conference on Innovations in Applied Artificial Intelligence*, S. 144–153, Ottawa, 2004. Springer.

[92] D. Jannach, K. Leopold, C. Timmerer, H. Hellwagner. A Knowledge Based Approach for Multi-Step Media Adaptation. In *Proc. of the 5th International Workshop on Image Analysis for Multimedia Interactive Services*, Lisabon, 2004.

[93] D. Jannach, K. Leopold, C. Timmerer, H. Hellwagner. A Knowledge-Based Framework for Multimedia Adaptation. *Applied Intelligence*, 24(2):109–125, 2006.

[94] Japan Electronics and Information Technology Industries Association. Exchangeable Image File Format for Digital Still Cameras: Exif Version 2.2, 2002.

[95] L. M. Jeon, V. Lavrenko, R. Manmatha, J. Jeon. A Model for Learning the Semantics of Pictures. In *Advances in Neural Information Processing Systems.* MIT Press, 2003.

[96] W. Kießling. Foundations of Preferences in Database Systems. In *Proc. of the 28th international conference on Very Large Data Bases*, S. 311–322. VLDB Endowment, 2002.

[97] P. Klante, J. Krösche, S. Boll. AccesSights – A Multimodal Location-Aware Mobile Tourist Information System. In *Proc. of the 9th International Conference on Computers Helping People with Special Needs*, S. 287–294, Paris, 2004. Springer.

[98] G. Klyne, F. Reynolds, C. Woodrow, et al. (Hrsg.). Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0. W3C recommendation, W3C, 2004.

[99] A. Kobsa, J. Koenemann, W. Pohl. Personalised Hypermedia Presentation Techniques for Improving Online Customer Relationships. *Knowledge Engineering Review*, 16(2):111–155, 2001.

[100] H. Kosch, J. Heuer. MPEG-7. Gesellschaft für Informatik - Informatiklexikon, 2005. www.gi-ev.de/service/informatiklexikon, zuletzt zugegriffen am 15.06.2009.

[101] J. Krösche, S. Boll, J. Baldzer. MobiDENK - Mobile Multimedia in Monument Conservation. *IEEE MultiMedia*, 10:72–77, 2004.

[102] O. B. Kwon. "I Know What You Need to Buy": Context-aware Multimedia-based Recommendation System. *Expert Systems with Applications*, 25(3):387–400, 2003.

[103] M. Langheinrich, L. Cranor, M. Marchiori. APPEL: A P3P Preference Exchange Language. W3C Working Draft, 2002.

[104] G. Lavee, L. Khan, B. Thuraisingham. A Framework for a Video Analysis Tool for Suspicious Event Detection. *Multimedia Tools Applications*, 35(1):109–123, 2007.

[105] V. Lavrenko, S. L. Feng, R. Manmatha. Statistical Models for Automatic Video Annotation and Retrieval. In *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2004.

[106] H. Lee, H. Lee, J. Nam, B. Bae, M. Kim, K. Kang, J. Kim. Personalized Contents Guide and Browsing Based on User Preference. In *Proc. of the AH2002 Workshop on Personalization in Future TV*, 2002.

[107] Y. Lee, S. Oh, W. Woo. A Context-Based Storytelling with a Responsive Multimedia System (RMS). In *Proc. of the International Conference on Virtual Storytelling*, S. 12–21, 2005.

[108] H. Lei, D. Sow, J. Davis, G. Banavar, M. Ebling. The Design and Applications of a Context Service. *SIGMOBILE Mobile Computing Communication Review*, 6(4):45–55, 2002.

[109] Y. Li, K. Bontcheva, H. Cunningham. SVM Based Learning System For Information Extraction. In *Proc. of the Sheffield Machine Learning Workshop*, Sheffield, 2005. Springer.

[110] C. Linnhoff-Popien. *CORBA: Kommunikation und Management.* Springer, 1998.

[111] D. Liu, K. Hua, K. Vu, N. Yu. Fast Query Point Movement Techniques with Relevance Feedback for Content-Based Image Retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 99(1):700–717, March 26-31 2006.

[112] J. B. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. of the 5th Berkely Symposium on Mathematical Statistics and Probability*, S. 281 – 297. Berkely University of California Press, 1967.

[113] J. Makkonen, H. Ahonen-Myka, M. Salmenkivi. Topic Detection and Tracking with Spatio-Temporal Evidence. In *Proc. of the annual European Conference on Information Retrieval*, S. 251–265, Pisa, 2003. Springer.

[114] B. Manaris, P. Roos, P. Machado, D. Krehbiel, L. Pellicoro, J. Romero. A Corpus-based Hybrid Approach to Music Analysis and Composition. In *Proc. of the 22nd Conference on Artificial Intelligence*, Vancouver, 2007.

[115] B. S. Manjunath. *Introduction to MPEG-7, Multimedia Content Description Interface*. John Wiley and Sons, Ltd., Jun 2002.

[116] F. Manola, E. Miller (Hrsg.). RDF Primer. W3C recommendation, W3C, 2004.

[117] P. Matevz, T. Jurij, M. Marko, K. Andrej. Personal Content Recommender Based on a Hierarchical User Model for the Selection of TV Programmes. *User Modeling and User-Adapted Interaction*, 15(5):425–457, November 2005.

[118] P. Melville, R. J. Mooney, R. Nagarajan. Content-boosted Collaborative Filtering for Improved Recommendations. In *Proc. of the 18th National Conference on Artificial Intelligence*, S. 187–192, Menlo Park, 2002. American Association for Artificial Intelligence.

[119] H. Menge, E. Pertsch. *Langenscheidts Taschenwörterbuch Latein*. Langenscheidt, 39. Auflage, 1987.

[120] L. Meyer. Three Scenarios for TV in 2015. *International Journal of Digital Economics*, 62:93–108, 2006.

[121] D. Michel. Transkription, Nachbabylonisch: Schrift- und Spracherkennung heute. *iX*, 2:98–99, 2009.

[122] S. E. Middleton, N. R. Shadbolt, D. C. De Roure. Ontological User Profiling in Recommender Systems. *ACM Transactions on Information Systems*, 22(1):54–88, 2004.

[123] S. Mieth, F. Fuchs, E.-P. Stoffel, D. Weiss. Reasoning on Geo–Referenced Sensor Data in Physical Infrastructures. In *Proc. of the Workshop Semantic Web meets Geospatial Applications*, Girona, 2008.

[124] G. Mishne. AutoTag: a Collaborative Approach to Automated Tag Assignment for Weblog Posts. In *Proc. of the 15th International Conference on World Wide Web*, S. 953–954, New York, 2006. ACM.

[125] K. Mitchell. *Supporting the Development of Mobile Context-Aware Systems.* Doktorarbeit, Lancaster University, 2002.

[126] National Information Standards Organization. Understanding Metadata. Technical report, NISO Press, 2004.

[127] C. Niederée. *Personalisierung, Kooperation und Evolution in digitalen Bibliotheken.* Doktorarbeit, Technische Universität Hamburg-Harburg, 2002.

[128] M. Nilsson. ID3 Tag Version 2.4.0 - Main Structure. Informal standard, November 2000.

[129] M. Nilsson. ID3 tag version 2.4.0 - Native Frames. Informal standard, November 2000.

[130] T. Ojala, J. Korhonen, M. Aittola, M. Ollila, T. Koivumäki, J. Tähtinen, H. Karjaluoto. SmartRotuaari – Context-aware Mobile Multimedia Services. In *Proc. of the 2nd International Conference on Mobile and Ubiquitous Multimedia*, Norrkoping, 2003. ACM.

[131] T. O'Reilly. Database War Stories 3: Flickr. radar.oreilly.com/archives/2006/04/database-war-stories-3-flickr.html, 2006. zuletzt zugegriffen am 15.06.2009.

[132] J. V. Ossenbruggen, F. Nack, L. Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web, Part 1. *IEEE Multimedia*, 11:38–48, 2004.

[133] J. V. Ossenbruggen, F. Nack, L. Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web, Part 2. *IEEE Multimedia*, 12:54–63, 2005.

[134] P. Pan, C. Kastner, D. Crow, G. Davenport. M-Studio: an Authoring Application for Context-aware Multimedia. In *Proc. of the 10th ACM international conference on Multimedia*, New York, 2002. ACM.

[135] J. Parsons, P. Ralph, K. Gallagher. Using Viewing Time to Infer User Preference in Recommender Systems. In *Proc. of the National Conference on Artificial Intelligence*, 2004.

[136] P. F. Patel-Schneider, P. Hayes, I. Horrocks (Hrsg.). OWL Web Ontology Language Semantics and Abstract Syntax. W3C recommendation, W3C, 2004.

[137] A. Pigeau, G. Raschia, M. Gelgon, N. Mouaddib, R. Saint-Paul. A Fuzzy Linguistic Summarization Technique for TV Recommender Systems. In *Proc. of the IEEE International Conference of Fuzzy Systems*, S. 743–748, St-Louis, 2003.

[138] S. Pokraev, J. Koolwaaij, M. van Setten, T. Broens et al. Service Platform for Rapid Development and Deployment of Context-Aware, Mobile Applications. In *Proc. of the IEEE International Conference on Web Services*, S. 639–646, Washington DC, 2005. IEEE.

[139] E. Prudhommeaux, A. Seaborne (Hrsg.). SPARQL Query Language for RDF. W3C recommendation, W3C, 2008.

[140] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proc. of the Computer Supported Cooperative Work*, S. 175–186, North Carolina, 1994. ACM.

[141] Reuters. YouTube knackt 100-Millionen-Marke, 2006. zuletzt zugegriffen am 15.06.2009.

[142] D. Riecken. Introduction: Personalized Views of Personalization. *Communications of the ACM*, 43(8):26–28, 2000.

[143] RIF Working Group. Rule Interchange Format. W3C working group, W3C, 2009.

[144] D. Rogers, J. Hunter, D. Kosovic. The TV-Trawler Project. *International Journal of Imaging Systems and Technology*, 13(5):289–296, 2003.

[145] B. Sarwar, G. Karypis, J. Konstan, J. Riedl. Item-based Collaborative Filtering Recommendation Algorithms. In *Proc. of the International World Wide Web Conference*, S. 285–295. ACM, 2001.

[146] C. Sauter. *Dynamische Dienstkomposition in Ubiquitous Computing Umgebungen.* Doktorarbeit, Ludwig-Maximilian-Universität München, 2007.

[147] A. Scherp. *A Component Framework for Personalized Multimedia Applications.* Doktorarbeit, Carl von Ossietzky Universität Oldenburg, 2007.

[148] A. Scherp, S. Boll. Generic Support for Personalized Mobile Multimedia Tourist Applications. In *Proc. of the 12th annual ACM International Conference on Multimedia*, S. 178–179, New York, 2004. ACM.

[149] A. Scherp, S. Boll. mobileMM4U - Framework Support for Dynamic Personalized Multimedia Content on Mobile Systems. In *Proc. of the Techniques and Applications for Mobile Commerce*, S. 204–215, Essen, 2004.

[150] B. N. Schilit, M. M. Theimer. Disseminating Active Map Information to Mobile Hosts. *IEEE Network*, 8:22–32, 1994.

[151] P. Schubert. *Virtuelle Transaktionsgemeinschaften im Electronic Commerce. Management, Marketing und Soziale Umwelt.* Josef Eul Verlag, Lohmar Köln, 2000.

[152] R. Shacham, H. Schulzrinne, S. Thakolsri, W. Kellerer. Ubiquitous Device Personalization and Use: The Next Generation of IP Multimedia Communications. *ACM Transactions on Multimedia Computing, Communications & Applications*, 3(2):12, 2007.

[153] U. Shardanand, P. Maes. Social Information Filtering: Algorithms for Automating Word of Mouth. In *Proc. of the Conference on Human Factors in Computing Systems*, S. 210 – 217, 1995.

[154] G. Smith. *Tagging: People-powered Metadata for the Social Web (Voices That Matter).* New Riders Press, December 2007.

[155] B. Smyth, P. Cotter. A Personalized TV Listings Service for the Digital TV Age. *Journal of Knowledge-Based Systems*, 13(2-3):53 – 59, 2000.

[156] A. Stahl. *Learning of Knowledge-Intensive Similarity Measures in Case-Based Reasoning.* Doktorarbeit, Technische Universität Kaiserslautern, 2003.

[157] M. Steinacher. Einführung in das Content Management: Einleitung - Metadaten. PlaNet-ET Konsortium, Universität Innsbruck, 2002. e-campus.uibk.ac.at/planet-et-fix/M8/8.3.1_Metadaten/10meta_einl.html, zuletzt zugegriffen am 15.06.2009.

[158] R. Steinmetz. *Multimedia-Technologie. Grundlagen, Komponenten und Systeme.* Berlin Springer, 2000.

[159] T. Strang, C. Linnhoff-Popien, K. Frank. CoOL: A Context Ontology Language to Enable Contextual Interoperability. In *Proc. of the International Conference on Distributed Applications and Interoperable Systems*, Paris, 2003. Springer.

[160] F. Suchanek, G. Kasneci, G. Weikum. YAGO: A Core of Semantic Knowledge - Unifying WordNet and Wikipedia. In *Proc. of the 16th International World Wide Web Conference*, S. 697–706. ACM, 2007.

[161] A. Tasch, O. Brakel, C. Ihl, R. Kerl. Personalization of Public Profiles in Virtual Lifestylecommunities: The Case of jetzt.de. In *Proc. of the 2nd Interdisciplinary World Congress on Mass Customization and Personalization*, München, 2003.

[162] Technical Advisory Service for Images. Metadata Overview. Technical report, Technical Advisory Service for Images, Nov 2006.

[163] L. Terveen, W. Hill. *Beyond Recommender Systems: Helping People Help Each Other*, chapter in: HCI in the New Millennium, S. 487–509. Addison-Wesley, 2001.

[164] H. S. Thompson, D. Beech, M. Maloney, N. Mendelsohn (Hrsg.). XML Schema Part 1: Structures Second Edition. W3C recommendation, W3C, 2004.

[165] C. Timmerer, H. Hellwagner. MPEG-21 Multimedia Framework. Gesellschaft für Informatik - Informatiklexikon, 2008. www.gi-ev.de/service/informatiklexikon, zuletzt zugegriffen am 15.06.2009.

[166] S. Tsekeridou. MPEG-7 MDS-Based Application Specific Metadata Model for Personalized Multi-Service Access in a DTV Broadcast Environment. In *Proc. of the IEEE International Conference on Multimedia and Expo*, 2005.

[167] B. L. Tseng, C.-Y. Lin, J. R. Smith. Using MPEG-7 and MPEG-21 for Personalizing Video. *IEEE MultiMedia*, 11(1):42–53, 2004.

[168] A. Vakali, M.-S. Hacid, A. Elmagarmid. MPEG-7 based Description Schemes for Multi-level Video Content Classification. *Image and Vision Computing*, 22(6):367–378, 2004.

[169] P. van Beek, J. Smith, T. Ebrahimi, T. Suzuki, J. Askelof. Metadata-driven Multimedia Access. *IEEE Signal Processing Magazine*, 20(2):40– 52, 2003.

[170] M. van Setten, S. Pokraev, J. Koolwaaij. Context-Aware Recommendations in the Mobile Tourist Application COMPASS. *Adaptive Hypermedia and Adaptive Web-Based Systems*, 3137:235–244, 2004.

[171] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, F. Ciravegna. MnM: Ontology Driven Semi-automatic and Automatic Support for Semantic Markup. In *Proc. of the International Conference on Knowledge Engineering and Knowledge Management Knowledge Patterns*, S. 379–391, 2002.

[172] P. Viappiani, B. Faltings, P. Pu. The Lookahead Principle for Preference Elicitation: Experimental Results. In *Proc. of the International Conference on Flexible Query Answering Systems*, S. 378–389, 2006.

[173] E. Vielmetti. The (R)evolution of Useful Web Services. *IEEE Communication Magazine*, 37:92–94, 1999.

[174] W3C Semantic Web Activity. www.w3.org/2001/sw. zuletzt zugegriffen am 15.06.2009.

[175] P. Walmsley, D. C. Fallside (Hrsg.). XML Schema Part 0: Primer Second Edition. W3C recommendation, W3C, 2004.

[176] J. Wang, A. P. de Vries, M. J. Reinders. Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. In *Proc. of the 29th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 501–508, 2006.

[177] Q. Wang, W.-T. Balke, W. Kießling, A. Huhn. P-News: Deeply Personalized News Dissemination for MPEG-7 based Digital Libraries. In *Proc. of the 8th European Conference on Digital Libraries*, S. 256–268, 2004.

[178] Y. Wang, M. Ding, C. Zhou, T. Zhang. A Hybrid Method for Relevance Feedback in Image Retrieval Using Rough Sets and Neural Networks. *International Journal of Computational Cognition*, 3(1):78–87, March 2005.

[179] A. Ward, A. Jones, A. Hopper. A New Location Technique for the Active Office. *IEEE Personnel Communications*, 4(5):42–47, 1997.

[180] D. Weiss. Zone Services — A New Approach of Location–based Services. In *Adjunct Proc. of the 4th International Conference on Pervasive Computing*, S. 235–240, Dublin, 2006.

[181] D. Weiss. Middleware–Ansatz zur Unterstützung von personalisierter und ortsabhängiger Werbung. In *Proc. of the 4. GI/ITG KuVS Fachgespräch Ortsbezogene Anwendungen und Dienste*, S. 37–41, München, 2007.

[182] D. Weiss. Modelling Context–Aware Multimedia Community Content on Mobile Devices. In *Proc. of the OTM Workshops*, S. 834–844, Vilamoura, 2007.

[183] D. Weiss, M. Duchon, F. Fuchs, C. Linnhoff–Popien. Context–Aware Personalization for Mobile Multimedia Services. In *Proc. of the 6th International Conference on Advances in Mobile Computing & Multimedia*, S. 267–271, Linz, 2008.

[184] D. Weiss, S. Helas, J. Martens. Context–Aware Adaptation of Mobile Multimedia Presentations. In *Proc. of the 1st international Workshop on Context-aware Adaption Mechanisms for Pervasive and Ubiquitous Services*, Oslo, 2008.

[185] D. Weiss, I. Krämer, G. Treu, A. Küpper. Zone Services — An Approach for Location–based Data Collection. In *Proc. of the 3rd IEEE International Workshop on Mobile Commerce and Services*, San Francisco, 2006.

[186] D. Weiss, J. Scheuerer, M. Wenleder, A. Erk, M. Gülbahar, C. Linnhoff–Popien. A User Profile–based Personalization System for Digital Multimedia Content. In *Proc. of the 3rd ACM International Conference on Digital Interactive Media in Entertainment and Arts*, S. 281–288, Athen, 2008.

[187] T. Wilton-Jones. Extensible Rendering Architecture - Technology White Paper. Technical report, Opera Software ASA, Norway, 2005.

[188] Wireless Application Forum. WAG UAProf. W3C recommendation, OMA, 2001. www.openmobilealliance.org/tech/affiliates/wap/wap-248-uaprof-20011020-a.pdf, zuletzt zugegriffen am 15.06.2009.

[189] M. Xu, C. Xu, L. Duan, J. S. Jin, S. Luo. Audio Keywords Generation for Sports Video Analysis. *ACM Transactions on Multimedia Computing, Communications & Applications*, 4(2):1–23, 2008.

[190] S. Yu, L. Al-Jadir, S. Spaccapietra. Matching User's Semantics with Data Semantics in Location-Based Services. In *Proc. of the of 1st Workshop on Semantics in Mobile Environments*, Zypern, 2005.

[191] J. Zhang, P. Pu. A Recursive Prediction Algorithm for Collaborative Filtering Recommender Systems. In *Proc. of the ACM Recommender Systems*, Minneapolis, 2007.

[192] J. Zimmermann, K. Kurapati, A. L. Buczak, D. Schaffer, J. Martino, S. Gutta. TV Personalization System: Design of a TV Show Recommender Engine and Interface. In M. M. Liliana Ardissono, Alfred Kobsa (Hrsg.) *Personalized Digital Television: Targeting Programs to Individual Viewers*, S. 27–52. Kluwer, 2004.