Swoozy

Intelligente Interaktionen für das semantische Fernsehen

Matthieu Deru

Swoozy

Intelligente Interaktionen für das semantische Fernsehen

Matthieu Deru

Dr. Matthieu Deru
DFKI GmbH
Saarland Informatics Campus, Geb. D 3_2
D-66123 SAARBRÜCKEN

matd@mat-d.com
www.mat-d.com
twitter.com/iMatD

Dieses Buch ist eine überarbeitete Fassung der 2016 an der Naturwissenschaftlich-Technischen Fakultät I der Universität des Saarlandes vorgelegten Dissertation "*Swoozy: Intelligente und benutzerzentrierte Interaktionen für das semantische Fernsehen*"

Eine eBook Version dieses Buches kann unter folgender Webseite heruntergeladen werden:
http://www.mat-d.com/books

1. Auflage

Inhaltsverzeichnis

Einführung

Stolz kündigte mein Grundschullehrer an, dass die Schüler zum Computer gehen dürfen. Dieser Computer war an einen Fernseher angeschlossen und, wenn wir lange gewartet hatten bis eine quietschende Kassette an ihr Ende gelaufen war, bekamen wir faszinierende Grafiken in Form von pädagogischen Spielen auf dem Bildschirm angezeigt. Mit einer grünen Schildkröte und der Programmiersprache LOGO konnte man das pixelige Tier in Bewegung setzen oder sogar etwas virtuell zeichnen. Später, als ich in der Stadt zur Schule ging, besaßen sie dort einen Computerraum voller MO-5 und To7 – der ganze Stolz der französischen Firma Thomson und unserer Grundschule. Waren wir als Schüler während des Tages brav genug, durften wir sogar Grammatikübungen mit dem TO7 und seinem legendären Lichtgriffel lösen; es war eine spannendere und spielerischere Art und Weise seine Schulaufgaben zu machen und dabei sogar etwas Neues zu lernen. Eines Tages kam unser Grundschullehrer mit einem mysteriösen Karton in die Klasse. Nach dem Auspacken stellte er eine Box auf den Tisch. Diese war ganz weiß, mit einer Maus ausgestattet und hatte einen eingebauten Schwarz-Weiß-Bildschirm; es war ein Macintosh. Zum ersten Mal brauchte man kein Fernsehgerät mehr, um „mit einem Computer zu arbeiten“. In der Tat, man konnte sogar die kleine Kiste unter dem Arm transportieren, sich irgendwo anders hinstellen und damit weiterarbeiten, in den späten 80ern eine echte Sensation.

Ein Wendejahr in meiner Faszination für Computer und Fernsehen war 1991, da sich in diesem Jahr das Fernsehen und die Art zu informieren und informiert zu werden grundlegend verändert haben. Im Januar 1991 bekam die ganze Welt live zu sehen, wie im Irak Krieg geführt wurde. CNN war der Sender, den man auf jeden Fall anschauen musste; faszinierend einerseits, mit dieser live „geupdateten" Laufschrift, die jede Aktion der Armee und der Generäle zeitnah einblendete, aber auch beängstigend durch realistische und noch nie zuvor gezeigte Kriegsbilder.

Das Medium „Fernsehen" ist dadurch in die Ära des Informationsüberschusses geraten, mit ihrer starken Auswirkung auf Politik und Volksmeinung. Dies wird von Politologen als CNN-Effekt[1] bezeichnet. Später entstand durch die Emergenz von Internet und von Blogs das Pendant dazu, der sogenannte Al-Jazeera-Effekt: immer schnellere Informationen, mehr Daten aus Twitter und Blogs und immer mehr Livevideos usw.

Im Sommer 1991 während einer Reise mit meinen Eltern nach New York sah ich bei FAO Schwarz an der 5th Avenue ein Gerät von Sony mit dem man mittels eines Stiftes magisch auf dem Fernseher malen konnte. Das Gerät - in Form eines weißen Tablets - habe ich in Europa allerdings nie wieder gesehen, vielleicht auch, weil Ende 1991 eine ganz neue Welle den Edutainment- Markt überschwemmen würde, die der Spielekonsolen. Meines Erachtens nach war der „Spielekonsolenboom" der Auslöser der ersten richtigen Interaktivität vor und am Fernseher. Ab diesem Zeitpunkt konnte man wirklich „etwas Lustiges" mit dem Fernseher anstellen. Damals war es als Teenager fast Pflicht zu wissen,

[1] http://en.wikipedia.org/wiki/CNN_effect

bis zu welchem Level man mit dem blauen Igel oder dem Klempner gelangt war oder wie viele Bits (8, 16, 32, 64) eine Spielekonsole besaß. Das waren die neuen Maßstäbe der Interaktivität. Es war bunt, cool und deutlich interessanter als das normale Fernsehen mit seinen statischen „Programmen“ und den langweiligen Sendungen.

Als Reaktion auf diese neue Spielekonsole-Welle bemerkten viele Fernsehsender, dass die Einschaltquoten sanken. 1992 starteten die französischen Fernsehsender Antenne 2 (heute France 2) und FR3 (heute France 3) den Verkauf der ersten, echten, mit dem Live-Fernsehinhalt verbundenen, interaktiven Systeme *Quizako* und *Multipoints*[2][3].

Es waren zwei kleine Geräte, *Quizako*, in Form einer Fernbedienung mit vier farbigen Tasten und das *Multipoints*, das die Form eines Taschenrechners besaß. Mit beiden konnte man live und interaktiv an einer Quizsendung oder einem Gewinnspiel teilnehmen. Über die Tasten konnte man seine Antworten validieren. Wenn diese richtig waren, erklang die Marseillaise (!) - damals stand Sounddesign noch nicht so im Mittelpunkt. Während einer Sendung wurde ein Barcode über das normale Videobild bzw. Fernsehbildsignal vom Fernsehsender eingeblendet. Platzierte man den *Quizako* oder *Multipoints*[4] [5] auf diesem Barcode, konnte man seine Punkte einsammeln und direkt auf dem Gerät sehen, wie viele Punkte man gewonnen hatte. Wurden viele Punkte während einer Sendung gewonnen, konnte man sie

[2] http://www.petrif.fr/le-quizako-televisator-2-et-giga-2858849.html

[3] http://archives-lepost.huffingtonpost.fr/article/2010/03/29/2008904_television-il-y-avait-plus-d-interactivite-il-y-a-une-vingtaine-d-annees.html

[4] http://www.ina.fr/video/PUB3774429042

[5] http://www.ina.fr/video/PUB3774448040

als Gutscheine einlösen, z.B. für einen gratis Hamburger bei Quick (eine Fast Food-Kette) oder CDs im Virgin Megastore oder sogar für eine Reise. Heute fast 20 Jahre danach und ohne dabei nostalgisch klingen zu wollen, scheinen die Second Screen-Angebote d.h. Tablet-basierte Applikationen (Apps), die parallel zu einer Fernsehsendung Zusatzinformationen anbieten, eher öde. Mit diesen Apps kann man höchstens noch „virtuelle Freunde“ oder „virtuelle Sterne“ während Sendungen gewinnen, im besten Falle wird einem heutzutage durch die Möglichkeit des gelegentlichen Versendens einer SMS während der Sendung eine gewisse Interaktivität vorgegaukelt; aber eins ist sicher: so spannend wie damals, wird es wohl nie mehr sein. Im darauf folgenden Jahr 1993 wurden Videospiele populärer als je zuvor. Die Fernsehsender reagierten schnell mit einem kleinen Kobold namens „Hugo“, den sie auf die Fernsehbildschirme brachten. Wie magisch! Mittels der Tasten seines Telefons zu Hause konnte der Kandidat, der mit dem gleichen Telefon live in die Sendung zugeschaltet war, diesen Hugo in einer virtuellen Szenerie bewegen; alle anderen Zuschauer konnten darüber zwar lachen, durften aber nur zuschauen. Eine neue Art die Zuschauer zu unterhalten, wurde durch den Kandidaten mittels der Fernsteuerung von Hugo erreicht, was zugleich aus meiner Sicht einmalig und faszinierend war. Diese faszinierende Verbindung von animierten Pixeln und der Möglichkeit mit diesen zu interagieren, veranlasste mich dazu, stundenlang die Teletextseiten der Fernsehkanäle zu durchforsten. Manche dieser Seiten waren sogar animiert und erlaubten „rudimentäre Spiele“. Ähnlich zum deutschen BTX gab es in Frankreich Minitel, eine Art französisches BTX oder Internet, das noch bis 2012 (!) funktionierte. Nachdem man über sein Telefon 4 Zahlen (meistens „3615“) eingewählt hatte, konnte man Minitel mit einem Telematik-System verbinden und interaktiv auf das Telefonbuch,

Spiele, Kochrezepte, Horoskope und Nachrichten zugreifen. Fast alle Fernsehsendungen und Marken hatten ihre eigene „3615“ und man konnte dort interessante Zusatzinformationen nachschlagen. Sieht es heute etwa anders aus? Nein! Zu jeder Sendung wird statt der „3615“ eine „WWW“-Adresse eingeblendet. Das Fernsehen hatte Minitel populär gemacht und gezeigt, dass neben den damals teuren Computern, der Bürger kostengünstig (Minitel war staatlich gefördert und die Anschaffung daher kostenlos) langsam in eine neue Informationsära eintreten konnte.

Im Jahre 1995 bekam ich meinen ersten Computer mit Windows 95, aber sehr schnell merkte ich, dass im Gegensatz zu Spielekonsolen, diese sog. PCs nicht so intuitiv und interaktiv zu bedienen waren, wie es in der Werbung suggeriert wurde. Die Betriebssysteme waren zwar bunt (dank SVGA, Super Video Graphics Array, das die Bildauflösung und Farbanzahl für Computergrafiken bzw. Bildschirmauflösungen bestimmte), man konnte auch überall hinklicken und es passierte meistens etwas; aber manchmal auch nicht. Allzu oft landete man als „Quittung “ für einen zu sorglosen Umgang mit dem Gerät auf einem blauen Bildschirm, der für einen Totalabsturz des PCs stand. Bei den Spielekonsolen kam das nicht vor.

Aus meiner Sicht verlief der Einstieg in die PC-Ära für den „Jedermann“ holprig, denn Computer blieben etwas Mysteriöses, weitgehend unausgereift und unintuitiv zu bedienen. Damals besaß fast jeder von meinen Freunden eine Spielekonsole, aber Computer waren „Geräte“ für Leute in Büros, um damit komplexe Berechnungen durchzuführen oder für einen „Daniel Düsentrieb“, der Spaß dran hatte, diesen auseinanderzubauen und kuriose Zusatzkomponenten namens „RAM“ oder „Motherboard“ einzubauen.

Wenn ein Wort die späten 90er geprägt hat, war es das Wort „Multimedia“. Es tauchte überall auf. Man konnte CD-ROMs (die die veralteten Disketten als Datenträger ersetzten) mit genügend Speicherkapazität erwerben und davon ansprechende Animationen mit Ton abrufen. Auch das Abspielen, Stoppen und Wiederholen von Videos war problemlos möglich. Dazu passend konnte man sogar hochwertige Lautsprecher und Mikrofone für den Computer erwerben. Erstmals konnte man seine Stimme mittels eines Computers aufnehmen. Eine Software war in der Lage zu bestimmen, ob man eine gute Aussprache in einer Fremdsprache hatte. Mit Softwares, wie Cubasis oder Cakewalk, war es möglich, Musik über den Computer zu komponieren und mittels des MIDI-Formats zu Keyboards zu übertragen. Parallel boomten die Angebote im Fernseh- und Entertainment-Bereich: jeder begann eifrig eine Satellitenschüssel auf seinem Dach zu installieren, um Neuigkeiten aus der ganzen Welt zu empfangen, allerdings erst nach stundenlanger Installation, Durchforsten komplexester Menüs und „Herumärgern“ mit den korrekten Einstellungen für den Receiver. Damals waren Begriffe wie „First User Experience“ oder „Intuitive Bedienung“ Fremdwörter; man war froh, wenn der Empfang funktionierte. Ähnlich unangenehme Erlebnisse hatte man bei der Benutzung der ersten PCs. Daraus entstand womöglich das volksmündliche und weitverbreitete „Never Touch a Running System!“-Sprichwort, das bis heute gerne benutzt wird, um den vermeintlich folgenschweren Veränderungen an einem stabilen Systemzustand vorzubeugen. Mittels des Satellitenempfangs war es möglich, von MTV auf CNN und zurück auf Eurosport zu „zappen“. Man konnte nahezu alles erfahren und erlernen: mit dem Telekolleg neue Sprachen, mit Wolfgang Back Neues aus der Computerwelt, mit MTV die angesagte Rap-, Rock- und Popmusik und auf dem neuen kommerziellen Kanal Premiere war es sogar

möglich, die Kameraperspektiven während eines Formel-Eins-Rennen frei mit seiner Fernbedienung auszuwählen. Durch das Satellitenfernsehen wurde die Tür zu einer vorerst kostenlosen, neuen und zugleich faszinierenden Medien- und Wissenswelt weit geöffnet.

Eines Tages, Ende 1996, fiel mir eine Diskette mit den Buchstaben AOL bzw. der Beschriftung America Online in die Hände. Damit und mit einem Gerät namens „Modem“ - das genauso komische Geräusche wie der Minitel machte - konnte man angeblich auf eine neue Art mit Leuten und Rechnern der ganzen Welt in Kontakt treten: das Internet war das neue „Highlight“. Der Besitz einer eigenen Email-Adresse war „in“, fast ein Statussymbol.

Parallel zu diesen Entwicklungen wurden erste intelligente Fernsehgeräte, nicht zu vergleichen mit den Smart-TVs von heute, auf den Markt gebracht. In den USA wurde sogar im Zuge des „Furby"[6] (ein intelligentes Plüschtier, welches voll mit Sensoren ausgestattet war), der erste fernsehgekoppelte Dinosaurier namens Barney[7] von Microsoft, verkauft. Dieser „hüpfte“ während einer Fernsehsendung und kommentierte was lief – zumindest während spezieller Kindersendungen. Aus heutiger Sicht ist vielleicht dieser Barney das erste Fossil des sich damals noch nicht abzeichnenden „Internet of Things“. In den frühen 2000ern wurde das Internet immer präsenter und in den USA tauchte die erste Mischform zwischen Internet und Fernsehen mit dem sehr stark von Microsoft „gepushten“ *WebTV*-System auf. Dies kann man als ersten richtigen Schritt in Richtung interaktives TV bezeichnen. In der Tat konnte man damit fernsehen, ins Internet gehen, Mails abrufen

6 http://www.furby.de
7 http://barney.wikia.com/wiki/ActiMates

und all das ganz einfach vom Sofa aus. Diese Innovation wurde bald durch die sog. Mediacenter abgelöst, die eine teilweise doch „recht unausgegorene" Zwischenform von PC, Spielekonsole und Recordern waren. Man spürte regelrecht den Wunsch der amerikanischen Computerindustrie in die Wohnzimmer vorzudringen, eben in den Bereich in dem bereits mächtige Firmen aus Japan, wie Sega, Nintendo und später Sony, Jahre zuvor erfolgreich einen regelrechten Hype mit ihren Spielekonsolen ausgelöst hatten. Es fehlte an cleveren Bedienkonzepten und meistens waren auf diesen Mediacentern-Geräten nur abgespeckte Versionen von Betriebssystemen anderer Plattformen installiert, mit all den Problemen, die eine solche Portierung mit sich brachte.

Eine ganz neue Erfolgsgeschichte begann in Cupertino (Kalifornien) mit einem Handy und dem magischen Wort „App", das Generationen prägen würde. Man sprach nicht mehr von *Frontends*, *KIT-Systemen* oder *Executables*. Auf einmal war Softwareprogrammierung cool und im Trend; manch einer wurde dabei über Nacht dank katapultierten und wütenden Vögeln zum Millionär (Angry-Birds). Gekoppelt mit der gerade entstehenden völlig neuen Macht neuer Akteure, wie z.B. Google, konnten diese Apps gemäß dem Motto „für alles gibt es eine App" angeblich alle Probleme der Menschen lösen. Das dachten auch die Hersteller von Fernsehern und übernahmen prompt das App-Konzept: eine neue Ära von interaktiven Fernsehern war geboren. Leider geriet mittlerweile der Fernseher, das ehemalige Hauptgerät zum Medien- und Informationskonsum, aus dem Fokus. Es wurde zu einem Gerät zwischen PC, Tablet und Handy, das „irgendwas" anzeigt und fast schon vergessen monolithisch im Wohnzimmer thront und im Hintergrund Töne von sich gibt. Ursache dafür ist die Art und Weise, wie

Menschen heute Medien konsumieren, die sich durch das Internet und die bereits erwähnten Apps mittels mobiler Endgeräte rasch und komplett verändert haben. Man „geht“ auf Vine oder YouTube und wählt dort seine „Channels“.
Parallel dazu überprüft man was Freunde bei Facebook hinterlassen haben. Das eigentliche Live-Fernsehprogramm wird nur noch mit einem Auge betrachtet. Langweilige Filmszenen oder Reportagen werden mittels eines schnellen Twitter-Updates oder einer Wikipedia-Suche zeitlich überbrückt.

Fernsehen und Internet als Medien für Informationen unterscheiden sich grundlegend. Das Fernsehen ist ein kanalisiertes halb-interaktives Wissensmedium, d.h. es kann nur auf Videoinhalte zugegriffen werden, die tatsächlich ausgestrahlt und vom Fernsehsender vorbereitet werden. Das Internet dagegen ist durch seine Supra-(Inter) Reaktivität bzw. Schnelligkeit und Vielfalt an Multimediaangeboten in der Lage, einen ungefilterten oft zu groben Informationsüberschuss zu liefern. Beide Medien sind komplementär und lassen sich wunderbar zusammenfügen. Die Art und Weise, welche Internet- und/oder Datendienste sinnvoll mit der Aktivität „Fernsehen“ verknüpft werden sollen und wie zukünftig mit diesen Informationen interagiert wird, wird die neue Herausforderung der kommenden Jahre sein.

Was soll nun diese „kleine Lebensgeschichte“ zeigen?

1. Die Veränderungen in den Bereichen Computer, Fernsehen und Informationssystemen waren schon immer eng miteinander verknüpft.
2. Interaktivität ist ein zentrales Konzept, das genauso mit dem Fernsehen wie mit den Computersystemen funktionieren sollte.
3. Wünschenswert aus Benutzersicht wäre, dass alles einfach und schnell funktioniert; analog dem App-Gedanken: für jede Frage sollte eine App existieren, die eine Antwort anbietet.

Jahrelang haben die drei Bereiche gegenseitig von den Innovationen in puncto Interaktivität, Multimedia und Internet profitiert. Heute geschieht eine langsame technische Konvergenz beider Bereiche in Form von Systemen (Set-Top-Boxen, Konsolen oder Smart-TVs), die den inneren Wunsch der Zuschauer, immer mehr zu erfahren und unterhalten zu werden, realisieren können.

Mit diesem Werk und durch die Realisierung von Swoozy bzw. die technische Konkretisierung des semantischen Fernsehens, möchte ich der Vision vom Fernsehen als „intuitiven, interaktiven und spielerischen Zugang zum Wissen“ neue Impulse geben.

Warum dieses Buch ?

Wenn man von interaktiven Systemen spricht, denkt man sehr schnell an Computer, Wearables, Tablets oder Spielekonsolen, die selbstverständlich mit dem Internet verbunden sind. Paradoxerweise, wie im letzten Abschnitt skizziert, war eigentlich das Fernsehen das erste Medium, das Interaktionen mit Multimediainhalten für jedermann ermöglichte, lange bevor es das Internet, wie wir es in seiner heutigen Form kennen, gab. Leider war auch das Fernsehen ein Medium, welches sehr spät von App-Konzepten und Internetbrowsern profitierte. Versuche, wie z.B. Internetbrowser mit dem Fernsehgerät zu koppeln, blieben sehr oft nur Insellösungen.

Erst die sogenannten Smart-TVs mit den TV-Apps und der Versuch seitens der Fernsehsender, insbesondere über HbbTV, eine gewisse Interaktivität über die Fernsehgeräte zu erzielen, waren erste Anzeichen, dass der Aufruf interaktiver und sendungspezifischer Informationen technisch realisierbar war.

Jedoch sind diese Möglichkeiten interaktionstechnisch sehr begrenzt und werden von dem nicht IT-affinen Benutzer oft erst gar nicht benutzt.

Durch Konkurrenz insbesondere durch Dienste, wie Netflix, verändert sich zusätzlich auch der Zugang zu den Sendungen und auch die Interaktivität: der Zuschauer bekommt lineares und nicht-lineares Fernsehen auf einen Schlag angeboten und muss sich damit zurechtfinden.

Die interaktiven und die intuitiven Bedienungskonzepte bleiben oft auf der Strecke, zur Frustration der Zuschauer bzw. Fernsehgerätebenutzer.

Basierend auf diesen Erkenntnissen und getrieben durch mein persön-

liches Interesse für das Fernsehen und die Fernsehproduktion, stellte ich mir die Frage, ob es technisch möglich wäre, ein neues interaktives System zu entwickeln, dass parallel zur Anzeige des Fernsehbildes deutlich mehr Informationen zur ausgestrahlten Sendung einblenden könnte als herkömmliche TV-Apps.
Natürlich sollten diese Informationen in Echtzeit und im richtigen Kontext jederzeit vom Zuschauer aufrufbar gemacht werden.
Daraus entstand die Idee das System namens *Swoozy* zu entwickeln und somit drei Welten zu vereinigen: die multimodale Interaktion über eine dedizierte Benutzerschnittstelle, die spannende Welt der (aus meiner Sicht zur Zeit immer noch unterschätzten) semantischen Wissensdatenbanken (Web 3.0) und letztendlich die intelligente Extraktion von Semantik aus Bildfolgen.
Dieses Buch und die technische Umsetzung von *Swoozy*, so meine Hoffnung, sollen beweisen, dass nur eine geschickte Kombination der parallelen Benutzung der Medien Fernsehen und Internet bzw. des semantischen Web, mit dem Fokus auf einer Suche und benutzerzentrierten Interaktionen, die intuitiv vorm Fernsehgerät durchgeführt werden können, eine Lösung für das zukünftige interaktive Fernsehen bilden kann. Parallel dazu vermittelt dieses Buch auch technische Informationen über Ansätze der künstlichen Intelligenz, die dazu beitragen können, eine präzisere Extraktion von Semantik aus Sendungsinhalten, kontextorientiert und benutzerzentriert, durchzuführen.
Zusätzlich soll das Buch den Lesern detaillierte Informationen über folgende Herausforderungen, die während der Realisierung des semantischen Fernsehens im Vordergrund stehen, geben:

- Welche Möglichkeiten bieten heutige Smart-TVs in puncto Interaktion, Design und funktionaler Erweiterbarkeiten an? Können

diese im Kontext des semantischen Fernsehens auch benutzt werden?

- Die Extraktion und Gewinnung von Semantik aus einem Video bzw. Live-TV-Programm durch automatische Annotationsverfahren und kombinierte Extraktionsverfahren aus den Bereichen Text- und Videoanalyse steht im Mittelpunkt der Realiserung des *Swoozy*-Systems. Ziel ist festzustellen, welche Informationen zurzeit von existierenden DVB-Datenströmen extrahiert und wie diese mit Diensten des semantischen Web, wie z.B. DBpedia oder Wikidata, angereichert werden können.

- Das Design und die Realisierung einer neuen innovativen grafischen Fernsehbasierten Benutzerschnittstelle ermöglicht erweiterte Interaktionen mit Videoinhalten. Zusätzlich stellt diese in interaktiver Form die gefundenen heterogenen Ergebnisse aus dem Web grafisch dar. Wie können sich diese von den aktuellen TV-Apps abgrenzen?

- Wie lässt sich die neue Architektur auch innerhalb eines Fernsehproduktionssystems einsetzen? Dafür wurden dedizierte Werkzeuge für die redaktionelle Unterstützung bei Aufgaben, wie die Annotation oder semantische Beschreibung der Inhalte, erstellt.

In diesem Buch werden dem Leser all diese Fragen unter einem wissenschaftlichen aber auch einem technischen Aspekt beantwortet.

Danksagung

Ein Buch über ein neues Thema zu schreiben, ist immer eine Herausforderung. Und das geht natürlich nicht ohne eine moralische, fachliche und technische Unterstützung! Aus dem Grund möchte ich mich bei den Personen, die mich während der Entstehungsphase dieses Buches begleitet haben, herzlich bedanken. Ganz besonderen Dank an Herrn Prof. Dr. Wolfgang Wahlster für seine Unterstützung bei meinen Forschungsarbeiten zu den Themen Semantic Web, intelligente Benutzerschnittstellen und KI, die mit der technischen Realisierung des semantischen Fernsehens zu einer erfolgreichen Promotion geführt haben. Ebenfalls besonders danken möchte ich Herrn Prof. Dr. Thorsten Herfet, der ohne Zögern und mit Enthusiasmus, seine Bereitschaft für die Begutachtung meiner Dissertation gezeigt hat.

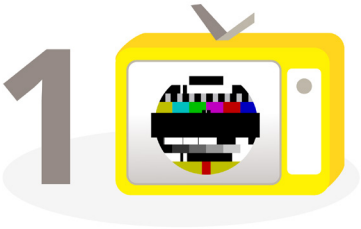
Des Weiteren möchte ich mich bei folgenden Kollegen bedanken:
Simon Bergweiler, Dr. Boris Brandherm, Christof Burgard, Dr. Dietmar Dengler, Christian Hauck, Dr. Jens Haupert, Gerd Herzog, Reinhard Karger, Sönke Knoch, Marcel Köster, Sylvia Krüger, Heike Leonhard, Dr. Robert Neßelrath, Dr. Alassane Ndiaye, Prof. Dr. Günter Neuman, Renato Orsini, Peter Poller, Daniel Porta, Dr. Norbert Reithinger, Armindo Ribeiro.

Die Welt der Fernsehproduktion und ihren internen Prozessen ist sehr komplex, aus dem Grund bedanke ich mich auch ganz herzlich bei den zahlreichen Kollegen der Fernsehsender SR, ZDF, ARD und Mirabelle TV, die während Führungen und lehrreichen Sitzungen, mir akribisch und auf spannender Art und Weise die Möglichkeiten gegeben haben hinter die Kulissen zu schauen.

Ein spezieller Dank den Herren Hanke, Mootz und Rinner vom SR und Herrn Kirchknopf vom ZDF.

Dem CeBIT Innovation Award 2013 Komitee, welches nicht nur 2012 während der Jury-Sitzung in Berlin an Swoozy geglaubt hat, sondern auch die Möglichkeit gegeben hat, das System auf der CeBIT dem breiten Publikum vorzuführen.

Ganz großer Dank geht an meine Eltern Monique und Jean-Paul Deru für ihr langjähriges Vertrauen sowie an meine wunderbarste Freundin Anna-Viktoria Rauch, die mich durch ihren außergewöhnlich guten Zuspruch ständig motiviert hat. An dieser Stelle nochmals vielen lieben Dank!



Einleitung

Motivation

Die „klassische Art fernzusehen“ befindet sich zurzeit in einer Transformationsphase. Sie wird durch IP-TV, Video on Demand (VOD) und Internet erweitert. Die Kombination von Internet und Fernseher ist immer beliebter geworden, wobei die Interaktionsmöglichkeiten des Zuschauers mit den Fernsehinhalten sehr eingeschränkt geblieben sind. Diese Fernsehlösungen bieten dem Benutzer zwar eine gewisse Möglichkeit mit den dargestellten Inhalten zu interagieren (z.B. durch Kamerawechsel, wie bei Free Viewpoint Television oder durch die Einblendung von Zusatzinformationen oder TV-Apps), aber die dort angewandten Eingabemodalitäten (z.B. Sprache oder Fernbedienung) sind für eine intensive und explorative Benutzung von Wissensdatenbanken am Fernseher oder beim Videostreaming ungeeignet. Parallel dazu haben sich in den letzten Jahren neue interaktive Fernseher und *Connected-TV*-Lösungen, wie z.B. Apple-TV, Smart-TVs von Samsung oder LG oder Android-TV-Set-Top-Boxen auf dem Markt durchgesetzt. Diese entwickeln sich immer schneller zu einem universalen Medienzentrum. Dabei sind sie nicht immer miteinander kompatibel und bieten teils unterschiedliche Interaktionsmöglichkeiten an.

Terminologien und Ausprägungen des interaktiven Fernsehens

Heute zählt man über zwanzig unterschiedliche Hersteller von Smart-TV-Geräten, die verschiedene Ausprägungen des interaktiven Fernsehens anbieten. Diese sehr heterogene Landschaft lässt sehr viel Spielraum für Terminologien, die sehr durch Marketing geprägt sind und kaum den tatsächlichen technischen Mehrwert des Fernsehgeräts widerspiegeln. Folgender Abschnitt gibt einen genaueren Überblick über diese Lösungen und Varianten des Fernsehens. Die Technologien zur Übertragung bzw. zum Empfang des Videosignals, die Interaktionsformen mit dem Fernsehbild und die Verbindung zu Internetdiensten werden näher betrachtet, vorgestellt und klassifiziert. Abbildung 1.1 fokussiert diese Fernsehvarianten und zeigt eine mögliche Anordnung der verschiedenen, teilweise kommerziell geprägten Begriffe. Die subtilen Unterschiede zwischen diesen einzelnen Varianten werden im Folgenden näher vorgestellt.

Die aktuellen Fernsehgeräte können in die drei Oberkategorien *IP-TV*, *Connected TV* und *klassisches Fernsehen* klassifiziert werden. Beim *klassischen Fernsehen* können nur lineare Inhalte empfangen und angezeigt werden. Die Ausstrahlung der Fernsehprogramme wird durch digitale Verfahren unterstützt. Der Empfang der Programme wird von einem Digital- bzw. DVB-Receiver ermöglicht. DVB (Digital Video Broadcasting)[1] ist ein standardisiertes Übertragungsverfahren, das für den Satellitenempfang (DVB-S), die Übertragung über Kabelnetze (DVB-C) und den terrestrischen Empfang (DVB-T) entwickelt wurde.

[1] `https://www.dvb.org`

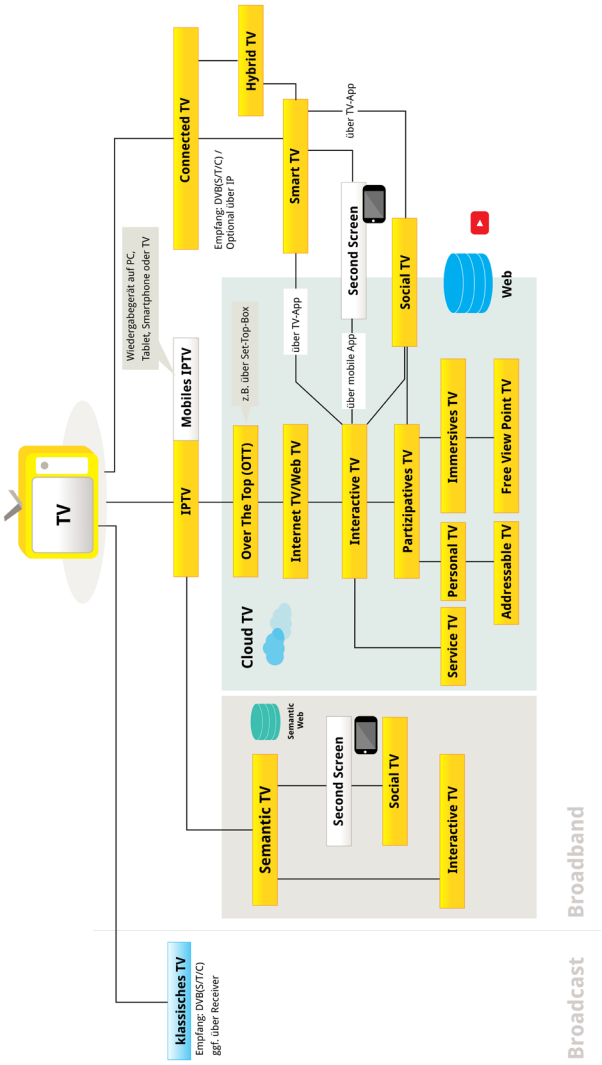


Abbildung 1.1: Klassifizierung der Fernsehvarianten und -technologien

Meistens ist der DVB-Receiver schon im Fernsehgerät integriert oder hat die Form einer an das Fernsehgerät angeschlossenen Set-Top-Box (kurz STB). Der Benutzer hat beim klassischen Fernsehen keine Möglichkeiten Zusatzmedien, eine Websuche oder eine TV-App zu starten. Hierbei spricht man vom *Broadcast-Modus*, da der Empfang der Fernsehprogramme ohne Benutzung des Internets realisiert wird. Im Gegensatz zum klassischen Fernsehen sind *Connected-TVs* mit dem Internet (*Broadband*) verbunden. Die Begriffe *Smart-TV* oder *Connected-TV* werden allgemein als Synonyme für Fernsehgeräte, die mittels einer speziellen Software und spezieller Betriebssysteme interaktiv mit dem Internet verbundene Zusatzfunktionen anbieten, betrachtet. Es besteht jedoch ein feiner Unterschied zwischen *Connected-TV* und *Smart-TV*. Smart-TV bezeichnet eine Gerätekategorie und ein Betriebssystem, das auf die Fernsehhardware abgestimmt ist. Der Fokus bei *Smart-TV* liegt auf der Benutzerinteraktion (z.B. die Steuerung der Lautstärke per Gesten oder Sprachbefehle) und der Internetfähigkeit, wie z.B. durch die Verwendung von Apps und dem Empfang von einer Digitalen Quelle (DVB). Beim *Connected-TV* dagegen liegt der Fokus eher auf der Möglichkeit, dass das Fernsehgerät erstens Zugriff auf das hauseigene Netzwerk (Stichwort: Heimvernetzung) erhält (das Musik- oder Videostreaming kann über DLNA[2] (Digital Living Network Alliance) oder uPNP[3] (Universal Plug and Play) durchgeführt werden) und zweitens Multimediainhalte aus dem Internet abspielen kann. Ein klassischer Fernseher, der keine Internetverbindung besitzt, kann durch den Anschluss einer zusätzlichen Set-Top-Box (wie z.B. Apple TV) oder eines HDMI-Sticks (z.B. Google Chromecast oder Amazon Fire TV Stick) zu einem *Connected-TV* gemacht werden. Je nach Hardwaretyp kann

[2] http://www.dlna.org
[3] https://openconnectivity.org

es jedoch so sein, dass die parallele Anzeige von einem DVB-Signal und die Benutzung, z.B. von Apps, nicht möglich ist. Aus diesem Grund können nur Smart-TVs als „echte" *Connected TV*s eingestuft werden. Bevor der Begriff *Smart-TV* sich kommerziell eingeprägt hat, u.a. durch Werbeaktionen von Samsung, sprach man von *Hybrid-TV*, einer Variante des klassischen Fernsehens mit beschränkten, fest in der Firmware des Gerätes einprogrammierten Internetzugriffsmöglichkeiten (z.B. Wetterdienste oder Online-basiertem Electronic Program Guide). Der Begriff *Hybrid-TV*, der eigentlich eine Gerätekategorie bezeichnet, wird sehr oft mit dem HbbTV-Standard (Hybrid Broadband Broadcast Television), der Informationen parallel zu einer Sendung einblenden kann, verwechselt (siehe Kapitel 4). Jedes aktuelle *Smart-TV* ist in der Lage, HbbTV-Inhalte anzuzeigen.

In seiner heutigen Ausprägung kann ein *Smart-TV* nicht nur DVB-Signale, sondern auch IP-basierte Videostreams (IP-TV), dank dedizierter VOD-Apps, wie z.B. Netflix, empfangen. Parallel dazu zeigt eine neue Tendenz, dass Zuschauer nicht nur das *Smart-TV* als Wiedergabegerät für die Fernsehprogramme benutzen. Mobile Endgeräte (Tablets oder Smartphones) bzw. Second Screens, Spielekonsolen und Set-Top-Boxen sind genauso wie *Smart-TV*s in der Lage, u.a. dank IP-TV, Fernsehprogramme wiederzugeben. Im Gegensatz zum digitalen Empfang über DVB, werden bei IP-TV Fernsehprogramme in linearer und nicht-linearer Form über Internet- und Streaming-Protokolle empfangen. Der Fernsehkonsum von Videoinhalten auf verschiedenen Geräten wird als *Multiscreen-TV* bezeichnet. Empfängt ein Benutzer, unabhängig von dem TV-Angebot seines Internetproviders, über seinen Fernseher dank einer App, ein IP-TV-Angebot, das ihm zusätzliche Dienste, wie VOD oder den Zugriff auf Mediatheken anbietet, spricht

man von *Over-The-Top (OTT)*-Angebot. Bekannte OTT-Angebote sind Netflix, Hulu oder Amazon Video. Diese OTT-Angebote können über eine Set-Top-Box aufgerufen bzw. dargestellt werden oder die Form einer dedizierten Website (*Web-TV*) annehmen.

Der Begriff *interaktives Fernsehen* bezeichnet die Möglichkeit für einen Zuschauer, neben dem Empfang des Fernsehsignals (über IP oder DVB), proaktiv über einen Rückkanal (meistens über die Internetverbindung) und über ein Gerät (Fernbedienung, Second-Screen, App) Zusatzdienste am Fernseher aufzurufen: es findet also, im Gegensatz zum klassischen Fernsehen, eine bidirektionale Kommunikation statt. Der eigentliche Empfang der TV-Programme kann entweder über IP oder DVB erfolgen. Bei *Smart-TV* findet diese Interaktivität durch die Benutzung von Apps oder Zusatzdiensten, wie *HbbTV*, statt. Ausgehend von einer Studie des CSA[4] (Conseil Supérieur de l'Audiovisuel - Höherer Rat des audiovisuellen Sektors in Frankreich) können die Begriffe *Social TV* oder *partizipatives Fernsehen* mit dem Oberbegriff *interaktives Fernsehen* gleichgestellt werden. Bei *Social-TV* spielen sowohl die *Second Screen* als auch die *Smart-TV*s eine zentrale Rolle: der Zuschauer kann über dedizierte TV- bzw. mobile Apps seine Meinungen über eine Sendung mit Freunden teilen oder parallel zu dieser Sendung an interaktiven Spielen teilnehmen. Gute Beispiele für partizipatives Fernsehen bilden interaktive Quizsendungen oder Twitter-Botschaften der Zuschauer, die während der Ausstrahlung durch den Fernsehsender eingeblendet werden. Eine Weiterentwicklung des *partizipativen Fernsehens* stellt die Möglichkeit des Zuschauers dar, einen direkten

[4] http://de.www.csa.fre05d.systranlinks.net/Etudes-et-publications/Les-etudes-thematiques-et-les-etudes-d-impact/Les-etudes-du-CSA/La-television-participative-ou-television-sociale-en-2014

Einfluss auf das Livegeschehen einer Sendung zum Ausstrahlungszeitpunkt zu nehmen, wie z.B. bei der Sendung „Rising Star“[5] oder beim Fernsehsender Okto[6]. *Personal-TV* kann als Unterspezifikation des *interaktiven Fernsehens* betrachtet werden, da der Zuschauer selber seine Programme und Themen auswählen und somit personalisierte Sendungen zusammenstellen kann.

Eine neue Tendenz zeigt sich im sog. *Adressable-TV*[7], das in Europa allerdings noch in den Kinderschuhen steckt und hauptsächlich zur Einblendung geolokalisierter Werbespots innerhalb eines linearen TV-Programms dient. Diese Technik kann jedoch nur mit Hilfe von *Smart-TV*s umgesetzt werden.

Eine besondere Form des *interaktiven Fernsehen*s ist das sog. *Service-TV*, wie es in der Trendstudie *TV 2020* des Beratungsunternehmens für strategische Zukunftsfragen ZPunkt GmbH [Com11] publiziert wurde. Dank *Service-TV* können Zuschauer (oder besser gesagt Bürger) über das Fernsehgerät und eine TV-App, einen schnelleren Zugriff auf Behörden (wie z.B. Bürgerdienste oder das Finanzamt) erhalten und über die Fernbedienung an Bürgerbefragungen teilnehmen. *Service-TV* ist in bestimmten Ländern Teil von e-Government-Initiativen. Mit *immersivem Fernsehen* bezeichnet man eine Form des Fernsehens, bei welcher der Zuschauer proaktiv und völlig frei mit dem angezeigtem Videoinhalt interagieren kann, z.B. mit der freien Auswahl einer Kameraperspektive, wie dies beim *Free View Point TV* der Fall ist. Zusätzliche Geräte, wie 3D-Brillen, können den Effekt der Immersion verbessern.

[5] http://www.rtl.de/thema/rising-star-t8518.html
[6] http://www.okto.tv
[7] https://www.sevenonemedia.de/documents/20182/171614/Addressable+TV+Basisinformationen.pdf

Die Benutzung von Internet am Fernsehen, kombiniert mit Second-Screen-Angeboten und interaktiven Möglichkeiten seitens des Zuschauers, Zusatzinhalte über das Web aufzurufen, kann als *Cloud-TV* bezeichnet werden. Pendant zum *Cloud-TV*-Ansatz bildet das vorgestellte *Semantic-TV*. Bei der letzteren Variante bilden die Zugriffe auf das semantische Web und die Gewinnung einer semantischen Beschreibung aus dem Videoinhalt den Kernaspekt.

Das Fernsehen im Jahre 2016

Bei heutigen interaktiven Fernsehlösungen werden viele Aspekte, wie Internetzugriffe, Benutzbarkeit und Zugang zum Web 3.0 (das semantische Web, bei dem Informationen mit einer eindeutigen Beschreibung versehen werden, damit diese von Computersystemen richtig verstanden und interpretiert werden können) immer noch nicht berücksichtigt und, aufgrund komplexer Bedienung und mangelnder intuitiver Interaktionskonzepte, vernachlässigt. Zudem sind das „Konsumieren" von Fernseh- und Videoinhalten und das gleichzeitige Suchen in Wissensdatenbanken am Fernseher über das Internet mühsam und für den ungeduldigen Zuschauer nicht schnell und effizient genug.

Dazu kommt es oft vor, dass bei sehr vielen Fernsehlösungen, insbesondere bei Smart-TV, das mobile App-Prinzip 1:1 übernommen wurde. Visualisierung oder Interaktionsparadigmen, die zwar für mobile Endgeräte oder am PC als geeignet gelten (z.B. das WIMP (Windows, Icons, Menus, Pointing Device)-Konzept), wurden ohne jegliche Berücksichtigung der spezifischen Randbedingungen, die beim Fernsehschauen

entstehen (Fernbedienung, Anzeige an einem größeren Bildschirm und Entfernung vom Benutzer zum Fernsehgerät), auf Fernsehgeräte übertragen. Aus diesem Grund sind TV-Apps oder die Bedienung von Untermenüs, um eine kombinierte Suche am Fernseher zu starten, ungeeignet. Dazu kommt die Tatsache, dass bestimmte Zuschauer über eine Spielekonsole fernsehen (Playstation 4 oder Xbox One), andere werden die Smart-TV-Angebote über ein Fernsehgerät (oder normales Fernsehgerät mit einer zwischengeschalteten Set-Top-Box) benutzen. Dies führt dazu, dass man nicht nur von einer einzigen standardisierten Smart-TV-Lösung sprechen kann, sondern von einem hybriden TV-Mikrokosmos, da die Zusatzinformationen über verschiedene Kanäle (Apps, HbbTV, eigenes Betriebssystem) dem Zuschauer zur Verfügung gestellt werden können.

Aktuelle Marktforschungen zeigen diese verstärkte Verschmelzung zwischen TV und Internet. Eine Studie[8] der GfK (Gesellschaft für Konsumforschung) aus dem Jahr 2015 zeigt, dass „10 Millionen Personen die Internetfähigkeit ihres TV-Geräts nutzen". Laut der gleichen Studie sind alle diese Geräte Smart-TVs, von denen 93% im Zeitraum Januar-April 2015 HbbTV-fähig sind. Die im Jahr 2015 publizierte Marktstudie von SevenOneMedia (*Media Activity Guide 2015*[9]) [Sev15] und eine Ende 2013 publizierte PricewaterhouseCoopers Deutschland Studie (*Media Trend Outlook*) [10] verdeutlichen zusätzlich, dass die Wünsche der Zuschauer bezüglich der Fernsehsituation sehr stark in Richtung Konsum von kurzen Videoclips (ca. 56%), Nachschlagen von TV-

[8] http://www.tv-plattform.de/images/stories/archiv/2015/sfn_gfk.pdf

[9] https://www.sevenonemedia.de/web/sevenone/mag

[10] https://www.pwc.de/de/technologie-medien-und-telekommunikation/assets/whitepaper-smart-tv.pdf

sendungsbezogenen Themen über Mediatheken (ca. 58%) und Suchen von Informationen und Surfen im Internet (ca. 54%) tendieren. Die Studie [Sev15] stellt klar, dass etwa 64% der Internetbenutzung, die parallel zum Fernsehen durchgeführt wird (hierzu zählen die Benutzer von Second Screens d.h. Second Screeners), einen direkten Bezug auf eine auf das TV-Programm bezogene Informationssuche hat.

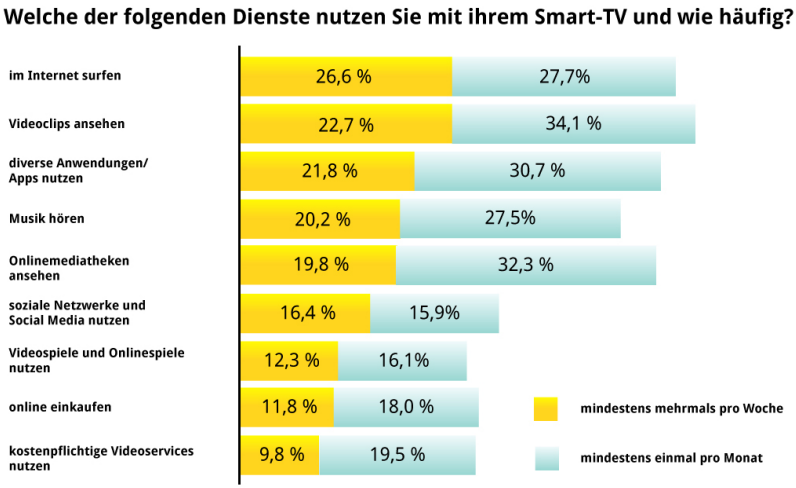


Abbildung 1.2: Studie von PricewaterhouseCoopers zum Thema Internetbenutzung am Fernsehen Quelle: PricewaterhouseCoopers. Grafisch adaptiert aus https://www.pwc.de/de/technologie-medien-und-telekommunikation/assets/whitepaper-smart-tv.pdf

Etwa 40% der Befragten suchen explizit nach Informationen zu Produkten, die sie während der TV-Sendung gesehen haben. Dies führt oft dazu, dass ein Produkt, kurz nachdem es im Fernsehen gezeigt wurde, über Onlineportale gekauft wird. Diese Ergebnisse korrelieren mit den Studienergebnissen, bei denen potentielle Smart-TV-Interessenten viel Wert auf eine Browserkomponente legen, die einen Zugriff auf eine Mediathek oder erweiterte Funktionalitäten, wie z.B. die Einblendung

von passenden sendungsbezogenen Inhalten, ermöglicht. Bei der Studie von PricewaterhouseCoopers Deutschland (siehe Abbildung 1.3) fällt auf, dass 37,3% der Befragten die unpraktische Bedienung der Smart-TV-Onlinefunktionalitäten bemängeln. 92,3% geben als Hauptkriterium für eine Nutzung des Smart-TV eine einfache und verständliche Bedienung an. Wenn dies nicht gegeben ist, bevorzugen etwa 60% der Befragten die Nutzung eines anderen Gerätes (z.B. ein Laptop oder ein mobiles Endgerät), um Recherchen im Internet durchzuführen, obwohl sie diese Suchen über ihre Smart-TVs hätten durchführen können.

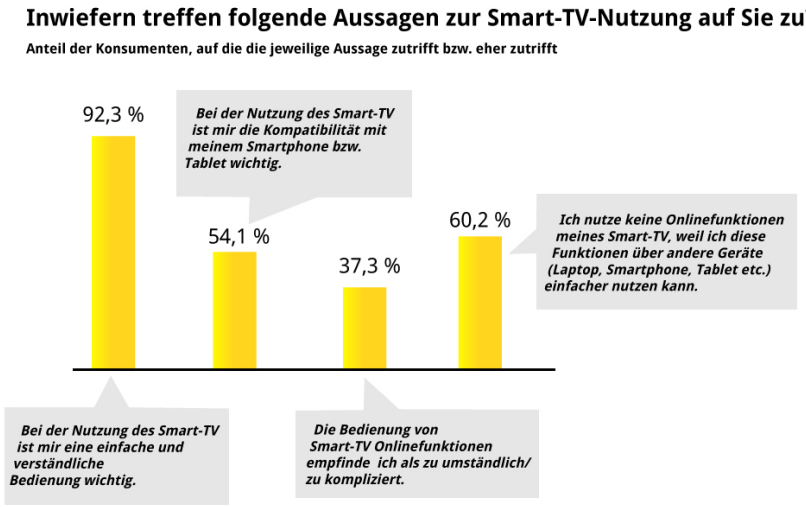


Abbildung 1.3: Studie von PricewaterhouseCoopers zum Thema Internetbenutzung am Fernsehen Quelle: PricewaterhouseCoopers. Grafisch adaptiert aus https://www.pwc.de/de/technologie-medien-und-telekommunikation/assets/whitepaper-smart-tv.pdf

Dies zeigt, dass erstens die aktuellen Erwartungen seitens der Konsumenten sehr hoch sind und zweitens, dass sich das Fernsehen als Medium in verschiedene Richtungen entwickelt. Diese Evolution wird nicht nur in Richtung des Internets erfolgen, sondern auch auf der Diensteebene und auf der Ebene der Integration und Verschmelzung derselben mit immer komplexeren Aufgaben, wie z.B. der Suche im Internet, dem Abruf von personalisierten Programmen, Video On Demand und neuen Interaktionsmöglichkeiten (partizipatives Fernsehen).
Diese Zahlen zeigen eine deutliche Diskrepanz zwischen den Erwartungen der Benutzer und der heutigen Ausprägung des Zugriffs auf verschiedene Wissenskanäle. Die Studie [Sev15], zeigt, dass zurzeit die Hälfte der Konsumenten gleichzeitig zum Fernsehen, parallel ein internetfähiges Gerät benutzt.

Der Weg zum semantischen Fernsehen

Die Idee, das Fernsehen mit dem Web zu verknüpfen, wurde mehrmals, insbesondere im Rahmen von drei größeren europäischen Projekten: NoTube[11], LinkedTV[12] und ViSTA-TV[13] erforscht (siehe Kapitel 5). Parallel dazu versuchen seit einigen Jahren die Smart-TV-Hersteller, die Idee, das Web mit dem Fernsehen zu verknüpfen, um u.a. Mehrwertdienste wie personalisiertes TV-Angebot anbieten zu können, voranzutreiben. Die Thematik des *„semantischen Fernsehens“* oder *„Semantic TV“* ist noch relativ neu. Eine erste Andeutung wurde im Artikel *Is Semantic Web Part of the Television Future?* [Eva09] gegeben. Die Be-

[11] http://www.notube.tv
[12] http://www.linkedtv.eu/
[13] http://vista-tv.eu

schreibung von Evain und die aktuellen Aktivitäten der EBU (European Broadcasting Union)[14] beschränken sich jedoch auf die Möglichkeit, bestehende Metadaten in einem professionellen Workflow (von der Produktion bis zur Ausstrahlung) mit Semantik zu versehen.
Die Arbeiten von John Domingue [MDD06] oder Lora Aroyao [ANM11] versuchen dagegen, dank des Semantic Webs, das Fernseherlebnis anzureichern und u.a. durch das automatische Auffinden von personalisierten und sendungsbezogenen Inhalten im Semantic Web. Diese Ansätze zeigen interessante Möglichkeiten für das interaktive Fernsehen und bilden einen ersten konkreten Schritt in Richtung des semantischen Fernsehens. Leider werden die Inhalte einer Sendung oder des Videomaterials nur über die von den Fernsehsendern übertragenen Daten bzw. Metadaten (meistens EPG) beschrieben. Dies bedeutet zugleich, dass der eigentliche Inhalt innerhalb eines Videos (samt Entitäten) bei den vorgestellten Ansätzen nicht durch Echtzeitextraktion gewonnen wird und somit nur Bruchteile des ausgestrahlten Videos mit einer kompletten semantischen Beschreibung versehen ist. Im Gegensatz zu den vorgestellten Ansätzen spielen bei dem vorgestellten *Swoozy-System*, die Fragen der Live-Extraktion von Semantik aus Videos, die benutzerinitiierte Suche im Semantic Web und der Gesteninteraktionen vor dem Fernseher eine zentrale Rolle.
Vergleicht man das aktuelle *Connected-TV* mit dem *Semantic-TV*, so kann festgestellt werden, dass beim letzteren die Benutzung vom Semantic Web und seinen Diensten in Vordergrund steht. Die Extraktion von Semantik, d.h. welche Entitäten sich in einem Video befinden, ist das Hauptmerkmal von *Semantic-TV*. Bei diesem Ansatz wird ein Medium nicht global beschrieben, sondern jedes einzelne Bildelement enthält eine Beschreibung. Als Ergebnis bekommt der Zuschauer eine

[14] `https://tech.ebu.ch/semanticweb_ebu`

feingranulare semantische Beschreibung des gesamten Inhaltes eines ausgestrahlten Videos. Diese Analyse wird vom *Semantic-TV* in Echtzeit realisiert, da es auch lineares Fernsehprogramm abspielen kann.

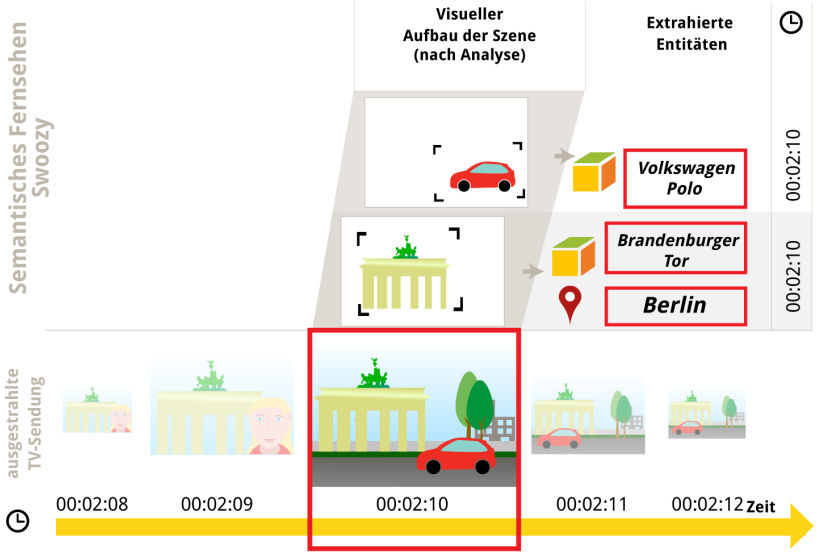


Abbildung 1.4: Vorhaben bei Semantic TV

Die Abbildung 1.4 zeigt dieses Prinzip genauer und verdeutlicht, in wie fern sich dieses von den bisherigen Semantic-TV-Ansätzen, wie NoTube oder ViSTA-TV, unterscheidet. Bei den letzteren wird die globale Beschreibung des Mediums bzw. Videos benutzt, wobei beim Swoozy-Ansatz, die einzelnen Bilder innerhalb des Videos als Eingangspunkt für die semantische Analyse und Extraktion benutzt werden.
Im Gegensatz zum klassischen Fernsehen können *Connected-TV* und *Semantic-TV* lineare und nicht lineare Inhalte darstellen (siehe Abbildung 1.5) und verfügen über Funktionen, ein Video anzuhalten oder zurückzuspulen. Das *Over-The-Top*-Prinzip kann von beiden Varian-

ten gewährleistet werden. Beim *Semantic-TV* kann die Interaktivität mit dem dargestellten Inhalt mehrere Formen besitzen: mit dedizierten Eingabegeräten oder mit einem Second Screen. Der Zugriff auf Fernsehsender-Mediatheken wird zwar von den beiden unterstützt, eine feinere Analyse des tatsächlichen Videoinhalts wird nur vom *Semantic-TV* durchgeführt.



		klassisches TV	Connected-TV Smart-TV	**Semantic TV**
Interaktion und grafische Darstellung	**Linearität**	nur lineares Fernsehen	linear und nicht linear (über Apps)	linear und nicht linear
	Interaktivität	sehr beschränkt / über Zusatzgerät	✓ mittel	✓ sehr hoch
	Aufruf von Zusatzdiensten	EPG / Teletext	✓ über Apps	✓
	OTT-Prinzip für die Anzeige von Medieninhalten	✗	✓	✓
Online Funktionalitäten – Zugriff auf Webinhalte	**Internetfähigkeit**	✗	✓	✓
	Einbindung von Second Screens	✗	✓	✓
	Social TV - Aspekte	✗	✓	✓
	Zugriff auf Fernseh-sender-Mediatheken	✗	✓	✓
Online Funktionalitäten – Semantic Web	**Extraktion von Semantik aus Videomaterial**	✗	✗	✓
	Zugriff auf einzelne Entitäten eines (Live)-Videos	✗	✗	✓
	Benutzung von Web 3.0 für die Mediensuche	✗	✗	✓
	Einbindung von (Live)-Annotationen	✗	✗	✓

Abbildung 1.5: Vergleich Semantic TV zum klassischen Fernsehen und zu Connected-TVs

Wissenschaftliche Ziele

Die verschiedenen Technologien und Fernsehvarianten bezüglich der Art fernzusehen, werfen mehrere wissenschaftliche Fragen auf. Diese Fragen lassen sich in folgende Kategorien anordnen:

Interaktionen vor dem Fernseher

Parallel zu einer Sendung möchte der Zuschauer schnell und gezielt Informationen zum vorgestellten Thema erhalten. Bei diesem Schritt passiert ein Interaktionsbruch, denn die Fernbedienung, das richtige Sprachkommando oder das passende Untermenü müssen zuerst gefunden werden, damit der Zuschauer eine URL oder einen Suchbegriff in seinen im Fernseher eingebauten Internet-Browser eingeben kann. Zwar wird das Gesuchte angezeigt, aber die Webseite oder die Ergebnisliste sind meistens schwer lesbar und die Interaktionsmöglichkeiten beschränken sich nur auf ein paar mit der Fernbedienung selektierbaren Links. Eine Suche im semantischen Web ist so nicht durchführbar. Die Gestaltung und Visualisierungsmöglichkeiten des Zugriffs und der Interaktionen an einem Fernseher folgen anderen Richtlinien als herkömmliche Software-Benutzerschnittstellen: hier gilt es primär eine benutzerzentrierte und dedizierte Umgebung zu gestalten, die für jedermann intuitiv benutzbar sein sollte.
Zusätzlich stellen sich die Fragen der Wissensrepräsentation und deren Ausprägung in Verbindung mit einem Fernsehsystem. Bei solchen Systemen sind Benutzerschnittstellen oft nur ein Synonym für eine rein grafische Darstellung, die mittels Darstellungstechnologien (z.B. HTML5, AIR) auf eine Fernsehhardware kostengünstig portiert wurden. Hier stellen sich folgende Fragen:

- *Welche Kriterien müssen erfüllt werden, um eine intuitive und benutzerfreundliche Interaktion vor dem Fernseher zu gewährleisten?*
- *Kann man sich von dem aktuellen TV-App-Paradigma lösen, indem man dem Benutzer eine andere, intuitivere Interaktion- und Auswahlmöglichkeit anbietet?*

Mehrbenutzerbetrieb

Fernsehen kann als soziale Aktivität bezeichnet werden, da mehrere Teilnehmer gleichzeitig ein gemeinsames Programm anschauen und dadurch Interaktionen zwischen den Zuschauern, aber auch mit dem Fernsehgerät, entstehen können. Dabei steht die Frage des Mehrbenutzerbetriebs vor dem Fernseher im Vordergrund. Lösungen, die z.B. innerhalb des DICIT-Projekts [MOM$^+$08] entstanden sind, fokussieren eher die Erkennung von Fragen und Befehlen, die sprachlich vom Zuschauer vor dem Fernsehgerät geäußert wurden. Dafür werden die Befehle oder die Fragen des jeweiligen Zuschauers mit Daten aus dem EPG korreliert und mittels Websuche-Anfragen teils beantwortet.
Bei dem Ansatz des semantischen Fernsehens Swoozy liegt jedoch der Fokus auf Aspekten des Mehrbenutzerbetriebs, der sowohl grafisch- als auch interaktions-technisch in einer einheitlichen Form berücksichtigt werden muss. Dabei müssen zuerst folgende wissenschaftliche Aspekte geklärt werden:

- *Wie können mehrere Benutzer gleichzeitig in einer nicht disruptiven Art und Weise und mit minimalsten Interaktionsschritten mit den ausgestrahlten und am Fernseher dargestellten Inhalten interagieren?*

- *Wie können mehrere Benutzer miteinander das gleiche System benutzen, ohne dabei andere Zuschauer zu stören?*
- *Können Ansätze wie Multiscreen-TV oder Second-Screen Lösungen für diese Problematik bieten?*

Multi-Screen TV, kognitive Last und App-Design

Die kognitive Last bezeichnet die mentale Anforderung an einer Person während einer Lernaktivität. Dabei spielen das Gedächtnis und die Fähigkeit während einer Aufgabe parallele Informationen wahrzunehmen und diese zu speichern, eine essenzielle Rolle. Diese Theorie trifft beim sog. Multiscreen-Fernsehen zu, insbesondere während der Benutzung von Second Screens. Die Benutzung dieser, parallel zu einer Sendung, zwingt den Zuschauer sehr oft den Blick vom TV-Bildschirm zu seinem mobilen Gerät zu wenden, mit der Gefahr den tatsächlichen Ablauf seiner Sendung aus den Augen zu verlieren. Die Studie [HJC12] belegt, dass die Benutzung von Second Screen, parallel zu einer Sendung, einen direkten Einfluss auf die zeitliche Länge und die Frequenz des Blickwechsels hat. Hierbei spielen die Konzipierung und der visuelle Aufbau (Design) einer Second-Screen-App eine wesentliche Rolle. In [NEJ16] wird empfohlen, eine Balance zwischen Informationsdarstellung und Interaktionskomplexität auf die Second-Screen zu berücksichtigen. Komplexe Interaktionsmöglichkeiten oder auf Second-Screens überfrachtete dargestellte Informationen können dazu führen, dass die Aufmerksamkeit auf das erste Medium (den Fernseher) deutlich reduziert wird. Die Studien [GLDG+14], [HJC12], [DMKA15] [BEJ+14] und [VM14] adressieren dieses Problem der Aufmerksamkeit beim *Multi-Screen*-TV.
Bei dem kontinuierlichen Wechsel der Augen vom Fernseher zum

Second Screen spielt die Akkommodation der Augen eine oft unterschätzte Rolle. Unter Akkommodation versteht man die „Fähigkeit des Auges, die Brechkraft dynamisch anzupassen und dadurch Gegenstände in unterschiedlicher Entfernung scharf zu sehen“[15]. Dieser Wechsel kann bei bestimmten Zuschauern auf Dauer zu einer Müdigkeit führen und somit das *Multiscreen*-TV-Erlebnis verschlechtern.
Erschwerend kommt hinzu, dass heutige Second-Screen-Angebote der Fernsehsender sich untereinander visuell sehr stark unterscheiden. Jeder Fernsehsender bietet seine eigene App an, die mit einem eigenen Corporate Design und unterschiedlichen Funktionalitäten bzw. Menüpunkten versehen ist. Der Zuschauer muss sich bei jeder Second-Screen-App ein neues mentales Modell merken. Dies führt dazu, dass die kognitive Last erhöht wird, insbesondere, wenn der Zuschauer „zappt“. Er wird jedes Mal gezwungen, die richtige App des jeweiligen Senders auf seinem Second-Screen zu finden und muss sich daran erinnern, welche Interaktionen innerhalb der App durchgeführt werden müssen, um zur gewünschten Information zu gelangen. Diese Aspekte führen zu folgenden Fragestellungen:

- *Wie können ein Benutzerschnittstellen-Design und die Interaktion so gestaltet werden, dass die kognitive Last bei einer parallelen Nutzung einer Second Screen App und dem Fernseher reduziert werden kann?*
- *Wie sollte die Benutzerschnittstelle konzipiert werden, damit trotz unterschiedlicher Entfernungen des Zuschauers zum Fernsehgerät, das Fernseherlebnis auf jedem Gerät einheitlich bleibt?*
- *Welchen Designansätzen muss gefolgt werden, damit die Aufmerksamkeit der Zuschauer erhalten bleibt?*

[15] `http://www.lasikon.de/auge/akkommodation`

- *Wie kann ein Zuschauer innerhalb seines „Multiscreen"-Erlebnisses, das mit dem semantischen Fernsehen gekoppelt ist, multimodale Interaktionen durchführen?*

Live-Extraktion von Semantik aus Bildfolgen

Die Definition von Semantic-TV fokussiert primär die Möglichkeit, aus einer Bildfolge den Sinn und die darin enthaltenen Elemente zu extrahieren und nachträglich zu klassifizieren. Zuschauer können diese Annotationen später als Eingabe für eine semantische Suche benutzen (Stichwort: *Grabbables* (siehe Kapitel 6)). Dafür müssen Verfahren gefunden werden, die nicht nur in der Lage sind, aus dem gesamten Video die einzelnen Elemente zu extrahieren, sondern auch deren Beschreibung in Form von Annotationen oder Metadaten zurückzuliefern. Um dies zu realisieren, können mehrere Quellen parallel benutzt werden, z.B. die in dem DVB-Stream mitgelieferten Tabellen und insbesondere der EPG (Electronic Program Guide) oder die Untertitel, falls diese vom Fernsehsender mitausgestrahlt werden. Bei dieser Betrachtung stehen folgende Fragen im Mittelpunkt:

- *Welche Live-Quellen sollen für die Extraktion benutzt werden?*
- *Welche Ansätze und Extraktionsmechanismen liefern in einem Live-Kontext eine ausreichende semantische Beschreibung des Videomaterials und den darin enthaltenen Elementen?*
- *Wie können die extrahierte Semantik bzw. die Annotationen innerhalb des semantischen Fernsehens weiterwendet werden?*
- *Das Live-Fernsehen ist ein lineares bzw. kontinuierliches Medium, da der Zuschauer die Videowiedergabe nicht beeinflussen kann. Wie*

kann zusätzlich zu dem Live-Videosignal gewährleistet werden, dass die extrahierten Annotationen nahezu zeitgleich und kontextbezogen für den Benutzer dargestellt werden?

Verschmelzung des Semantic Web mit dem Fernsehen

Als Ergebnis der Live-Extraktion entstehen Annotationen, die mit semantischen Diensten, wie DBPedia, Wikidata oder dem Google Knowledge Graph, in Relation gebracht werden können. Ziel ist es, angewandte Konzepte zur Suche im semantischen Web (Web 3.0) durch angepasste und benutzerzentrierte Interaktionen zu vereinfachen, um dem Zuschauer eine deutlich höhere Interaktivität und einen direkten Zugriff auf Daten direkt vom Fernseher anzubieten. Damit das Konzept vom semantischen Fernsehen realisiert werden kann, muss eine Abstraktionsebene zwischen der tatsächlichen technischen Ausprägung, der grafischen Darstellung und ihrer Bedeutung skizziert werden. Dem Paradigma *„No presentation without representation„* [MW98] [RBE+05] und dem *Spot*- und *Spotlet*-Konzept [SDB09] folgend, wird durch die technische Realisierung des Swoozy-Systems gezeigt, wie eine Semantifizierung der Benutzeroberfläche eine nahtlose Verbindung mit dem Semantic Web ermöglichen kann, um bestehende heterogene Medien (Bilder, Texte, TV-Ausschnitte und Videos) aus verschiedenen Quellen zugreifbarer zu machen.

- *Welche Visualisierungsansätze ermöglichen eine benutzerzentrierte Darstellung semantischer Ergebnisse?*
- *Wie fein granular muss eine semantische Beschreibung erfolgen, damit eine Suche über das Web 3.0 möglich ist? Und welche Voraus-*

setzungen müssen dafür erfüllt werden?

- *Wie kann eine intuitive semantische Suche gestartet werden und welche Abstraktionsebenen müssen beachtet werden, damit ein Benutzer, trotz Komplexität der Suchanfrage, korrekte und kontextorientierte Ergebnisse von den Web 3.0-Diensten erhält?*

Technische Herausforderungen

Als erste technische Herausforderung bei der Realisierung eines neuen Systems, ist es, eine einheitliche und generische Plattform zu schaffen, die nicht nur die beschriebene Hardware-Problematik löst, sondern auch die Heterogenität der einzelnen Webdienste und Zugriffe betrachtet und durch eine benutzerfreundliche Bedienung handhabbar macht. Die zweite Herausforderung besteht darin, aktuelle Live-Sendungen mit intelligenten Prozessen entweder durch die Analyse von EPG, Audiodeskription und Teletextdaten oder durch automatische Bilderkennungsverfahren zu annotieren und somit, trotz noch nicht vorhandener *semantischer TV*-Infrastruktur seitens der Fernsehsender, dem Zuschauer ein verbessertes und „partizipatives" Fernseherlebnis anzubieten. Dazu passend wird parallel zur Erstellung einer intelligenten Fernsehapplikation (hier als Swoozy Client bezeichnet) die Entwicklung einer auf einem DVB-basierenden Softwarelösung zur Live-Generierung von Annotationen von TV-Inhalten realisiert. Im letzten Textabschnitt dieses Buches werden zwei Werkzeuge vorgestellt, die eine Fernsehredaktion beim Annotieren und *„Semantifizieren"* von Videos und Sendungen innerhalb des Produktionsworkflows (Ausstrahlung, Fernsehender, Produktion und Redaktion) unterstützen.

Technische Fragestellungen

Das semantische Fernsehen stellt mehrere Herausforderungen aus drei Themenbereichen dar: Interaktionen mit dem Fernseher, das intuitive Suchen von neuen Medien im Web und der automatischen Verknüpfung von Wissen mit TV-Inhalten. Dabei spielen sowohl wissenschaftliche als auch technische Fragen eine wichtige Rolle. Zunächst müssen die heutigen fernseh- und medienproduktionsspezifischen Aufgaben näher betrachtet werden, wie z.B. die Annotation und Metadatengenerierung während des Produktionsworkflows. Zusätzlich muss bestimmt werden, welche ersten Schritte in Richtung einer semi-automatischen Semantifizierung der ausgestrahlten Programme durchgeführt werden müssen. Folgende Auflistung umfasst die technischen Herausforderungen und Fragestellungen, um das Konzept Semantic TV zu realisieren.

- *Welche aktuellen Darstellungsmöglichkeiten (App oder integriertes System) sind für heutige Smart-TVs am geeignetsten?*
- *Verschiedene Hersteller besitzen verschiedene Betriebssysteme und Entwicklungsplattformen (auch SDK – Software Development Kit genannt), die sich nicht nur rein architekturtechnisch unterscheiden, sondern auch programmiertechnisch (API). Dabei muss identifiziert werden, welche APIs und Anbindungen von jedem Hersteller angeboten werden, um eine intuitive Benutzerschnittstelle und ein Benutzererlebnis (engl. User Experience (UX)) zu gestalten und diese mit bestehenden Interaktionsformen zu verbinden.*
- *Inwiefern können über DVB ausgestrahlte Video- und Audiodaten mittels einer Wissensdomäne mit Semantik und Annotationsverfahren angereichert werden?*

- *Wie können Webdienste auf Beschreibungs- und Repräsentationsebene vereint werden, damit der Benutzer diese über das Fernsehsystem ansteuern kann?*
- *Welche Interaktionsformen können realistisch bei der Benutzung von einem Fernsehsystem benutzt werden?*
- *Wie lässt sich eine performante Wissensextraktionskomponente in Kombination mit DVB-Streams realisieren und integrieren?*
- *Welche Maßnahmen müssen Produktion und Fernsehsender treffen, damit das semantische Fernsehen eine Realität wird?*
- *Welche Werkzeuge existieren, um eine Semantifizierung der Mediendaten zu realisieren und welche Nach- und Vorteile bringen die einzelnen Softwarelösungen mit sich?*

Aufbau des Buches

1. Die Einleitung stellt die wissenschaftlichen und technischen Ziele der Realiserung von Swoozy dar, in dem u.a. die Motivationen von mir angesichts der verschiedenen Marktstudien und heutigen technischen Herausforderungen im Bereich interaktives Fernsehen skizziert werden.

2. Im Kapitel *„Das semantische Web“* werden die Grundlagen des Web 3.0 näher skizziert. Dieses Kapitel vermittelt technische und theoretische Hintergrundinformationen über das semantische Web, die dort angewandten Technologien und liefert einen ausführlichen Überblick über Annotationen und Verwendung von Metadaten im Hinblick auf eine Benutzung mit semantischen Webdiensten. In diesem Kapitel werden Dienste, u.a. DBpedia, Google Knowledge Graph, Wikidata und Yago, samt Zugriffsmöglichkeiten auf Wissensdomänen mittels Abfragesprachen, wie SPARQL und APIs, detailliert vorgestellt.

3. Multimodale Interaktionen werden im Kapitel *„Verwandte Arbeiten“* genauer betrachtet, insbesondere wie eine Kombination aus multimodalen Interaktionen, Suche und visueller Darstellung semantischer Ergebnisse erreicht werden kann. Dabei werden nicht nur die verschiedenen Technologien und Ansätze detailliert vorgestellt, sondern auch ein Überblick über mögliche Kombinationen gegeben. Auch werden innerhalb des Kapitels verschiedene Verfahren zur Videoanalyse und Wissensextraktion vorgestellt, die Ansatzpunkte zur Erstellung des Swoozy-Systems bilden. Zusätzlich werden spezifische Cloud-basierte Dienste bzw. API vorgestellt, die bei einer Echtzeitanalyse der Sendungen zum

Einsatz kommen können.

4. Im Kapitel *„Technische Grundlagen zum digitalen interaktiven Fernsehen“* wird ein Überblick über die aktuellen, technischen Ausprägungen von Smart-TV gegeben. Hierbei werden die APIs und technischen Möglichkeiten gängiger kommerzieller Fernsehhardwareplattformen untersucht und bewertet. Zusätzlich werden die Konzepte und Grundlagen von HbbTV detailliert vorgestellt. Des Weiteren werden die unterschiedlichen DVB-Tabellen, die für Swoozy relevante Informationen beinhalten, aufgelistet. Dafür werden die Grundlagen des DVB-T/S-Empfangsverfahrens erläutert. Im letzten Abschnitt werden mittels einer Übersichtstabelle und einer Analyse die Vor- und Nachteile jeder Plattform veranschaulicht.

5. Anhand des Kapitels *Verwandter Arbeiten* und des letzten Kapitels *„ Technische Grundlagen zum digitalen interaktiven Fernsehen“* werden die Ansätze analysiert und diskutiert. Hierbei ist das Ziel, alle Verfahren zu vergleichen und ihre Einsatzmöglichkeiten und Potenziale innerhalb des Swoozy-Frameworks zu bewerten.
   Zusätzlich werden verschiedene von mir realisierte Ansätze und Demoszenarien im Bereich semantischer Interaktion, z.B. mit den Systemen CoMET, Calisto und Cirius und deren Teilkomponenten, insbesondere die grafisch gestützte Repräsentation von semantischen Suchkomponenten, die sogenannten Spotlets sowie Forschungsprojekte im Gebiet des interaktiven Fernsehens, vorgestellt. Letztere bilden einen wichtigen Baustein für die konkrete und technische Realisierung der semantischen Interaktion im Zusammenhang mit dem Fernsehsystem.

6. Nachdem die Bausteine identifiziert worden sind, werden im Kapitel *Konzept und Realisierung von Swoozy: das semantische Fernsehen* die technische Architektur und die Grundlagen des Swoozy-System zum semantischen Fernsehen präsentiert. Basierend auf den Erkenntnissen aus den letzten beiden Kapiteln werden im zweiten Abschnitt alle technischen Details zur Analyse und Wissensextraktion aus TV-Programmen dargestellt. Dabei wird beschrieben, welche Komponenten für die Audio- und Videoextraktion benutzt werden und welche Ansätze implementiert worden sind. Zusätzlich wird die technische Machbarkeit einer Cloud-basierten Anbindung der Extraktion und der resultierenden semantischen Verknüpfung mit Live-TV-Inhalten über die DVB-Standards näher betrachtet. In diesem Abschnitt werden auch das Zusammenspiel zwischen Swoozy und textbasierte Extraktionskomponenten, wie z.B. NEMEX, vorgestellt.

   Im nächsten Schritt werden die Designentscheidungen und die grafische Umsetzung der Benutzerschnittstelle vorgestellt. Dabei werden die Designprozesse und die Motivationen skizziert, die dazu beitragen, eine kontextorientierte Darstellung der Ergebnisse zu realisieren. Des Weiteren wird das Zusammenspiel zwischen dem Swoozy-Server und den Clients erläutert und wie diese Komponenten in Kombination mit neuen Interaktionsformen angesteuert werden können, um eine Suche durchzuführen.

7. Nachdem das Prinzip und die Implementierung von Swoozy näher erklärt wurden, werden zwei Werkzeuge (Swoozy Livana und Swoozy SKRPTR) dargestellt, die den redaktionellen Annotations-Workflow innerhalb eines Fernsehsenders erleichtern sollen. Somit wird einem Redakteur eine schnelle und effiziente Möglich-

keit zur Verfügung gestellt, bereits während der Produktion, seine Beiträge (Sendungsvideos) mit semantischer Information zu versehen und diese nachträglich zu bearbeiten.

8. Im Kapitel *„Anwendungen und realisierte Demonstratoren"* werden mehrere Ausprägungen und Versionen des Swoozy-Systems vorgestellt, die für verschiedene Szenarien entwickelt wurden.

9. Im letzten Kapitel *„Fazit und zukünftige Arbeiten"* werden Erweiterungsmöglichkeiten und die sich daraus ergebenden neuen technischen Implikationen für das semantische Fernsehen vorgestellt inklusive möglicher Monetarisierung von Live TV-Inhalten mit Hilfe von *Grabbables*, einer spezifischen für Swoozy entwickelten interaktiven grafischen Darstellung semantischer Annotationen.

Anforderungen

Verschiedene Anforderungen unterschiedlicher Natur werden während der Entwicklung und Ausarbeitung des Swoozy-Systems gestellt. Diese können auf vier Ebenen klassifiziert werden (siehe Abbildung 1.6). Auf der Datenebene muss das zu entwickelnde System in der Lage sein, die semantischen Daten aus unterschiedlichen heterogenen Wissensdatenquellen zu verwalten und korrekt zu kombinieren. Dafür muss gewährleistet werden, dass die ausgewählten Dienste semantisch beschrieben und miteinander kompatibel sind. Gegebenenfalls müssen diese mittels einer semantischen Beschreibung vereinheitlicht und von einer dedizierten Komponente verwaltet werden.
Zusätzlich muss überprüft werden, welche Programmierschnittstellen für die Suche (Stichwort: REST-basierte APIs) im Semantic Web

am geeignetsten sind und qualitativ hochwertige multimediale Medienobjekte rückliefern. Parallel dazu müssen die Video- und Text-Extraktionsverfahren in der Lage sein, ausreichende semantische Informationen und Beschreibungen zurückzuliefern, damit eine Weiterverwendung, z.B. durch weitere Webdienste ermöglicht wird . Die Frage der Annotationen der Medienobjekte spielt ebenso eine wichtige Rolle, wie die Echtzeitanreicherung und Extraktion von Daten aus dem Fernsehsignal, wie z.B. dem EPG-Datenstrom. Diese Anreicherung sollte womöglich (semi)-automatisch und in Echtzeit erfolgen, ohne dabei das Fernseherlebnis des Zuschauers zu beinträchtigen. Jedoch, um eine bessere Unterstützung bei dem Annotationsvorgang anzubieten und die Qualität der generierten semantischen Annotationen zu überprüfen, müssen den Fernsehredakteuren einfach zu bedienende Werkzeuge zur Verfügung gestellt werden. Ausgehend von den semantischen Daten muss das semantische Fernsehen in der Lage sein, dem Zuschauer die gefundenen Medienobjekte in einer verständlichen und nachvollziehbaren Art und Weise zu präsentieren. Hierfür muss eine grafische Unterstützung der Suche und der Aufbereitung der Ergebnisse konzipiert werden. Hier kommen Design- und Interaktionsparadigmen, wie z.B. das 10-Foot Design, ins Spiel, welche die optimale Gestaltung der grafischen Elemente bestimmen und in welcher Form diese dem Zuschauer am Fernseher angeboten werden können. Anzeigeverfahren, wie Überblendungen, d.h. die Möglichkeit über das Live-Fernsehbild noch zusätzliche Grafiken einzublenden, müssen überprüft werden.

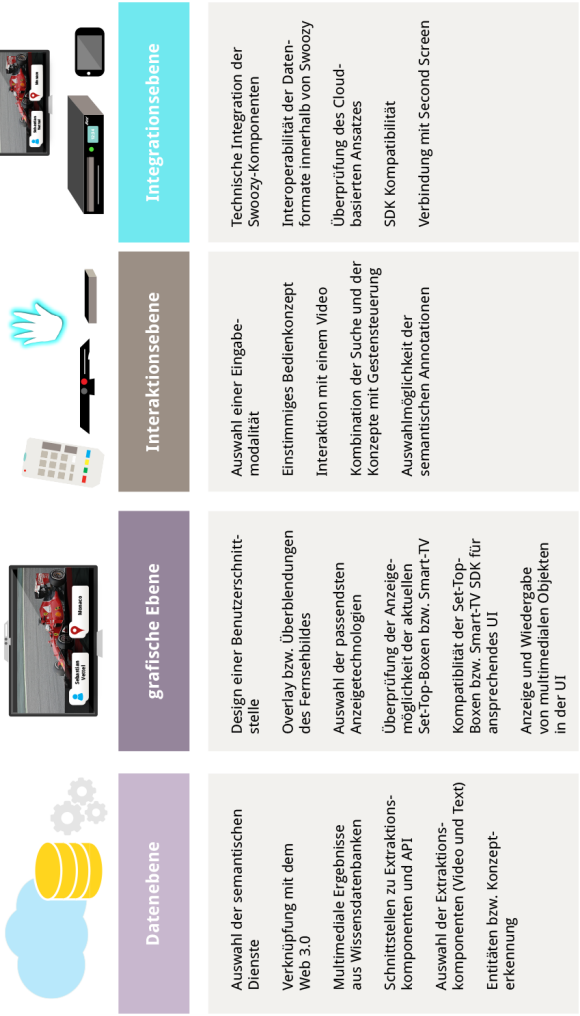
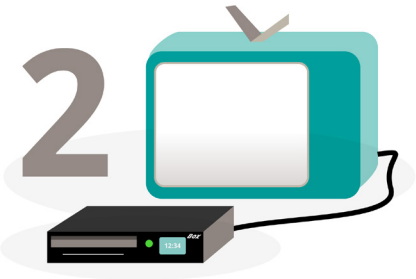


Abbildung 1.6: Anforderungsebenen für Swoozy

Zusätzlich müssen die Technologien zur Darstellung dieser Grafiken erkundigt und die SDKs der jeweiligen Fernsehgerätehersteller analysiert werden, um zu erörtern, ob die gleichzeitige Anzeige oder Wiedergabe mehrerer Videosignale möglich ist.

Die Präsentationsebene spielt die zentrale Rolle bei der grafischen Ausgabe des semantischen Fernsehens. Nichtsdestotrotz sollte das Swoozy-System dem Zuschauer eine Möglichkeit bieten mit den dargestellten Informationen zu interagieren und ein Bedienkonzept vorweisen. Hierfür sollte ein einstimmiges Bedienkonzept zur Interaktion mit einem Video, zur Selektierung und Suche der Konzepte und Auswahl der gefundenen Informationen am Bildschirm angeboten werden.

Als letzter Punkt der Anforderungen steht die technische Integration des vorgestellten Ansatzes. Unter Integration versteht man die Möglichkeit, die Interaktion-, Visualisierungs- und Datenebenen innerhalb eines einheitlichen Frameworks zu integrieren und dieses in eine für sich abgeschlossene Hardware- bzw. Softwareumgebung laufen zu lassen. Dafür müssen die aktuellen Fernseh- und Hardwaretechnologien, wie z.B. Set-Top-Boxen oder Smart-TV, analysiert werden und die Möglichkeiten diese mit Programmiersprachen zu erweitern, überprüft werden.



Das Semantische Web: Web 3.0

Grundlagen

Die Geschichte des Semantic Web

Im Jahre 1969 präsentierten Allan M. Collins und Ross Quillian in [CQ69] das erste Verfahren, um Faktenwissen mittels semantischer Netze in einem Computersystem zu speichern und danach suchen zu können. Damals handelte es sich um eine sehr vereinfachte 3-Stufen-Hierarchie, bei der Tiere, Vögel und Fische in zwei Unterkategorien mit verschiedenen Eigenschaften (kann fliegen/kann schwimmen) kategorisiert wurden. Somit konnte man erste Folgerungen der Form: „Ein Kanarienvogel ist gelb, gehört zur Familie der Vögel, die wiederum dem Oberbegriff *„Tiere“* zugeordnet werden kann“, ableiten. Durch Relationen, wie *isA (ist ein)*, konnten nachträglich bestimmte Rückschlüsse gezogen werden. Die Autoren fanden einen erheblichen Unterschied zwischen der Anordnung im menschlichen Gedächtnis und der eher hierarchiebasierten Anordnung dieses Konzepts. Zum Beispiel wurden Hunde direkt von Menschen als *„Tier“* bezeichnet, obwohl, gemäß ei-

ner strikten Kategorisierung, der Hund erst als *„Säugetier“* bezeichnet werden müsste. Diese erste Strukturierung des Wissens hatte gezeigt, dass es möglich ist, Wissen in Computersystemen zu speichern. In den 50er bis 60er Jahren waren diese Strukturen sehr dokumentbezogen. Als Anhaltspunkt für die Kategorisierung wurde lange die Metapher der alphabetischen Anordnung der Themen, gemäß des Prinzips des *„Bibliothekskatalogs“*, für die Suche nach einer Information benutzt. Dieser Suchmodus wurde innerhalb der ersten Systeme verwendet. Dies warf die Frage auf, *wie* oder besser formuliert *wann* tatsächlich ein physisches Objekt (z.B. Tier, Person oder Gegenstand) zu einem *„Dokument“* wird, d.h. wann ein Objekt eine Analyse und Klassifizierung benötigt und bis zu welchem Komplexitätsgrad bzw. Granularität.
Um diese Frage zu beantworten, stützt sich Michael Buckland auf die Arbeiten der französischen Dokumentationsexpertin Suzanne Briet und identifizierte die folgenden vier Kriterien: *Intention*, *Prozess*, *Perzeption* und *Indizierbarkeit* (kann das Objekt in eine Kategorie oder Klassifizierung hinzugefügt werden?). Diese bestimmen wann ein Objekt zu einem *„Dokument“* wird. Die Kriterien haben eine direkte Auswirkung auf die Auffindbarkeit und Klassifizierbarkeit eines Objekts [Buc97].
In „Weaving the Web“ [BLHL+01] und „Spinning the Semantic Web: Bringing the World Wide Web to its Full Potential“ [FWL02], teilt der Erfinder des World Wide Web, Tim Berners Lee, seinen Traum mit, dass eines Tages *„Maschinen in der Lage sein werden, alle Daten des Web - Inhalte, Links und Transaktionen zwischen Personen und Computern - zu analysieren“*.
Das kann nur erfolgreich realisiert werden, indem die heutigen Konzepte des Web, wie z.B. Hypertextlinks, mit Zusatzinformationen angereichert werden, damit die Webseiten und zugleich deren Inhalte (Bilder, Texte, Videos, Webseiten, Adressen...) intelligenter erfasst und

angeordnet werden können. Diese Zusatzinformationen (oder Metadaten) werden zusätzlich zum Inhalt oder zur Datei hinzugefügt und beinhalten eine semantische Beschreibung (z.B. ein Konzept) des Inhalts. Parallel zu seinem Inhalt kann ein Text Informationen über den Autor, das Erstellungsdatum oder die Anzahl der angehängten Bilder beinhalten. Eine Auflistung dieser Zusatzinformationen findet sich auf der Webseite Schema.org[1].

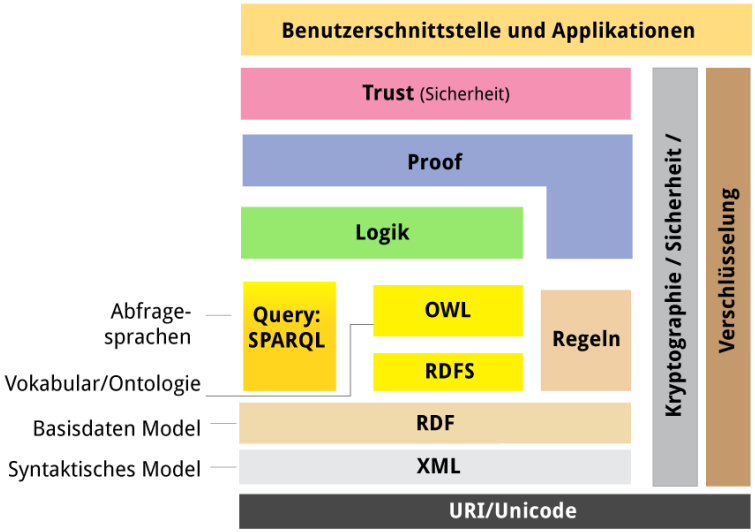


Abbildung 2.1: Semantisches Schichtendiagramm. Quelle: Grafisch adaptiert von http://de.slideshare.net/swadpasc/semantic-web-from-the-2013-perspective

Verschiedene Standards, wie z.B. RDF (Resource Description Framework), OWL (Web Ontology Language) vom W3C oder Suchsprachen, wurden definiert, um diese Inhalte zu beschreiben, sodass die Inhalte

[1] `http://www.schema.org`

von Suchalgorithmen im Web wiedergefunden werden können. Damit diese Beschreibungen nicht willkürlich vergeben werden, müssen bestimmte Grundstrukturen und Hierarchien respektiert werden, sodass Begriffe, aber auch Konzepte, die gleiche Bedeutung für den Menschen als auch für ein Computersystem besitzen. Dafür verwendet man eine Ontologie, die eine Hierarchie und Klassifikation des Wissens in formaler Sprache beschreibt, ähnlich der Grundidee von Allan M.Collins (in [CQ69]). Die neue Form des Sammelns von Wissen und den damit verbundenen Suchmechanismen bezeichnet man als das *„Semantic Web"* oder auch *Web 3.0* [WDT+06]. Mit dem *Semantic Web* ist es nicht nur für Suchalgorithmen möglich, neue Inhalte oder Medien in Wissensdatenbanken zu finden, sondern es können sich neue Relationen bilden, die neue Inhalte hervorbringen. Damit erhält das Web eine neue Dimension. Es wurde eine erweiterte Ebene geschaffen, die bei der Suche und deren Ergebnissen die gleiche inhärente Bedeutung für den Menschen wie auch für die Maschine hat.

Ressourcen und Links

Einer der Haupteigenschaften des Webs ist es, Dokumente über Links zu verknüpfen und eine nahtlose Navigation und Suche zu ermöglichen. Dabei spielen die sog. *Ressourcen* eine wichtige Rolle. Unter einer *Ressource* versteht man ein Dokument, das unter einem URI (engl. Uniform Resource Identifier) abgerufen werden kann. Ein URI kann direkt zur einer (X)HTML-Seite hinzugefügt werden, damit sie später gefunden und neu verlinkt werden kann. Dieser URI ist eine eindeutige Identifikation für ein Element oder eine Ressource im Web. Im Web werden diese URIs als URL (Universal Resource Locator) bezeichnet oder umgangssprachlich als *Link*. Um diesen Relationen zwischen den

Ressourcen einen Sinn zu verleihen, müssen die Zusammenhänge spezifiziert werden, damit sie für ein System bzw. eine Maschine genauso verständlich sind wie für einen Menschen. Aus diesem Grund wird ein wissensbasiertes Beschreibungsformat benutzt, welches sowohl die Relationen als auch die Ressourcen beschreibt. Dazu zählen RDF oder auch OWL, die als Grundlage des Semantischen Web dienen.

Ontologien und Wissensmodellierungen

Eine Ontologie definiert ein abstraktes Modell von Begriffen, die gemeinsam von Systemen und Menschen verstanden und interpretiert werden können. Diese Begriffe (auch Konzepte genannt) und die damit verbundenen Relationen beschreiben ein „Wissen" über eine festgegebene Domäne oder einen Anwendungsbereich. Somit ist es möglich, Zusammenhänge zwischen Konzepten festzustellen und über eine Beschreibungslogik (engl. Description Logic oder kurz DL), diese besser aufzufinden und Schlussfolgerungen zu ziehen. Dies bezeichnet man als *Reasoning*. Ontologien und Wissensdatenbanken sind Bestandteile des Semantic Webs, wobei ein Unterschied zwischen den beiden existiert. Eine Ontologie kann mit dem aus der Beschreibungslogik definierten Begriff *T-Box – Terminologische Axiome* gleichgestellt werden. Die T-Box beinhaltet Terminologien, Vokabulare und Konzepte und definiert die Ontologie. Die T-Box kann von Reasoning-Verfahren benutzt werden. Die A-Box (Assertionen Axiomen) entspricht den Daten oder Instanzen [May06]. Eine Wissensdatenbank (engl. Knowledge Base) wird von einer A-Box (eigentlich Fakten über Instanzen) definiert, wobei eine Wissensdatenbank logischerweise immer eine Referenz zu einer Ontologie bzw. T-Box besitzt. Die A-Box kann zur Laufzeit mit

beliebigen Instanzen erweitert werden, wobei die T-Box oft statisch bleibt. Eine Wissensdatenbank ist also die Zusammenstellung von einer T-Box mit einer A-Box (siehe Abbildung 2.2).

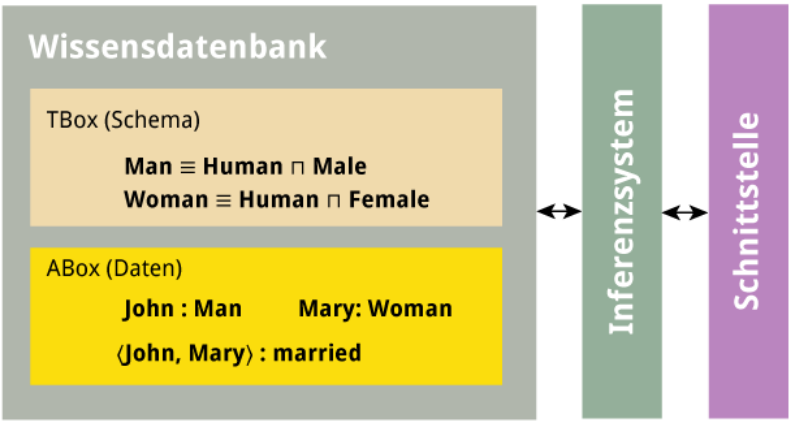


Abbildung 2.2: Aufteilung der T-Box und A-Box bei einer Wissensdatenbank. Quelle: Grafisch adaptiert von http://kde.cs.uni-kassel.de/conf/iccs05/horrocks_iccs05.pdf

In [MSS+03] wird eine Wissensdatenbank als Sammlung von Objektbeschreibungen, die zu einer gegebenen Ontologie referenziert, beschrieben. Um das Schließen von neuen Aussagen zu ermöglichen, werden Reasoning-Verfahren benutzt. Reasoning bedeutet *„das Ziehen von vernünftigen, begründeten, nachvollziehbaren Schlüssen“* [May06]. Diese Schlüsse dienen wiederum als Grundlage für die Überprüfung weiterer Schlussfolgerungen. Reasoners können auf Wissen von der T-Box und der A-Box beruhen und benutzen die Prädikatenlogik, Beschreibungslogik aber auch Inferenzen, um die Gültigkeit einer Aussage und Ihrer Vollständigkeit zu überprüfen. Beispiele von Reasoners werden im Abschnitt über OWL gegeben.

RDF

Das Akronym RDF steht für *Resource Description Framework*. Eine Ressource ist eine eindeutig definierte Entität, die im Web wieder auffindbar ist, z.B. über eine URL. Damit jeder Bezeichner und jede Ressource (Dokument, Datei oder Metadaten) eindeutig wiedergefunden werden können, benutzt man eine URI-ähnliche Notation. RDF erlaubt auch eine Form der Reifikation [May06]. Dies bedeutet, dass Graphen bzw. Triples in andere Graphen integriert und verkettet werden und somit eine „Aussage über eine Aussage" bilden können. Die Verschachtelung kann wiederum als Knoten für weitere Graphen benutzt werden. RDF-Dokumente stützen sich für die Beschreibung der Bezeichner auf die XML-Syntax. Somit lassen sich RDF-Dokumente serialisieren (eine Umwandlung des Formats in ein einfaches sequenzielles Format, wie z.B. XML oder JSON) und sind für die Benutzung in Zusammenhang mit REST-Aufrufen (Representational State Transfer - eine vereinfachte Realisierung von Datenaustausch innerhalb des Webs) [Fie00]) gut geeignet. Die abstrakte visuelle Darstellung von RDF und Relationen werden mittels einer Darstellung mit einem Graph realisiert. Dieser gibt einen besseren Überblick über die Verknüpfung als eine rein textuelle Dokumentstruktur. Diese Graphen werden in drei Elemente (Tripel) formuliert: *Subjekt*, *Prädikat*, *Objekt*. In folgendem Beispiel ist dem Subjekt „Steven Spielberg" das Prädikat „regisseur_von" und das Objekt dem Film „Jurassic Park" zugeordnet.

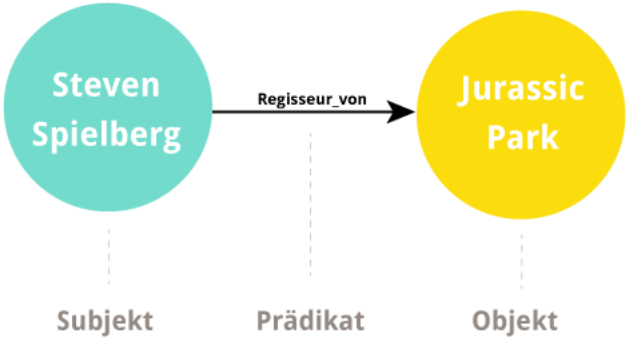


Abbildung 2.3: Subjekt, Prädikat und Objekt-Relation

Damit Relationen zwischen Konzepten und Ressourcen nicht willkürlich neu erschaffen werden müssen, muss ein sogenanntes Vokabular benutzt werden, das vorsieht, welche neue Relationen oder welche Konzepte vererbt werden können. Diese Definition von erlaubten Relationen und gültigen Konzepten wird in RDFS beschrieben. RDFS steht für RDF-Schema und ermöglicht die Einführung neuer Klassen und Eigenschaften zu RDF, sodass auch Subklassen sowie die Definitionen von Werten und Gültigkeitsbereichen entstehen können. Neue Vokabulare können so definiert werden, wie z.B. *hasDirected* oder *FilmDirector*. Dafür wurden die Klassen *rdfs:Resource*, *rdfs:Class* und Prädikate, wie *rdfs:domain*, *rdfs:subClassOf* oder *rdfs:range*, eingeführt [Bir06]. Die Abbildung 2.4 aus der RDF Spezifikation vom W3C[2] stellt den Aufbau einer Klassenhierarchie mittels RDFS dar. Hier werden mittels der Prädikate *rdfs:subClassOf* drei neue Unterklassen von *MotorVehicle* definiert.

[2] `https://www.w3.org/TR/2000/CR-rdf-schema-20000327`

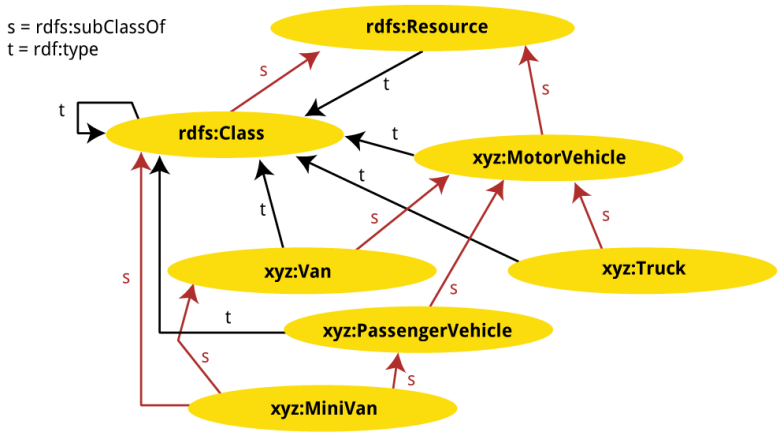


Abbildung 2.4: Klassenhierarchie Quelle: Adaptiert aus der W3C RDF Spezifikation 1.0 (https://www.w3.org/TR/2000/CR-rdf-schema-20000327)

Die entsprechende Ausprägung dieser Klassenhierarchie wird wie folgt in RDF/XML repräsentiert:

Listing 2.1: Serialisierung der Klassenhierarchie in RDF

```
<rdf:RDF xml:lang="en"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<!-- Note: this RDF schema would typically be used in RDF instance
     data
     by referencing it with an XML namespace declaration, for
         example
     xmlns:xyz="http://www.w3.org/2000/03/example/vehicles#".
         This allows
     us to use abbreviations such as xyz:MotorVehicle to refer
     unambiguously to the RDF class 'MotorVehicle'. -->

<rdf:Description ID="MotorVehicle">
```

```
 <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
 <rdfs:subClassOf
  rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdf:Description>

<rdf:Description ID="PassengerVehicle">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
 <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>

<rdf:Description ID="Truck">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
 <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>

<rdf:Description ID="Van">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
 <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
</rdf:Description>

<rdf:Description ID="MiniVan">
 <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
 <rdfs:subClassOf rdf:resource="#Van"/>
 <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
</rdf:Description>

</rdf:RDF>
```

Folgende Abbildung 2.5 verdeutlicht die klare Trennung der Schema- und Datenebene. Die spezifischen Instanzen der Konzepte werden auf Datenebene definiert (RDF/XML), wobei die Klassen im RDF- Schema ein Vokabular definieren, wie z.B. *FilmRegisseur* oder *Film*. Die Relationen zwischen diesen Klassen können durch Eigenschaften, wie z.B. *Regisseur_von*, definiert werden.

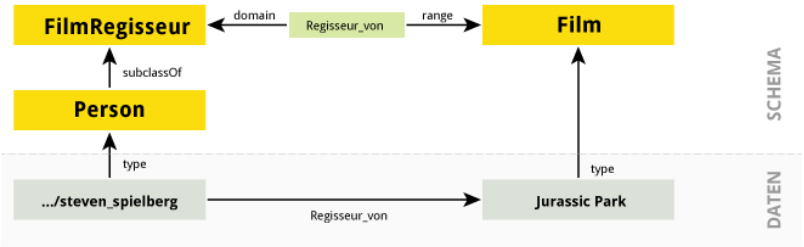


Abbildung 2.5: Schema- und Datenebene

Ein RDF/S-Schema kann somit in drei verschiedene Abstraktionsebenen kategorisiert werden [BKVH03]:

1. Die *syntaktische Ebene*: hier sind diese Schemata XML-basierte Dokumente
2. Die *strukturelle Ebene*: hier sind die Schemata Sets von Triples
3. Auf der *semantischen Ebene* bilden sie ein oder mehrere Graphen mit einer teilweise definierten Semantik

Bei der syntaktischen Ebene können komplexe Anfragen nur sehr umständlich formuliert werden. Da die XML- Syntax für RDF nicht eindeutig ist (verschiedene Konzepte oder Relationen können unterschiedlich modelliert werden), wäre es nötig, bevor eine tatsächliche Anfrage formuliert wird, die Gesamtheit eines RDF-Dokuments zu analysieren, um jede einzelne Relation zu extrahieren. Dies lässt sich dadurch erklären, dass innerhalb des RDF-Dokuments nicht über Knoten von Bäumen gesucht wird, da das RDF-Modell ein Graph sein kann, mit verschiedenen Ebenen oder Verschachtelung der Relationen. Auf der strukturellen Ebene besteht die Gefahr, dass jeder RDF nur als Triple betrachtet wird. Das bedeutet, dass bestimmte Relationen nicht

direkt erkannt werden, wenn diese nicht explizit als Triple modelliert worden sind. Die Relation kann dann nur nach Überprüfung der Prädikate, wie z.B. *rdfs:subclassOf*, Aufschluss geben, ob es sich um eine Subklasse handelt und somit eine mögliche Relation bestätigen. Wenn z.B. Steven Spielberg, modelliert als Triple vom Typ *director* ist und *director* eine Subklasse von *person* ist, kann auf strukturelle bzw. Triple-Ebene nicht direkt Rückschluss gezogen werden, dass Steven Spielberg eine Person ist, weil die Beziehung *Steven Spielberg isA person* nicht explizit als Triple definiert war. Dieses Beispiel zeigt, dass komplexere Anfragen nur auf semantischer Ebene realisiert werden können, z.B. mit einer Abfragesprache, die den kompletten Graph betrachtet, wie SPARQL[3].

RDFa

Die Serialisierung von RDF zu einem lesbaren XML-Dokument wird meistens als RDFa[4] bezeichnet. Dadurch kann gewährleistet werden, dass die Begriffe und das Vokabular in den XHTML-Code der Webseite hinzugefügt und trotzdem weiterhin von Entwicklern und Maschinen benutzt werden können, z.B. in HTML-Seiten oder DOM-Elementen. Dieser Ansatz wird sehr oft mit Microformats verglichen, da hier die ähnliche Einbettung von Attributen innerhalb von Tags stattfindet. Das angewandte Vokabular kann erweitert werden. Obwohl RDFa eine Unterstützung bei der Semantifizierung der Daten und der Tags in einem XHTML-Dokument erleichtert, ermöglicht diese nur eine beschränkte Anzahl von Auszeichungssprachen (engl. Markup).

[3] `https://www.w3.org/TR/rdf-sparql-query`
[4] `https://rdfa.info`

RDFa Lite

RDFa Lite 1.1 [W3C12c] ist eine vereinfachte Form von RDFa mit dem Fokus auf Einsetzbarkeit durch Webentwickler. Obwohl die Philosophie der Integration von Attributen bei XHTML-Tags geblieben ist, wurden bei RDFa diese Attribute auf fünf reduziert (*vocab, typeof, property, resource* und *prefix*). Als Vokabular wird meistens das von Schema.org vorgeschriebene benutzt.

OWL

Die Sprache OWL (Web Ontology Language) ist eine weitere Ausprägung von RDFS und wurde vom W3C verabschiedet, um Ontologien für das Semantic Web besser definieren zu können [PH05] [SET09]. OWL bietet ein ausführliches Vokabular, um Eigenschaften und Klassen zu beschreiben und benutzt u.a. RDF. Rückschlüsse können mittels Reasoning auf OWL-Daten durchgeführt werden[Hor05]. OWL untergliedert sich in drei Unterkategorien:

OWL Lite erlaubt nur die Verwendung von einfachen Kardinalitäten (z.B. 0,1) in Zusammenhang mit festgegebenen Bedingungen (engl. Constraints). OWL Lite wird oft für das Generieren von Taxonomien oder Klassifikationen benutzt.

OWL DL (Description Logic) bietet zusätzlich zu den Funktionalitäten von OWL Lite die Möglichkeit über Beschreibungslogik Klassenrelationen zu bestimmen. Zusätzliche „Constraints“, wie z.B. eine maximale Rechenzeit, können angegeben werden. Bei OWL DL wird immer gewährleistet, dass der Reasoner anhalten und ein gültiges Ergebnis zurückliefern wird. [Web10]

OWL Full wurde dafür ausgelegt, eine präzise und komplette semantische Beschreibung der RDF-Graphen zu gewährleisten, was dazu führen kann, dass bestimmte nicht- entscheidbare (engl. undecidable) Probleme oder Halteprobleme auftreten können. In OWL Full werden nur die Logik der Ausdrücke bzw. Relation überprüft.

Durch OWL können nicht nur Relationen zwischen Klassen bzw. Konzepte und Entitäten ausprägungsvoller ausgedruckt werden, sondern auch logische Zusammenhänge, z.B. durch Kardinalitäten oder Restriktionen[5] [Lov07] [Pow09]. Im Dezember 2009 wurde vom W3C die OWL 2-Initiative ins Leben gerufen, um bestimmte Einschränkungen und manche mit OWL 1 nicht realisierbare domänenspezifische Relationen — diese spielen z.B. im medizinischen Bereich (bei Inferenzen oder der Vererbung von Konzepten bei Symptomenbeschreibung) eine wichtige Rolle — ausdrücken zu können. Ausführlichere Details und Anwendungsgebiete von OWL 1 und OWL 2 können in [HKR09] und [W3C12a] nachgelesen werden. Ähnlich wie bei OWL 1 gibt es bei OWL 2 drei verschiedene Ausprägungen bzw. Profile:

OWL2 EL beruht auf der leichtgewichtigeren Beschreibungslogiksprache EL++[6]. OWL 2 EL wird z.B. bei der medizinischen Ontologie SNOMED CT [Don06] benutzt und liefert in weniger als einer Minute eine komplette Klassifikation von 10^5 medizinischen Datensätzen (Klassen und Eigenschaften) [HKR09].

OWL 2 QL basiert auf der Beschreibungssprache DL-Lite und wird meistens bei Applikationen, die sehr viele Abfragen und Datenmengen analysieren müssen, benutzt. Bei OWL 2 QL können

[5] http://www.linkeddatatools.com/introducing-rdfs-owl
[6] https://www.w3.org/2007/OWL/wiki/EL

die Abfragen mittels der Abfragesprache SQL (Structured Query Language) durchgeführt werden [Chn13].

OWL 2 RL ermöglicht über sog. Regeln (engl. *Rules* RL) das Reasoning schneller durchzuführen [W3C09]. Dieses Profil dient dazu, eine erweiterte Beschreibung und Ausprägungen von RDF(s)-Graphen zu realisieren [W3C12b].

OWL bildet die Grundlage für Reasoners. Diese benutzen die mittels OWL definierte Ontologie, um Schlussfolgerungen durchzuführen bzw. um Aussagen mittels einer Beschreibungslogik zu überprüfen. Aktuell existieren mehrere OWL-Reasoner, darunter: FaCT++ [TH06], JFact (eine Java-Version von FaCT++)[7], HermiT [MGS08] [8] [HMW12] [GHM+14], Pellet [SPG+07] [9] und Racer [HHMW12].

Semantische Informationen im Web

Folgender Abschnitt beschreibt das Konzept der Metadaten und Vokabulare, die benutzt werden können, um Medienobjekte im Web auffindbar zu machen. Um dies zu realisieren, müssen Zusatzinformationen zu diesen Medienobjekten hinzugefügt werden. Im folgenden Abschnitt werden zuerst die Begriffe *Metadaten* und *Vokabulare* erläutert und in einem weiteren Schritt ihre Benutzung im Semantic Web.

[7] `http://jfact.sourceforge.net`
[8] `http://www.hermit-reasoner.com/publications.html`
[9] `https://github.com/Complexible/pellet`

Metadaten

Unter Metadaten versteht man wortwörtlich „Daten über Daten" [Ros04] oder anders formuliert, die Beschreibung des Inhalts eines Dokumentes oder Medienobjektes. Diese Beschreibung kann entweder die Form eines assoziierten Eintrags (z.B. der Name des Autors) annehmen aber auch mittels vorgefertigter Tags (z.B. Bild, Berg, außen...) oder Wörtern (z.B. Berge, See, sonnig, Bodensee...) definiert werden. Diese Metadaten können auf verschiedener Ebene verwaltet und erweitert werden. Eine andere Definition von Metadaten findet sich in [BI98]: *„eine beschreibende Information, die zur Indizierung, Zusammenführung, Klassifizierung und Verbesserung der Ressourcenzugriffe einer Bibliothek oder eines Museums dient"*. Metadaten bilden das fundamentale Element für die Beschreibung oder Klassifikation von Medien, elektronischen Dateien und Gegenständen (wie z.B. Bücher).

In [DHSW02] werden die verschiedenen Metadaten-Typen in jeweils unterschiedliche Kategorien eingebunden:

Eingebettete Metadaten Bei den eingebetteten Metadaten werden Zusatzinformationen vom Entwickler oder von einer Applikation zu einer Datei hinzugefügt. Die Informationen bleiben immer mit der Datei konsistent und erst nach Zugriff einer Applikation oder des Betriebssystems können diese verändert werden. Die Zugriffe auf die eingebetteten Metadaten sind nicht destruktiv und können erweitert werden. Digital Asset und Media Asset Management-Systeme benutzen eingebettete Metadaten, um u.a. eine erweiterte Suche zu ermöglichen. Dabei werden die Metadaten der jeweiligen Dateien gescannt und auf Schlüsselwörter (engl. Keywords) überprüft. Oft werden eingebettete Metadaten

aus verschiedenen Quellen zu einer Datei hinzugefügt. Ron Roszkiewicz [Ros04] unterscheidet dabei verschiedene Ebenen der Verwendung dieser Metadaten:

- Auf Betriebssystemebene bei der Suche von Dokumenten auf Festplatten (wie z.B. beim Spotlight in MacOS X)
- Auf Applikationsebene: eine Software produziert Metadaten als „Nebenprodukt" (wie z.B. bei Word, bei dem der Benutzername des Dokumenterstellers, die Seitenanzahl oder Veränderungsdatum automatisch mitgespeichert werden)
- In Zusammenhang mit Graphiken und Content Assets, z.B. für Content Management Systeme

Assoziierte Metadaten Diese Metadaten werden nicht direkt in die Datei gespeichert, sondern werden extern von einem Server aus verwaltet und gespeichert. Die Metadaten sind für externe Websuchmaschinen nicht sichtbar. Um dies zu umgehen, ermöglichen z.B. neue Protokolle (Open Archives Initiative Protocol Metadata Harvesting Protocol[10] – kurz OAI-PMH) das Suchen und Finden dieser Daten. Dabei muss die Problematik des Schutzes von Daten berücksichtigt werden und es sollte insbesondere bei Copyright-geschützten Dateien darauf geachtet werden, welche Drittanwender diese letztendlich benutzen dürfen.

Third Party – Metadaten von Dritten Diese Art von Metadaten wird von externen Applikationen über extra dedizierte Webdienste bereitgestellt. Die Absicht bei der Offenlegung dieser Metadaten ist es, eine bessere Sichtbarkeit bei Suchmaschinen zu erzielen, aber auch Entwicklern die Möglichkeit zu geben, Informationen

[10] `http://www.openarchives.org/OAI/openarchivesprotocol.html`

mit existierenden Webseiten zu kombinieren und somit neue soziale netzwerk-basierte Dienste anzubieten, wie z.B. das Finden eines neuen Freundeskreises anhand der Musikrichtung oder anderer Themengebiete.

Sehr oft limitiert sich die Verwendung dieser Metadaten auf Webseiten, die sehr viel Multimediainhalte anbieten, wie z.B. Blogs, und leidet darunter, dass die Metadaten nur über externe Server verfügbar und aufrufbar sind. Hierbei spielt die Datenfreigabepolitik des jeweiligen Anbieters eine wichtige Rolle. Konkret bedeutet dies, dass, wenn z.B. eine Suchmaschine bestimmte Daten löscht oder die internen Suchalgorithmen anders gestaltet, Drittanwender (oder sog. "Third Party Metadaten"-Konsumenten), z.B. bei einer Webseite, nicht mehr auf die gewünschten Informationen zugreifen können. In [For13] wird das Vorgehen der Integration der Third Party Metadaten von Twitter[11] und Facebook (mit dem OpenGraph) in bestehenden Webseiten dargestellt. Beide Anbieter, Twitter und Facebook, benutzen das Dublin Core Vokabular [W3C16b], um die Inhalte eines Tweets oder Facebook-Eintrags zu beschreiben.

Automatische Metadaten Diese Metadatentypen werden sehr oft von Programmen oder Betriebssystemen ohne Einwirken des Benutzers automatisch generiert und zu Dateien hinzugefügt. Die meisten Bild- und Textverarbeitungsprogramme fügen z.B. den Namen des Autors, die Anzahl der Wörter oder die Auflösung der Bilder in die bearbeitete Datei automatisch ein. Bei diesen Programmen wird dem Benutzer nur beschränkt die Möglichkeit eingeräumt, diese Metadaten manuell zu verändern. In

[11] `https://dev.twitter.com/docs/cards`

[DSW06] werden folgende Prinzipien vorgestellt, die bei der automatischen Generierung von Metadaten eingehalten werden müssen, u.a. damit diese Metadaten zugreifbar und nachträglich verwendbar bleiben:

- Lizenz für die Metadaten (Open Source / Public Licence)
- Metadaten müssen für Software oder Apps lesbar und veränderbar sein
- Falls Metadaten nicht direkt in einer Datei eingebettet sind, sollten diese separat in eine lesbare Datei definiert werden oder alternativ über eine definierte URL oder Schnittstelle zur Verfügung gestellt werden.
- Wenn möglich, können Metadaten auf andere intern oder extern gespeicherte Daten verweisen

Listing 2.2: Beispiel einer Twitter-Card

```
<meta name="twitter:card" content="summary">
<meta name="twitter:site" content="@iMatD">
<meta name="twitter:creator" content="@iMatD">
<meta name="twitter:title" content="Android min3d quick tip:
    Changing textures of 3DS and OBJ 3D models at runtime">
<meta name="twitter:description" content="In the previous
    tutorials about min3D for Android we took a closer look on
     how to optimize the work flow from Blender or Daz3D to
    min3D.">
<meta name="twitter:image" content="http://www.mat-d.com/site/
    wp-content/uploads/min3d_change_textures.jpg">
```

Bei der Beschreibung der Metadaten muss gewährleistet werden, dass die Medienobjekte ausreichend und korrekt beschrieben worden sind.

Bestimmte Felder, wie z.B. eine Zusammenfassung oder Kurzbeschreibung eines Bildes oder Videos, können manchmal fehlen. Bei der Klassifizierung dieser Datei muss beachtet werden, dass diese nicht unter eine subjektive Anordnung fallen. Wenn z.B. Metadaten zu einem Bild hinzugefügt werden, muss gewährleistet werden, dass diese und die dazugehörige Beschreibung mit dem, was auf dem Bild zu sehen ist, übereinstimmen. Dieser Aspekt ist nicht zu unterschätzen, besonders wenn es sich um Buchkritiken, journalistische Werke oder Zusammenfassungen von Reportagen handelt. Oft werden Metadaten in sog. Content- und Asset-Management-Systeme zu ungenau eingepflegt, was dazu führen kann, dass bestimmte Medien nicht mehr im Datenpool gefunden werden können. In den folgenden Abschnitten werden verschiedene Metadatenformate samt Ausprägungsformen vorgestellt. Zusätzlich wird definiert, in welchem Multimediabereich diese für eine semantische Annotation eingesetzt werden können.

Microdata

Das W3C beschreibt Microdata wie folgt [W3C13a]: „Microdata ermöglicht das Einbetten von maschinenlesbaren Daten in HTML-Dokumente, sodass diese sowohl von Suchmaschinen als auch von Menschen besser lesbar sind“. Dabei sind Microdata kompatibel zu anderen Datenformaten, wie z.B. RDF und JSON. Technisch bedeutet dies, dass jedes HTML-Tag innerhalb eines HTML-Dokuments mit semantischen Informationen annotiert werden kann. Dadurch ist es möglich, über die Verwendung der Attribute *itemscope /itemtype* und *itemprop* innerhalb eines HTML-Tags festgelegte Vokabulare zu definieren. Dieses Vokabular verleiht den eigentlichen Sinn des Tags bezogen auf die jeweiligen Key-Value Paare, was sich wie folgt skizzieren lässt:

Listing 2.3: Beispiel von Microdata in einem HTML-Dokument

```
<section itemscope itemtype="http://data-vocabulary.org/Person">
  <h1 itemprop="name">Matthieu</h1>
  <p><img itemprop="photo" src="http://www.mat-d.com/site/about/
      matd.jpg"></p>
  <p><a itemprop="url" href="http://www.mat-d.com/">MyWebpage</a
      ></p>
</section>
```

itemscope und *itemtype* definieren hier, dass der Abschnitt (*section*) eine Person definieren wird. Das Attribut *itemprop* (Eigenschaft) definiert, dass der Inhalt der Tags einen Personennamen beinhalten wird, versehen mit einem Foto und einer definierten URL. Ähnlich wie bei den anderen Vokabularen existieren auch Werkzeuge, wie z.B. der Microdatengenerator[12] [W3C13a], welche die Benutzung und Einbettung dieser in HTML- Dokumente für Webentwickler erleichtern sollen.

Microformats

Microformats ermöglichen genau wie Microdata bestimmte Informationen über Personen, Produkte oder Geokoordinaten als Erweiterung des Basis-HTML-Codes hinzuzufügen. Diese zusätzlichen Informationen können von Suchmaschinen benutzt werden, um ihre Indexierung zu verfeinern[13].

Bei Microformats stehen die Einfachheit der Benutzung und die direkte Integration in HTML in Vordergrund. Um dies zu realisieren, wird innerhalb von HTML-Tags ein *class* Attribut hinzugefügt, das die Beschreibung des jeweiligen Eintrages definiert.

[12] `http://www.microdatagenerator.com`
[13] `http://microformats.org/wiki/Main_Page`

Folgender Code zeigt die Ausprägung von Microformats innerhalb eines HTML-Dokuments.

Listing 2.4: Microformats in einem HTML-Dokument

```
<div id="hcard-Matthieu" class="vcard">
 <a class="url fn n" href="http://www.mat-d.com">
   <span class="given-name">Matthieu</span><span class="family-
       name">Deru</span></a>
 <div class="org">DFKI</div>
 <div class="adr">
  <div class="street-address">Stuhlsatzenhausweg</div>
  <span class="locality">SAARBRUECKEN</span>
  <span class="postal-code">66123</span>
  <span class="country-name">Germany</span>
 </div>
</div>
```

Hierbei liefern z.B. die Attribute *fn* (Formated Name) oder *org* Zusatzinformationen zu den jeweiligen *div*- und *span*-Elementen. Durch die Verwendung von Microformats in Kombination mit HTML können sog. vCard und vcalendar für Web-basierte Visitenkarten und Eventlisten generiert und diese mit zusätzlicher Semantik versehen werden, ohne dabei das grafische Aussehen der Seite zu verändern. Weiterführende Informationen zur Integration von Microformats in HTML-Dokumente können detailliert in [Daw10] und [SET09] nachgelesen werden.

Dublin Core Metadata Initiative (DCMI)

Der Dublin Core (kurz DC) Standard wurde von der Initiative *„Dublin Core Metadata Initiative“* etabliert und versucht seit 1995 Beschreibungen, Vokabulare und Metadaten-Modelle für Interoperabilität zu etablieren, um Ressourcen korrekt zu beschreiben, sodass diese von

einem intelligenten Suchsystem benutzt werden können. Der Dublin Core wurde 2006 zum ISO 15836 Standard. Ziel dieser Initiative ist es, Videos, Bilder oder Texte im Internet mittels der gespeicherten Metadaten besser auffindbar zu machen.
Mittlerweile wird das vom Dublin Core vorgeschlagene Vokabular und dessen Beschreibungen in Applikationen benutzt, bei denen Metadaten eine zentrale Rolle spielen, wie z.B. bei Adobe Photoshop (siehe Abbildung 2.6) und Adobe Bridge. Diese Informationen werden auf der Betriebssystemebene verwendet und unterstützt. Ein weiteres Merkmal der Dublin Core Initiative ist, dass verschiedene Vokabulare für unterschiedliche Domänen (z.B. Medizin, Photographie und Finanzwesen) angeboten werden. Dublin Core spezifiziert fünfzehn sogenannte Eigenschaften (*contributor, coverage, creator, date, description, format, identifier, language, publisher, relation, rights, source, subject, title* und *type*) oder *DCMI Metadata Terms*[14,15]. Diese werden für die Beschreibung von Ressourcen benutzt und darunter findet man die Möglichkeit, die Angaben zu den Autoren (*dc:contributor* oder *dc:creator*), das Erstellungsdatum von Dokumenten (*dc:date*), ihre Sprache (*dc:language*) oder eine Beschreibung näher zu präzisieren. DCMI Metadata Terms können für die Erhaltung von Urheberrechtsinformationen im Kontext Copyrightmanagement benutzt werden.
Ebenso wie bei Microdata oder anderen Vokabularen können die Dublin Core Metadaten in HTML/XHTML-Dokumente eingebettet werden (siehe Quelltext 2.5). Dabei wird vorgeschlagen, dass ein separates RDF- bzw. XML- Dokument, das über das Element <*link*> zum HTML-Dokument verlinkt wird, benutzt werden soll[16] [Pow03]. Zusätzlich

[14] `http://dublincore.org/documents/dcmi-terms`
[15] `http://dublincore.org/documents/dcmes-qualifiers`
[16] `http://de.selfhtml.org/html/kopfdaten/meta.htm{#}dublin_core`

können Dublin Core Metadaten direkt in RDF eingebettet werden. Diese Einbettung ermöglicht eine Grundkompatibilität zu anderen Vokabularen, wie z.B. vCard [DRS02].

Listing 2.5: Dublin Core innerhalb eines HTML-Dokuments

```
<head profile="http://dublincore.org/documents/dcq-html/">
<title>Expressing Dublin Core in HTML/XHTML meta and link elements
    </title>
<link rel="schema.DC" href="http://purl.org/dc/elements/1.1/" />
<link rel="schema.DCTERMS" href="http://purl.org/dc/terms/" />
<meta name="DC.title" lang="en" content="Expressing Dublin Core
in HTML/XHTML meta and link elements" />
<meta name="DC.creator" content="Andy Powell, UKOLN, University of
     Bath" />
<meta name="DCTERMS.issued" scheme="DCTERMS.W3CDTF" content
    ="2003-11-01" />
<meta name="DC.identifier" scheme="DCTERMS.URI"
content="http://dublincore.org/documents/dcq-html/" />
<link rel="DCTERMS.replaces" hreflang="en"
href="http://dublincore.org/documents/2000/08/15/dcq-html/" />
<meta name="DCTERMS.abstract" content="This document describes how
qualified Dublin Core metadata can be encoded
in HTML/XHTML &lt;meta&gt; elements" />
<meta name="DC.format" scheme="DCTERMS.IMT" content="text/html" />
<meta name="DC.type" scheme="DCTERMS.DCMIType" content="Text" />
</head>
```

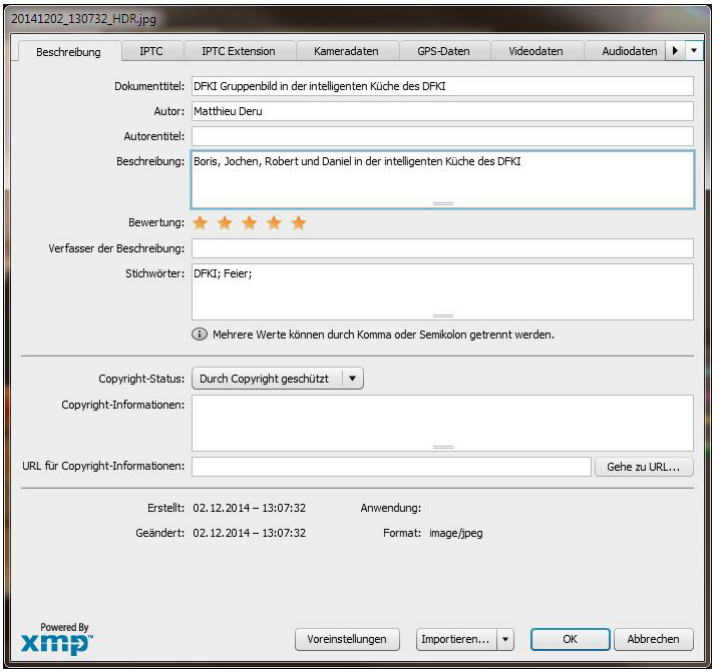


Abbildung 2.6: DCMI-Eingabepanel in Adobe Photoshop

EXIF

Das EXIF (engl. *Exchangeable Image File Format*) ermöglicht parallel zu den eigentlichen JPEG- oder TIFF-Bilddaten, zusätzliche Informationen, wie z.B. Datum, Brennweite, GPS-Koordinaten, im Header einer Bilddatei zu speichern. Diese Zusatzinformationen können entweder direkt von der Kamera (im Gegensatz zu IPT/XMP-Informationen, die manuell eingetragen werden müssen) selbst hinzugefügt werden (z.B. Blendeneinstellung, ISO-Werte, Farbtiefe, Marke des Gerätes usw.) oder nachträglich von einer mobilen App, Software (z.B. GIMP oder iPhoto) oder im letzten Falle direkt vom Betriebssystem hinzugefügt werden. Mittels der mitgespeicherten Metadaten lassen sich die GPS-Koordinaten

über den Aufnahmeort speichern, sodass es später möglich ist, die Bilder mittels einer Kartenanzeige wieder dem korrekten geografischen Ort zuzuordnen. Online-Foto-Austausch-Plattformen, wie z.B. Flickr, erlauben über API-Zugriffe[17] verschiedene EXIF-Daten aus online verfügbaren Flickr-Bildern zu extrahieren. Diese Information kann für eine erweiterte Suche (z.B. mit Geokoordinaten) genutzt werden. Diese Möglichkeit besteht bei Bildverwaltungsprogrammen wie z.B. Adobe Bridge. Gleichzeitig bietet EXIF die Möglichkeit, Informationen über Urheberrechte mit zu speichern. Es gibt in der EXIF-Spezifikation keine Möglichkeit, Regionen oder Teilbereiche eines Bildes zu annotieren [HNSS07].

IPTC-Information Interchange Model

Inhalte und Metadaten zu Medienobjekten (Bilder, Texte oder Videos) können durch Metadaten, die mittels des Information Interchange Models (kurz IPTC-IMM) beschrieben wurden, angereichert werden. Diese Medienobjekte können so mittels des IPTC-Models annotiert werden. Dieses Modell wurde 1990 vom Internal Press Telecommunication Council (IPTC) zum Standard gemacht und diente hauptsächlich zum Austausch von Daten zwischen Redaktionen und Journalisten. Es wurde anfangs als IPTC-NAA (Newspaper Association Of America) Information Interchange Model und später als *„IPTC Metadata“* bezeichnet. Das Modell wurde ein paar Jahre später für die Beschreibung und Annotationen von Fotos und Texten verwendet. Adobe Systems band diese Metadaten 1994 erstmals in digitale Bilddateien ein. Sie werden bis heute als *IPTC Headers* bezeichnet und können mit den gängigen Bildverarbeitungsprogrammen (z.B. Photoshop, Illustrator, Acrobat)

[17] `https://www.flickr.com/services/api/flickr.photos.getExif.html`

oder einer Medien-Management-Software (z.B. Adobe Bridge) bearbeitet und gespeichert werden. Bietet das Programm keine direkte Möglichkeit diese Metadaten zu integrieren, können Drittprogramme, wie z.B. Plugins, diese Metadaten in Bilddateien speichern [IPT10].
Der IPTC-Standard kann in zwei Kategorien unterteilt werden: Core und Photo Extension. Beide werden nur noch unter einer Bezeichnung vom IPTC selber geführt, dem sog. IPTC-Photometadata Standard. Auf der Webseite des IPTC[18] werden immer noch beide präsentiert. IPTC-Core beinhaltet die grundlegenden Metadaten und Vokabulare/-Begriffe, die benutzt werden, um ein Bild korrekt zu beschreiben und zu verwalten.
IPTC-Photo Extension ermöglicht die Einbettung zusätzlicher Metadaten innerhalb eines Bildes. Der Ersteller eines Mediums kann Angaben zu Copyrights und Bearbeitungsschritten textuell zu einer Bilddatei hinzufügen. Diese Veränderung bzw. das Einpflegen der Metadaten erfolgt über sog. Panels, die als Zusatzfenster in einer IPTC-kompatiblen Applikation, wie z.B. Adobe Creative Cloud[19] oder ACDSeePro[20], aktiviert werden können. Diese Panels unterscheiden zwischen IPTC-Core und IPTC-Extension. IPTC-Metadaten können innerhalb von JPEG-, TIFF-, JPEG2000- oder PNG-Bilddateien gespeichert werden. Bilddateien, die in PCX oder GIF gespeichert sind, können keine Metadaten beinhalten. Die gängigsten Bildbearbeitungsprogramme können beide IPTC-(Core und Photo Extension) Informationen lesen und speichern.

[18] `https://iptc.org/`
[19] `http://www.adobe.com/de/products/cs6.html`
[20] `http://www.acdsee.com/de/index`

XMP

Im Jahr 2001 wurde von Adobe Systems das XMP-Format (Extensible Metadata Platform Format) vorgestellt, das Felder und teilweise Eigenschaften des IMM-IPTC-Standards übernahm, jedoch mit einer XML-basierten Syntax. Somit wurde gewährleistet, dass das XMP-Format leicht erweiterbar ist und doch die Möglichkeit besteht, die IPTC-Header zu beinhalten, damit der Workflow nicht unterbrochen wird. XMP kann EXIF-Daten integrieren, jedoch keine binären Metadaten (kleine Vorschaubilder, sog. Thumbnails können jedoch trotz Einschränkung direkt in XMP mittels Base24 eingebunden werden).
In Zusammenarbeit mit dem IPTC wurde 2005 die „IPTC Core Schema for XMP“ — manchmal auch IPTC4XMP genannte — Spezifikation publiziert. Diese beschreibt das Einbetten von Metadaten in verschiedene Multimedia-Datei-Typen, wie z.B. JPEG, TIFF, PNG, HTML, PDF, SVG oder sogar Photoshop PSD. Wichtig ist dabei zu beachten, dass bei XMP zwar Felder oder Eigenschaften des IPTC-Fotostandards übernommen worden sind (IPTC4XMP), jedoch ist XMP eine Entwicklung, die primär von Adobe vorangetrieben wird. Das IPTC stellt sog. Panels für Adobe-Produkte zur Verfügung, damit neben XMP auch ältere IPTC-IMM Informationen mit der Datei mitgespeichert werden können [IPT05] [Ado04]. Auf Wunsch können Dritte über die C++, Java, Actionscript-basierte XMPCore API weitere Panels in Adobe Produkte integrieren, sodass andere Arten von Metadaten oder Vokabular direkt über das XMP-Format eingepflegt und innerhalb eines Mediums mitgespeichert werden können.
Wichtig bei der Weiterentwicklung des XMP-Formats ist, dass zusätzliche Informationen wie Urheberrechte während eines Transfers (vom PC zu einem Content-Management-System oder einer Applikation)

erhalten und jederzeit abrufbar bleiben. Jede Software von Adobe unterstützt zurzeit das XMP-Format und kann Dateien generieren bzw. speichern, die dieses Format inklusive Metadaten direkt integrieren. Alle von XMP verwendeten Tags basieren auf dem Dublin Core und IPTC-Vokabular, andere können pro Anwendungsbereich oder Domäne problemlos hinzugefügt werden.
Obwohl Adobe die Rechte an der XMP-Marke hat und die Spezifikation des Formats selbst festlegt, bleibt es ein offenes Format, das von anderen Softwareherstellern oder von Applikationen, wie z.B. Google Picasa, Microsoft Windows und ACDSee Pro, benutzt wird.

JSON-LD

In den letzten Jahren stellen immer mehr REST-Schnittstellen von Web-Diensten, wie Twitter, Facebook oder Wikidata, ihre Ergebnisse im JavaScript Object Notation-Format (kurz: JSON) zur Verfügung. JSON ist ein Format, „das für Menschen einfach zu lesen und zu schreiben und für Maschinen einfach zu parsen (Analysieren von Datenstrukturen) und zu generieren ist“[21]. Durch seine einfache Strukturierung kann dieses Format in verschiedenen Programmiersprachen benutzt und für das Speichern von größeren Datenmengen (z.B. in NoSQL-Datenbanken wie MongoDB[22]) eingesetzt werden. Die Kombination von REST-basierten Schnittstellen zur Abfrage von Webdiensten und die Rückgabe der Ergebnisse dieser Dienste als JSON-Format gehört aktuell zu den grundlegenden Mechanismen zur Entwicklung von Web-Applikationen. Diese werden auch als *„Restful Webservices“* bezeichnet [Rod08] [RAT11].

[21] `http://www.json.org/json-de.html`
[22] `https://www.mongodb.com/json-and-bson`

Als JSON für Linked Data oder kurz JSON-LD bezeichnet man eine Möglichkeit ein bestehendes Dokument (z.B. ein HTML-Dokument) mit Kontextinformationen, erweiterten Links zu Ressourcen, welche die Grundlagen des Linked Data-Konzepts bilden und einer semantischen Beschreibung anzureichern. Da JSON-LD ausdrucksstark genug ist, können auch weiterhin RDF-Konzepte benutzt werden [LG12].

Listing 2.6: Beispiel einer JSON- LD Definition in einem HTML-Dokument

```
<script type="application/ld+json">
    {
        "@context": "http://schema.org",
        "@type": "Person",
        "@id": "http://dbpedia.org/resource/Nelly_Furtado",
        "name": "Nelly Furtado",
        "gender": "Female",
        "birthPlace": "Canada",
        "birthDate": "1978-12-02"
    }
</script>
```

Somit wird eine Kompatibilität zu den bestehenden RDF-basierten Wissensdatenbanken ermöglicht und ein einfacherer Anschluss von Dokumenten zum Semantic Web gewährleistet. Semantische Webdienste, u.a. der Google Knowledge Graph (siehe nächsten Abschnitt) setzt JSON-LD als Ergebnisformat ein. JSON-DL wird von Google als „Markup für strukturierte Daten“ bezeichnet (engl. structured data markup), bei der Vokabulare, wie schema.org benutzt werden, um die verschiedenen Schlüssel-Wert-Paare zu definieren.

Die Benutzung eines vordefinierten Vokabulars wird mit dem Schlüsselwort *@context* angegeben [Lan14]. Der Quellcodeauszug 2.6 zeigt eine JSON-LD-Struktur, die innerhalb des Script-Tags eines HTML-

Dokuments integriert wurde. JSON-LD ist mit JSON kompatibel, was bedeutet, dass schon existierende JSON-Dokumente oder HTML-Dokumente mit semantischen Zusatzinformationen angereichert werden können, ohne die ursprüngliche Struktur oder die Tags (wie es bei RDFa oder Microformats der Fall ist) verändern zu müssen. Die Tatsache, dass aktuell der Knowledge Graph von Google JSON-DL als bevorzugtes Ergebnisformat fundiert, weist darauf hin, dass zukünftig andere semantische und Linked Data-Dienste dieses Format adoptieren werden. Insbesondere SEO (Suchmaschinenoptimierer) sehen durch die Einbettung von JSON-LD bessere Chancen Webseiten in der Google-Suchergebnisliste besser zu platzieren[23,24,25].

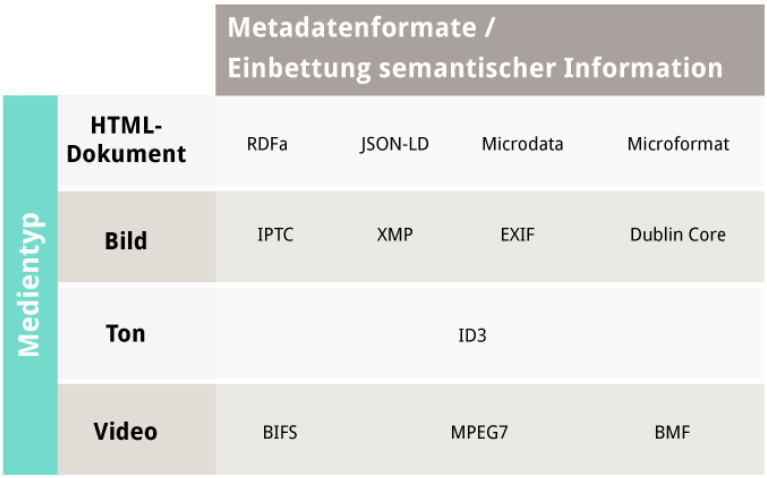


Abbildung 2.7: Metadaten für Medienobjekte

[23] http://www.whitespark.ca/blog/post/62-the-json-ld-markup-guide-to-local-business-schema
[24] http://manu.sporny.org/2013/json-ld-is-the-bees-knees
[25] https://blog.codeship.com/json-ld-building-meaningful-data-apis

Die Abbildung 2.7 fasst zusammen, welche Metadaten bzw. semantischen Informationen in die verschiedenen Medienobjekttypen eingebettet werden können. Die bereits vorgestellten Formate werden insbesondere für Medien wie Bilder oder Text-basierte Dokumente (z.B. HTML-Dokumente) eingesetzt. Bei Videos können komplexere Beschreibungsstrukturen wie das BMF (Broadcast Metadata Exchange Format)[26], das MPEG-7 [MSS02] oder das *BIFS* (Binary Format for scene description)[27] Format verwendet werden.

Vokabulare

Vokabulare sind bei der Beschreibung von Medienobjekten, insbesondere bei spezifischen Anwendungsdomänen, wichtig. Die Eigenschaften eines jeweiligen Objekts oder der Ressource und deren Beschreibung müssen einheitlich definiert werden, damit u.a. ein Austausch der Information ohne Wissensverlust möglich ist. Zudem sind einheitliche Vokabulare (auch „Schemes“ oder „Controlled Vocabulary“ genannt) von Bedeutung, um eindeutig Medien zu annotieren. Des Weiteren dienen Vokabulare dazu, einheitliche Begriffe zu spezifizieren.

Für die lokalen Annotationen, z.B. von Fotos oder Videos, können Vokabulare benutzt werden, die von den gängigsten Bildverarbeitungsprogrammen vorgeschlagen werden. Somit können die Medienobjekte innerhalb eines Produktionsworkflows besser annotiert werden. Die Annotationen können nachträglich dazu dienen, die Medien innerhalb einer Multimediadatenbank besser wiederzufinden. Als Alternative

[26] `https://www.irt.de/de/themengebiete/produktion/bmf.html`

[27] `http://mpeg.chiariglione.org/standards/mpeg-4/scene-description-and-application-engine`

können vordefinierte Vokabularsets benutzt und nachträglich als Erweiterung zu Anwendungen wie Adobe Bridge[28,29] heruntergeladen werden. Diese Vokabulare können mittels Online-Tools wie dem Polythematic Structured Subject Heading[30], dem System der Library Of Congress (CSH)[31], den Linked Open Vocabularies (LOV)[32] oder sogar dem Keyword Catalog[33] verfeinert werden. Damit wird gewährleistet, dass die Beschreibung eines Mediums einheitlich bleibt. Diese Vokabulare lassen sich online benutzen und erlauben eine einheitliche Beschreibung von Dokumenten.

Schema.org

Die Erweiterungen der Dokumente, u.a. mit RDFa und Microdata, zeigen, dass in HTML noch weitere Attribute oder (Attribut-Paare) mit einer Bedeutung (engl. „Role“) hinzugefügt werden können (siehe Quelltext 2.7). Bei der Beschreibung dieser Attribute muss ein bestimmtes Vokabular beachtet werden, damit dieses einheitlich und von verschiedenen Systemen richtig verwaltet (Relationen) werden kann. Um dies zu erreichen, haben 2011 die Suchmaschinen Bing, Google, Yahoo! und Yandex gemeinsame Bezeichner (engl. Markups) für eine bessere Anzeige der Suchergebnisse aber auch, um eine bessere Treffer- und Pertinenzquote bei der Suche zu erreichen, vereinbart [DM11]. Dazu können die bekannten HTML-Tags mit Attributen erweitert werden. Die Webseite von Schema.org[34] präsentiert die komplette

[28] `http://www.controlledvocabulary.com`
[29] `http://vimeo.com/35592159`
[30] `http://psh.ntkcz.cz/skos/en`
[31] `http://id.loc.gov`
[32] `http://lov.okfn.org`
[33] `http://keyword-catalog.com`
[34] `www.schema.org/docs/full.html`

Hierarchie der benutzbaren Begriffe und Vokabulare.

Listing 2.7: Angaben der Geokoordinaten von Paris mittels des Schema.org Vokabulars in einem HTML-Dokument

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head><meta http-equiv="Content-Type" content="text/html; charset=
    utf-8" />
<title>Paris</title>
</head>
<body>
<div itemscope itemtype="http://schema.org/Place">
  <a href="paris.html" itemprop="url"><div itemprop="name">Paris</
     div></a>
has following geocoordinates (WGS84) : 48.856638 , 2.352241
<div itemprop="geo" itemscope itemtype="http://schema.org/
    GeoCoordinates">
    <meta itemprop="latitude" content="48.856638" />
    <meta itemprop="longitude" content="2.352241" />
</div></div>
</body>
</html>
```

Das Hinzufügen weiterer Attribute in HTML ermöglicht Entwicklern eine leichtere Integration in schon vorhandene Webseiten und die Benutzung des Vokabulars für spezifisch gezielte Wissensdomänen, wie z.B. Medizin oder den E-Commerce- Bereich (z.B. mit dem Vokabular von GoodRelation). Somit kann gewährleistet werden, dass die generierten HTML-Dokumente Wissen und Kontextinformationen zur passenden Domäne integrieren und auf dieser Basis maschinell weiterverarbeiten. Ein weiteres Merkmal von Schema.org ist, dass dieses vorgeschlagene Vokabular innerhalb von RDFa Lite direkt referenziert und benutzt werden kann. Um Entwickler bei der Integration von RDF und der Benutzung von Schema.org Vokabularen zu unterstützen,

werden verschiedene Werkzeuge (z.B. in den Programmiersprachen Javascript oder PHP) angeboten[35]. Diese Werkzeuge haben als Ziel, die Vokabulare von Schema.org und Linked Data zusammenzufügen. Das Bestreben hinter dieser Initiative ist es, sog. Mapping Tools mit der Vokabularebene zu vereinheitlichen, damit eine bessere Verknüpfung und Suche von Relationen ermöglicht wird. Für Webentwickler und Content-Publisher stehen Online-Tools wie Schema-Creator[36] zur Verfügung, die über ein Webformular das Hinzufügen von Microdata für Bücher, Filme, Personen, Produkte, Events und Firmeneinträge mit dem Schema.org Vokabular ermöglichen. Somit können erweiterte Informationen direkt zu einer HTML-Seite hinzugefügt werden. Parallel dazu können Entwickler das Vokabular von Schema.org in HTML5-Dokumente einbetten und in Blogging-Plattformen wie Wordpress integrieren[37].

GoodRelations

Beim Vokabular von Schema.org lassen sich nicht alle Eigenschaften eines Produkts oder Gegenstandes ausführlich und präzise genug für eCommerce-Applikationen abbilden. Um dies zu verbessern, wurde die „GoodRelations Ontologie“ erstellt. Diese Initiative wurde im Rahmen des vom Bundesministerium für Bildung und Forschung geförderten Projekts „Intelligent Match“[38] gestartet und von Martin Hepp weitergeführt. GoodRelations definiert sich selbst als eine eCommerce-Erweiterung für Schema.org [Hep08]. Fokus der GoodRelations-Initiative ist primär für verschiedene Produktfamilien, Preise, Eigenschaften

[35] http://schema.rdfs.org/tools.html
[36] http://schema-creator.org/book.php
[37] http://journal.code4lib.org/articles/6400
[38] http://www.intelligent-match.de

und Händler/Herstellerangaben soweit zu spezifizieren, dass sog. Semantic SEO (Search Engine Optimizer - d.h. Suchmaschinenoptimierer) Produkte von Online-Shops besser gefunden und referenziert werden können. Dadurch werden Produkte auch für Crawler oder Preisvergleichsplattformen sichtbarer. Das GoodRelations-Vokabular kann über RDF/Microdata direkt in HTML5 integriert werden. Obwohl die Basisontologie von GoodRelations schon sehr ausspezifiziert ist, lässt sie sich dennoch erweitern. Dadurch können nicht nur Standardgüter besser definiert, sondern komplexere Güter und deren Subkomponenten, wie z.B. ein Auto, mit allen Ersatzteilen spezifiziert werden. Somit stellt GoodRelations neben der Standardontologie eine Ontologie für den Verkauf von Autos, Booten, Musik, oder einen Restaurant- oder touristischen Führer als Erweiterung zur Verfügung. Diese GoodRelations Ontologieerweiterungen werden pro Business-Anwendungsbereich untergliedert und als RDF zum Download[39] angeboten. Weitere ausführlichere Details zur Verwendung von GoodRelations als Vokabular findet man unter [HSB07].

Suchen und Finden im Web 3.0

Die vorherigen Abschnitte haben beschrieben, welche Zusatzinformationen zu Ressourcen, Medien oder sogar HTML-Dokumenten hinzugefügt werden können. Im folgenden Abschnitt wird das Konzept der semantischen Webdienste näher dargestellt und erklärt, mit welchen Mitteln Entwickler in Wissensdatenbanken suchen können.

[39] `http://www.productontology.org/`

Abfragesprachen

In diesem Abschnitt werden drei technische Möglichkeiten vorgestellt, Daten in Wissensdatenbanken zu finden und zu extrahieren. Bei MQL handelt es sich jedoch um eine heute nicht mehr verfügbare Abfragesprache. Diese wurde trotzdem in diesem Abschnitt berücksichtigt, da sie Grundlage für weitere neuere Methoden ist und sich der Zusammenhang zu den Diensten Google Knowledge Graph, Wikidata oder Freebase, die im nächsten Abschnitt näher vorgestellt werden, besser darstellen lässt.

SPARQL

Ähnlich wie bei den relationalen Datenbanken und der Sprache SQL (Simple Query Language) existieren innerhalb des semantischen Web Mechanismen, die Graphen und die damit zusammenhängenden Daten mittels Abfragesprachen suchen und finden können. Die bekannteste dieser Sprachen ist SPARQL. Die Syntax von SPARQL ist sehr nah an die von SQL angelehnt. Laut der W3C-Spezifikation [W3C13b] [W3C16a] bildet SPARQL die Standard-Abfragesprache für RDF. Der Name SPARQL steht als Akronym für *Simple Protocol And RDF Query Language* und ermöglicht die Suche in RDF-basierten Dokumenten bzw. Graphen.

Software- und Webbasierte-Schnittstellen, die einen Zugriff auf Wissensdatenbanken mittels der Abfragesprache SPARQL erlauben, werden als *SPARQL Endpoints* bezeichnet. Die in den nächsten Abschnitten vorgestellten semantischen Webdienste bieten alle eine solche Server-seitige SPARQL-Schnittstelle (engl. *Endpoint*) an. Somit kann gewährleistet werden, dass einheitliche Abfragen formuliert werden.

Dafür muss das Vokabular der jeweiligen Wissensdomäne im Vorfeld bekannt sein, damit die Terminologie und die zu suchenden Begriffe richtig aufgefunden werden können.

Ein besonderes Merkmal von SPARQL besteht darin, dass mittels einer einzigen Abfrage auf mehreren unabhängigen und heterogenen RDF-Datenquellen oder Graphen eine Suche realisiert werden kann. Die daraus resultierenden Ergebnisse werden kombiniert und als serialisierte Struktur (in XML, JSON, CSV, TSV oder als neuer RDF-Graph) ausgegeben [BP08].
Folgender Quellcode-Abschnitt 2.8 zeigt die Formulierung einer einfachen SPARQL-Abfrage (Query) über DBpedia, die eine Suche über die Sängerin *Nelly Furtado* durchführt.

Listing 2.8: SPARQL–Abfrage über DBpedia für die Sängerin Nelly Furtado

```
PREFIX d: <http://dbpedia.org/ontology/>
SELECT ?albumName, ?comment WHERE
{
     ?albumName d:artist :Nelly_Furtado.
     ?albumName rdfs:comment ?comment.
     FILTER (lang(?comment) = 'de')
}
```

MQL

Bei MQL handelte es sich um eine Abfragesprache für die Suche innerhalb der Freebase-Daten. Bei dieser Abfragesprache wurden die Abfragen in JSON-Strukturen gepackt und zur REST-Schnittstelle von Freebase gesendet. Die Syntax für die Abfragen war mit Absicht einfach gehalten worden. Trotzdem ließen sich komplexere Abfragen

mittels Operatoren wie *Not* und über sog. *Wildcards* durchführen. Zur Offlinenutzung bot Freebase eine Abbildung der Datenbank in Form einer Datei-RDF. Dieser Dump konnte mittels Werkzeugen wie Protegé[40] (einem kostenfreien Ontologie-Editor) geladen und per SPARQL abgefragt werden.

Besonders erwähnenswert ist die Tatsache, dass MQL aufgrund seiner einfachen Ausdrucksform und leichten Einsetzbarkeit, insbesondere bei AJAX-basierten Abfragen, für die Abfragen innerhalb relationaler Datenbanksysteme geeignet war. In bestimmten Fällen konnte MQL die traditionelle Abfragesprache SQL ersetzen (MQL-to-SQL) [Bou11]. Seit März 2015 werden die bisherigen angesammelten Daten von Freebase an die Wikidata-Wissensdatenbank transferiert. Das hat zur Folge, dass Freebase samt MQL heute nicht mehr zur Verfügung stehen.

Listing 2.9: MQL-Abfrage zur Bildersuche für die Sängerin Rihanna

```
[{
  "name": "Rihanna",
  "type": "/people/person",
  "/common/topic/image": [{
    "id": null
  }]
}]
```

Obwohl Freebase und MQL nicht mehr existieren, sind die damals angewandten Konzepte für die aktuelle Entwicklung des semantischen Webs immer noch von Bedeutung. Zurückblickend kann festgestellt werden, dass Freebase einen Meilenstein für den Google Knowledge Graph und Wikidata bildete. Ähnliche Konzepte wie die vereinfachten Abfragen und die Ergebnisrückgaben als JSON-Strukturen tauchen heute im Google Knowledge Graph mit JSON-LD wieder auf. Genauso

[40] `http://protege.stanford.edu`

sieht es bei den Datensätzen aus, die teilweise vom Google Knowledge Graph und von Wikidata übernommen wurden.

API-basierte Suchen

Einige semantische Webdienste bieten Suchmöglichkeiten über sog. REST-APIs an. Das bedeutet, dass der Entwickler, um eine Suche zu formulieren, keine Abfragesprache wie SPARQL mehr benutzen muss, sondern er dem Webdienst eine Liste von Parametern, die er in einer URL und über die HTTP-Befehle POST oder GET übermitteln kann. Bei Wikidata und DBPedia können die Rückgabenstruktur-Formate (z.B. JSON, XML, RDF) innerhalb der Abfrage-URL als Parameter festgelegt werden. Trotz dieser Einfachheit sind oft die Parameter für die Suche innerhalb der Wissensdatenbanken von den jeweiligen Anbietern sehr beschränkt. Nur bestimmte freigeschaltete Operationen können benutzt werden. Bei dem Google Knowledge Graph kann der Entwickler nur eine Zeichenkette als Eingabeparameter angeben[41]. Ein Beispiel für eine solche Abfrage wird im Präsentationsabschnitt des Google Knowledge Graph vorgestellt.
Zusätzlich kann die komplette Mächtigkeit einer Abfragesprache wie SPARQL nicht über eine URL-Parameterangabe abgedeckt werden kann. Das Zusammenschließen von Teilmengen wie z.B. eine Union in SPARQL oder die Formulierung komplexerer Abfragen wie *„Suche alle amerikanische Bundespräsidenten, die während ihrer Amtszeit einen Nobelpreis gewonnen und mindestens 5 Bücher geschrieben haben*" über Parameterangaben erfordern ein Grundwissen über die Daten und die interne Strukturierung.
Eine interessante Alternative bietet das Projekt Wikidata Query (WDQ)[42].

[41] `https://developers.google.com/knowledge-graph`
[42] `https://wdq.wmflabs.org/api_documentation.html`

Hier wird versucht mit einfachen Zeichenketten, die als URL-Parameter angegeben werden, komplexe Suchen von Wikidata-Objekten (engl. item) zu ermöglichen.
Obwohl WDQ neben SPARQL offiziell als Zugriffsmöglichkeit auf Wikidata-Daten aufgelistet ist, meldet der Initiator von WDQ, Markus Manske[43], dass er WDQ bald nicht mehr unterstützen wird und legt Entwicklern nahe, zukünftig die SPARQL-Schnittstelle von Wikidata[44] zu benutzen.

Listing 2.10: WDQ-Abfrage

```
claim[106:177220] and BETWEEN[569,1988,1990] and claim[21:6581072]
     and claim[412:186506] and claim[106:33999]
```

Die Abbildung 2.8 zeigt das Beispiel einer WDQ-Abfrage für die Suche nach allen Sängerinnen (claim 106:177220), die gleichzeitig auch Schauspielerinnen sind, die zwischen 1988 und 1990 geboren (claim 569) sind und mit einer Mezzosopranstimme singen.

[43] http://opendata.stackexchange.com/questions/7055/querying-wikidata-wdq-vs-wdqs-sparql
[44] https://query.wikidata.org

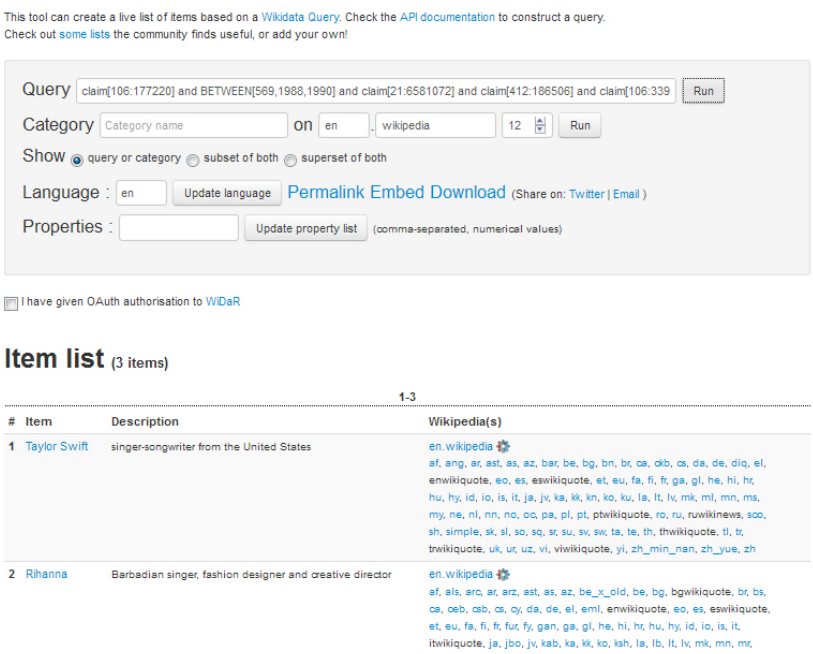


Abbildung 2.8: Anzeige der Ergebnisse der WDQ-Abfrage

Wissensdatenbanken und semantische Webdienste

Im Glossar vom W3C[45] wird ein Webdienst wie folgt beschrieben: „*Ein Webservice ist ein Software-gestütztes System, das konzipiert wurde, um interoperable Maschinen-zu-Maschinen-Kommunikation über ein Netzwerk zu unterstützen. Es besitzt eine Schnittstelle, die über ein maschinenbearbeitbares Format beschrieben wird.*“ Ein Webservice kann unterschiedliche Aufgaben lösen, von der einfachen Abfrage von Wetterdaten bis zur Durchführung komplexer Rechnungen oder Datenbankenzugriffe.

[45] `http://www.w3.org/TR/ws-gloss/{#}webservice`

Werden strukturierte Informationen von diesen Diensten zurückgeliefert, können diese als Datenquellen bezeichnet werden. Diese Daten werden von einer klar definierten Schnittstelle (API) abgerufen und dank eines maschinenlesbaren Formats von einem oder mehreren Clients konsumiert, verarbeitet und interpretiert. Beim Semantic Web spielen Webservices, um Ergebnisse dieser strukturiert zusammenzufügen, eine zentrale Rolle. Um dies zu realisieren, wurden Plattformen und Mechanismen entwickelt, die Ergebnisstrukturen aus unterschiedlichen heterogenen Quellen im Netz miteinander verknüpfen und - dank einheitlicher Beschreibungssprachen und Vokabular - auch kohärent abfragen können. Dies wird u.a. mittels der vorher vorgestellten RDF- und OWL-Formate realisiert. Mit diesen können Wissensdomänen repräsentiert und in Form von Graphen modelliert werden. Dank Abfragesprachen wie z.B. SPARQL können Suchen gestartet werden. Das resultierende Ergebnis kann serialisiert werden und einem Client per Netzwerkzugriff über ein maschinenlesbares Format (JSON oder XML) zur Verfügung gestellt werden. Die Funktionsweise des Semantic Web ermöglicht neue Relationen zwischen den Daten zu schaffen und diese wiederzufinden.

Um von den intelligenten Funktionalitäten profitieren zu können, haben Entwickler innerhalb ihrer Software die Möglichkeit, entweder ihre eigenen Datenstrukturen samt semantischer Modellierung (Ontologie) zu benutzen oder externe, von Drittanbietern zur Verfügung gestellte Daten über Abfragen aufzurufen. Die letzte Variante erfolgt über die Benutzung von sog. maschinenverstehbaren Datenstrukturen, die in der Regel über APIs von Drittanbietern aus unterschiedlichen Fachbereichen (Biologie, Filmindustrie, Physik, Informatik, Musik usw.) im Web zur Verfügung gestellt werden. Im folgenden Abschnitt werden Dienste vorgestellt, welche die semantische Suche unterstützen und qualitativ

gute, für eine Multimedia-bezogene Weiterverwendung verwertbare Ergebnisse liefern.

DBpedia

Einer der SPARQL Endpoints und Wissensdatenbanken im Semantic Web wird von der Plattform DBpedia angeboten. DBpedia stellt strukturierte Informationen aus Wikipedia frei zur Verfügung. Diese können von Entwicklern benutzt werden, um die Informationen semantisch zu verarbeiten oder neue Relationen zu entdecken.
Die DBpedia-Initiative entstand aus den Forschungsgruppen der Freien Universität Berlin, der Universität Leipzig und des Hasso Plattner Instituts in Potsdam [LIJ+14]. Das DBpedia-Datenset aus dem Jahre 2016 beinhaltete für die englische Version 4,58 Millionen „Objekte“ und 583 Millionen Fakten. Mit allen Sprachen und Verknüpfungen besitzt DBpedia in seiner 2014er-Version etwa 3 Milliarden Einzeldaten bzw. Einträge, die aus Wikipedia extrahiert wurden. DBpedia beinhaltet neben Wikipedia-Daten Verweise auf Datensätze anderer Dienste, wie z.B. OpenCyc, GeoNames (siehe nächsten Abschnitt), Musicbrainz[46] für Musikwissen oder Eurostat[47] für statistische Daten [FEM+16].
Über verschiedene Kommunikationsschnittstellen[48] (SPARQL EndSparql Clients, RDF Browser oder den normalen HTML Browser) können Daten abgefragt werden [LIJ+14]. Die Abbildung 2.9 zeigt den DBPedia-Eintrag des Rennfahrers Sebastian Vettel.

[46] https://musicbrainz.org/
[47] http://ec.europa.eu/eurostat/de
[48] http://DBpedia.org/sparql

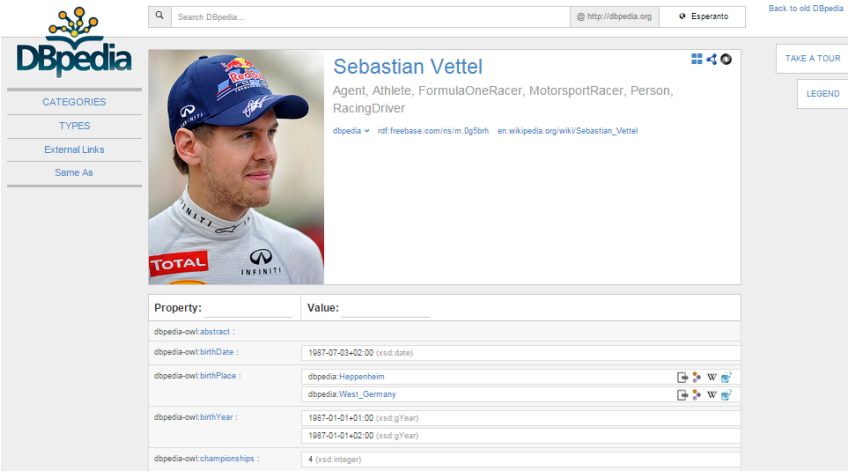


Abbildung 2.9: Screenshot der DBpedia-Seite von Sebastian Vettel

Freebase

Freebase war bis 2015 die weltweit größte von Benutzern gepflegte Wissensdatenbank und eine viel benutzte Wissensquelle für das Semantic Web. Obwohl ähnlich wie bei Wikipedia jeder Besucher Einträge (siehe Abbildung 2.10) hinzufügen konnte, wurden die neu hinzugefügten Daten immer vom Freebase-Team überprüft [BEP+08]. Im Jahre 2015 wurde Freebase von Google akquiriert und weiterbetrieben. Freebase trug dazu bei, die internen Google Knowledge Graph-Daten zu erweitern und zu vervollständigen.

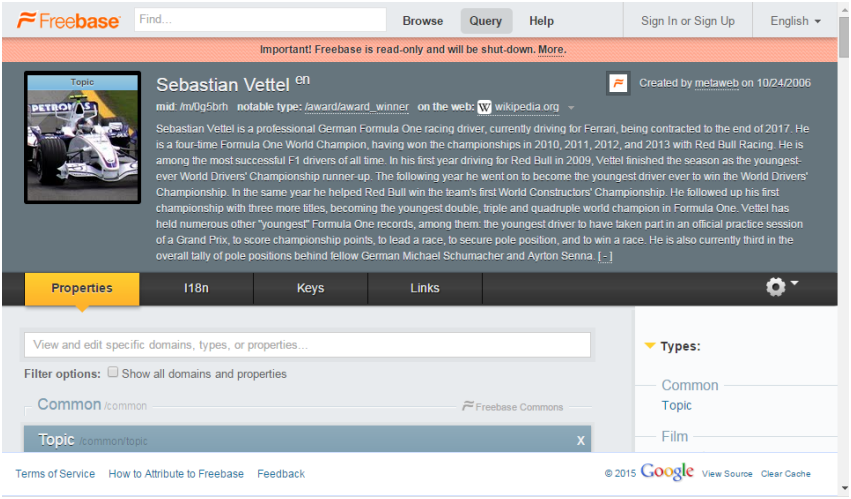


Abbildung 2.10: Freebase Seite über Sebastian Vettel

In August 2016 wurde Freebase offiziell geschlossen, wobei es ein Zeit lang immer noch möglich war, Offline-Dumps der letzten Version von Freebase herunterzuladen. Innerhalb von Freebase wurden die Relationen über festdefinierte IDs erstellt und Einträge in Kategorien geordnet.

Diese Ordnung führte zu Inkonsistenzen, da für einen Antrag (z.B. eine Person kann sowohl Schauspieler, als auch Sänger sein) mehrere passende Kategorien benutzt werden können. Da Freebase zu den beliebtesten Datenquellen für das semantische Web gehörte und somit zum Linked Data zählte, ist es fraglich, wie DBPedia, die auch auf Freebase verwiesen hat, zukünftig diese Information verwalten wird. Die Publikation „From Freebase to Wikidata: The Great Migration" [TVS+16] erklärt detailliert, welche Herausforderungen bei der Übernahme der Freebase-Daten durch Wikidata entstanden sind.

Google Knowledge Graph

Der Google Knowledge Graph ist eine Wissensdatenbank, die 2012 von Google ins Leben gerufen wurde, um die klassischen Google-Ergebnisse mit semantischen Informationen anzureichern. Hierbei werden nicht wie bei einer herkömmlichen Websuche nur Schlagwörter gesucht und gefunden, sondern der Sinn des gesuchten Begriffes wird mitberücksichtigt. Im Google Knowledge Graph werden die gesuchten Begriffe als „Sachen" („Things") und nicht als „Zeichenketten" („Strings") betrachtet[49]. Google Knowledge Graph benutzt nicht nur öffentliche Wissensdatenbanken, wie Wikipedia oder die CIA Factbook[50], um die Suche zu erweitern, sondern auch die Daten aus dem 2015 aufgekauften Freebase-Dienst. Darüber hinaus können auch neue Relationen und Daten direkt aus den von Google „gecrawlten" Webseiten extrahiert werden, wenn diese sog. Knowledge Graph Cards integrieren. Hierfür muss der Herausgeber einer Webseite Zusatzinformationen im JSON-LD Format zu dem Quellcode der Webseite hinzufügen. Diese Zusatzinformationen bzw. Eigenschaften, sollten mit dem Schema.org Vokabular übereinstimmen[51]. Somit können diese Informationen z.B. über Firmen oder Veranstaltungen mit dem Knowledge Graph verknüpft werden und als Ergebnis innerhalb der Google-Ergebnisseite erscheinen. Entwickler von Webseiten können mittels des Werkzeugs Structured Data Testing Tool (SDTT)[52] überprüfen, ob die hinzugefügten Zusatzinformationen von einem Google-Dienst für den Knowledge Graph richtig interpretiert und gecrawlt

[49] https://googleblog.blogspot.de/2012/05/introducing-knowledge-graph-things-not.html
[50] https://www.cia.gov/library/publications/the-world-factbook/
[51] https://developers.google.com/search/docs/guides/intro-structured-data
[52] https://search.google.com/structured-data/testing-tool

werden können. Hinweise, wie der Google Knowledge Graph intern funktioniert, insbesondere über den Knowledge Vault finden sich in mehreren wissenschaftlichen Publikationen wie [DGH+14] [Ede12] [SVT+12] und im Patent der Firma Metaweb [HF09] [53], der ehemaligen Betreiberin von Freebase.

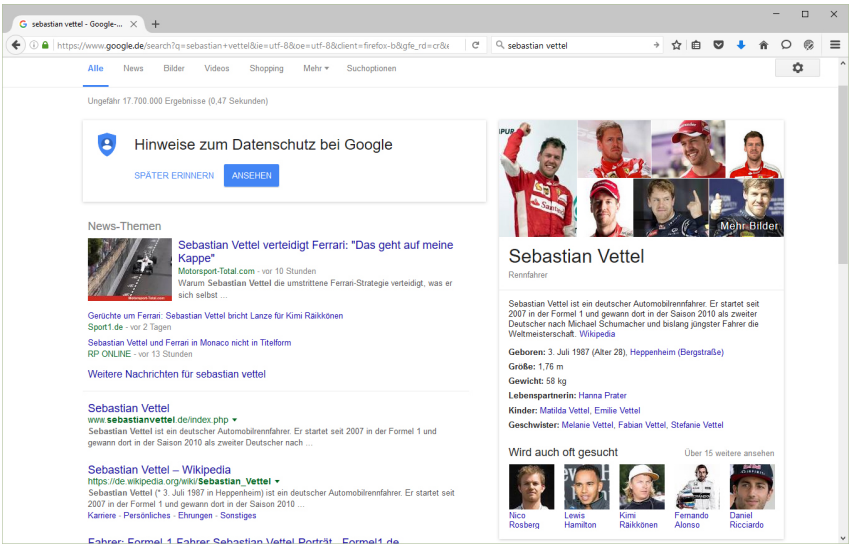


Abbildung 2.11: Visuelle Darstellung des Google Knowledge Graph Eintrags über Sebastian Vettel (rechts) innerhalb der Google-Suchergebnisse

Die Benutzung von Google Knowledge Graph wird über eine REST-API[54,55] realisiert und liefert als Ergebnis eine JSON-LD-Struktur. Diese Ergebnisstruktur kann analysiert und in Form einer HTML-Seite dargestellt werden. Folgender Quelltext zeigt die Anfrage und das Ergebnis

[53] http://www.google.com/patents/US7502770
[54] https://developers.google.com/knowledge-graph
[55] https://developers.google.com/knowledge-graph/reference/rest/v1

einer Suche nach dem Formel-1-Fahrer Sebastian Vettel mit der Google Knowledge Graph REST-API.

Listing 2.11: Google Knowledge Graph–Abfrage für den Formel 1 – Fahrer Sebastian Vettel

```
https://kgsearch.googleapis.com/v1/entities:search?query=sebastian
    +vettel&key=...&limit=1&indent=True
```

Listing 2.12: Ergebnis der Abfrage als JSON-LD-Format

```
{
  "@context": {
    "@vocab": "http://schema.org/",
    "goog": "http://schema.googleapis.com/",
    "EntitySearchResult": "goog:EntitySearchResult",
    "detailedDescription": "goog:detailedDescription",
    "resultScore": "goog:resultScore",
    "kg": "http://g.co/kg"
  },
  "@type": "ItemList",
  "itemListElement": [
    {
      "@type": "EntitySearchResult",
      "result": {
        "@id": "kg:/m/0g5brh",
        "name": "Sebastian Vettel",
        "@type": [
          "Person",
          "Thing"
        ],
        "description": "Racing driver",
        "image": {
          "contentUrl": "http://t0.gstatic.com/images?q=tbn:
              ANd9GcQFfaPn0OtI1PryFBQRG9V6mxaNvWA0edWaswKx8QTt1Msea3ZD
              ",
```

```
          "url": "https://en.wikipedia.org/wiki/Sebastian_Vettel",
          "license": "http://creativecommons.org/licenses/by-sa
              /4.0"
        },
        "detailedDescription": {
          "articleBody": "Sebastian Vettel is a German racing
              driver, currently driving in Formula One for
              Scuderia Ferrari. He is a four-time Formula One
              World Champion, having won the championship in 2010,
               2011, 2012, and 2013 with Red Bull Racing. ",
          "url": "http://en.wikipedia.org/wiki/Sebastian_Vettel",
          "license": "https://en.wikipedia.org/wiki/Wikipedia:
              Text_of_Creative_Commons_Attribution-ShareAlike_3.0
              _Unported_License"
        },
        "url": "http://www.sebastianvettel.de/"
      },
      "resultScore": 656.906921
    }
  ]
}
```


OpenCyc

OpenCyc ist eine frei verfügbare Wissensdatenbank, die zum Cyc-Projekt [Len95] gehört. Das Cyc-Projekt wurde 1984 von Doug Lenat gestartet und hat bis heute, u.a. mit dem Unternehmen Cycorp Inc.[56], das Ziel, Alltagswissen zu beschreiben, in dem Konzepte über Relationen miteinander verknüpft werden[57] [MWK+05] [CSS08] [SGKM04]. Neben der Wissensdatenbank (bzw. Ontologie) bietet Cyc eine eigene CycL(Cyc Language) Wissensrepräsentationssprache [MCWD06]

[56] http://www.cyc.com/
[57] http://www.pro-pix.de/presentation/Cyc.pdf

[Ste05], die für Reasoning benutzt wird. Parallel zu OpenCyc koexistieren zwei weitere Versionen der Wissensdatenbank: EnterpriseCyc für kommerzielle Applikationen und ResearchCyc für Forschungszwecke wobei jede dieser Versionen Unterschiede bezogen auf die Anzahl der Konzepte und der Reasoning-Möglichkeiten[58] aufweisen.

Abbildung 2.12: Beispiel einer OpenCyc-Ausgabe über Barack Obama

Der Inhalt der Wissensdatenbank von Cycorp kann als OWL-basierte Ontologie heruntergeladen und über festdefinierte URL-Endpunkte abgefragt werden. Diese Endpunkte liefern als Ergebnis eine RDF-Repräsentation des jeweiligen gesuchten Konzepts zurück. Diese Konzepte beinhalten eindeutige und festgelegte Felder, die auch mehr-

[58] `http://www.cyc.com/platform`

sprachig aufrufbar sind. Alternativ kann die Wissensdatenbank über eine Java API[59] oder eine Webschnittstelle abgefragt werden.
Die letzte Version von OpenCyc stammt aus dem Jahre 2012 und ihr Aktualisierungsgrad ist unbekannt, was u.a. bedeutet, dass viele aktuelle Persönlichkeiten des öffentlichen Lebens (Sportler oder berühmte Schauspieler) fehlen. Abbildung 2.12 zeigt den Eintrag über Barack Obama innerhalb der OpenCyc Wissensdatenbank.

Sindice

Sindice war eine semantische Suchmaschine, die vom Digital Enterprise Research Institute (DERI) [ODC+08] entwickelt wurde. Sindice, das sich selbst als einen Index für das Semantic Web bezeichnete, suchte im Netz (aber auch in sozialen Netzwerken wie Twitter) nach Inhalten und Dokumenten, die annotiert worden sind und von den jeweiligen Betreibern in unterschiedlichen Formaten wie RDF, HCARD, Microformats verfügbar gemacht wurden.
Über eine spezifische API können die Sindice-Suchfunktionalitäten aufgerufen und über Parameter näher spezifiziert werden. Diese API lieferte aggregierte und konsolidierte Ergebnisse in Form von ATOM-, RDF- oder JSON-Strukturen zurück. Innerhalb dieser Daten wurden über die Metadata-Felder Zusatzinformationen wie z.B. der Titel, der direkte Link oder die letzte Aktualisierung des Dokuments zurückgeliefert. Alternativ zur Verwendung der Sindice API kann die Abfragesprache SPARQL benutzt werden.
Eine kommerzielle Ausprägung von Sindice wird von der Sindice-Gründer Firma Sindicetech[60] angeboten, wobei hier mehr der Fokus

[59] `http://dev.cyc.com`
[60] `http://www.sindicetech.com`

auf der Erstellung von privaten und öffentlichen Knowledge Graphs für Unternehmen liegt[61].

Wikidata

Im Gegensatz zu Freebase, das von Google gepflegt wurde, bietet Wikidata[62] eine Community betriebene Wissensdatenbank mit der Open Data - Common Creative 0 Lizenz an. Wikidata enthält über 58 Millionen Datensätze und 13 Millionen referenzierte und verlinkte Objekte bzw. Daten [FEMR15]. Diese Daten werden in der Regel von Wikimedia und Wikipedia extrahiert und regelmäßig von Besuchern und Autoren der Wikipedia-Community (engl. Contributors) eingepflegt. In [VK14] beschreibt der Autor die wichtigsten Merkmale von Wikipedia und Wikidata und ihre Unterschiede. Im Gegensatz zu Wikipedia ermöglicht Wikidata das einheitliche Einpflegen multilingualer Inhalte auf einer gemeinsamen Basis. Für jeden Wikipedia-Eintrag koexistieren verschiedene Seiten, die meistens von der Community übersetzt wurden.

Dies ist bei Wikidata nicht der Fall, wie folgendes Beispiel skizziert: Ein Eintrag über die Anzahl der Bewohner von Berlin kann sowohl auf der Seite „Städte in Deutschland“, als auf der „Berlin“-Seite auftauchen, was zu Dateninkonsistenzen führen kann, wenn nachträglich die Einwohnerzahlen für alle multilingualen Einträge, d.h. alle Wikipedia-Seiten in den jeweiligen Sprachen, verändert werden müssen. Um dieses Problem zu umgehen, speichert Wikidata die Daten auf einer globaleren Ebene, sodass die Daten letzten Endes über alle Einträge konsistent bleiben. Dies gilt genauso für Relationen bzw. Aussagen

[61] `http://www.dataversity.net/end-support-sindice-com-search-engine-history-lessons-learned-legacy-guest-post/`

[62] `https://www.wikidata.org`

(engl. Statements), die nur einmal ausformuliert werden müssen und dadurch direkt für verschiedene Sprachen verfügbar sind.
Die in Wikidata gesammelten Daten können als einzelne Einträge exportiert werden, z.B. in den Formaten JSON, XML oder RDF. Jeder Wikidata-Eintrag (engl. Wikidata item) besitzt eine feste ID, die sprachunabhängig und einmalig ist, sodass das Referenzieren eines Eintrages eindeutig ist. Die Mehrsprachigkeit und die Übersetzung der Terminologien werden über sogenannte Labels ermöglicht (siehe Abbildung 2.13).

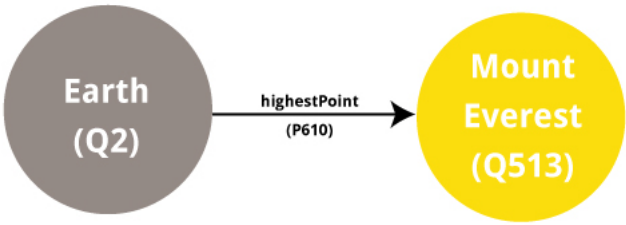


Abbildung 2.13: Grafische Darstellung der Relation „highestPoint“ in Wikidata

Jeder Wikidata-Eintrag (engl. *item*) wird mit einem Wert (*value*) über eine festgelegte Eigenschaft (*property*) verlinkt [VBB+14]. Die *items* werden mit dem Präfix Q definiert, die dazugehörenden Eigenschaften mit dem Präfix P, wobei die textuelle Ausprägung der Eigenschaft mehrsprachig sein kann. Zum Beispiel besitzen die Eigenschaften *Höhe*, *Height* und *Hauteur* die gleiche Referenznummer und zwar Q208826. Folgendes Beispiel (siehe Abbildung 2.13) zeigt, wie die Relation „highest Point“ als P610-Eigenschaft in Wikidata gespeichert ist.

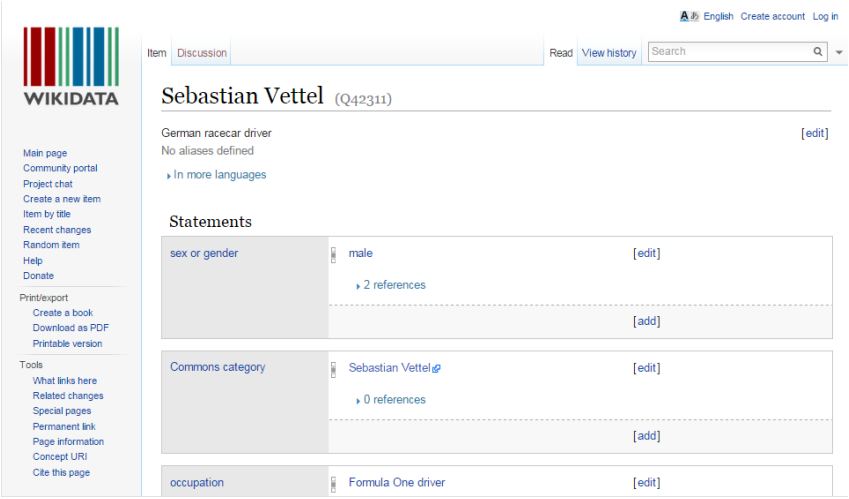


Abbildung 2.14: Wikidata-Seite über Sebastian Vettel

Eine Relation wird durch eine Aussage (engl. Statement), die sich in Eigenschaft und Wert aufteilen lassen, ausgedrückt. Dies wird im Beispiel der Abbildung 2.14 der Wikidata-Seite über Sebastian Vettel deutlich. Hier wird die Eigenschaft *occupation (P106)* mit dem Wert *Formula One driver (Q10841764)* eindeutig referenziert. Durch diese eindeutige Identifizierung (Q und P) ist es möglich, weitere Relationen aufzulösen, diese Daten miteinander zu kombinieren und über diese neue Verlinkungen zu identifizieren. Diese Herangehensweise folgt dem Linked Data-Paradigma, welches ermöglicht, dass auch andere Dienste wie z.B. Google Books[63] und MusicBrainz[64] innerhalb von Wikidata-Referenzen verlinkt werden können und somit auffindbar werden.

[63] `https://books.google.com`
[64] `https://musicbrainz.org`

YAGO

YAGO steht für *„Yet Another Great Ontology"* und ist eine gemeinsame Entwicklung der Gruppe DBWeb von dem Max Planck Institut für Informatik in Saarbrücken[65] und der Universität Télécom ParisTech[66] in Paris. Das Projekt startete 2006 und setzt sich laut der YAGO-Webseite als Ziel: „das Aufsetzen einer bequemen durchsuchbaren, groß angelegten Wissensbasis, basierend auf gemeinsamen Fakten, in einer maschinen-bearbeitbaren Repräsentation". Die YAGO-Wissensbasis wird so aufgebaut, dass die Daten von Wikipedia extrahiert und mittels der Konzepte innerhalb von WordNet[67] oder GeoNames[68] vereinheitlicht werden. Die Unifikation zwischen WordNet und Fakten aus Wikipedia ermöglicht eine Vereinheitlichung der Konzepte beider Ontologien mit einer Präzision von 97% [SKW07]. WordNet wird zusätzlich benutzt, um Relationen, Hyponymie bzw. Hypernymie innerhalb eines Satzes, Begriffsfelder und Kategorien, die von Wikipedia extrahiert worden sind, besser zu identifizieren. Bei Wikipedia werden Extraktionsverfahren angewandt, um aus den Wikipedia-Infoboxes Kategorien (z.B. Sport, Medizin) und Relationen (z.B. *Date of birth*) einer Entität zu extrahieren. Die Mehrsprachigkeit der extrahierten Konzepte wird dadurch realisiert, dass die einzelnen Wikipedia-Artikel in den jeweiligen Sprachen analysiert werden. Um eine bessere Unterstützung der Mehrsprachigkeit von geographischen Entitäten zu gewährleisten, wird der Dienst GeoNames benutzt [MBS14] [SKW07]. Im Gegensatz zu anderen Wissensdatenbanken werden bei YAGO temporelle und räumliche Informationen zu den extrahierten Entitäten hinzugefügt.

[65] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago
[66] http://dbweb.enst.fr/research
[67] https://wordnet.princeton.edu
[68] http://www.geonames.org

Dies bedeutet, dass bei jedem Konzept auch das Extraktionsdatum bzw. der Änderungszeitpunkt [HSBW13] und bei jedem geografischen Ort (sog. Geo-Entitäten) die Geokoordinaten, Postleitzahlen und ggfs. die Namen in verschiedenen Sprachen mitgespeichert werden. Die Daten von YAGO können entweder offline über einen sog. Dump oder online über eine dedizierte SPARQL-Endpunkt-Schnittstelle[69] abgefragt werden.

Übersichtstabelle

Die folgende Tabelle (siehe Abbildung 2.15), die teilweise Kriterien aus [FEMR15] berücksichtigt, fasst die existierenden Wissensdatenbanken bzw. semantischen Dienste zusammen. Im Hinblick auf die Benutzung dieser Dienste innerhalb des semantischen Fernsehens spielt die Menge und die Aktualität der Daten eine wichtige Rolle. Insbesondere bei Schauspielern oder Sportlern müssen diese Daten ständig aktuell gehalten werden. Bei OpenCyc stellt dies ein großes Problem dar, denn die letzte Aktualisierung der Daten stammt aus dem Jahr 2012. Neuere Ereignisse wurden in diese Wissensdatenbank nicht miteinbezogen. Der Google Knowledge Graph, Wikidata und DBpedia bieten dagegen aktuellere Daten sowie Links zu Medien (Bilder und Videos) innerhalb der Suchergebnisse. Diese Suchergebnisse werden über dedizierte REST-Schnittstellen geliefert. Die Rückgabestrukturen dieser Dienste bzw. Formate wie JSON (für Wikidata), JSON-LD (im Falle vom Google Knowledge Graph) oder XML ermöglichen eine bessere Integration in HTML-basierte Applikationen. Zusätzlich trägt die Einfachheit der Verwendung dieser Formate und der Abfragesprachen dazu bei, die Akzeptanz dieser Dienste bei der Entwicklergemeinschaft von Web 3.0-

[69] `https://io.datascience-paris-saclay.fr/exampleView.php?ex_id=8`

Applikationen zu erhöhen. Bei Wikidata oder dem Google Knowledge Graph ist es, u.a. dank ihrer gut dokumentierten REST-Schnittstelle für Entwickler, um eine semantische Abfrage durchzuführen, nicht mehr zwingend notwendig, eine Abfragesprache wie SPARQL zu benutzen. Ein weiterer Aspekt dieser Betrachtung liegt bei der Mehrsprachigkeit. Unter Mehrsprachigkeit versteht man, dass Konzepte, Relationen und Entitäten in mehreren Sprachen verfügbar sind und die interne Modellierung einheitlich und sprachunabhängig bleibt. Dies wird insbesondere bei YAGO, Google Knowledge Graph, Wikidata und DBpedia berücksichtigt. OpenCyc dagegen bietet nur Relationen und Entitäten in englischer Sprache. Nach Betrachtung der einzelnen Dienste werden die semantischen Dienste Wikidata und Google Knowledge Graph als Datenquelle für die semantische Suche innerhalb von Swoozy benutzt.

		Wissensdatenbanken bzw. semantische Dienste			
		Aktiv			
		Wikidata	DBpedia	Google Knowledge Graph	yago select knowledge
Daten	Anzahl Datensätze	24 Mio. Einträge	4.58 Mio. Instanzen	1,6 Mrd. Triples [2,3]	10 Mio. Entitäten 120 Mio. Fakten
	Modellierungs-sprache/ Format	JSON	Turtle (RDF) JSON-DL	unbekannt	Turtle (RDF) TSV
	Mehr-sprachigkeit	✓ (mehr als 100)	✓ (mehr als 100)	✓ Mehrsprachig	✓ Mehrsprachig (10 Sprachen)
	Multimedia-inhalte	✓ Ja, über Wikipedia	✓ Ja, über Wikipedia	✓	✓ Ja, über Wikipedia
Zugriffe und Schnittstellen	API ggfs. Abfragesprache	✓ REST-API	✓ REST-API/SPARQL	✓ REST-API	✓ REST-API/SPARQL
	Rückgabe-formate	JSON/XML/RDF	JSON	JSON/JSON-LD	SRX JSON CSV
	Einpflegen / Korrigieren der Daten	Community-basiert	Community-basiert	von Google	Direktes Feedback durch Benutzer
	Benutzungs-lizenz	CC0 1.0 Universal Public Domain Dedication	CC Attribution ShareAlike 3.0	Google API Benutzungs-vereinbarungen	CC BY 3.0 Unported
	offline Dumps	✓	✓	✗	✓
Anmerkung		benutzt Wikipedia	extrahiert und benutzt andere Quellen u.a. GeoNames, MusicBrainz, DBLP oder YAGO	benutzt teilweise Wikipedia, Wikidata Daten wurden von Freebase übernommen	extrahiert Wissen aus Wikipedia benutzt u.a. WordNet

[2] Stand 2014 (Knowledge Vault)
http://www.cs.cmu.edu/~nlao/publication/2014.kdd.pdf

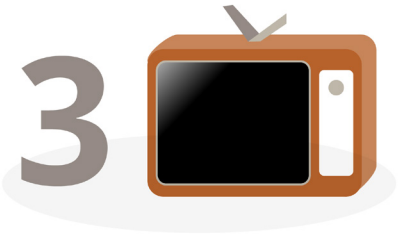
[3] Stand 2012
570 Millionen Entitäten / 18 Billionen Fakten

Abbildung 2.15: Übersichtstabelle der Wissensdatenbanken und semantischen Dienste (1/2)

		Wissensdatenbanken	
		Aktiv	Inaktiv
		OpenCyc/ OpenCyc KB	**Freebase**[1]
Daten	Anzahl Datensätze	~239.000 Begriffe ~2093000 Triples	1.9 Mrd. Triples
	Modellierungs-sprache/ Format	OWL	N-Triples RDF
	Mehr-sprachigkeit	✗ nur Englisch	✓ Mehrsprachig
	Multimedia-inhalte	✗	✓ Bilder
Zugriffe und Schnittstellen	API ggfs. Abfragesprache	✓ CycL/Java API	✓ REST-API/MQL
	Rückgabe-formate	RDF Cyc Core Java API	JSON/XML
	Einpflegen / Korrigieren der Daten	Update-Zyklus unklar Geplegt von Cycorp	eingestellt
	Benutzungs-lizenz	kommerziell frei	CC BY CC0
	offline Dumps	✓	✓
Anmerkung		referenziert auf Wikipedia oder DBPedia	von Google eingestellt

[1] Seit August 2016 nicht mehr verfügbar

Abbildung 2.16: Übersichtstabelle der Wissensdatenbanken und semantischen Dienste (2/2)



Wissenschaftliche Arbeiten zum interaktiven Fernsehen

Einleitung

Im folgenden Kapitel werden die verwandten Arbeiten im Bereich Interaktionen mit Multimediasystemen, der Extraktion von Wissen aus Multimediainhalten sowie aktuelle Werkzeuge zur Annotation von Medien detailliert vorgestellt. Im ersten Abschnitt wird zunächst der Fokus auf natürliche Gesteninteraktionen mit Fernsehsystemen (z.B. Spielekonsolen) gerichtet, indem Hardwarekomponenten und deren Interaktionsparadigmen einzeln vorgestellt werden. Dabei werden die Prinzipien, die Analyse, die Erkennung des Benutzers und die Interpretation seiner Bewegungen durch diese Komponenten näher betrachtet. Zusätzlich werden die Programmierschnittstellen und Bibliotheken für diese Technologien analysiert, um die Defizite der existierenden Interaktionsformen aufzudecken und die geeignetsten Technologien für das angestrebte Ziel eines semantischen Fernsehens

zu finden. Des Weiteren werden die Leitlinien (Leitfaden in Form konkreter Empfehlungen für die Interaktionen) detailliert betrachtet, um aus ihnen das für den Swoozy-Ansatz geeignetste Interaktionsparadigma zu ermitteln. Ein weiterer Fokus wird auf der Extraktion von Semantik aus Multimediadaten gesetzt. Diese Verfahren werden im Abschnitt *Semantifizierung von Mediendaten* detailliert vorgestellt. Hier werden nicht nur die Video- und Bildbasierten Analysealgorithmen wie z.B. OCR (Optical Character Recognition) unter die Lupe genommen, sondern auch Informationsextraktionswerkzeuge aus dem kommerziellen und wissenschaftlichen Bereich, die es ermöglichen, nicht annotierte Multimediadaten in Form von semantischen annotierten Begriffen oder Taxonomien zu klassifizieren.

Gesteninteraktionen: Technologien und Designleitlinien

Die Art und Weise, wie der Benutzer mit einem Fernsehsystem interagiert, bestimmt meistens, wie das Interaktionserlebnis (engl. User Experience kurz UX) eines Systems gestaltet werden muss. Dieses Interaktionserlebnis kann durch die Benutzung von Gesteninteraktionen in Zusammenhang mit einer Benutzerschnittstelle verbessert und angereichert werden.

Gesten

Laut Duden wird eine Geste als *„spontane oder bewusst eingesetzte Bewegung des Körpers, besonders der Hände und des Kopfes, die jemandes Worte begleitet oder ersetzt [und eine bestimmte innere Haltung aus-*

drückt]" bezeichnet. Eine Gestik wird als *„Gesamtheit der Gesten [als Ausdruck einer charakteristischen inneren Haltung]*" definiert.

Klassifikation und Typisierung der Gesten

Als ausgeführte Geste versteht man die gleichzeitige Bewegung des Armes und/oder der Hand während eines kommunikativen Akts. Oft sind Gesten mit einer sprachlichen Äußerung verbunden und ergänzen diese. In [McN92], [Que94] und in [PSH97] findet man eine Taxonomie der Gesten, die als Grundlage der Arbeiten im Rahmen der Mensch-Maschinen-Schnittstellen dienen. Gesten werden in folgende Kategorien klassifiziert:

- Ikonische Geste (engl. iconic): Diese Gestenart symbolisiert simulierte Aktion oder Halten eines Gegenstands. Hier wird mit den Händen nachgeahmt, dass ein Gegenstand gehalten wird. Wenn ein Sprecher von einem Regenschirm spricht, wird er die Hände so halten, als würde er tatsächlich den Regenschirm halten [McN92]. Die Größe eines Objektes kann ebenso mit Handgesten angedeutet werden.

- Metaphorische (manchmal auch als pantomimische bezeichnet) Geste: Diese Gestenart ist ähnlich der ikonischen Geste, aber vermittelt eine abstraktere Idee. Wenn ein Sprecher eine Gruppe symbolisieren möchte, öffnet und schließt er beide Hände, als würde er eine (virtuelle) Kugel in den Händen halten. Das abstrakte Konzept einer „Gruppe" wird somit symbolisiert.

- Deiktische bzw. zeigende Geste: Hier steht die eindeutige Referenz durch eine gezielte und gerichtete Zeigegeste im Fokus. Darunter versteht man eine natürliche Zeigegeste mit einer Re-

ferenz auf einem realen physikalisch existierenden (z.B. Haus, Baum) oder virtuellen Gegenstand (Icon, Bild, Text). Deiktische Gesten werden oft in Zusammenhang mit multimodalen Dialogsystemen benutzt, um entweder eine Aktion innerhalb eines Systems auszulösen oder einen virtuellen grafischen Gegenstand sprachlich eindeutig zu referenzieren. Ein Beispiel davon ist das *„Put-That-There"*-System [Bol80][1]. Bei diesem System wird ein Sprachbefehl (*„Platziere das Boot hier hin!"*) mit einer Zeigegeste auf einer geografischen Karte kombiniert. Die Zeigegeste wird über einen virtuellen Pointer auf einer Leinwand grafisch dargestellt. Diese Art der Multimodalität war ebenso Bestandteil des XTRA-Systems [Wah98] [KAR+86], das einem Benutzer die Möglichkeit gab, ein Formular per deiktischer Gesten auszufüllen und parallel zur Gesteninteraktion Fragen wie: *„Can I add my annual $15.00 ACL dues to these membership fees?"* zu stellen. Ein weiteres Beispiel aus [Neß16] zeigt ein Szenario bei dem eine deiktische Geste in Richtung eines Kinoplakats mit einer sprachlichen Aufforderung zum System: *„Gib mir Informationen zu diesem Film!"* kombiniert wird. Im System SmartWeb [SEH+07] [Wah04] kann die Frage *„Wie viele Spiele hat dieser Spieler gewonnen?"* mit einer deiktischen Referenz auf einen Fußballspieler gestellt werden.

- Rhythmisierende Gesten (engl. beats) werden für die Betonung innerhalb eines gesprochenen Satzes benutzt, um die einzelnen Wörter zu betonen und somit die Wichtigkeit dieser visuell stärker hervorzuheben.

- Kohäsive Geste werden in [Aky03] als „Sonderformen, die dazu dienen, disjunkte aber logisch zusammen gehörende Aussagen

[1] https://www.youtube.com/watch?v=0Pr2KIPQOKE

zu markieren“ bezeichnet.

Eine weitere Taxonomie, die von [Que94] [Que95] [QMB+02] vorgeschlagen wurde, unterscheidet zwei wesentliche Gestenklassen: die kommunikativen Gestenklassen und die manipulativen Gesten.

- Unter manipulativen Gesten werden Interaktionen, die mit konkreten Gegenständen der Umgebung durchgeführt (wie z.B. die Rotation eines Objekts) werden, bezeichnet. Diese Kategorie kann in vier weitere Kategorien unterteilt werden, wie es in [KS05] präsentiert wird. Die erste davon enthält 2D-Interaktionen, die über eine 2D-basierte Benutzerschnittstelle durchgeführt werden können. 3D-Gesten können mit Zusatzinformationen wie z.B. dem Fingerdruck (im Falle eines Multitouch-Tischs), der Schnelligkeit der Gestenausführung und der räumlichen Parameter dieser, angereicht werden. Eine weitere Klasse bildet die Interaktion mit sog. greifbaren Benutzerschnittstellen (engl. Tangible User Interfaces), die es erlauben, physikalische haptische Objekte zu bewegen und dadurch gleichzeitig eine Aktion in einem System auszulösen bzw. visualisieren zu lassen. Ein Beispiel dafür bildet das Multitouch-basierte ReacTable System [JGAK07]. Durch die Bewegung und Manipulation von physikalischen Objekten (z.B. ein Holzwürfel, der auf der Oberfläche des Multitouchtischs platziert ist), werden gleichzeitig die Visualisierung und das akustische Feedback des Systems verändert. Zu der letzten Unterkategorie gehören die Gesten, die Interaktionen, die eine gezielte Auswirkung auf physikalische Gegenstände ermöglichen wie z.B. die Ansteuerung eines Roboterarms über die Microsoft Kinect bzw. Tiefenbildkameras [DZ14] oder der Fingertrackinglösung Leap Motion [WBRF13]. Feingranularer betrachtet, werden

die 3D-Gesten, die ohne Zusatzgerät durchgeführt werden, als Freihandgesten bezeichnet. Die Gesten, die mittels einer Hardware (z.B. einer Fernbedienung oder WiiMote) realisiert werden, werden in der Literatur als Gerätegesten bezeichnet.

- Kommunikative Gesten sind meistens mit Sprache verbunden und haben ein kommunikatives Ziel. Diese sind wiederum in zwei Kategorien unterteilt [KS05]. Eine weitere Kategorie der kommunikativen Gesten bilden, laut [QMB+02] , die Semaphore-basierenden Gesten (engl. semaphoric gestures), was man im Deutschen in etwa als lexikalisierte Gesten übersetzen könnte. Laut Duden ist das Semaphor ein *„Mast mit verstellbarem Flügelsignal zur optischen Zeichengebung“*. In [QMB+02] werden jedoch Semaphoren als Gestensysteme bezeichnet, die ein Wörterbuch oder eine Referenz von statischen oder dynamischen Hand- oder Armbewegungen definieren. Der Daumen nach oben bedeutet ein „OK“ und kann somit als Semaphore (im Sinne von Quek [Que94]) betrachtet werden. Diese können wiederum als kommunikative Gesten betrachtet werden, denn durch die Benutzung dieser, wenn diese eindeutig von einem System identifiziert wurden, können Benutzer vordefinierte Funktionen auslösen. Zusätzlich wird in [QMB+02] eine feingranulare Unterscheidung zwischen diesen Gestentypen und den manipulativen gemacht. Bei den lexikalisierten Gesten ist kein Feedback nötig. Diese können jedoch kulturell geprägt und benutzerabhängig sein (siehe Beispiel der Anwendung von Handvokabularen in Süditalien[2]). Akten (engl. acts) können wiederum in zwei weiteren Kategorien eingeordnet werden. Die mimetische Geste bezeichnet eine

[2] http://www.portanapoli.de/kultur/gesten-neapel-italien

imitierte Geste, die vom Benutzer identisch zu einer vorher bekannten oder gesehenen Geste nachgemacht bzw. wiederholt wird. Eine deiktische Geste entsteht, wenn ein Benutzer mit dem Finger oder mittels eines Zusatzgerätes auf eine grafische Oberfläche zeigt. Aus diesem Grund können innerhalb der deiktischen Gesten zwei zusätzliche Kategorien identifiziert werden, welche die Taxonomie von Quek erweitern könnten: das natürliche Zeigen (engl. natural pointing) und das simulierte Zeigen (engl. simulated pointing) mit Hilfe eines Stiftes oder eines Zusatzgeräts [Sch91] [Sch87].

Als Erweiterung der Taxonomie von Quek werden in [RV08a][3] drei zusätzliche Gestenkategorien vorgeschlagen: die symbolisch-manipulativen Gesten, die kontinuierlichen Gesten und die Bewegungsverfolgung. Der Vorschlag ist sehr von der Mensch-Computer-Interaktion geprägt und trennt sich von der klassischen semiotischen Kategorisierung. Die symbolisch-manipulativen direkten Gesten besitzen eine direkte grafische Auswirkung auf das manipulierte Objekt (z.B. die Position oder die Größe eines Bildes, das über ein Touchscreen virtuell verändert werden kann). Dagegen müssen manipulative abstrakte Gesten wie z.B. die *Pinch-To-Zoom*-Geste beim Multitouch vorher vom Benutzer bekannt sein. Diese Gesten sind oft System- und Hardwarespezifisch. Die *Pinch-To-Zoom*-Geste auf dem „Home"-Bildschirm besitzt auf einem Android-Gerät eine andere Funktion als auf einem iOS-Gerät. Bei den indirekten symbolischen manipulativen Gesten muss eine schon bekannte Geste durchgeführt werden. Ein Beispiel davon ist das Zeichnen eines X auf einem Touchscreen (ähnlich dem Graffiti System von Palm [CM08]) zum Löschen eines grafischen Objekts. Die Abbildung

[3] `http://www.info-design.net/laborbuch/2011/02/kassifizierung-von-gesten-fuer-hci`

3.1 zeigt eine mögliche Erweiterung der ursprünglichen Taxonomie von [Que95] durch die neuen Gestenbeschreibungen aus [RV08a] und den multimodalen Interaktionen aus [Sch87] [SW91]. 3D-Gesten werden in der Literatur als „Mid-Air Gestures“ [BVLG11] [ADWMH+12] bezeichnet. Hierbei muss differenziert werden, dass eine *Mid-Air*-Interaktion auch dann ausgeführt werden kann, wenn der Benutzer eine Wii-Mote (siehe nächsten Abschnitt) oder sein mobiles Endgerät als Eingabegerät bzw. Fernbedienung benutzt.

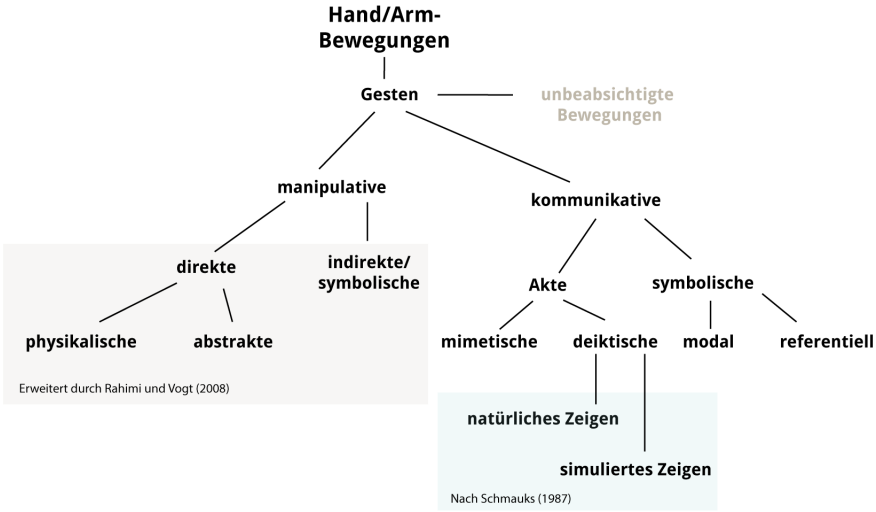


Abbildung 3.1: Taxonomie der Gesten

In der Benutzerstudie von Microsoft [ADWMH+12] und in Hinblick auf die Benutzung von Tiefenbildkameras wurde eine neue Klassifikation vorgeschlagen (siehe Abbildung 3.2 und das erklärende Video[4]). In Abbildung 3.2 wurden die Begriffe ins Deutsche übersetzt, außer der Begriff „semaphoric“, der mit dem Neologismus „semaphorisch“ (die-

[4] https://vimeo.com/63116278

ses könnte auch mit dem Begriff „lexikalische Gesten“ gleich gesetzt werden) ersetzt wurde.

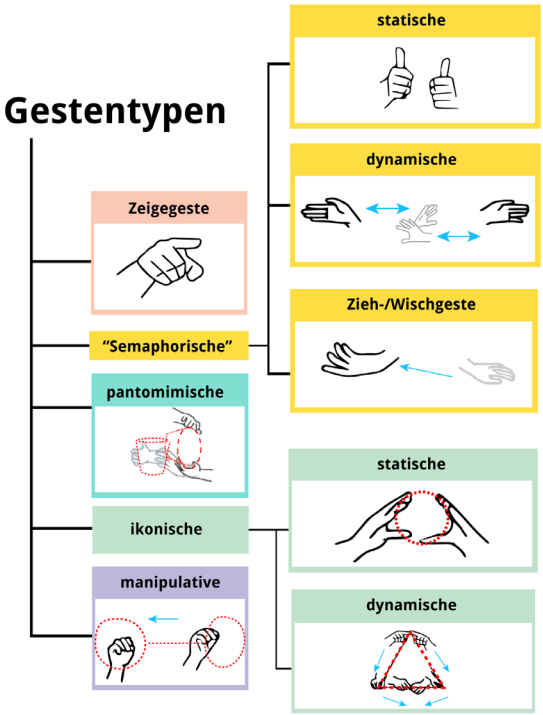


Abbildung 3.2: Taxonomie der Gesten nach ADWMH+12. Grafisch adaptiert und übersetzt.

Weitere Interaktionsmöglichkeiten und Klassifikationen innerhalb von 2D bzw. 3D-Umgebungen werden in [BK14] und [PD15] vorgestellt.

Ausführung von Gesten

Gesten sind dynamische Aktionen, die einen genauen Ablauf und einen temporalen Rahmen besitzen. Nach [McN92][5] folgt die Ausführung von drei Hauptschritten (Vorbereitung, Hauptgeste und Retraktion), die laut [Aky03] und der Webseite Doccheck.com[6] wiederum in fünf Schritte unterteilt werden können.

- Vorbereitung (engl. Preparation) (optional)
- Vorhaltung (engl. Prestroke-hold) (optional)
- Hauptgeste (engl. Stroke)
- Halten der Geste (engl. Stroke Hold) (optional)
- Beendung der Hauptgeste (engl. Poststroke hold) (optional)
- Einnehmen der Ruheposition (engl. Retraction) (optional)

Auswertung von Gesten

Die Auswertung von Gesten folgt einer Reihenfolge von Schritten und Analysen, die im Vorfeld durchgeführt werden müssen, bevor die eigentliche Ausführung eines gestenbasierten Kommandos durch ein System erfolgen kann.

[5] http://mcneilllab.uchicago.edu/analyzing-gesture/intro_to_annotation.html

[6] http://flexikon.doccheck.com/de/Gestik

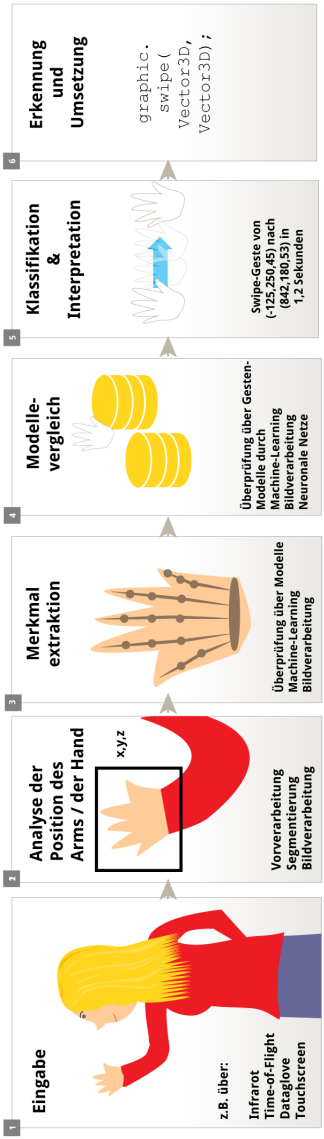


Abbildung 3.3: Ablauf einer Gestenerkennung und -auswertung

Im ersten Schritt erfolgt die Gesteneingabe durch den Benutzer (siehe Abbildung 3.3, die grafisch aus [Aky03] adaptiert und ergänzt wurde). Bei diesem Schritt unterscheidet man zwischen einer Marker-basierten und Marker-losen Erkennung. Bei der ersten Variante werden optisch wiedererkennbare Markierungen (z.B. kleine weiße Aufkleber) oder elektro-haptische Sensoren an der Hand befestigt. Hierfür muss die Hand des Benutzers im Vorfeld mit einem Zusatzgerät ausgestattet werden. Ein Beispiel davon ist der Handschuh *Data Glove* [ZLB+87] [Qua90]. Bei der Marker-losen Erkennung wird über das Eingabegerät mittels physischer Eigenschaften z.B. über Infrarotlicht (im Falle der Kinect oder des Leap Motions Kontrollers) über eine Kamera [GAS09] oder über die Analyse des magnetischen Feldes wie beim bekannten *Put-That-There"*-Szenario [Bol80] (hier wurde ein Polhemus-Sensor benutzt [SH82]), die Position der Hand oder des Armes eines Benutzers aus seiner Umgebung „extrahiert". Diese Geräte werden im folgenden Abschnitt detaillierter präsentiert. Im zweiten Schritt werden die Rohdaten der Eingabegeräte extrahiert und in Teilbereiche (im Falle eines kamerabasierten Systems) oder Signale (im Falle von akustischen oder piezoelektrische Verfahren) isoliert und separat betrachtet. Dies erfolgt mittels Bildverarbeitungsmethoden oder über eine Signalanalyse durch neuronale Netzwerke oder Markov-Modelle [NW+96]. Im nächsten Schritt werden die Merkmale der extrahierten Bild- bzw. Signalinformationen verarbeitet und mittels Modellen verglichen. Wird, wie in der Abbildung 3.3 dargestellt, eine Hand erkannt, wird das Skelettmodell dieser benutzt und die aktuelle Position in ein 3D-Koordinatensystem gesetzt und angeglichen. In Schritt 4 wird das aktuelle Handmodell mit vorhandenen Modellen verglichen. Der Vergleich ruht nicht nur auf einem Positionsvergleich innerhalb des 3D-Koordinatensystems, sondern nimmt zusätzlich die Dauer der aus-

geführten Geste in Betracht. Im letzten Schritt wird die Geste, falls diese von einem multimodalen Dialogsystem analysiert wird, klassifiziert und mittels der extrahierten Parameter ausspezifiziert. Diese Parameter werden in einem letzten Schritt zu einem Programm weitergereicht und die damit verbundene Aktion ausgeführt. Auch wenn heute Gesten im Bereich der Benutzerschnittstellen allgegenwärtig sind, können ihre Analyse und Interpretation durch folgende Faktoren eingeschränkt werden:

- Die technische Beschränkung des Eingabegerätes (Okklusion, Lichteinflüsse, Präzision der Erkennung).
- Die Interpretation der Gesten (*Wie reagiert das System und welche Aktionen müssen durchgeführt werden, damit diese kohärent bleiben und schnell erlernbar sind*)?
- Das Anwendungsszenario, d.h. in welcher Umgebung (Fahrzeug, vor einem Bildschirm) und mit welcher Körperhaltung (stehend vs. sitzend) die Geste ausgeführt werden muss.
- Kulturelle Aspekte (*„Welche Gesten sind kulturell geprägt? Und welche sollen daher vermieden werden?"*)
- Patentrechtliche Probleme. Im Rahmen einer kommerziellen Benutzung von bestimmten Eingabegeräten kann die Benutzung spezifischer Gesten einem Patent unterliegen. So kann die *Slide-Geste* zum Entsperren eines iPhones nicht in ein anderes kommerzielles System mit der gleichen Metaphorik und Funktion benutzt werden. Die Erfassung und Benutzung von personalisierten 3D-Gesten sind im Falle der Microsoft Kinect Bestandteil eines Patents [ZKLM11]. Zehn weitere Fälle von patentierten

Gesten können unter der Gizmodo-Webseite[7] nachgeschlagen werden.

Diese aufgelisteten Randbedingungen müssen in allen Fällen designtechnisch berücksichtigt werden, insbesondere in einem starken benutzerzentrierten Szenario wie bei der Interaktion mit einem Fernsehgerät.

Bezug zum Fernsehen

Den Fernseher per Geste anzusteuern, wurde schon in den frühen 90ern, u.a. mit einem System von Mitsubishi [FW95], erforscht. Dabei wurde mit einer Kamera und einer Bildanalyse die Handfläche eines Benutzers erfasst. Die aus heutiger Sicht anekdotische Hauptmotivation bestand darin, dass Fernbedienungen leicht verloren gehen können und durch den Einsatz von Gestensteuerung einfach und schnell ersetzt werden könnten. Grafisch wurde die Position der Hand über das Fernsehbild überlagert und als „virtuelle Hand" bzw. Hand-Icon dargestellt. Durch diese Gesten konnte der Benutzer sog. Schaltelemente (Sliders) bewegen und somit die Fernsehkanäle auswählen und die Lautstärke einstellen. Die vorhin durchgeführte Betrachtung hat gezeigt, dass Gesten in verschiedenen Klassen eingeordnet werden können und teilweise kulturell geprägt sind. Der Benutzer sollte so weit wie möglich keine Zusatzgeräte oder Marker während der Interaktion tragen.

Für die Benutzung von Gesten innerhalb des fernsehbasierten Systems Swoozy werden natürliche metaphorische manipulative 3D-Gesten benutzt, die eine erweiterte Fokussierung und Auswahl von

[7] http://io9.gizmodo.com/10-physical-gestures-that-have-been-patented-1251848412

virtuellen grafischen Elementen ermöglichen. Bei der Auswahl der geeignetsten wurden die erkannte Problematik der Müdigkeit und die kulturellen Aspekte berücksichtigt.
Laut der Klassifikation in [ADWMH+12] ist die "Grab'n'drop"-Interaktion, die innerhalb von Swoozy benutzt wird (siehe Kapitel 6), eine pantomimische Geste.
Basierend auf diesen theoretischen Erkenntnissen und Verfahren werden im folgenden Abschnitt verschiedene Technologien beschrieben, die speziell für die Gesteninteraktionen mit dem Fernseher oder der Spielekonsole dediziert sind. Diese Technologien und Verfahren ermöglichen, Gesten von Benutzern zu erfassen und zu analysieren. Die Auswahlkriterien für diese Technologien sind die Verfügbarkeit, die Kosten, die Programmierbarkeit und die schnelle Einsetzbarkeit der verschiedenen Lösungen und Hardwarekomponenten im Kontext einer Integration in das Swoozy-System. Am Ende des Abschnitts werden weitere mögliche einsetzbare Hardwarekomponenten, die jedoch nicht gestenspezifisch sind, zur Interkation mit dem Fernseher vorgestellt.

10 foot-Design

Der Begriff *10 foot-Design* spielt im Kontext Fernsehen und Design von Benutzerschnittstellen für größere Bildschirme eine zentrale Rolle.
Unter *10 foot-Design*, oft als *10 foot-UI* oder *10 foot-Experience* gekürzt, versteht man Designleitlinien, welche größere, gut erkennbare Schriften und Schaltelemente bei der grafischen Konzipierung einer Benutzerschnittstelle einfordern. Diese Schaltelemente, z.B. Menüs oder Texte, sollten idealerweise von einer Entfernung von 10 Füssen (ca. 3 Meter) gut erkennbar bleiben und somit die Interaktion mit dem

Fernseher intuitiver gestalten. Zusätzlich stellen die benutzerzentrierte Ergonomie, eine angemessene Bildschirmgröße und die leichte Bedienung eines virtuellen Cursors die Grundlagen des *10 foot-Design*s dar[8,9,10] [Lal13]. Letzteres beruht teilweise auf dem Fitts' Gesetz[11] und den Studienergebnissen von Scott MacKenzie [Mac92]. Folgende Abbildung zeigt ein Beispiel von Benutzerschnittstellen kommerzieller Systeme, die dem *10 foot-Design*-Paradigma folgen.

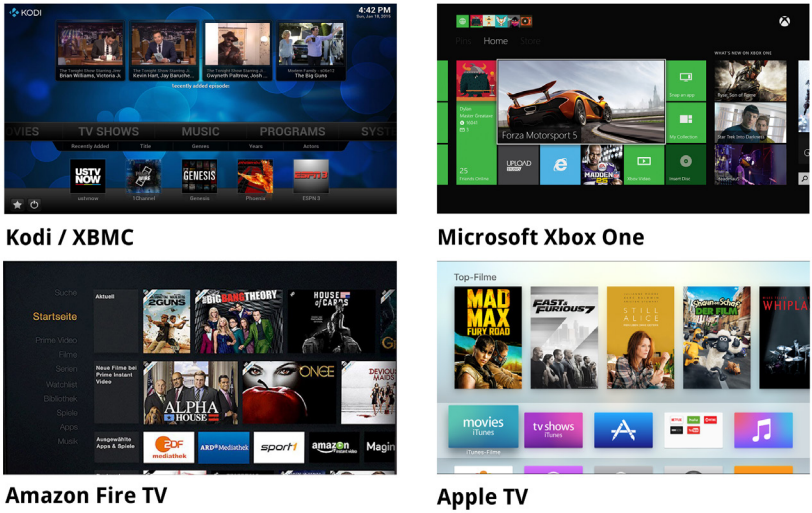


Abbildung 3.4: Beispiele von 10 foot-Design Benutzerschnittstellen. Quelle: kodi.tv, microsoft.com, amazon.com und apple.com.

8 http://blogs.adobe.com/dreamweaver/2015/11/designing-user-experiences-for-the-10-foot-ui.html

9 https://www.google.com/design/spec-tv/design-principles/designing-for-tv.html

10 http://uxpamagazine.org/tv-or-not-tv

11 https://www.interaction-design.org/literature/book/the-glossary-of-human-computer-interaction/fitts-s-law

Microsoft Kinect: Tiefenbildkamera zur Körpersteuerung

Prinzip

Die *Kinect*-Technologie von Microsoft wurde erstmals im November 2010 auf den Markt gebracht. Sie ermöglicht neben Spracherkennung über ihre vier Mikrofone die Erfassung von Benutzergesten über eine sog. Tiefenbildkamera. Die Kamera ist dabei in der Lage, mehrere sich bewegende Personen zu analysieren. In ihrer ersten Version war die Kinect nur für die Spielekonsole Xbox 360 vorgesehen. Hierbei wurden interaktive Spiele wie z.B. Kinect Sports angeboten, bei denen Sportbewegungen durchgeführt werden mussten, damit Spieler Punkte sammeln konnten. Schon wenige Stunden nach ihrer Markteinführung wurde die Kinect durch mehrere Computerspezialisten und Entwickler gehackt. Eine Analyse der USB-Datenpakete führte dazu, dass die Bilddaten samt Skelettpunkten über Reverse-Engineering extrahiert werden konnten. Ziel dieses Vorgehens war es, die Kinect an einen PC-Rechner anschließbar zu machen (über USB) und die extrahierten Daten z.B. für 3D-Scannen oder die Ansteuerung einer Maus zu verwenden.

Ab Sommer 2011 wurde von Microsoft eine Windows-kompatible Kinect zum Kauf angeboten. Parallel dazu wurde eine freiverfügbare SDK (Software Development Kit bzw. Anwendungen, um eine Software zu erstellen) freigegeben, die es ermöglichte, sowohl die 3D-Skelettinformationen der einzelnen Benutzer per Software über eine Schnittstelle mittels der Programmiersprache C# programmtechnisch zu erfassen und zu bearbeiten als auch die sog. Mikrophone-Arrays zu kontrollieren und deren Signal zu analysieren. Diese Freigabe ermöglichte eine preiswerte Realisierung von Kinect-kompatiblen Ap-

plikationen und die Erfassung von aggregierten Umgebungsdaten (Spracheingabe, Geräuschkulisse und 3D), die u.a. in multimodalen Dialogsystemen eine essentielle Rolle spielen. Im nächsten Abschnitt wird beschrieben, welche Hardwarekomponenten in der Kinect 1 und 2 eingebaut sind und welche Art von Daten mit Tiefenbildkameras gewonnen werden können.

Technische Realisierung

Abbildung 3.5: Kinect 1 (links) und die Kinect 2 (rechts)

Die Technologie der Kinect und die damit verbundenen Bewegungsverfolgungsverfahren beruhen auf dem Prinzip des sog. optischen Trackings. Dabei erfasst eine erste Kamera (RGB (Rot Grün Blau) – Kamera) das zweidimensionale Bild der Umgebung. Diese Daten werden als reine Pixelinformationen (X, Y- Koordinaten und RGB Werte) gespeichert, ähnlich dem Prinzip einer Webcam oder eines CMOS-Chips einer Digitalkamera. Zu diesem Zeitpunkt fehlt die Erfassung der dritten Dimension, um daraus ein korrektes und brauchbares Tiefenbild zu generieren. Zwar könnten stereoskopische [IDS12] [Hoi11] [Pri07] Verfahren zur Nachberechnung der Tiefe benutzt werden, diese verlangen jedoch eine rechen- und zeitintensive parallele Analyse der jeweiligen erfassten Bilder [Bla13].
Um trotzdem an diese fehlende Rauminformation zu gelangen, wirft die Kinect – parallel zum normalen RGB-Bild – ein Muster (Gitterpunk-

ten) aus Infrarotlicht in den Raum. Dies bezeichnet man als strukturiertes Licht, denn die Muster erscheinen als Punkte-Matrix (auch als Punktwolke bezeichnet) und können in einem weiteren Schritt von einer zweiten Kamera (diesmal einer Infrarotkamera) erfasst und analysiert werden. Das von der Kinect projizierte Muster ist durch ein Patent der Firma PrimeSense urheberrechtlich geschützt [AMY10] [Shp10].
Die Abbildung 3.6, die von den Webseiten[12,13] von Kinect-Entwicklern entnommen wurde, zeigt das Muster, das von der Kinect projiziert wird.

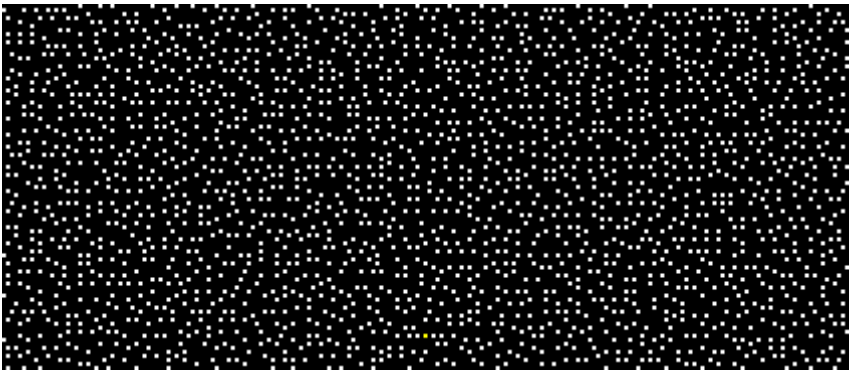


Abbildung 3.6: Infrarot-Punkte-Matrix von der Kinect 1. Quelle: [AMY10] und [Shp10]

Wenn das Infrarotlicht an einer Stelle im Raum reflektiert wird, können die Koordinaten (X, Y, Z) dieser reflektierenden Stelle über ein Triangulierungsverfahren bestimmt werden [Zha12] [HSXS13]. Die Berechnung und Analyse des Musterbildes werden vom Chip PS1080-A2 des Herstellers PrimeSense durchgeführt und als Ergebnis mit dem RGB-Kamerabild (mit einer UXGA-Auflösung von 1600x1200 Bildpunk-

[12] http://azttm.wordpress.com/2011/04/03/kinect-pattern-uncovered
[13] http://www.futurepicture.org/?p=116

ten) zusammengefügt und in RGB-D- (RGB + Depth/Tiefeninformation) Daten umgewandelt (siehe Abbildung 3.7).

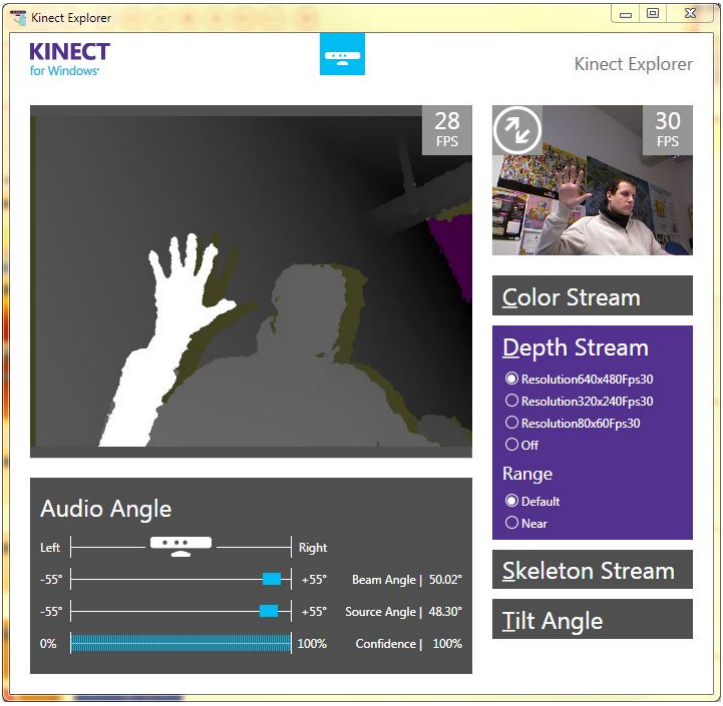


Abbildung 3.7: RGB-D-Bild der Kinect 1 über die Software Kinect Explorer

In der Praxis können diese RGB-D-Daten als Graustufenbild ausgegeben und von einem Algorithmus analysiert werden. Dabei gilt: je dunkler ein Pixel ist, desto näher ist es an der Kamera. Wenn ein Pixel schwarz ist, bedeutet es, dass das entsprechende Objekt zu nah an der Kamera ist. Die Folge ist, dass die Strahlen und deren Entfernung nicht mehr korrekt von der Linse und der IR-Kamera erfasst werden können [Bel14]. Nach Berechnung und Bildvergleich steht nicht nur die reine Bildinformation zur Verfügung, sondern auch die räumliche

Position der einzelnen Pixel [Mac11]. Bei der ersten Version (Kinect 1) besaß die Kamera eine Auflösung von 640x480 Pixel und konnte Kamerabilder mit einer Framerate von 30 Bildern pro Sekunde (FPS) aufnehmen. Die Genauigkeit der Kinect 1 beträgt bei den X, Y- Koordinaten etwa 3mm und in der Tiefe etwa 1cm. Der Arbeitsbereich, den der Sensor erfasst, reicht von ungefähr 0.5 bis 5 Metern [KE12].
Die Kinect 2 (auch Kinect for Xbox One genannt) benutzt ein anderes Verfahren[14], das sog. Time of Flight[15] (kurz. *ToF*-Laufzeitverfahren). Die Time of Flight-Kamera besitzt eine Auflösung von 512x424 Pixel und benutzt dabei Lichtimpulse, die für jeden Bildpunkt die Zeit, die das Licht für den Weg zu einem Objekt (Hindernis) und zurück benötigt hat, messen. Dadurch wird eine dreifach höhere Präzision als bei der Kinect 1 erzielt[16]. Durch die Anwendung von ToF wird es möglich, dass sich Benutzer einerseits weiter entfernt von der Kamera aufhalten können und andererseits mehrere Teilnehmer gleichzeitig verfolgt werden. Die Präzision ist dabei so hoch, dass Falten in der Kleidung des Benutzers erkannt werden[17]. Beide Versionen der Kinect (v1 und v2) können über eine SDK angesteuert und die damit erzeugten Tiefenbilddaten von einem Drittprogramm oder einer Softwarekomponente analysiert werden. Als Ergebnis dieser Analyse erhält der Kinect-Entwickler eine Folge von Punkten (X, Y, Z-Koordinaten) die, wenn diese grafisch zusammen verbunden werden, das menschliche Skelett abbilden. Über die Kinect SDK kann die räumliche Position von 20 Skelettpunkten (d.h.

[14] https://blogs.technet.microsoft.com/microsoft_blog/2013/10/02/collaboration-expertise-produce-enhanced-sensing-in-xbox-one
[15] https://social.msdn.microsoft.com/Forums/en-US/7d4e7c16-1a18-4dca-a312-843f1d6bc172/kinect-for-xbox-one-uses-a-timeofflight-depth-camera-how-does-it-work?forum=kinectv2sdk
[16] http://www.gamasutra.com/blogs/DanielLau/20131127/205820/The_Science_Behind_Kinects_or_Kinect_10_versus_20.php
[17] https://futurezone.at/science/forscher-freuen-sich-auf-neue-kinect/24.598.506

Hand, Schulter, Ellenbogen, Hüfte, Knie und Fuß) abgefragt werden. Die Abbildung 3.8 zeigt eine Visualisierung vom menschlichen Skelett innerhalb einer Testapplikation innerhalb der Kinect-SDK. Alle Skelettpunkte werden von der Kinect-API als Paare von Positionswerten (X, Y, Z-Koordinaten und RGB-Kamerabild) zurückgegeben.

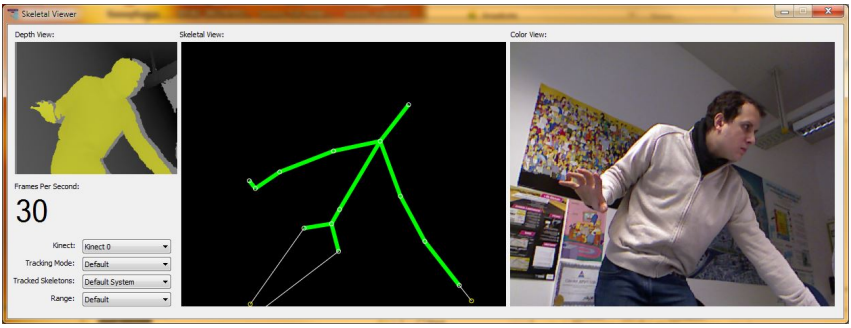


Abbildung 3.8: Bild von Kinect-Skelettpunkten (Kinect 1)

Leitlinien

Die Verwendung der Kinect außerhalb des Spielekontextes und in Zusammenhang mit PC-basierten Applikationen warf anfangs eine Vielzahl von Fragen bezogen auf die Interaktion und Benutzung des Gerätes auf. Aus diesem Grund stellte Microsoft parallel zur Herausgabe des Kinect SDK ein sehr ausführliches Leitlinien-Dokument [Mic15] von über 140 Seiten zur Verfügung, das u.a. alle Interaktionsformen der Kinect wie Sprache, Gesten sowie multimodale und kollaborative/-kooperative Benutzereingaben beschreibt und auch UX-Vorschläge in Form von Bildern und Grafiken vorstellt. In der folgenden Auflistung werden mit Hinblick auf die konkrete Entwicklung des semantischen Fernsehsystems *Swoozy*, die wichtigsten UX-Leitlinien exemplarisch vorgestellt. Die Leitlinien und deren Erkenntnisse können auf andere

gestenbasierte Ansätze und Hardware wie z.B. die Asus Xtion Pro[18] oder Softkinetic[19] transferiert und benutzt werden. Die Hauptinteraktionsmöglichkeit der Kinect ist die Gestensteuerung. Unter Gestensteuerung versteht man die Erfassung der Bewegung des kompletten Körpers eines Benutzers. Bei der Kinect wird zwischen zwei Modi unterschieden, die eine direkte Auswirkung auf die Erkennung der Gesten haben. Einerseits dem „normalen" Modus, bei dem der Benutzer stehend mit dem System interagieren kann, und andererseits dem sitzenden Modus, bei dem der Benutzer sitzend und in einem relativ nahen Bereich der Kinect interagieren kann (von 0.4 bis 2 Meter bei der Kinect 2). Bei dem *seated*-Modus werden nur die oberen Körperteile für die Bild- und Bewegungsanalyse benutzt. Die Leitlinien unterscheiden beide Modi und liefern ausführliche Informationen über die Ergonomie und die Position des Körpers, die es zu beachten gilt, wenn der Benutzer mit einem Kinect-basierten System interagiert. Die Gesten können über die SDK individuell vordefiniert werden und in Kombination mit einer Benutzerschnittstelle bestimmte Aktionen auslösen. Die Herausforderung für Entwickler besteht darin, die Gesten, die für eine Aktion benötigt werden, besonders komfortabel zu gestalten. Das führt dazu, dass der Benutzer sehr leicht mit dem System interagieren soll und nicht erst das System erlernen muss. Um Ermüdungserscheinungen bei der Interaktion zu minimieren, dürfen keine „größeren Gesten" (z.B. die Arme hoch in die Luft strecken, um ein Element in der oberen Ecke eines Menüs zu bedienen) vom Benutzer gefordert werden. Gleiches gilt für Gesten, die mehrmals ausgeführt werden müssen. Hierfür sollte eine Position der Hand in der Nähe des Körpers ausgewählt werden.

[18] `https://www.asus.com/de/Multimedia/Xtion_PRO_LIVE`
[19] `http://www.softkinetic.com`

Zusätzlich sollten sich die Gesteninteraktionen untereinander immer klar unterscheiden, um Fehlinterpretationen zu vermeiden. Kombinierte Gesten wie z.B. das Winken und das gleichzeitige *Wischen (engl. Sliden)* sind zu vermeiden ebenso eine längere Benutzung beider Hände, um repetitive Aufgaben (wie z.B. Zoomen oder das *„Pinchen"* von Bildern) auszuführen.
Wegen der physikalischen Begrenzungen der Kinect (Sichtfeld der Kamera) und der Okklusionsproblematik (d.h. Körperteile, die nicht von der Kamera erfasst werden können, weil andere Körperteile sich vor diesen befinden) müssen bei der Konzipierung und Implementierung von Kinect-basierten Benutzerschnittstellen folgende Punkte beachtet werden:

1. Überlappende Körperteile wie z.B. eine Hand vor der Brust werden schlecht erkannt und infolgedessen wird die Bestimmung der Skelettposition ungenauer. In diesem Fall ist es sinnvoll, den Benutzer darüber zu informieren, dass er eine andere Körperstellung einnehmen sollte. Diese Aufforderung kann z.B. über eine kleine Animation realisiert werden.

2. Die Kinect 1 kann 30 Bilder pro Sekunde erfassen, die Kinect 2 dagegen 60 Bilder pro Sekunde. Trotzdem sollten schnelle Interaktionen vermieden werden, um erstens die Erkennungsleistung nicht zu überfordern und zweitens aufgrund der potentiellen Ermüdung des Benutzers durch das Ausführen schneller Bewegungen. Lange andauernde Posen für die Selektierung eines Buttons (z.B. 5 Sekunden die Hand hochhalten) sollten sparsam und gezielt eingesetzt werden.

3. Bestimmte Handgesten können kulturell bedingt negativ konnotiert sein. Hier wird empfohlen, falls Applikationen international

eingesetzt werden, die Bedeutung der Gesten vorher zu überprüfen und ggf. die Interaktionsgesten leicht zu verändern. Ein prominentes Beispiel ist das sog. „Zero-Zeichen" (Daumen und Zeigefinger formen dabei eine Null), das in manchen Ländern als Beleidigung betrachtet wird.

4. Lange „scrollbare" Listen oder sog. „slidebare" Schaltflächen, die mehrmals eine schnelle Handgeste nach unten (oder nach oben) verlangen, sollten generell vermieden werden. Dieser Aspekt sollte schon direkt bei der Konzeption der Benutzerschnittstelle (UI) mitberücksichtigt werden.

5. Die Spracheingabe spielt bei der Kinect eine wichtige Rolle und gibt Entwicklern die Möglichkeit, Systeme zu implementieren, die multimodal zu bedienen sind. Die Leitlinien konzentrieren sich auf die kombinierte Benutzung von Sprache und Gestik. Dabei wird nur die Erkennung von Sätzen oder sprachlichen Äußerungen in Form von Sprachkommandos berücksichtigt und nicht die Kombination derselben, z.B. bei einer deiktischen Geste. Grund dafür ist, dass eine rein sprachzentrierte Interaktion ohne eine Alternativeingabe über Gesten im Falle einer Fehlerkennung der Spracheingaben zu Frustration beim Benutzer führen könnte.

   Bei der Auswahl der Kommandos sollten die Entwickler Wert auf klare und präzise Kommandos und Vokabular legen. Die Anzahl der Kommandos pro Bildschirm oder Menüpunkt sollte mit Absicht gering gehalten werden, denn der Benutzer wird während seiner Interaktion nicht in der Lage sein, sich längere Befehle zu merken.

   Aus diesem Grund wird dazu geraten, jederzeit die möglichen Kommandos in textueller Form auf der Benutzeroberfläche ein-

zublenden. Dies gewährt eine Unterstützung der Interaktion und liefert dem Benutzer ein direktes Feedback. Es ist ratsam, erkannte Kommandos anzuzeigen, um gegebenenfalls den Benutzer zu einer Korrektur der Eingabe durch Wiederholung des gesprochenen Wortes oder Kommandos aufzufordern. Bei einer Fehlerkennung durch das System können dabei Vorschläge à la *„meinten Sie…?“* für den Benutzer sinnvoll sein.

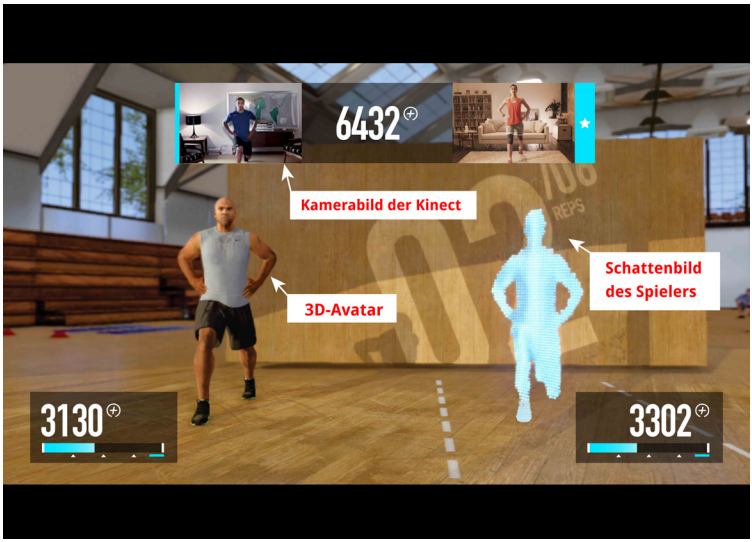


Abbildung 3.9: Avatar-Darstellung, Schattenbild und Kinect-Kamerabild eines Benutzers in Nike Training für Xbox. Quelle: Nike.com

6. Bei jeder Interaktion sollte dem Benutzer entweder ein visuelles oder ein akustisches Feedback gegeben werden. Dies lässt sich z.B. in Form einer Farbveränderung der per Geste auswählbaren Schaltelemente innerhalb der Benutzerschnittstelle oder eines Tons (z.B. Einrast-Sound) realisieren. Da sich die Körperstellung der einzelnen Benutzer während der Zeit verändern kann, wird

empfohlen, dem Benutzer ein visuelles Feedback über seine aktuelle Position (aus Sicht der Kamera) zu geben. Im Spiel *Nike Training* wird ständig die Position des Benutzers in Form eines virtuellen „Schattenbildes" eingeblendet (siehe Abbildung 3.9).

Einen weiteren räumlichen Marker bildet die Einblendung einer halbtransparenten Silhouette bzw. der Umrisse des Spielers innerhalb der Bedienungsoberfläche. Diese Darstellung begrenzt die virtuelle Zone (eigentlich die Sichtzone der Kinect), in welcher sich der Benutzer während einer Interaktion aufhalten muss. Wird diese Zone nicht respektiert, wird der Benutzer aufgefordert, seine Körperposition dementsprechend zu verändern, sodass das Personentracking der Kamera problemlos fortgeführt werden kann. Als Alternative, die Positionierungsprobleme des Benutzers leichter zu beheben, kann das Spiegelbild oder besser gesagt das, was die Kamera vom Raum sieht in die grafische Benutzeroberfläche eingeblendet werden (siehe Abbildung 3.9).

7. **PHIZ: Kinect Physical Interaction Zone**

   Nach den ersten Entwicklungen von Kinect-gestützten Applikationen entstand das von Microsoft selbst geprägte Konzept der *Physical Interaction Zone* (kurz PHIZ) (siehe Abbildung 3.10). Diese Zonen grenzen den physischen Bereich ein, in dem Aktionen von Benutzern ausgelöst werden können und in denen ein sehr schnelles Feedback gegeben werden sollte.

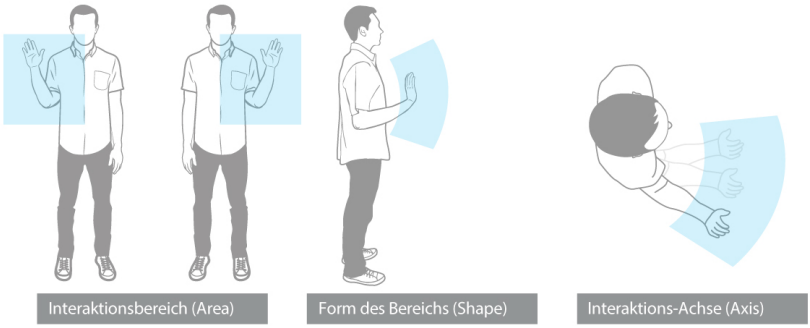


Abbildung 3.10: Physical Interaction Zones bei der Kinect. Quelle: Adaptiert von Microsoft – Kinect For Windows Human Interface Guidelines.

Die PHIZ befinden sich im Normalfall in einer Zone, die von der Mitte des Kopfes bis zum Nabel reicht und in der sich die Hand bewegen kann. Diese Zonen sind benutzerabhängig, d.h. die Distanzen und Skelettpunkte, die von der Kinect-Kamera berechnet und erkannt werden, sind je nach Geschlecht und Alter des Benutzers verschieden. Die Erkennung der Zonen kann durch ein Kalibrierungsverfahren verfeinert werden.

Diese Kalibrierung kann entweder automatisch erfolgen, indem der Winkelgrad und das Sichtfeld (Fokus) automatisch über die in der Kinect eingebauten Motoren per Software verändert oder pro-aktiv vom Benutzer initiiert werden, indem er sich neu im Raum und vor der Kamera positioniert. Diese Neupositionierung kann über die grafische Benutzerschnittstelle unterstützt und interaktiv aufgefordert werden.

	Kinect Region (px)		Button (px)
	Width	Height	Width/Height
UHDTV	7680	4320	880
(W)QHD	2560	1440	294
1080p	1920	1080	220
WSXGA+	1680	1050	214
HD+	1600	900	184
WXGA+	1440	900	184
WXGA	1366	768	157
720p	1280	720	147
XGA	1024	768	157
SD	720	480	98
HVGA	480	320	66

Abbildung 3.11: Kinect-Skala Quelle: Microsoft – Kinect For Windows Human Interface Guidelines.

Während der Designphase einer gestengesteuerten Benutzeroberfläche muss die Größe der UI-Elemente in Korrelation zu der PHIZ an die Bildschirmauflösung angepasst werden. Hierfür muss eine Skala benutzt werden: tatsächlich bleibt die Entfernung zum Bildschirm in der Regel konstant, jedoch nicht die Auflösung der Anzeige, die sich von HVGA (480x320) auf UHDTV (7680 x 4320) erweitern kann (siehe Abbildung 3.11).

Bei steigender Auflösung werden die grafischen Elemente immer kleiner und somit schwieriger für den Benutzer per Gesten zu treffen bzw. zu selektieren. Aus diesem Grund wird ein Skalierungsfaktor zwischen Anzeigeauflösung, d.h. dem, was der Benutzer sieht, PHIZ, d.h., welche UI-Elemente vom Benutzer per Geste leicht erreicht werden können, und Interaktion, d.h.,

welche Gesten der Benutzer ausführen muss, definiert.

Dieser Faktor hilft dabei, die richtigen Dimensionen der Benutzerschnittstelle zu bestimmen. Die Problematik der Gestaltung des UI in Hinblick auf Benutzerinteraktionszonen und Höhe der Auflösung wird 10-foot Design genannt und detaillierter in [Sam13] skizziert. Ohne diese Skalierung müsste der Benutzer vor seinem Bildschirm größere Gesten ausführen, um die einzelnen Elemente (virtuell) auswählen zu können. Als Lösungsvorschlag wird in [Mic15] eine Möglichkeit zur Normierung dieser Skalierung vorgeschlagen.

Fazit

Die Tiefenbildkamera-basierten Systeme wie die Kinect ermöglichen es gestengesteuerte Benutzerschnittstellen zu realisieren. Obwohl diese Systeme relativ neu auf dem Markt sind, werden den Entwicklern verschiedene Ansätze angeboten, um benutzerfreundliche Bedienkonzepte zu implementieren und diese konkret mit fernsehbasierten Benutzerschnittstellen zu verknüpfen. Zusätzlich wurde gezeigt, dass die technischen Randbedingungen einer Tiefenbildkamera (Sichtwinkel) und deren Benutzung (z.B. PHIZ, physische Müdigkeit) während der Konzipierung gestenbasierter grafischer Benutzerschnittstellen berücksichtigt werden müssen. Die Realisierung eines gestenbasierten Systems ist als einheitliches Modul zu betrachten. Dies bedeutet zugleich, dass die grafische Benutzerschnittstelle und ihre Steuerung per Gesten ein stimmiges Bedienungskonzept vorweisen müssen und keinesfalls getrennt berücksichtigt werden können.

Leap Motion

Prinzip

Neben der vorgestellten Tiefenbildkamera, die den kompletten Körper zur Eingabesteuerung benutzt, bildet die Leap Motion eine kostengünstige Alternative, um eine grafische Oberfläche mit den Händen oder Fingern zu bedienen (engl. *Fingertracking*).

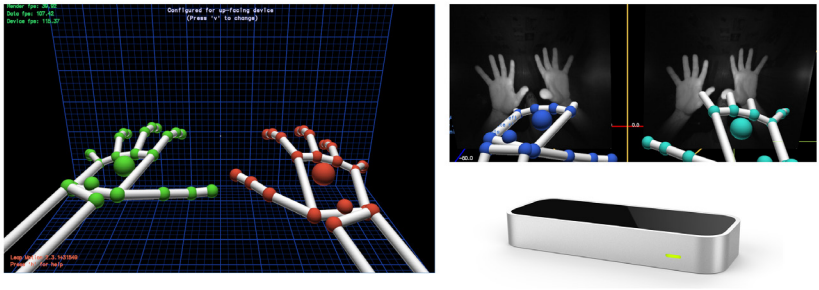


Abbildung 3.12: Leap Motion Gerät (rechts). Schwarz-Weiß Bild der Rohbilder der Leap Motion Kamera mit Skelett-Visualisierung der Hand (links)

Die Leap Motion wird sehr oft mit der Kinect verglichen, obwohl es Unterschiede bei der internen Hardware und der technischen Realisierung der Erkennung gibt. Das Leap Motion-Gerät lässt sich per USB an einen Computer anschließen und ermöglicht die Erkennung und Berechnung der Positionen von Händen, Fingern aber auch kleineren Werkzeugen (z.B. eines Bleistifts) in einem 3D-Raum. Zusätzlich können die Gesteninteraktionen (*Circle, Swipe, Key Tap* und *ScreenTap*) über den eingebauten Gesten-Klassifikator (dieser enthält per Default nur eine beschränkte Liste an vordefinierten Gesten, die ohne Lernphase erkannt werden können) erfasst werden.

Technische Realisierung

In der Leap Motion sind zwei Schwarz-Weiß-Kameras und drei Infrarotstrahler[20] eingebaut. Die Leap Motion kann als stereoskopisches Eingabegerät bezeichnet werden, da die Position der Hände mittels der zwei Kameras berechnet wird. Anders als bei der Microsoft Kinect wird keine Punktwolke oder Muster von den Infrarot-LEDs produziert. Die Verarbeitung erfolgt in der Leap Motion Controller Hardware selbst. Bei den ersten Versionen des Leap Motion Controllers betrug die Ausgaberate der Sensordaten etwa 300 Bilder pro Sekunde (engl. FPS). Aktuell beträgt die Ausgaberate, je nach Konfiguration und Firmware, ca. 115 Bilder pro Sekunde[21]. Zusätzlich gibt es eine Latenzzeit von 30 Millisekunden[22], um leichte Störfaktoren wie z.B. das Zittern der Hand zu reduzieren und somit dem Benutzer ein besseres Interaktionserlebnis zu gewährleisten. Die erfassten Positionen der Hand werden von der internen Hardwarekomponente analysiert und per USB 2 oder USB 3 zum Rechner weitergeleitet. Aus dieser Analyse entsteht eine Abbildung der Position der Hände und der Finger in einem 3D-Koordinatensystem, das von einer Applikation als Eingabe benutzt werden kann. Zu einer besseren Erkennung der Finger müssen sich die Hände optimaler Weise in einer Entfernung von 25 bis 60 Millimetern über dem Eingabegerät befinden (siehe Abbildung 3.13). Die Präzision des Leap Motion Controllers beträgt etwa 0.7 mm [GJP+14].

[20] http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/
[21] https://community.leapmotion.com/t/sample-period-frames-frequency/3281
[22] http://blog.leapmotion.com/understanding-latency-part-1

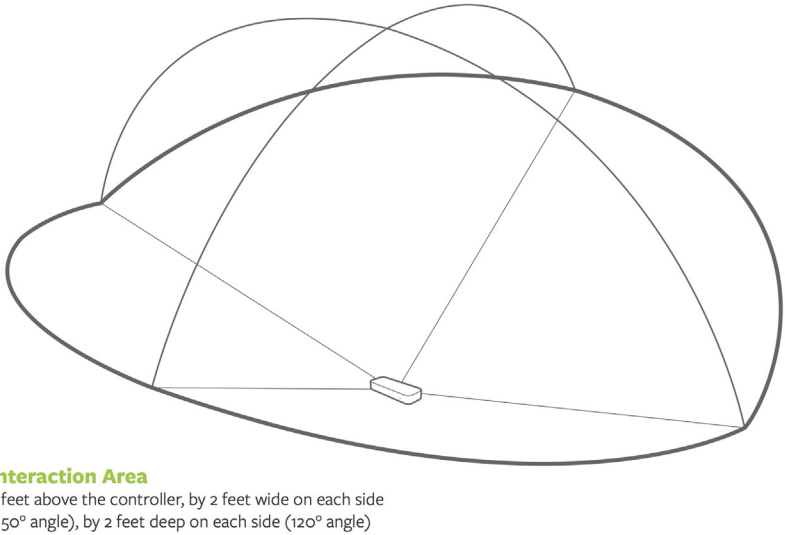


Abbildung 3.13: Sichtfeld der Leap Motion. Quelle: LeapMotion.com

Werden Hände oder Finger erkannt, werden die jeweiligen 3D-Positionen in eine JSON-Struk-tur (ein Datenformat für den Datenaustausch zwischen Anwendungen) vom Leap Motion Controller zusammengestellt und können von anderen Programmen weiterverarbeitet werden. Vorteil dieses Ansatzes ist, dass sich diese 3D-Positionen sehr leicht in den unterschiedlichsten Programmiersprachen (z.B. C# in Kombination mit Unity, ActionScript, Javascript, Perl, Python, Objective-C oder Android) verarbeiten lassen. Das offizielle SDK von Leap Motion bietet den Zugriff auf Roh- und ungefilterte Daten und Schwarz-Weiß-Bilder der eingebauten Kamera. Somit ist es möglich, eine bessere Erkennung zu erzielen. Zusätzlich ermöglicht dies den Programmierern parallel zur reinen Infrarot-basierten-Erkennung weitere Bildanalyseverfahren für die Erkennung zu benutzen. Die Leap Motion Applikationen können entweder frei im Netz von den jeweiligen Entwicklern zur Verfügung

gestellt oder über die Plattform *Leap Motion Airspace* zum Verkauf angeboten werden. Innerhalb der Leap Motion dedizierten *AppStores* können Benutzer Applikationen herunterladen, die exklusiv nur über Fingertracking angesteuert werden. Diese Applikationen reichen von Spielen über Musik bis zu pädagogischen Apps. Durch ihren Formfaktor ist die Leap Motion primär für eine Navigation vorm Bildschirm geeignet, obwohl erste Prototypen und Interaktionssysteme zeigten, dass sich diese im Automobilkontext[23] oder in Kombination mit Tablets im Industrie 4.0 Szenario [HBPH14] verwenden lässt.

Interaktion und UX-Leitlinien

Ähnlich wie bei der Kinect von Microsoft liefert die Firma Leap Motion Guideline-Dokumente, damit Entwickler ihre Apps intuitiver gestalten können. Diese Empfehlungen sind insofern interessant, als sie zum ersten Mal Richtlinien zur besseren Gestaltung von grafischen Benutzerschnittstellen mit Fingertracking-Lösungen ausspezifizieren. Da die Interaktion mit der Leap Motion den Benutzer leicht verwirren kann (er benutzt kein physisches und haptisches Medium wie eine Maus, um einen Cursor zu bewegen, sondern nur seine Finger) fokussieren die Leitlinien auf zwei Kernaspekte der Benutzerschnittstelle: einerseits die Treffsicherheit und andererseits die Abgrenzung von der traditionellen Mausinteraktion durch die Berücksichtigung folgender Elemente:

- **Menügestaltung und Interaktionsformen**

  Die grafische Platzierung der einzelnen Schaltelemente (Buttons, Menü) muss so gestaltet werden, dass der Benutzer jederzeit in

[23] `http://www.gizmag.com/volkswagen-golf-r-touch/35472/`

der Lage sein sollte, diese Elemente zu treffen und zu selektieren, ohne dass diese Interaktion falsch vom System interpretiert werden kann. Dazu müssen die Schaltelemente besonders groß und innerhalb einer klar definierten Zone erreichbar sein. Diese Zonen werden als *Touchzone* oder *Poke* bezeichnet. Eine Alternative zu großen Schaltflächen ist die sogenannte Aufteilung nach dem Nähe-Prinzip (*Proximity-based Highlighting*). Hierfür werden deutlich größere Zonen um jedes Schaltelement definiert. Durch diesen Ansatz werden die Probleme bezüglich des Treffens von Schaltelementen behoben[24].

Eine weitere Möglichkeit Menüs benutzerfreundlich zu gestalten, besteht darin, diese in gröbere quadratische Zonen oder in radiale Zonen zu unterteilen[25], wie es in Abbildung 3.14 zu sehen ist. Diese Unterteilung ist besonders gut geeignet, wenn eine Wahl zwischen mehr als vier Einträgen vom Benutzer getroffen werden muss und wenn ein zentrales Hauptelement eine vordefinierte und eindeutige Funktion besitzt wie z.B. die Einblendung des Hauptmenüs. Die Anordnung nach dem Prinzip der radialen Menüs ermöglicht zusätzlich eine schnellere und treffsichere Auswahl der einzelnen Elemente und eine bessere Memorierbarkeit.

- **Feedback und Interaktionsschritte**

  In den Leitlinien der Leap Motion wird darauf bestanden, dass der Benutzer jederzeit ein Feedback der ausgeführten Interaktion erhalten sollte. Hierfür gibt es verschiedene visuelle Möglich-

[24] `https://developer.leapmotion.com/documentation/csharp/practices/Leap_Orientation_and_Tutorial_Guidelines.html`

[25] `http://www.pushing-pixels.org/2012/07/25/the-usability-of-radial-menus.html`

keiten, dieses Feedback darzustellen, wie z.B. die Veränderungen der Farben und der Bildschärfe, das Skalieren des ausgewählten UI-Elements oder die neue räumliche Platzierung des Schaltelements auf der Z-Achse. Ist ein UI-Element auswählbar, sollte dieses für den Benutzer grafisch über eine Einblendung hervorgehoben werden. Bei jeder Leap Motion-Applikation sollte die traditionelle „Mausklick-Metapher" als „Finger-Tap Interaktion" implementiert werden[26].

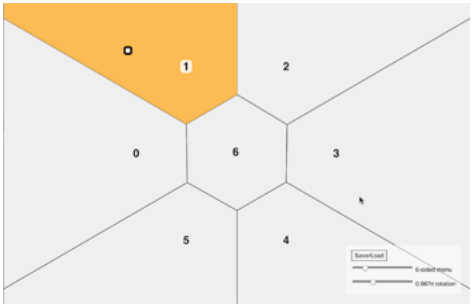


Abbildung 3.14: Beispiel eines radialen Menüs für die Bedienung und Auswahl mit der Leap Motion

Ohne dabei allzu detailliert zu sein, vermitteln diese Leitlinien[27,28] Entwicklern einen guten Ansatz zur Erstellung von Applikationen, die primär mit Fingertracking zu benutzen sind. Hierbei muss beachtet werden, dass aufgrund des Formfaktors der Leap Motion bestimmte Interaktionen mehr oder weniger gut erkannt werden. Auch eventuell auftretende Probleme wie die Okklusion von Fingern, besonders die

[26] http://adamleaps.tumblr.com/post/59536096658/recognizing-taps-with-the-leap-motion-api
[27] https://developer.leapmotion.com/documentation/csharp/devguide/Leap_Overview.html
[28] https://developer.leapmotion.com/documentation/csharp/practices/Leap_Menu_Design_Guidelines.html

des Daumens, müssen bei der Design- und Implementierungsphase berücksichtigt werden.

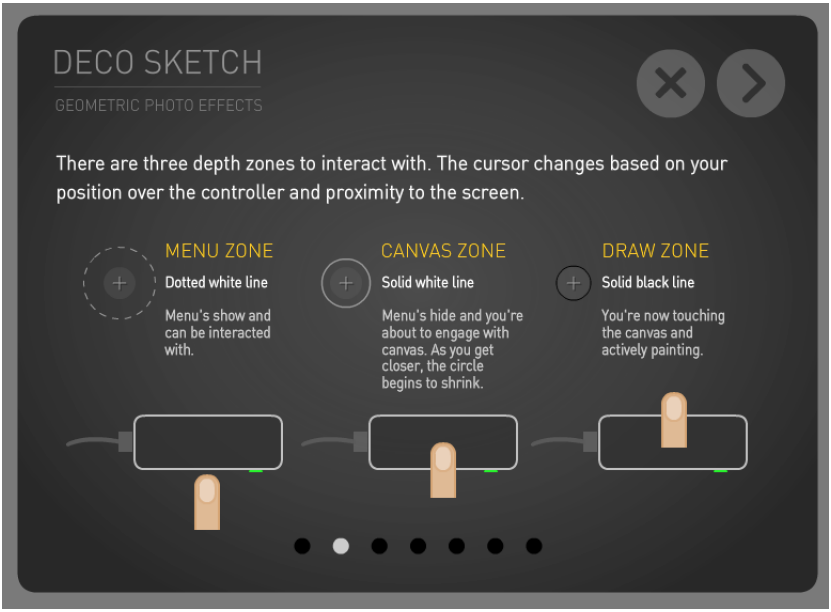


Abbildung 3.15: Zusatzmenüs werden in der Applikation *Deco Sketch* in drei sog. Depth Zones eingeblendet

Neuere Konzepte wie die *Depth Zones*, interaktive Zonen, die es ermöglichen, je nach Z-Raumposition der Hand Kontextmenüs einzublenden, sollten mit Vorsicht benutzt werden (siehe Abbildung 3.15). Es stellt sich dabei schnell heraus, dass die Verwendung des 3D-Raums als Interaktionszone, eine völlig neue Interaktionsmetapher darstellt. Leider wurden bisher sehr viele dieser Apps nicht im Hinblick auf ihre Benutzbarkeit untersucht, sodass es zum heutigen Zeitpunkt schwer zu beurteilen ist, welche der verwendeten Metaphern bzw. 3D-Fingergesten für den Benutzer am intuitivsten und präzisesten für eine Benutzung am Bildschirm sind. Die Abbildung 3.16 zeigt ein Beispiel einer Be-

nutzerinteraktion mit der für die Leap Motion adaptierten Software *Gravilux*[29].

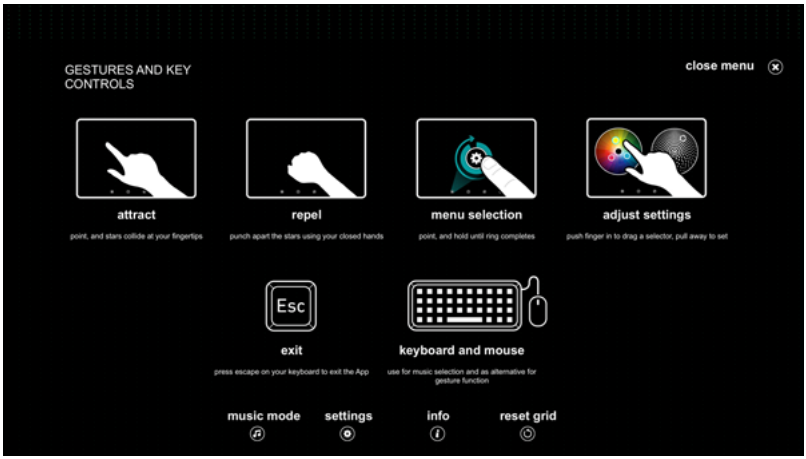


Abbildung 3.16: Hand-Interaktionen zur Ansteuerung der Applikation *Gravilux*

Fazit

Die Leap Motion bildet eine sehr interessante Hardware-Lösung zur Ansteuerung von Software per Finger oder Handbewegung. Anders als bei einem Datenhandschuh (engl. Dataglove) [Qua90] braucht der Benutzer nicht mit Zusatzgeräten ausgestattet zu werden. Die Erkennung der Hände wird über Infrarot und ein Marker-loses Tracking durchgeführt. Das angebotene SDK ermöglicht innerhalb mehrerer Programmiersprachen, die Leap Motion schnell und effizient mit Softwares (Apps) und neuartigen grafischen Benutzerschnittstellen zu verknüpfen. Hier muss, ähnlich wie bei der Kinect, aufgepasst werden, dass bestimmte grundlegende Interaktionsmetaphern beachtet werden. Um den Entwickler zu unterstützen, diese besser zu integrieren, wurden

[29] https://youtu.be/7GTf0WLOfHQ

Leitlinien publiziert. Diese sind nicht sehr detailliert ausgeführt, vermutlich, um die Kreativität der Entwickler nicht gleich einzuschränken - zumal diese preiswerte Fingertrackinglösung für den breiten Markt noch nicht lange verfügbar ist und wenig Feedback zur Benutzerbarkeit innerhalb von Spielen und Grafikprogrammen gesammelt werden konnte. Dies bedeutet gleichzeitig, dass die Entscheidungsmöglichkeiten bei der Gestaltung von Leap Motion-gestützten Applikationen sehr variieren können und immer ein gewisser Freiheitsgrad bei der Konzipierung besteht. Für eine Verwendung im Home-Entertainment-Bereich bildet die Leap Motion eine gute und kompakte Alternative zur Microsoft Kinect, wobei das eingeschränkte Interaktionsfeld der Kamera die Interaktionsmöglichkeiten im Raum sehr stark reduziert.

MYO-Armband

Prinzip

Das MYO ist ein Armband, entwickelt von der Firma Thalmic Labs[30], das auf dem Verfahren der Elektromyographie (EMG) beruht. Bei diesem Verfahren leitet eine auf der Haut angebrachte Elektrode die elektrische Aktivität der Muskelfasern an ein Computersystem weiter. Bei der standardmäßigen Verwendung im medizinischen Bereich wird diese Aktivität gemessen, um Diagnosen über Krankheiten der Nerven oder Muskeln zu erkennen. Auch im Bereich der Sportmedizin oder in der Biomechanik (z.B. für die Parametrisierung von Exoskeletten) kommt diese Art der Aktivitätsanalyse zum Einsatz.

[30] `http://www.thalmiclabs.com`

Abbildung 3.17: MYO-Armband der Firma Thalmic Labs. Quelle: https://www.thalmic.com/press

Die Elektromyographie ist die *„Studie der Muskelfunktion auf der Grundlage der Analyse der von Skelettmuskeln ableitbaren elektrischen Signale“*[31]. Weitere medizinische Details über die Elektromyographie, ihre Anwendung und die Vorteile von Oberflächenelektroden werden in [TCV00] [DL97] [Bas74] [Hah10] und [Kon05] ausführlicher präsentiert.

Die Idee, elektrische Signale der Muskeln zu erkennen und zu benutzen, um ein System per Gesten zu steuern, wurde schon im Jahre 2008 in [STMB08] und [STM+09] erstmals erwähnt und im Rahmen einer Untersuchung als Methode evaluiert. Die benutzte Hardware hatte damals schon die Form eines Armbands mit fest verdrahteten Oberflächenelektroden. Dabei spielte die Anbindung eines Gestenklassifikators eine wichtige Rolle. Die Stimuli (Signale) wurden anhand zweier Klassifikationsverfahren bestimmt. Die Aufgabenstellung war dabei, eine von vier Fingerinteraktionen (*Extend-curl, Tap, Press* und *Lift*) mittels maschinellen Lernverfahrens zu erkennen. Diese Vorgehensweise besitzt eine besonders gute Erkennungsrate (95%) und kann laut

[31] `http://www.hgk-koeln.de/medizin/kliniken/neurologie/neurophysiologische-diagnostik`

Entwicklern für die Früherkennung von Muskelintensionen und für die Ansteuerung einer Prothese benutzt werden. Das Gleiche gilt für die Erkennung von emotionalen Zuständen, die anhand winziger Kontraktionen der Gesichtsmuskeln analysiert werden, auch wenn diese nicht direkt mit dem bloßen Auge sichtbar sind. Die Analyse der Kontraktionen im Gesicht einer Person öffnet neue Interaktionsmöglichkeiten wie z.B. im Spracherkennungsbereich, indem die EMG-Signale benutzt werden, um gesprochene Sätze unabhängig vom akustischen Signal zu erkennen [MHMSW05]. Der erste Ansatz, EMG als Eingabemodalität für ein Computer-System zu benutzen wurde bereits in [CKL+96] präsentiert und hat zur Entwicklung einer neuen Schnittstelle geführt, dem sog. muCi (Muscle Computer Interface). In [Fre14] wird erläutert, wie EMG-Signale des MYO-Armbandes für die Bewegungserkennung klassifiziert und benutzt werden können. Dabei können physische Eigenschaften wie Fett, Gewebe, Haare oder Schweiß des Benutzers sehr stark die Signale verändern und im schlimmsten Fall verfälschen. Da das MYO-Armband nicht intrusiv ist und nur über 8 Oberflächenelektroden das EMG misst, sollten diese Faktoren vom Entwickler berücksichtigt werden. Ebenfalls findet man eine Diskrepanz bei den Strategien, die benutzt werden, um eine Bewegung (z.B. eine Wink-Geste) auszuführen. Von Person zu Person können diese Interaktionen auf muskulärer Basis anders erfolgen und somit jedes Mal andere Muskelaktivitäten verursachen. Die Herausforderung besteht darin, ein normiertes und klassifizierbares Signal für jede einzelne Geste zu erkennen. Dafür hat Thalmic Labs eine Tabelle erstellt (siehe [Fre14]), welche die besonderen Merkmale einer Geste – oder besser gesagt die daraus resultierende elektrische Aktivität – auflistet und als Anhaltspunkt für eine Erkennung dienen kann.

Technische Aspekte

Das MYO-Armband beinhaltet einen ARM Cortex M4 Prozessor und verfügt über ein haptisches Feedback über einzelne im Armband platzierte Minimotoren. Die Bewegungen der Hand und des Arms werden über ein Gyroskop, einen Beschleunigungssensor (engl. Accelerometer) und einen Magnetometer aufgezeichnet. Zusätzlich werden über einen Elektromagnet elektrische Impulse der einzelnen Muskeln (EMG) registriert und von der mitgelieferten Software ausgewertet. Diese Auswertung erfolgt nachdem die einzelnen Signale vom Armband über die Übertragungstechnologie Bluetooth 4.0 von einer Smartphone-App empfangen worden sind. Die Verwendung des MYO-Armbands zielt eher auf eine *„Hands-Free"* Benutzung im Unterhaltungsbereich, sodass z.B. Musikstücke während einer Fahrt auf dem Snowboard gewechselt oder Spielzeugdrohnen per Handgeste kontrolliert werden können. Durch die Verwendung des EMG-Verfahrens können Muskelaktivitäten im Voraus registriert und erkannt werden, sodass es möglich ist, Befehle intern bereits auszulösen, kurz bevor die eigentliche physische Geste vom Benutzer ausgeführt wird [KT13] [TK13]. Dank des SDK können u.a. Applikationen angesteuert werden, mit denen es möglich ist, auf gröbere Gesten zu reagieren.

Leitlinien

Um Entwickler bei Implementierung und Design von MYO-kompatiblen Applikationen zu unterstützen, publizierte der Hersteller ein kompaktes fünfseitiges Dokument [Tha14] mit Empfehlungen, welches die Grundregeln zur Verwendung des MYO-Armbands auflistet. Ähnlich wie bei der Microsoft Kinect wird beim MYO-Armband empfohlen, ein visuelles Feedback bei Aktionen einzublenden und längere Gestenin-

teraktionen, die zu einer frühzeitigen Ermüdung führen könnten, zu vermeiden. Ein weiterer interessanter Aspekt ist die notwendige Berücksichtigung des Armes an dem das Band angebracht wurde, bevor eine Gesteninterpretation überhaupt stattfinden kann. Die Abbildung 3.18 zeigt die zwölf verwendbaren Gesten (fünf pro Arm und zwei für die Rotation und Schiebegeste, die von beiden Armen durchgeführt werden können), die vom Armband interpretiert und in eine Applikation integriert werden können.

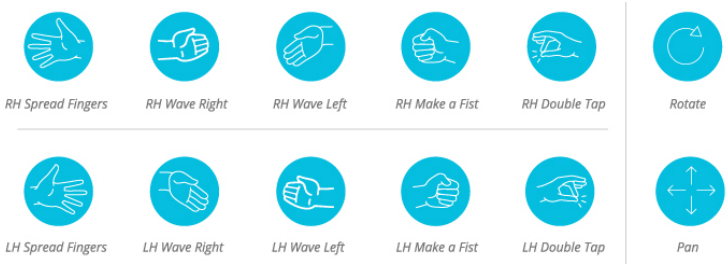


Abbildung 3.18: Auflistung der MYO-Gesten. Quelle: https://www.thalmic.com/press

Fazit

Das MYO-Armband stellt eine Alternative zur Microsoft Kinect oder der Leap Motion dar. Hierbei ergibt sich eine neue Problematik, nämlich die der notwendig gewordenen „Ausstattung des Benutzers mit einem Zusatzgerät" während einer Aktivität, wie z.B. des Fernsehens. Es ist fraglich, ob ein Benutzer während des Fernsehens ein zusätzliches Armband tragen möchte. Eine weitere Herausforderung bildet der Transfer der traditionellen Maus-basierten Interaktionen (Click, Scrollen) auf MYO-Gesten und wie diese in ein 3D-Koordinatensystem

integriert werden können. Beim MYO-Armband existiert kein Referenzenkoordinatensystem: das bedeutet, dass die räumliche Position der Hand nur relativ von einem fest vorgegebenen Startpunkt aus berechnet werden kann. Dies wirft die Frage nach der Realisierung einer präzisen Bedienung und gezielten Selektierung von Menüpunkten innerhalb einer grafischen Oberfläche auf. Aus diesem Grund kann das MYO- Armband in einem Fernsehszenario nur für Kommando-Gestik bzw. Auswahl-Gestik oder für das Auslösen fest definierter Funktionen wie z.B. die Lautstärkeveränderung oder den Kanalwechsel zum Einsatz kommen.

Wii-Remote

Prinzip und Interaktionsarten

Die Wii-Remote, auch kurz *WiiMote* genannt, ermöglicht eine Interaktion mit der Spielekonsole Wii von Nintendo. Die Verbindung zu dieser wird drahtlos über die Bluetooth-Technologie realisiert. Kurz nach Veröffentlichung der WiiMote wurde das Verbindungs- und Vermessungsprotokoll durch *Reverse Engineering* enthüllt. Ab diesem Zeitpunkt war es möglich, über frei verfügbare Programmierschnittstellen wie z.B. die C# Bibliothek WiiMoteLib[32] die von der WiiMote verschickten Daten mit einem PC/Mac oder sogar einem Smartphone analysieren zu lassen. So konnten Applikationen, unabhängig von der Nintendo Wii Konsole, die Eingaben der Wii-Remote bearbeiten und als Eingabe benutzen.

[32] `https://wiimotelib.codeplex.com/`

Abbildung 3.19: Wii Remote von Nintendo

Die Wii-Remote enthält einen 3-Achsen-Beschleunigungssensor (X, Y und Z). Dieser misst den genauen Zustand der Wii-Remote, indem er nicht nur die drei Achsen sondern auch die Drehung (engl. Rotation) und die Bewegungsgeschwindigkeit der Fernbedienung mitprotokolliert und zur Konsole per Bluetooth weiterreicht. Somit ist es möglich, die Bewegung abzufangen und nachträglich zu analysieren. Darüber hinaus besitzt die Wii-Remote neun dedizierte Tasten (inklusive Richtungs-Tasten) und auf der unteren Seite befindet sich die sog. B-Taste, die als Trigger-Steuerung bezeichnet wird.
Innerhalb von Wii-Spielen wird diese B-Taste als Taste zum *„Greifen“* benutzt, z.B. in *Zelda-Ocarina Of Time* (siehe Abbildung 3.21), um eine „Waffe“ zu bedienen, oder wie bei dem Bowlingspiel in Wii Sports, um virtuell eine Kugel zu halten.
Durch gleichzeitiges Betätigen der Tasten A und B kann der Spieler eine sog. „Grab-Interaktion“ durchführen wie innerhalb des *Mii-Konfigurators* (siehe Abbildung 3.20).

Abbildung 3.20: Greif- bzw. Grab-Interaktion mit der WiiMote. Quelle: Nintendo.com

Abbildung 3.21: Greif-Interaktion beim Spiel Zelda-Ocarina Of Time. Quelle: http://wii.mmgn.com/Lib/Images/News/Normal/Zelda-Skyward-Sword-coming-after-Ocarina-of-Time-3DS-1064630.jpg

Neben dem direkten visuellen Feedback, das grafisch auf dem Fernseher dargestellt wird, kann ein akustisches Feedback von der Konsole über die WiiMote erzeugt werden. Zusätzlich wird über Vibrationsmotoren ein haptisches Feedback ausgelöst, z.B. wenn der Spieler virtuell einen Gegner getroffen hat oder sich über einem Schaltelement befindet. Die Bestimmung der 3D-Position der WiiMote im Raum und die Berechnung eines In-Game Cursors erfolgt über eine Infrarotleiste. Diese Leiste ist fest installiert (z.B. über den TV-Bildschirm) und sendet kontinuierlich Infrarot-Lichtsignale. Die WiiMote selbst besitzt eine Infrarotkamera mit einer Auflösung von 1024x768 Pixeln. Die Position der WiiMote im Raum wird durch diese Infrarotkamera bestimmt, indem die räumliche Position der Infrarotleiste erfasst und relativ genau zum aktuell erfassten Kamerabild berechnet wird.

Dieses Prinzip lässt sich umgekehrt anwenden. Die WiiMote kann dabei fixiert bleiben und die Infrarotquelle bewegt sich. Mit diesem Prinzip lassen sich Szenarien wie ein interaktives Whiteboard[33] mithilfe von kleineren Infrarotstiften realisieren [Lee08]. Mit zwei Infrarotquellen kann dies als kostengünstige Headtracking- oder 3D-Fingertracking-Alternative [HSS+09] verwendet werden [DASLP10] [GDPM08].

Die Anbindung der WiiMote ohne Wii-Konsole über Bluetooth an einen PC oder Mac-Rechner wurde weder offiziell von Nintendo unterstützt noch untersagt. Dies erklärt, warum keine Leitlinien und Informationen für die Benutzung der WiiMote in Zusammenhang mit PC/Mac-basierten grafischen Benutzerschnittstellen existieren. Als Grundlage können sich Programmierer von Interaktionsparadigmen innerhalb der Wii-In-Game-Menüs inspirieren lassen. Diese nutzen das komplette Potenzial der WiiMote, inklusive akustischem Feedback und Vibration, und zeigen, welche Interaktionsmetaphern über die WiiMo-

[33] `http://johnnylee.net/projects/wii/`

te erzielt werden können.

Fazit

Die WiiMote mit ihrem Formfaktor und ihrer Haptik ähnelt am ehesten dem Konzept der klassischen Fernbedienung und ist für eine Benutzung in Zusammenhang mit einer Fernseh-basierten Benutzerschnittstelle geeignet. Zugleich bedeutet es für den Benutzer nur eine minimale kognitive Umstellung. Über Programmierschnittstellen können die Daten der WiiMote in unterschiedlichen Programmiersprachen analysiert und interpretiert werden. Diese Interpretationen der Benutzergesten können mittels Klassifikatoren [NA09] [Nes08] [NLS+11] [SvWMB09] verfeinert werden.
Das Prinzip der Gyroskop-basierten Fernbedienung, das zuerst bei der WiiMote zu finden war, wurde mittlerweile von vielen Fernsehherstellern wie z.B. LG mit seiner Magic Remote[34] oder Samsung mit der Smart Touch Control-Fernbedienung übernommen und gehört zu den Standardeingabegeräten zur Bedienung von sog. Smart-TVs. Ein direktes haptisches Feedback bekommt der Benutzer durch Vibrationen. Nachteil der Interaktion mit der Wii-Mote und der Fernbedienung ist, dass der Benutzer im Gegensatz zu der Kinect oder Leap Motion immer ein Zusatzgerät für die Steuerung der Benutzerschnittstelle benutzen muss.

[34] http://www.lg.com/us/tv-accessories/lg-AN-MR500-magic-remote

Weitere Verfahren zur Interaktionen mit dem Fernseher

Neben der Möglichkeit Gesten für die Interaktion mit dem Fernseher zu benutzen, gibt es weitere Hardware-Lösungen, die es theoretisch ermöglichen würden, das Fernseherlebnis anzureichern. Diese Technologien werden als disruptive bezeichnet, denn sie beeinflussen durch ihre unterschiedlichen Eingabemöglichkeiten die Art und Weise, wie Fernsehen konsumiert wird. Diese werden im folgenden Abschnitt präsentiert, wobei die Benutzung dieser spezifischen Hardware in Zusammenhang mit dem semantischen Fernsehen nicht Bestandteil der technischen Implementierung von Swoozy ist.

Eye-Tracking

In [PB06] wird Blickerfassung (engl. Eye-tracking) als „*Technik bei der die Augenbewegungen gemessen werden, damit man jederzeit erfahren kann, wohin eine Person an einem bestimmten Zeitpunkt schaut und die Sequenz in der die Augen der Person sich von einer Position zu einer anderen bewegen ermitteln kann*“ bezeichnet. Es gibt mehrere Eyetracker-Arten und Verfahren die Position der Augen zu bestimmen.

Abbildung 3.22: Eye-Tracker-Lösung von TheEyeTribe. Quelle: www.theyetribe.com

Die Eye-Tracker wie z.B. die von Tobii[35] oder The EyeTribe[36] funktionieren auf der in [Duc07] präsentierten Methode der Vermessung der Reflexion der Netzhaut, die extern von einer Infrarot Lichtquelle bestrahlt wird. Durch die Reflexion auf der Hornhaut und die Position der Pupillen ist es möglich, die Position der Augen zu bestimmen. Tobii und The EyeTribe bieten eine Programmierschnittstelle und sehr ausführliche Leitlinien an [Tob14]. Es ist denkbar Eye-Tracker im Kontext des Fernsehens zu benutzen, wie z.B. für die Selektierung eines grafischen Elements nach einem längeren Blick. Die Selektierung per Blick sollte jedoch nur als Unterstützung zu einer anderen Eingabemodalität wie z.B. der Sprache oder einer Zeigegeste dienen, da es einen Konflikt zwischen dem Blick, der auf die Videowiedergabe gerichtet ist, und dem Blick, der auf ein grafisches Element gerichtet ist, gibt. Wegen Ihrer technischen Einschränkung bzgl. der Distanz zwischen dem Auge und der Infrarot Lichtquelle sind Eye-Tracking eher für den Nahbereich einsetzbar.

35 http://www.tobii.com
36 http://theeyetribe.com

Google Glass at Work

Google Glass ist eine intelligente Brille (siehe Abbildung 3.23), die einen Zugriff über verschiedene Apps und Dienste interaktiv ermöglicht. Die erste Version war mit einem 640x320px Mikrodisplay, einer Digitalkamera, um Fotos und Videos aufzuzeichnen, sowie einem vollfähigen Android Betriebssystem ausgestattet. Eingaben können entweder per Sprache oder über einen Touchpad erfolgen. Zusätzlich besitzt diese Brille verschiedene Sensoren wie z.B. einen Lage- und Näherungs-Sensor, um den Benutzungskontext des Gerätes zu bestimmen und somit spezifische Apps und Informationsanzeigen auszulösen. Das hochauflösende Display ist äquivalent zu einem 25 Zoll HD Monitor, der von einer Entfernung von 2,43 m (8 Füßen) betrachtet wird[37].

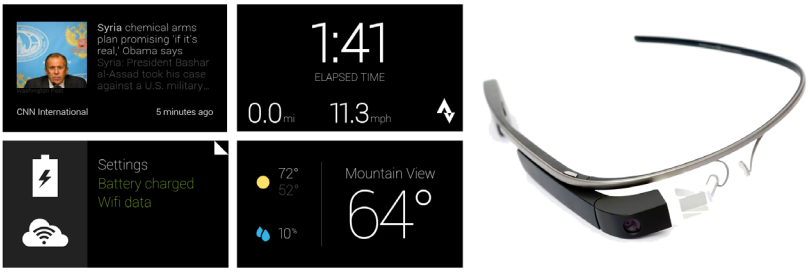


Abbildung 3.23: Google Glass At Work. Quelle: https://developers.google.com/glass/distribute/glass-at-work

Eine neuere für den professionellen Bereich angepasste Variante der Brille wird von Google unter dem Namen Google Glass at Work[38] angeboten.

Eine Augmented Reality (dt. erweiterte Realität) oder begleitende Ap-

[37] https://support.google.com/glass/answer/3064128?hl=en&ref_topic=3063354

[38] https://developers.google.com/glass/distribute/glass-at-work

plikation (engl. companion app) im Kontext einer parallelen Benutzung von Fernsehen wäre denkbar. Der mit solch einer Brille ausgestattete Zuschauer bekäme in einer Ecke seines Blickfeldes zusätzliche Informationen zum Fernsehprogramm. Richtet er seinen Blick in Richtung des Fernsehers, verschwindet diese eingeblendete Information. Somit ließen sich auch Kontextinformationen blickabhängig, personalisiert und benutzerzentriert einblenden.

Microsoft HoloLens

Die Microsoft HoloLens[39] ist eine Augmented-Reality Brille, die verschiedene Eingabemodalitäten unterstützt, wie z.B. die Gesten-, Blick (engl. Gaze)- und Sprachsteuerung, die einen direkten Einfluss auf die Anzeige von Informationen innerhalb der Brille haben [CLST15] [Got15]. Die Brille ermöglicht die Visualisierung und Manipulationen von Microsoft selbst bezeichneten Hologrammen, d.h. in das Sichtfeld des Benutzers hinzugefügte virtuelle 3D-Objekte, die über Gestensteuerung und Spracheingaben verändert und manipuliert werden können. Man spricht in diesem Fall von gemischter Realität (engl. Mixed-Reality) [MTUK95].
Die Einsetzbarkeit der HoloLens während des Fernsehens könnte auf zwei Ebenen gesetzt werden: einerseits könnten die Medienobjekte und das Fernsehprogramm innerhalb der Brillen dargestellt werden (dabei wird kein physikalisches Fernsehgerät mehr benutzt sondern nur noch eine holographische Projektion, das als Fernsehanzeige dient); andererseits könnte der Benutzer weiterhin sein Fernsehgerät benutzten und „klassisch“ fernsehen. Die eigentlichen Zusatzinformationen würden ihm dann grafisch über das reale Fernsehbild

[39] https://www.microsoft.com/microsoft-hololens

überlagert dargestellt werden. Das Fernsehbild wird, im Sinne von *Augmented Reality*, augmentiert bzw. mit zusätzlichen grafischen Elementen angereichert.

Abbildung 3.24: Microsoft HoloLens. Quelle: https://www.microsoft.com/microsoft-hololens/en-us

Die HoloLens könnte eine Antwort auf die Problematik der Mehrbenutzerfähigkeit des Fernsehers bilden. Zukünftig wäre denkbar, dass jeder Zuschauer seine eigenen projetzierten Zusatzinformationen in seiner HoloLens-Brille erhält, aber trotzdem gemeinsam das gleiche Fernsehprogram sieht. Somit wäre ein personalisiertes Fernseherlebnis möglich, ohne dabei die sozialen Aspekte der zwischenmenschlichen Kommunikation zu hindern. Dieser Aspekt des kooperativen und kollaborativen Interagierens wird von Microsoft als Kriterium für den Einsatz der HoloLens im professionellen Bereich gesehen.
Neben der Microsoft HoloLens existieren andere Technologien wie die Magic Leap[40], die ähnliche Mixed-Reality-Interaktion ermöglichen.

[40] `https://www.magicleap.com`

Smart Chair und smarte Textilien

Die Aktivitäten des Fernsehens finden die meiste Zeit sitzend statt. In [WCZ+14] [CBZL13] und [TLP97] wird die Idee vom Smart Chair präsentiert (siehe Abbildung 3.25).

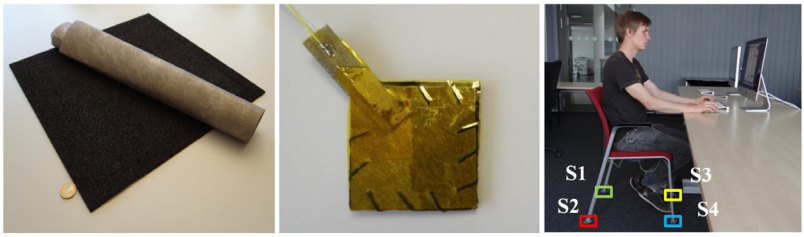


Abbildung 3.25: Beispiel des Smart Chairs mit Sensoren (S1 bis S4). Quelle: [CBZL13]

Dieser intelligente Stuhl ist in der Lage, durch eine eingebaute Sensorik die Bewegungen einer Person zu erfassen und zu klassifizieren. Es wäre denkbar, über eine Körperhaltungsklassifikation bestimmte Haltungen einer Person auf einer Couch zu identifizieren (z.B. nach vorne lehnen, als würde man mit erhöhtem Interesse etwas beobachten) und erst dann eine personalisierte grafische Schnittstelle über das Fernsehbild einzublenden. Dies ließe sich mit dem Proxemik-Ansatz und Mehrbenutzerbetrieb erweitern, ähnlich wie in [MG10] beschrieben. Über sog. smarte Textilien, die über die Couch überzogen werden, könnten Zusatzfunktionalitäten wie z.B. eine „Slide-Geste“ auf der Armlehne aufgerufen werden.

Übersicht: Eingabetechnologien

Die Auswahl einer Eingabetechnologie hängt sehr stark von ihrer Benutzung innerhalb eines fest vordefinierten Fernsehkontexts ab.
Die Abbildung 3.26 stellt die verschiedenen Eingabe- und Interaktionstechnologien, die für eine Benutzung im Fernsehkontext am geeignetsten sind, vor. In diesem Zusammenhang können vier Interaktionsbereiche definiert werden, wobei die Übergänge direkt von Größe und Auflösung des Bildschirms abhängig sind.
Der erste Bereich ist der sog. *Tactile-Bereich*. Dieser Bereich ist ausschließlich für sehr präzise oder Touch-basierte Interaktionen gedacht. Der Benutzer muss sich sehr nah am Fernsehgerät aufhalten, um eine Interaktion auszuführen. Diese Touch-Interaktion Problematik kann wiederum durch ein Second Screen oder die Verwendung von einem Zusatzgerät wie das Touchjet[41] der Firma Wave, das ein Fernsehgerät zu einem Touchscreen umwandelt, gelöst werden. Obgleich der Name des Bereichs, *Tactile-Bereich*, können im Fernsehkontext keine Touchscreens als Eingabegerät benutzt werden, da diese keine sinnvolle Interaktion mit Fernsehinhalten erlauben. Die Leap Motion und die Touchjet-Technologie stellen hierbei eine kleine Ausnahme dar. Obwohl eigentlich für den sehr nahen Bereich konzipiert, kann sie notfalls über ein verlängertes USB-Kabel auch in entfernteren Bereichen benutzt werden (z.B. auf einer Sofa-Armlehne).
Im zweiten Bereich, dem sog. *Nahbereich*, können Kamera-basierte Systeme die Gesten der Benutzer analysieren und diese in Interaktionen umwandeln. Die WiiMote ist für eine Benutzung im *entfernten Bereich* geeignet und verlangt vom Benutzer, im Gegensatz zur Microsoft Kinect oder der Leap Motion, nur wenige Körperbewegungen, um die

[41] `http://www.touchjet.com/wave`

Benutzerschnittstelle zu kontrollieren.
Im letzten Bereich, dem *sehr entfernten Bereich*, wird die Interaktion der Fernsehbedienoberfläche vom Interaktionsgerät teilweise abgekoppelt. Beide Geräte können, müssen aber nicht zwingend, miteinander interagieren.

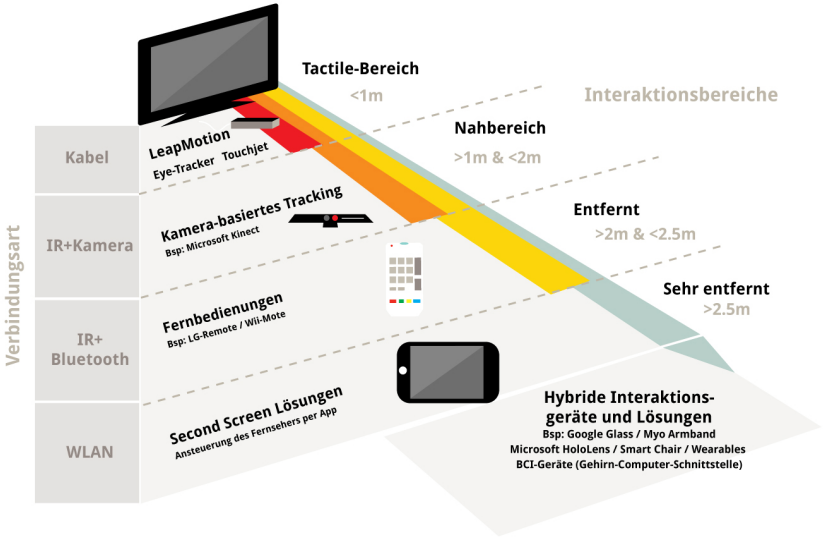


Abbildung 3.26: Übersicht: Eingabetechnologien im Home Entertainment-Bereich

Dazu gehören die Tablet-Applikationen (Second Screen), die sowohl den Fernseher kontrollieren (Lautstärke, Senderauswahl, Menü, usw.) als auch Apps starten können. Hybride Lösungen wie z.B. das MYO-Armband, Google Glass oder Microsoft HoloLens können zwar im Fernsehkontext benutzt werden, der Benutzer muss jedoch mit einem Zusatzgerät ausgestattet werden, d.h. einen zusätzlichen physischen Gegenstand tragen, damit eine Ansteuerung der grafischen Benutzerschnittstelle des Fernsehers realisiert werden kann. Diese Art der

Interaktion ist im Wohnzimmerszenario nicht wünschenswert und aus Interaktionssicht nicht effizient.

Die Tabelle 3.27 fasst alle vorgestellten Steuerungstechnologien zusammen, die im Kontext einer Benutzung mit einem Fernseher von Nutzen sind. Diese werden nach den Kriterien der Präzision, der Interaktionsart (z.B. Finger oder Hand) und der Kosten aufgelistet. Zusätzlich wird die Präsenz einer Programmierschnittelle und Leitlinien als weiteres Kriterium benutzt.

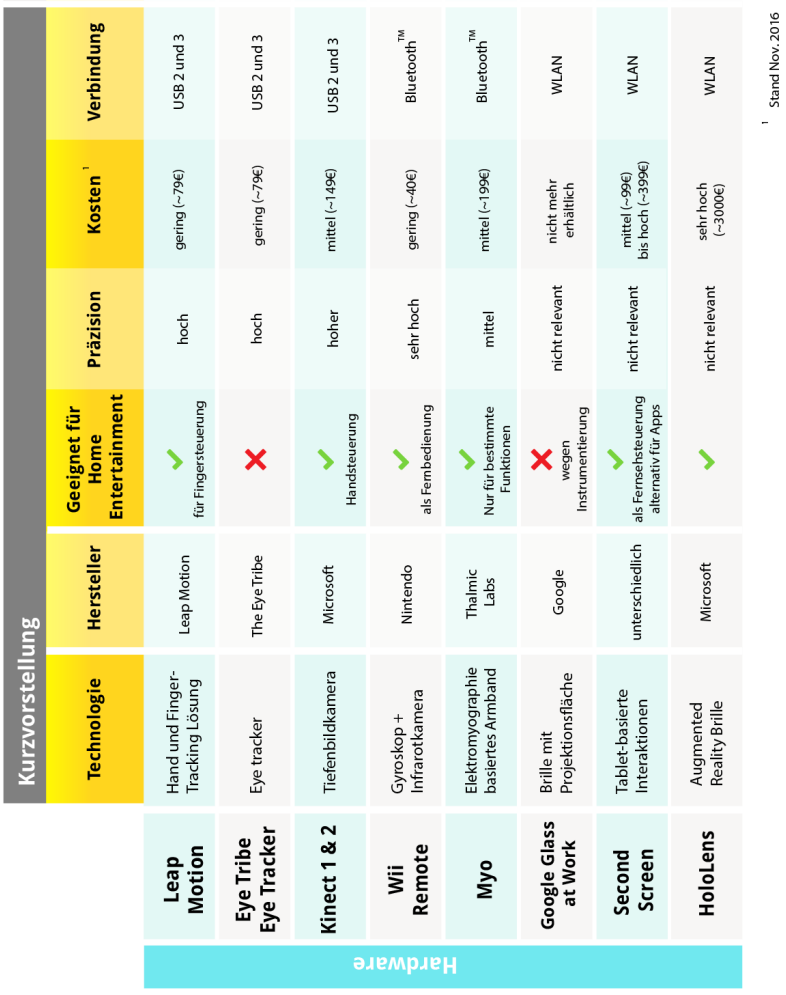


Abbildung 3.27: Vergleich der Technologien im Bereich Interaktionen mit dem Fernseher (1/3).

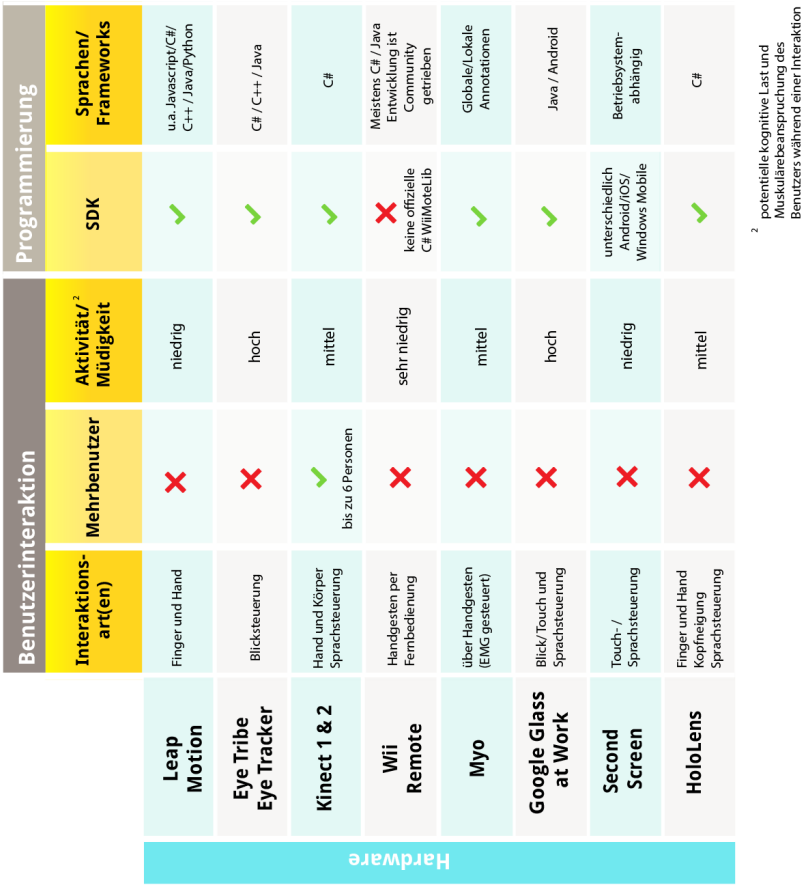


Abbildung 3.28: Vergleich der Technologien im Bereich Interaktionen mit dem Fernseher (2/3).

		UI-Design	Anmerkungen
		Guidelines	
Hardware	Leap Motion	✔ für 3D und VR	Gut für den Fernsehkontext einsetzbar jedoch USB-Kabel Genauigkeit der Erkennung könnte besser sein
	Eye Tribe Eye Tracker	✔	Nicht im Fernsehkontext benutzbar: Konflikt Videobetrachtung und Auswahl von UI-Elementen durch Benutzerblick
	Kinect 1 & 2	✔ sehr ausführliche	Gut für den Fernsehkontext einsetzbar Präzise Auswahl von UI-Elementen über Handgesten
	Wii Remote	✘	Sehr gut für den Fernsehkontext einsetzbar Präzise Auswahl von UI-Elemente über die WiiMote
	Myo	✔	Benutzer muss zusätzliche Hardware tragen: ungeeignet für eine Benutzung im Fernsehkontext
	Google Glass at Work	✔	Benutzer muss zusätzliche Hardware tragen: ungeeignet für eine Benutzung im Fernsehkontext
	Second Screen	nicht relevant	Steuerung des Fernsehers über dedizierte Apps Erweiterte Funktionalitäten dank Second Screen Apps
	HoloLens	✔	Steuerung einer interaktiven 3D-Projektion über Gesten in Umgebung des Benutzers. Benutzer muss zusätzliche Hardware tragen

Abbildung 3.29: Vergleich der Technologien im Bereich Interaktionen mit dem Fernseher (3/3).

Fazit und Analyse

Im letzten Abschnitt wurden technische Lösungen und Ansätze für eine benutzerzentrierte Interaktion vorgestellt. Hierbei wurde primär die Gestensteuerung als Hauptinteraktionsmöglichkeit untersucht. Viele Interaktionsprinzipien wie die der Microsoft Kinect oder der Wii Remote stammen aus dem Spielekonsolenbereich und sind für eine gelegentliche und spielerische Interaktion mit dem Fernseher sehr gut geeignet.

Bei allen Interaktionstechnologien werden sogenannte Leitlinien von den jeweiligen Herstellern vorgeschlagen. Diese werden in Form einer Dokumentation für Entwickler zur Verfügung gestellt und listen präzise auf, welche Gesteninteraktionen, Abstände und Eingabekombinationen benutzerfreundlich und leicht auszuführen sind. Hierbei liegt der Fokus auf Spielen. In den meisten Fällen wird die Verwendung des *10-foot Design* empfohlen, welches dank größerer Schriftarten und Schaltelementen (Buttons oder Leisten) eine leichtere Bedienung auf größeren Bildschirmen ermöglicht [Lev14]. Dieses Prinzip wurde für die Programmierung der Benutzerschnittstellen der Kinect von Microsoft oder der Leap Motion zu Grunde gelegt. Bei den Interaktionsformen muss beachtet werden, ob der Benutzer ein zusätzliches Gerät am Körper tragen möchte und welcher Mehrwert sich daraus für ihn ergibt. Bei dem MYO-Armband muss der Benutzer explizit ein Zusatzgerät tragen, um mit einem System interagieren zu können. Bei der Kinect, den Eye Tracker-Lösungen oder bei der Leap Motion müssen bestimmte Abstände eingehalten werden, damit die optische Erfassung und die Analyse funktionieren. Das kann unter Umständen bedeuten, dass der Benutzer während er fernsieht eine ganz bestimmte Position einnehmen muss und bestimmte Abstände einhal-

ten sollte. Zwar ist dies für kurze Interaktionssequenzen wie z.B. die Suche oder Auswahl eines Menüpunktes denkbar aber für intensivere Interaktionen mit dem Fernseher dem Benutzer nicht zumutbar. Bei längeren Interaktionen, die eine präzise Bedienung erfordern, können Ermüdungsprobleme auftreten, wie schon sehr früh in [FW95] beschrieben. Aus diesem Grund müssen, falls Gesteninteraktionen für die Bedienung benutzt werden sollen, die Interaktionen kurz und einfach gehalten werden.

Aufgrund ihrer nahen Verwandtschaft zum Fernbedienungsprinzip stellen die Wii Remote und die Gyro-Mäuse eine gute Alternative zur traditionellen Fernbedienung dar. Dieses Prinzip wurde ebenfalls von Fernsehherstellern wie LG oder Samsung übernommen. Damit lassen sich nicht nur 3D-Gesten ausführen, sondern es wird die wohl bekannte Mausbedienungsmetapher verwendet. Steuerelemente auf der Bedienoberfläche lassen sich über einen auf der Bedienungsoberfläche eingeblendeten Cursor auswählen und Zusatzfunktionalitäten wie die Lautstärkeregelung entweder traditionell über die Tasten oder per Sprachinteraktion aufrufen. Die Entscheidung, welche Eingabetechnologie und Hardware am geeignetsten für eine Interaktion vor dem Bildschirm sind, hängt sehr stark davon ab, inwiefern die ausgewählte Lösung mit der Benutzeroberfläche zusammen benutzt werden soll. Dabei steht die Frage der Benutzung eines Zusatzgerätes durch einen Benutzer im Vordergrund. Lösungen wie Kinect oder Leap Motion verlangen keine zusätzlichen Geräte, die am Körper des Benutzers befestigt werden müssen. Daher sind beide Technologien für eine Verwendung in Kombination mit einem Fernseher sehr gut geeignet. Nur die genaue Kalibrierung (Erfassung der Größe des Benutzers) und die Erfassung der räumlichen Positionierung (z.B. vor dem Fernseher mit einem bestimmten Abstand vom Benutzer oder auf der Armlehne

liegend) beider Lösungen könnten ein minimales Hindernis für eine intensive Benutzung vor dem Fernseher sein.

Semantische Extraktion

Unter der Extraktion der Semantik aus einem Medienobjekt (Text, Video, Bild) versteht man eine annähernd präzise Beschreibung des Inhalts in Form eines Konzepts oder einer gröberen Zuordnung (auch Klassifizierung genannt). Als Ergebnis eines Extraktions-Verfahrens wird eine detaillierte semantische Beschreibung des Medieninhaltes erzeugt. Diese kann als Startpunkt für eine nachträgliche semantische Suche über dedizierte Dienste dienen.

In folgenden Abschnitten werden zuerst die Möglichkeiten zur Extraktion von Wissen und Konzepten aus heterogenen Texten vorgestellt. Es wird die Textanalyse über dedizierte Cloud-basierte Dienste in Echtzeit erläutert. Im nächsten Schritt werden die technischen Herausforderungen und die Herangehensweise der visuellen Erkennung von Konzepten in Bildern und Videos skizziert, mit dem Ziel, eine weiterführende automatisierte Personenerkennung und OCR (engl. Optical Character Recognition)-Analyse durchzuführen. Das letztliche Ziel ist es, das Medium besser und präziser „semantifizieren“ und klassifizieren zu können. Parallel zu diesen Betrachtungen werden die Werkzeuge und Programmierschnittstellen, die im Rahmen der Entwicklung von Swoozy von Bedeutung sind, erläutert. Im letzten Abschnitt werden Annotationswerkzeuge aus dem wissenschaftlichen und professionellen Bereich detailliert vorgestellt. Diese Betrachtung ist aus dem Grund nötig, da, obwohl die vorgestellten Verfahren in der Lage sind, (semi)-automatisch Multimediadaten zu analysieren und zu

„semantifizieren“, es aus redaktioneller Sicht immer die Möglichkeit geben sollte, manuell Annotationen mit alternativen Wissensquellen aus dem Netz zu verknüpfen.

Analyse textueller Inhalte

Verfahren

Im folgenden Abschnitt werden Verfahren vorgestellt, die aus heterogenen und nicht vorannotierten Texten in der Lage sind, den (maschinenverständlichen) Sinn, die Konzepte, aber auch den Kontext zu erfassen und in strukturierter Form wiederzugeben. Dies wird über verschiedene morphologische, lexikalische, syntaktische und semantische Analysen und maschinelle Lernverfahren durchgeführt. Dabei spielen Verfahren aus der Computerlinguistik eine zentrale Rolle. Die dort angewandten Prinzipien der Textextraktion werden für gesprochene Texte (die von einer *Speech-To-Text*-Software stammen) und die Analyse von Audiostreams benutzt. Das Ergebnis der Analyse bilden strukturierte und annotierte Textstrukturen, die für eine weitere Verwendung z.B. durch eine sog. Wikification-Komponente [MC07] benutzt werden können. Durch diese Komponente werden die erkannten Entitäten mit neueren domänenspezifischen Wissensquellen wie z.B. Wikidata, DBpedia und dem Google Knowledge Graph verknüpft, wobei neue Relationen und thematische Verknüpfungen zwischen Konzepten identifiziert und angereichert werden.
Die Umwandlung von heterogenen Daten in maschinenverstehbare Strukturen kann über verschiedene Teilverfahren und Granularitäten bzw. Präzisionsstufen erfolgen. Diese Verfahren können den folgenden Kategorien zugeordnet und miteinander kombiniert werden:

- Thematische Zuordnung eines Textes
  Bei dieser Analyse wird versucht, die Hauptthematik (z.B. Sport, Musik, usw.) und deren Unterthematiken (z.B. Fußball, Oper, usw.) zu erkennen. Hierfür werden die Begriffe und Entitäten benutzt und untereinander verglichen, um die Zusammenhänge besser zu erfassen. Bei dieser Art der Analyse wird sehr oft auf existierende Wissensdatenquellen zugegriffen, um damit existierende Relationen hervorzuheben und thematisch miteinander zu verknüpfen. Eine Variante besteht bspw. darin, die Wikipedia-Links eines Konzepts zu verfolgen und aus den weiteren zusammenhängenden Kategorien die Thematik zu erkennen.

- Erkennung von Entitäten - Named-Entity Recognition (NER) und Konzepten
  Bei dieser Textanalyseaufgabe werden Entitäten, d.h. Begriffe, Namen von Personen, Unternehmen oder Orte aus einem nicht vorannotierten Text erkannt. Diese erkannten Entitäten werden nach Konzepten oder Gruppen (Unternehmen, Ort, Person) klassifiziert. Die Genauigkeit und Qualität der Erkennung hängt stark davon ab, welche grundlegenden Informationen und Domänenwissen im Hintergrund benutzt werden. Abbildung 3.30 zeigt, welche Entitäten und Konzepte mit dem Cloud-basierten Textanalysedienst AYLIEN[42] aus einem Text extrahiert werden können.

  Bei diesem Beispiel werden der Name einer Persönlichkeit, der Ort und der Eigenname des Produktes vom AYLIEN-System erkannt und klassifiziert. Die benutzten Verfahren und die zugrundeliegenden Algorithmen werden ausführlicher in [NPvdA14]

[42] `http://aylien.com`

[Bra11] [LXU11] beschrieben.

Optional kann sich die Entitäten-Extraktion auf *Entity Linking* stützen. Unter *Entity Linking* versteht man die Aufgabe, bestimmte Namen oder Begriffe mit Dokumenten zu verknüpfen, indem neben einer erzeugten Annotation ein direkter Link zur Wissensdatenbank hinzugefügt wird [LE+12].

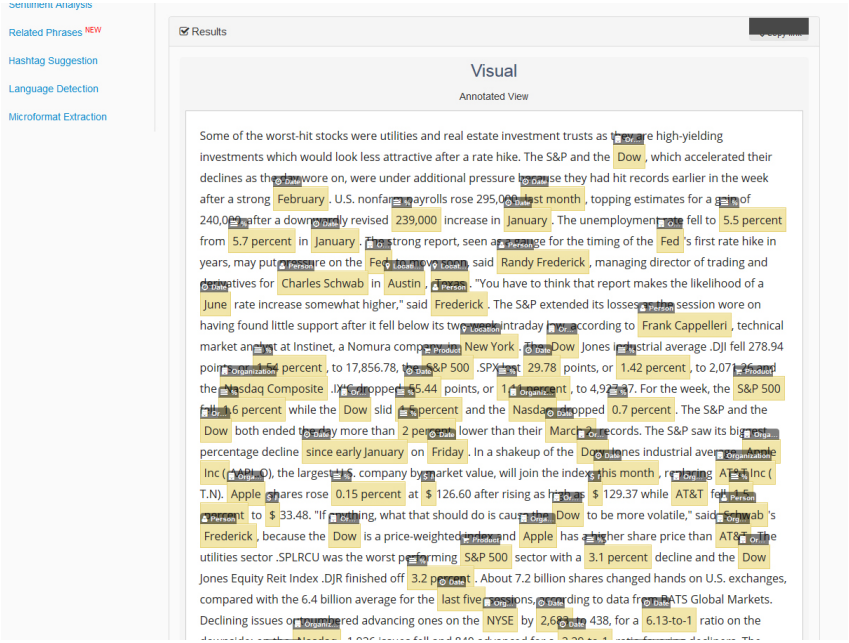


Abbildung 3.30: Beispiel einer Named-Entity Recognition. Quelle: www.aylien.com

- Textual Entailment

  Textual Entailment ist ein Verfahren, bei dem eine Relation zwischen zwei Textfragmenten erkannt und gewichtet wird. Textfragmente (engl. *Chunks*) bilden die einzelnen Komponenten (z.B. Worte, Bindeworte, Verben, Namen) eines Satzes. Die Her-

ausforderung von Textual Entailment ist es, aus zwei textuellen Ausdrücken eine Hypothese bzw. eine Schlussfolgerung zu bilden und daraus zu entnehmen, ob die angenommene Hypothese richtig oder falsch ist. Dies geschieht nach dem gleichen Prinzip, wie beim Menschen: auf Hintergrundwissen und auf Inferenzen basierend. Textual Entailment liefert als Ergebnis eine Hypothese, die entweder verifiziert oder falsifiziert wird, und einen Konfidenzwert, der ein Indikator für die Richtigkeit der Äußerung eines Textsatzes in Zusammenhang mit der ersten Hypothese bzw. ersten textuellen Äußerung ist [NPvdA14] [NBB+97] [WN07] [DGM06] [SVR12]. Die Thematik des Textual Entailment wurde u.a. im Rahmen des europäischen Projekts EXCITEMENT[43] ,[44] bearbeitet [MZD+14].

- Sentiment Analyse des Inhaltes - Opinion Mining
  Bei der *Sentiment Analyse* oder Stimmungserkennung wird versucht, die allgemeine Atmosphäre, Haltung oder Textaussage als positiv oder negativ zu bewerten. Um dies zu realisieren, werden die Grammatik aber auch die Konnotationen der einzelnen Worte bestimmt und mit der allgemeinen Gesamtstimmung des Textes verglichen. Beispielsweise werden Äußerungen wie *ist gestorben* oder *ist gesunken* als negative Äußerungen bewertet. Dies wird erreicht, indem bestimmte Nomen und Verben mit ihrem Eintrag in einem Wörterbuch mit Stimmungsgewichtungen (wie z.B. SentiWordnet[45] oder/und Synonymen-Datenbanken wie OpenThesaurus[46]) abgeglichen werden [ES06] [Seb02] [Den08]. Die Paragraph-übergreifende Erkennung von Relationen und der

[43] https://sites.google.com/site/excitementproject/
[44] http://hltfbk.github.io/Excitement-Open-Platform/
[45] http://sentiwordnet.isti.cnr.it
[46] https://www.openthesaurus.de

Vergleich der einzelnen Wörter mit dedizierten und domänenspezifischen Wörterbüchern (z.B. im Sport mit Begriffen wie "Massensturz", "Reifenplatzer", oder "Hypoglykämie") können dazu beitragen, den Präzisionsgrad der Analyse zu verbessern. Diese Analyse wird sehr oft in Kombination mit *Opinion Mining* (Meinungsanalyse) benutzt, um z.B. im Businessbereich das allgemeine Image einer Marke (*„Warum kaufen Kunden eher Produkt A als B?"*) besser zu analysieren und bestimmte Tendenzen einer Community zu erfassen. *Opinion Mining*-Verfahren werden zur Analyse textueller Kundenrezessionen benutzt, um damit die Zufriedenheit und Verbrauchermeinung besser in verschiedenen Kontexten (z.B. Blog-Post, Forum-Beitrag bei einem Onlineverkäufer) zu erfassen [Mul11]. Die Analyse der Meinungen bzw. Trends kann entweder durch cloud-basierte Dienste oder durch Textanalyseverfahren bzw. Werkzeuge wie z.B. SProUT[47] [DKP+04] oder NEMEX (siehe Kapitel 6) durchgeführt werden.

Cloud-basierte Werkzeuge für die Analyse textueller Inhalte

Bei der Extraktion wird sehr oft am Anfang des Analyseprozesses eine Domäne (oder Themenfeld) festgelegt. Die Anzahl der zur erkennenden Begriffe ist somit beschränkt und es können qualitativ bessere Ergebnisse als bei einer Open-Domain-Extraktion erzielt werden. Der Begriff *Open-Domain* bedeutet, dass die Extraktionskomponente weder ein „Vorwissen" über die Begriffe noch über die möglichen Wortkomponenten bzw. Entitäten besitzt. Dies bedeutet, dass während der Analyse eines Textes womöglich jede Thematik oder Domäne

[47] `http://sprout.dfki.de/index.html`

angesprochen und daraus eine Information extrahiert werden kann. Die Gefahr dabei ist, dass die Erkennung schlechter sein kann, oder, dass eine Disambiguierung der erkannten Entitäten nachträglich nötig wird. Im schlimmsten Fall können Begriffe oder Konzepte aufgrund mangelnden Domänenwissens oder semantischer Repräsentation gar nicht erst erkannt oder wegen ihrer Zweideutigkeit falsch klassifiziert werden.

Weltweit befassen sich sehr viele Forschungsprojekte [WHW09] [WW10] [ZDR15] [SZZ+15] [EFC+11] [QOP12] mit der Problematik der Korrektheit der erkannten Entitäten innerhalb einer offenen Domäne. Um dieses Problem teilweise zu lösen, werden bekannte Wissensquellen wie Wikipedia als Grundlage für die Erkennung benutzt. Existiert ein Begriff bei Wikipedia, kann indirekt abgeleitet werden, dass dieser und die damit zusammenhängenden Konzepte valide sind. Durch die Verlinkung zu DBpedia und die Analyse der Verknüpfungen innerhalb eines Wikipedia-Artikels kann die Entität präzisiert und thematisch angeordnet werden. Vorteil dieses Ansatzes ist, dass durch die ständig wachsende Anzahl der möglichen erkennbaren Entitäten innerhalb von Wikipedia immer neuere Entitäten eingepflegt und aktualisiert werden und multilingual zur Verfügung stehen.

Cloud-basierte Textanalysedienste stützen sich sehr stark auf dynamische Wissensdaten, um die Problematik der Erkennung von Entitäten bzw. Konzepten, Emotionen und Relationen in einer offenen Domäne zu lösen. Diese Eigenschaft ist essentiell, da erstens das angestrebte System verteilt und online läuft, d.h. es können keine lokalen Abbildungen von Wissensdaten gespeichert werden, und zweitens die möglichen extrahierten Entitäten eine konkrete Verknüpfung zu weiteren Ressourcen vorweisen müssen, damit passende online-basierte multimediale Daten, wie Videos, Bilder oder sogar Tweets, als Ergebnis

einer Extraktion in einem weiteren Schritt dem Benutzer angeboten werden können.
Im nächsten Abschnitt werden diese Dienste kompakt vorgestellt. Der Fokus bei der Auswahl der Dienste lag auf deren Leistungsvermögen bei der *Named Entity*-Extraktion und der Extraktion von Konzepten (z.B. Orte oder Firmennamen) und Emotionen bzw. Stimmungen aus einem EPG-artigen Text (Text der innerhalb von Telextext oder dem Electronic Programming Guide verwendet wird). Die Qualität der Ergebnisse wurde im Hinblick auf eine mögliche Anwendbarkeit innerhalb des geplanten Systems genauer betrachtet.

Stanford NLP - NER

Das Stanford NER (die Abkürzung NER steht für Named Entity Recognition) ist ein Java-basiertes Programm, das in der Lage ist, eine Sequenz von Wörtern innerhalb eines Textes zu klassifizieren und die Namen von Personen, Gegenständen oder Firmen zu extrahieren [FGM05]. Dieses Programm kann parametrisiert werden und die Merkmalsextraktoren zur Eigennamenerkennung verfeinert werden. Als Ergebnis der Texterkennung werden, je nach trainierten Modellen, die erkannten Entitäten in sieben Klassen (*Location*, *Person*, *Organization*, *Money*, *Percent*, *Date* und *Time*) eingeteilt. Trainierte Modelle für verschiedene Sprachen können ebenso von der Projektseite heruntergeladen werden.
Auch wenn die Benutzung von Stanford NER eine lokale Installation erfordert, bieten verschiedene Portierungen auf Programmiersprachen wie Javascript (in Form eines nodes.js-Packets (ner-server)[48]) oder

[48] `https://www.npmjs.com/package/ner-server`

PHP[49] die Möglichkeit, das Programm als eigenständigen Webdienst zu starten. Aus diesem Grund kann das Stanford NER-Werkzeug mehr oder weniger in der Kategorie der möglichen Cloud-basierten Dienste, die für das angestrebte Szenario des semantischen Fernsehens zum Einsatz kommen, klassifiziert werden.

Alchemy API

Alchemy[50] bietet eine sehr ausführliche API zur Textextraktion mit Verfahren, die teilweise aus dem Deep Learning-Bereich stammen. Ziel der Alchemy API ist es, neben Bildern und Videos auch Texte und Dokumente zu analysieren. Die API ist primär für den Business-Bereich gedacht, um damit Online-Marktforschung betreiben zu können d.h. es werden u.a. Tweets, Webseiten und Emails analysiert. Nachträglich werden diese Analyseergebnisse mit einer semantischen Datenbasis verglichen. Somit können Unternehmen die Wirksamkeit einer Werbekampagne von angebotenen Diensten oder die Resonanz eines Produkts im Netz besser messen. Unter der Produktserie *Alchemy Language* werden insgesamt zwölf über API-abrufbare Textanalysenfunktionen dem Entwickler zur Verfügung gestellt. Darunter befinden sich u.a. eine Entitätsextraktion, eine Meinungsextraktion und ein Konzept-Tagging in Verbindung mit semantischen Linked Data-Quellen wie DBpedia oder OpenCyc[51]. Die Ergebnisse der Anfragen werden in Form von JSON-, XML- oder RDF-Formaten zurückgeliefert, sodass eine Anbindung für eine Weiterverarbeitung innerhalb semantischer Applikationen möglich ist.

[49] `https://github.com/agentile/PHP-Stanford-NLP`
[50] `http://www.alchemyapi.com`
[51] `http://sw.opencyc.org`

Semantria

Semantria[52] bietet eine Unterstützung zur Sentiment- und Textanalyse über verschiedene REST-basierte APIs. Entwickler können über diese APIs direkt die Kategorisierung- und Entitätsextraktionsfunktionalitäten aufrufen. Tweets und Webseiten lassen sich auch als Eingabequelle für die Analyse benutzen. Semantria bietet eine Unterstützung für Excel in Form einer Bewertung von tabellarisch aufgesammelten Daten. Die Integration von Semantria in Softwareprojekte kann über die Programmiersprachen (C++, JAVA, PHP, .NET, Python, Ruby und Javascript) erfolgen. Semantria unterstützt sieben Fremdsprachen und ist in einem multilingualen Kontext verwendbar.

AYLIEN

AYLIEN[53] ist eine Cloud-basierte Lösung, die acht Algorithmen aus den Bereichen Natural Language Processing, Informationsextraktion und maschinelles Lernen in Form einer einheitlichen API anbietet. AYLIEN bietet Funktionalitäten wie Konzept-Extraktion, Klassifikation und Keyword-Tracking in Webseiten, Dokumenten und Tweets. Ähnlich Semantria ermöglicht AYLIEN über die Konzeptextraktion eine direkte Verlinkung zu DBPedia und Linked Data-Entitäten inklusive ihrer semantischen Typen, die von Schema.org stammen.
Als Erweiterung bietet AYLIEN ein Intentionserkennungsmodul, das eine Unterstützung bei Entscheidungen anbietet. Darüber hinaus können Marktveränderungen oder Tendenzen (Buzz) über Social Media erfasst, aufgelöst und entdeckt werden. Diese Funktionalitäten können genauso im Business-Bereich (z.B. im Börsengeschäft) als auch für

[52] `https://semantria.com`
[53] `http://aylien.com`

Online-Werbung (Stichwort: Impaktierung) benutzt werden. Zusätzlich zur der angebotenen API bietet AYLIEN eine direkte Einbindung von Google Spreadsheets an, sodass die Analyse und Extraktionen von Google Docs-basierten Dokumenten komplett online durchgeführt werden können.

Fazit

Die Extraktion von Entitäten und Konzepten aus Dokumenten oder Webseiten sind Verfahren, die in Zusammenhang mit den präsentierten Cloud-basierten APIs für Entwickler sehr schnell aufrufbar und integrierbar sind. Die Unterschiede bei den APIs liegen im Wesentlichen bei der Qualität der Auswertung der Texte und der Extraktion der Entitäten im multilingualen Kontext. Auch wenn viele Sprachen für die Analyse unterstützt werden, stellt sich sehr schnell heraus, dass die Korpora hauptsächlich für die englische Sprache trainiert wurden. Ein weiteres Problem besteht darin, dass teilweise die Konzept- und Namenserkennung und deren Verlinkung mit DBpedia oder Wikipedia sehr minimal gehalten werden, d.h. es wird nur ein Link auf die Entität zurückgegeben aber keine Verweise auf weitere Eigenschaften aus Linked Data. Insbesondere die Eigennamenerkennung für Persönlichkeiten oder Institutionen des deutschen und französischen Sprachraums liefert lediglich sehr unzuverlässige bis gerade passable Ergebnisse. Ein weiteres Unterscheidungsmerkmal bilden die teilweise sehr heterogenen Ergebnisse der Sentiment-Analyse. Je nachdem welche Sprache für die Analyse ausgewählt wurde, werden unterschiedliche Interpretationen zurückgegeben.

Semantische Extraktion aus Bildfolgen und Videoinhalten

Neben der Wissensextraktion aus textuellen Daten stellt im Multimedia-Bereich die Extraktion bzw. Erkennung von Entitäten oder Konzepten eine weitere zentrale Herausforderung dar. Ziel dieses Abschnitts ist es, die Verfahren und Mechanismen zu präsentieren, die in der Lage sind, Bilder bzw. Videoframes automatisch oder semi-automatisch zu analysieren und daraus eine korrekte semantische Beschreibung zu erstellen. Hierzu werden zuerst die grundlegenden Verfahren vorgestellt, die Schritte einer Video- bzw. Bild-basierten Erkennung erläutert und gezeigt, welche dieser Verfahren für die Realisierung des angestrebten Systems am geeignetsten sind.

Auch wenn automatische Verfahren in der Lage sind, Entitäten oder sogar Personen aus den jeweiligen Szenen zu erkennen, besteht – wie bei der textuellen Extraktion – aus redaktioneller Sicht die Notwendigkeit, die erkannten Entitäten zu überprüfen, zu validieren und ggf. nachzueditieren und zu ergänzen. Aus diesem Grund werden Bild- und Videoannotationswerkzeuge aus dem wissenschaftlichen und professionellen Bereich vorgestellt und miteinander verglichen. Diese Betrachtung hilft dabei, die Annotationsmöglichkeiten und Verknüpfungsmechanismen mit existierenden Ontologien oder Datenbanken besser nachzuvollziehen.

Aus Softwareentwicklersicht spielen die Möglichkeiten, Bild- und Videoanalysekomponenten innerhalb eines Systems softwaretechnisch zu integrieren, eine wichtige Rolle. Ähnlich der textuellen Extraktion werden im letzten Abschnitt aktuelle Werkzeuge und Cloud-basierte APIs zur Bild- und Videoanalyse und Extraktion präsentiert und miteinander

auf Qualität und Integrierbarkeit mit verschiedenen Programmiersprachen verglichen.

Prinzip

Spricht man von *Extraktion von Semantik* bei Videos oder Bildern, sind meistens die dahinterliegenden softwaregestützten Prozesse gemeint, die es ermöglichen in verschiedenen Schritten, ausgehend von einer Pixel-basierten Struktur (bzw. einem Videoframe oder Bildfolge) eine semantische Beschreibung des Bildinhaltes zu generieren. Diese Prozesse bestehen aus dem Durchlaufen mehrerer Stufen, bei denen verschiedene Verfahren eingesetzt werden, und liefern als Ergebnis eine Beschreibung des analysierten Bildes. Die benutzten Verfahren sind ausschlaggebend für die Granularität und Qualität der Erkennung. Die Präzision der Erkennung kann nachträglich durch Verfahren wie die Gewinnung von Kontextinformationen durch Aktivitätserkennung erhöht werden [Fre15]. Die Präzision kann dabei von der einfachen Erkennung eines Konzepts (z.B. einer Szene mit Personen) bis zur Erkennung der einzelnen Gesichter und deren Emotionen reichen.

Erkennungsschritte

Die Abbildung 3.31 stellt vereinfacht dar, welche Schritte für die klassische Analyse eines Bildes oder von Videomaterial vom Einlesen der Datei bis zur Generierung einer semantischen Beschreibung durchgeführt werden müssen.

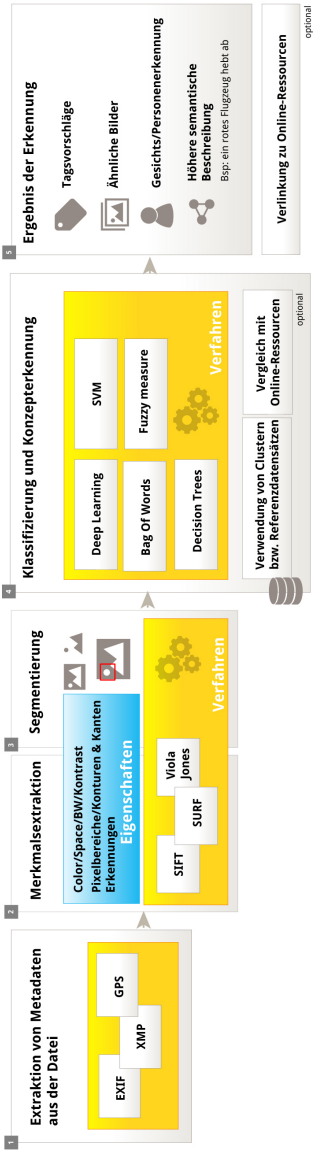


Abbildung 3.31: Prozess der Extraktion

Datei als Startpunkt (Schritt 1)

In der ersten Stufe (siehe Punkt 1 in Abbildung 3.31) besteht die Aufgabe darin, die Datei zu analysieren und zu versuchen erste Informationen zu extrahieren. Bilder oder Videos können Metadaten (in den Formaten XMP- oder EXIF) enthalten, die z.B. die GPS-Positionen des Aufnahmeortes oder sogar die Positionen von Gesichtern mitgespeichert haben. Auch der Name der Datei kann erste Informationen über den tatsächlichen Inhalt der Datei liefern. Diese Informationen können schon für eine erste Klassifikation benutzt werden [KSA13].

Merkmalextraktion (Schritt 2)

Damit das Bildmaterial vom Erkenner optimal erfasst und von den Algorithmen analysiert werden kann, müssen zunächst „Unreinheiten" entfernt und leichte kolorimetrische Anpassungen vorgenommen werden. Dieser Schritt der Analyse kann mit beliebigen Varianten und Bildparametrisierungen mehrmals wiederholt werden bis die Qualität des Bildes für die Analyse und Merkmalextraktion ausreichend ist. Üblicherweise wird bei den meisten Verfahren eine Schwarz-Weiß-Transformation des Bildes durchgeführt und über mehrere Durchläufe die Bildeigenschaften wie Kontrast, Schärfe oder Helligkeit, Sättigung und Farbwert (engl. Brightness/Saturation/Hue) verändert. Diese iterativen Analysen können dazu führen, dass im Enddefekt eine bessere Erkennungsrate erzielt werden kann.
Zusätzlich muss, bevor die tatsächliche Analyse durchgeführt wird, darauf geachtet werden, dass die Bilder eine „vernünftige" Auflösung besitzen. In diesem Schritt kann manchmal eine manuelle Sichtung oder Nachbearbeitung notwendig sein. Bei künstlichen Artefakten in Bildern (wie z.B. durch Weichzeichen, Linseneffekte, Zoomen oder

der Verwendung weiterer Filter) muss bedacht werden, dass solche Bilder nicht gleich pauschal als schlechte Kandidaten für eine weitere Analyse angesehen werden. Tatsächlich können gerade solche Effekte indirekt Aufschluss darüber liefern, welche Gegenstände im Vorder- oder Hintergrund platziert worden sind. Somit ist es möglich, Konzepte besser zu erkennen und visuell voneinander zu trennen. In [VG11] werden diese Eigenschaften für die räumliche Position von Konzepten innerhalb eines Bildes benutzt.

Das grundlegende Prinzip der Merkmalextraktion besteht darin, ein Bild über Computer- Vision-Algorithmen zu analysieren und bestimmte Pixel-Bereiche anhand von existierenden Referenzmustern (oder Bilddatenbanken) miteinander zu vergleichen und identifizieren zu lassen. Diese Muster dienen später als Suchmuster für das Clustering und werden in der Literatur als *Bag-Of-Features* bezeichnet [CDF+04][54]. Diese Clustering- und Such-Operationen werden über Algorithmen, die als Interest-Operator bezeichnet werden, realisiert. Einige Algorithmen wie z.B. Harris-Laplace dienen dazu, eine bessere Erkennung von Rändern zu erzielen. Alternativ können andere Verfahren wie z.B. der Förstner-Operator während des Erkennungsvorganges zum Einsatz kommen [RK06].

Zwei weitere sehr verbreitete Verfahren zur Merkmalerkennung sind:

1. SIFT (Scale Invariant Feature Transform) oder *skalier invariante Merkmalstransformation* wird für die Extraktion von Merkmalen im Bild benutzt und wurde von David Lowe entwickelt [Low99]. Diese Methode ermöglicht eine sehr schnelle Erkennung und Extraktion der Bildmerkmale. Selbst wenn das Bild gedreht, gestreckt oder die Skalierung verändert wurde, können durch das

[54] `http://gilscvblog.wordpress.com/2013/08/23/bag-of-words-models-for-visual-categorization`

SIFT-Verfahren die Merkmale (*Keypoints*) innerhalb der Bilder erkannt werden. Der Algorithmus ist seit 2004 in den USA als Patent angemeldet und kann ohne eine Lizenzierung durch die University of British Columbia nicht für kommerzielle Zwecke verwendet werden[55]. Alternativ kann der OpenSIFT-Algorithmus[56] [Hes10] benutzt werden.

2. SURF (Speeded Up Robust Features) wird sehr oft als eine freie und nicht kommerzielle Alternative für SIFT bezeichnet. Die Qualität der Erkennung von SURF wird gegenüber der von SIFT als überlegen betrachtet [BSP07]. SURF wurde von Herbert Bay [BTVG06] implementiert. Die Erkennung von Merkmalen wird über den *Haar-Cascade Klassifikator* [LKP03] [Abe13] – benannt nach dem ungarischen Mathematiker Alfréd Haar – realisiert. Alternativ können Verfahren wie die *Viola-Jones Object Erkennung* [VJ01] benutzt werden. Genau wie bei SIFT existieren alternative Versionen von SURF wie z.B. OpenSurf[57] welche innerhalb verschiedener Computer Vision-Frameworks wie z.B. OpenCV eingesetzt werden.

Neben SURF und SIFT existieren zahlreiche andere Merkmal-Klassifikatoren wie z.B. OpponentSIFT, Local Binary Patterns oder Harris Laplace Detektoren, die es ermöglichen, Merkmale aus einem Bild zu extrahieren [vdSGS10].

[55] `http://www.google.com/patents/US6711293`
[56] `https://robwhess.github.io/opensift/`
[57] `http://chrisevansdev.com/computer-vision-opensurf.html`

Segmentierung (Schritt 3)

Nach der Merkmalsextraktion kann eine Segmentierung der Bildbereiche durchgeführt werden. Bei diesem Schritt werden verschiedene Bildanalyseverfahren angewandt, u.a. die Kantenerkennung, bei der die Übergänge von einem Bildbereich zum anderen analysiert werden, oder die Analyse der Farbhistogramme [SM00], [HS85]. Je nach Relevanz der gefundenen Bildbereiche können diese Bereiche in Clustern eingruppiert werden. Auf dieser Basis wird in einem weiteren Schritt die Konzepterkennung durchgeführt, die ein oder mehrere Bildbereiche des ursprünglichen Bildes analysieren wird.

Klassifizierung und Konzepterkennung (Schritt 4)

Nachdem die Merkmale extrahiert worden sind, müssen sie in Gruppen zusammengefügt werden. Zu jeder Gruppe gehören ein oder mehrere Begriffe/Konzepte (z.B. Vogel, Baum, Auto) oder Domänen (z.B. Sport, Nachrichten).
Sie bilden den sog. *Bag-Of-Words* [Sch03a], der als Training Set benutzt werden kann. Damit lassen sich Verknüpfungen zwischen Bildern realisieren und Gruppierungen erstellen. Tatsächlich wird die Technik, um ein automatisches Tagging zu realisieren, oft als *Bag-of-Words*, *Bag-Of-Features*, *Bag-Of-Visterms* [JT05] oder sogar *Bag-Of-Keypoints* benannt. Dabei werden besondere Bild- und Videomerkmale und deren Eigenschaften, z.B. Ränder, Kanten, Bereiche (engl. Blobs), aus einem Bild extrahiert und in Clustern oder Gruppen (sog. „Bags“) angeordnet. Die Frequenz des Auftretens eines bestimmten Merkmals bestimmt seine Zuordnung in die korrespondierenden Cluster. Ein Bild kann zu mehreren Clustern gehören. Die Überlappung der Cluster definiert letztendlich den tatsächlichen Inhalt des Bildes. Diese Gruppierung

ergibt ein statistisches Modell, das als *Wort-Histogramm* visualisiert werden kann. Dadurch ist die Verteilung bestimmter Begriffe direkt erkennbar, denn die Frequenz des Auftretens von Begriffen oder erkannten Worten gibt den Wahrscheinlichkeitswert an. Je öfter ein Merkmal in einem Bild oder Video erscheinen wird, desto öfter wird der passende Begriff mitgezählt.

Abbildung 3.32: Prinzip von Bag-Of-Words

Diese Methode stammte ursprünglich von Verfahren, die in den Bereichen NLP und textueller Analyse eingesetzt wurden. Sie wird seit ein paar Jahren verstärkt im Bereich Computer-Vision eingesetzt [CDF+04] [NJT06].

Wenn neue, noch nicht bekannte Bilder oder Videos analysiert werden müssen, werden die Merkmale erstmals untereinander verglichen

und mit Annäherungsmethoden und einer Gewichtung wie z.B. dem k-Means [M+67] Clustering überprüft. Als Ergebnis erhält man eine Klassifikation der Merkmale, geordnet nach Wort bzw. Begriff.
Parallel dazu können alternative Klassifikationsmethoden während des Einordnungsvorganges zum Einsatz kommen. In den meisten Fällen werden SVMs (*Support Vector Machines*), neuronale Netzwerke (engl. *Neural Networks*), Entscheidungsbäume (*Decision Trees*) oder *Fuzzy Measures* benutzt [KSA13].
Ziel dieser Klassifikationsmethoden ist es, nicht nur die verschiedenen Merkmale der Cluster korrekt einzugruppieren, sondern diese untereinander zu vergleichen und einen Konfidenzwert parallel zu der gefundenen Gruppe zu ermitteln. Mittels des Konfidenzwerts lassen sich parallel zum Ergebnis Unterklassen oder z.B. mehrere Kategorien wie *Tier, Vogel, Papagei* bestimmen. Um Teile eines Bildes erfolgreich z.B. über das *Bag-of-Words*-Verfahren wiedererkennen zu können, müssen Trainingsdaten oder Referenzendatensätze benutzt werden. Diese Daten können manuell erstellt werden, indem Bilder oder Videos mit einem selbsterstellten Begriff-Wörterbuch (engl. *codebook*) oder semiautomatisch mit Hilfe von Benchmarking Sets (auch Training Sets genannt) wie ImageClef[58], ImageNet[59], oder TRECVid[60] indiziert werden. Während des Trainingsvorgangs müssen nicht nur die passenden Kombinationen aus Wörtern und Merkmalen erstellt werden, sondern es können, u.a. um das Training zu beschleunigen, auch nicht-passende (sog. positive Negative) Kombinationen von den Klassifikatoren miteinbezogen werden. Somit können schwer erkennbare Gegenstände besser differenziert werden. Durch die Frequenz und Anzahl der wie-

[58] `http://www.imageclef.org`
[59] `http://www.image-net.org`
[60] `trecvid.nist.gov`

derkehrenden Merkmale ist es möglich, die Erkennungserfolgsrate zu erhöhen.

Ergebnisse der Erkennung und Konfidenzwerte (Schritt 5)

Als Ergebnis der Klassifizierung wird eine Liste der erkannten Entitäten oder Konzepte zurückgeliefert. Die Grade der Granularität können sehr unterschiedlich sein. Die meisten Verfahren liefern sog. Tags. Alternativ können ähnliche Bilder zurückgeliefert werden, die aufgrund ihrer Ähnlichkeiten dazu dienen können, die Beschreibung des ursprünglichen Bildes zu verbessern. Das Ergebnis der Erkennung kann eine Wiedererkennung von Gesichtern oder Personen sein, wobei die Verfahren davon ausgehen, dass die Training Sets eine ausreichende Datenmenge besitzen, damit eine sinnvolle Erkennung durchgeführt werden kann. Diese Möglichkeit wird zur Erkennung von bekannten Persönlichkeiten genutzt, da die Referenzbilddatenmengen im Netz hoch sind.
Die höchste Granularitätsstufe, die nur mit kleineren Trainingssets und von keinen Online-basierten Suchmaschinen angeboten wird, bildet die komplette semantifizierte Beschreibung eines Bildes inklusive der abgebildeten Aktion (z.B. mit einer kompletten Beschreibung à la *ein Flugzeug landet* oder *eine Person läuft über die Straße*). Diese sehr genaue Information ist schwierig zu extrahieren, daher beruht diese Art der Extraktion meistens auf einem sehr aufwändigen, feingranularen und sehr gut aufgefüllten Training Set, das manuell angelegt wurde. Ansätze, die Deep Learning-Verfahren anwenden, sind aktuell teilweise in der Lage diese Probleme zu lösen.

Personen- und Gesichtserkennung in Video- und Bildmaterial

Im folgenden Abschnitt wird eine kompakte Übersicht gegeben, wie die Erkennung von Personen oder Gesichtern realisiert werden kann. Dabei müssen zwei Aspekte beachtet werden. Erstens muss die physische Entität *Person* über verschiedene Verfahren wie z.B. Skelett-, Pose- oder Mimik-Erkennung aus einem 2D-Bildmaterial extrahiert werden (Mehrere Forschungsprojekte haben sich als Ziel gesetzt, diese Posen auch mittels 3D-Verfahren und Skeletttracking zu ermitteln). Zweitens, nachdem eine Person erkannt wurde, muss auch ihr Gesicht analysiert werden. Erst danach kann eine Verlinkung der effektiven Position einer Person im Bild mit dem Namen durchgeführt werden.
Die Erkennung einer Person innerhalb eines Bildes kann mittels verschiedener Verfahren erreicht werden. In [GHGM14] wird das Verfahren der *K-poselet* vorgestellt, das aus einem Bild die Position einer Person erkennen kann. Dabei werden die einzelnen Darstellungen und Positionen getrennt aus verschiedenen Bildern gelernt und über einen Klassifikator zusammengestellt. Dies erfolgt während einer Trainingsphase. Nachdem die Punkte anhand des Trainingsets ermittelt worden sind, werden sie als Hauptmuster zur Erkennung von Posen und Personen benutzt[61].
In [RFZ05] wird ein prototypisches Verfahren präsentiert, das es ermöglicht, Personen und deren Position in einem Video über die Analyse der Kinematik zu detektieren. Hierbei wird ein visuelles Modell der Gliederpositionen erstellt. Dieses prototypische Verfahren kann für schnelle Bilder und Bewegungen in Filmen eingesetzt werden und die direkte Erkennung der Aktivität über die gerade durchgeführte Aktion

[61] `http://www.eecs.berkeley.edu/Research/Projects/CS/vision/shape`

(„Laufen“, „Springen“, „Sitzen“, usw.) der jeweiligen Person ermitteln. Mittels einer solchen Technik wird eine genauere Beschreibung einer Film- oder Sportszene möglich.

Weitere Verfahren und Algorithmen können bei der Gesichts- und Objekterkennung angewandt werden, wie z.B. Merkmal-Detektoren durch den Haar Cascade Classifier in Zusammenspiel mit der Viola Jones-Erkennung.

Parallel zur Bildanalyse-basierten Erkennung von Gesichtern lassen sich Personenbilder anhand von deren Beschriftung und der sie umgebenden, zusammenhängenden Texte wiedererkennen [BBEF05]. Zuerst wird versucht, das Gesicht zu erkennen, und in einem zweiten Schritt wird dieses mit einem in der Bildnähe platzierten Text in Relation gesetzt. Über die Berechnung der Wahrscheinlichkeit und Platzierung des Textes kann eine Relation zwischen Bild und Text vermutet werden. Dieses Verfahren funktioniert mit Zeitungsartikeln oder Nachrichtensendungen besonders gut. Aus diesem Grund ist es denkbar, letzteres Verfahren in Kombination mit einem Video und Einblendungen (z.B. während Nachrichtensendungen) für eine verbesserte Erkennung der Protagonisten oder des Kontextes einzusetzen.

Deep Learning und semantische Extraktion

Die vorhin vorgestellten Verfahren und Methoden zur Wissensextraktion und Semantik aus Bildern und Videos können zukünftig durch die Benutzung von neuronalen Netzwerken bzw. Deep Learning-basierten Ansätze ersetzt werden. Unter *Deep Learning* versteht man Methoden des maschinellen Lernens, die auf neuronalen Netzen beruhen [LBH15]. Neuronale Netze übertragen das aus der Gehirnforschung stammende Konzept von Neuronen und Synapsen in mathematische

Modelle. Neuronen sind Nervenzellen und die Synapsen bilden die Verbindungen zwischen den Neuronen. Mehrere zusammengeknüpfte Neuronen bilden ein neuronales Netzwerk. Jedes Neuron besitzt einen Schwellwert (meistens ein Wert von 0 bis 1). Diese Neuronenarten können eine Sigmoid-Funktion[62], wie folgende:

$$f(x) = \frac{1}{1 + e^{-x}}$$

benutzen. Die Verbindungen zwischen den Neuronen dagegen besitzen eine Gewichtung. Ein Neuron verarbeitet eine Eingabe (z.B. ein Pixelwert) und je nach interner Analysefunktion und resultierenden Schwellwerten wird er aktiviert und übergibt sein Ergebnis zum nächsten Neuron. Da meistens mehrere Neuronen gleichzeitig dieselbe Funktion ausführen (z.B. eine Convolution- oder ein Subsampling-Funktion) werden diese als Schicht (engl. Layer) [Hin07] betrachtet. Werden mehrere Schichten nacheinander verwendet, bezeichnet man diese als versteckte Schichten (engl. Hidden Layers [TP89]). Somit können verschiedene Merkmale bei jedem Durchlauf einer Schicht extrahiert bzw. klassifiziert werden. Nach einer festgegebenen Anzahl von Schichten (von 1 bis n) ist das neuronale Netz in der Lage z.B. mittels der Pixelinformation herauszufinden, zu welcher Klasse ein Bild gehört. Die Abbildung 3.33 stellt das Prinzip eines einfachen neuronalen Netzes schematisch dar.

[62] http://cs231n.github.io/neural-networks-1/

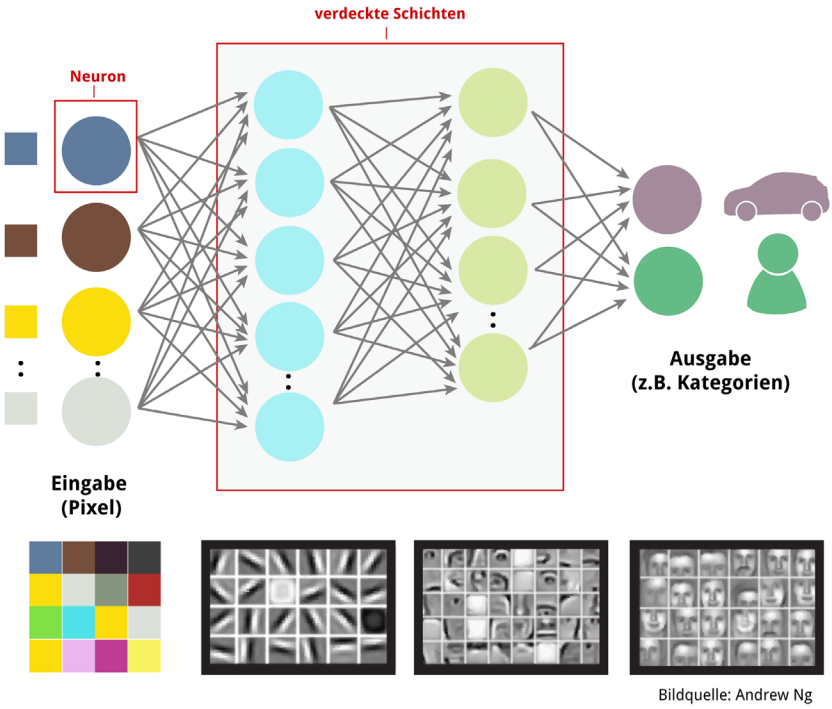


Abbildung 3.33: Prinzip eines neuronalen Netzes für die Bilderkennung. Quelle: Grafisch adaptiert aus http://www.nature.com/news/computer-science-the-learning-machines-1.14481. Quelle der Beispielbilder: Andrew Ng (http://www.andrewng.org)

Generell werden drei Typen von Lernverfahren mit neuronalen Netzen benutzt:

- Überwachtes Lernen (engl. supervised learning). Hier werden Eingabemuster bzw. Trainingssets benutzt [LBH15]. Diese sind schon vorbereitet und besitzen meistens eine textuelle Beschreibung (engl. label). Wenn ein Ergebnis aus einem Durchlauf festgestellt worden ist, wird dieses mit dem erwünschten Ergebnis verglichen. Der Unterschied zwischen dem erwünschten und

dem tatsächlichen Ergebnis kann berechnet und als Differenzvektor oder Fehlerquote bezeichnet werden. Basierend auf dieser Fehlerquote werden die ursprünglichen Gewichte des Netzes sowie die Verbindungsschichten verändert. Dies bezeichnet man als Rückwärtspropagierung (engl. backpropagation). Dieses Verfahren wird solange angewandt, bis das erwünschte Ergebnis korrekt bzw. die Fehlerquote niedrig ist.

- Eine Erweiterung vom überwachten Lernen ist das Bestärkende Lernen (engl. reinforcement learning) [Ben09]. Hier wird keine Fehlerquote berechnet, sondern nur bestimmt, ob die Ausgabe gegenüber der anfänglichen Hypothese richtig oder falsch war.
- Unüberwachtes Lernen (engl. unsupervised learning). Bei diesem Verfahren werden keine vordefinierten Angaben bezüglich der erwarteten Ausgabe gemacht. Hier ist das Ziel, neue Gruppen oder Cluster von Elementen zu bilden, die eine oder mehrere Gemeinsamkeiten vorweisen. Dieses Verfahren wird benutzt, um neue Gruppen, die im Vorfeld keine Beschreibung besaßen, zu entdecken. Bei Bildern kann das unüberwachte Lernen dazu führen, dass die Bilder besser segmentiert und andere Bildbereiche berücksichtigt werden können [SAFAKRM11] [CKN12].

Eine häufig benutzte Kategorie der neuronalen Netze bilden die sog. Convolutional Neural Networks (engl. kurz: CNNs oder im dt. faltendes neuronales Netzwerk), die bei Aufgaben der Bild- und Videoerkennung benutzt werden. Die Hauptcharakteristik der CNNs beruht darin, dass nicht jeder Pixel des Eingabebildes benutzt wird, sondern Ausschnitte des ursprünglichen Bildes. Jeder Pixel dieses Ausschnitts wird mit einem Neuron verknüpft. Die Faltung (engl. convolution) sind Filter wie z.B. Weichzeichnen, die auf diesen Ausschnitten benutzt werden.

Die Merkmale jeder dieser Ausschnitte werden extrahiert und können von weiteren Schichten analysiert werden.

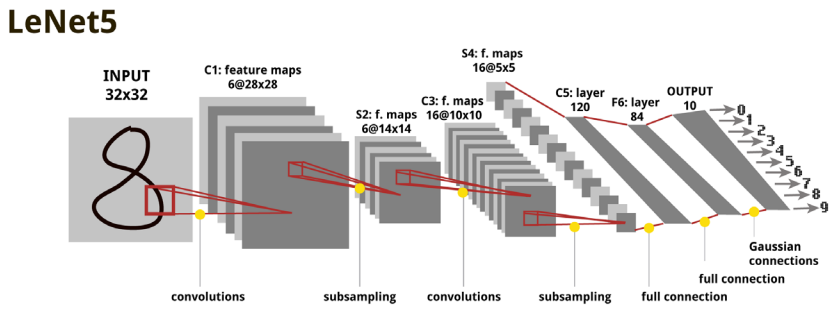


Abbildung 3.34: LeNet-5 neuronales Netzwerk von Yann LeCun. Grafisch adaptiert.

Am Ende ergibt sich eine Klassifizierung. Das bekannteste neuronale Netz ist *LeNet-5*[63] von Yann LeCun, das die Erkennung von handgeschriebenen Ziffern auf Basis der MNIST-Daten[64] durchführt und klassifiziert [BL+07] [LBBH98] [LJB+95]. Die Abbildung 3.34 zeigt den Aufbau von LeNet-5 mit seinen unterschiedlichen Schichten. CNNs lassen sich in den Bereichen der Bild- und Gesichtserkennung [Le13] [JSD+14] wie z.B. bei Deep Face [TYRW14] oder CelebFaces [LPLT15], der Texterkennung, z.B. als alternative OCR-Komponente wie in [CCC+11] [AH15] präsentiert, und der Videoerkennung einsetzen. Diese Ansätze gehen soweit, dass einzelne kleinste Bildelemente bzw. Pixelbereiche innerhalb eines Bildes semantisch annotiert werden können. Folgende Abbildung aus DenseCap[65] [JKFF16] zeigt das Ergebnis einer semantischen Analyse von zwei Bildern, in denen die einzelnen erkannten

[63] `http://yann.lecun.com/exdb/lenet/index.html`
[64] `http://yann.lecun.com/exdb/mnist/`
[65] `http://cs.stanford.edu/people/karpathy/densecap/browser`

Bildbereiche mit einem Label bzw. einer vollständigen Beschreibung versehen werden.

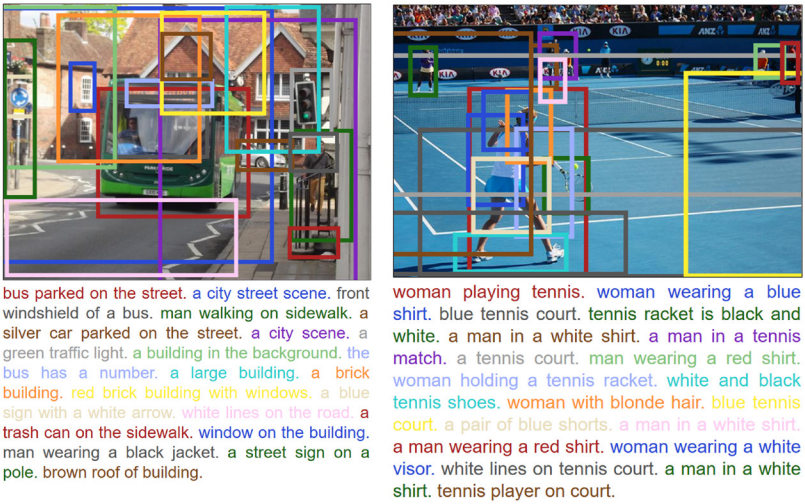


Abbildung 3.35: Bildschirmauszug der Webseite des Projekts „DenseCap“. Die erkannten Relationen und Informationen sind farblich markiert und textuell in der jeweiligen Farbe beschrieben

Im Bereich der Videoanalyse kann Deep Learning dazu benutzt werden, bestimmte Aktionen in Videos zu erkennen. Dafür werden spatio-temporale Eigenschaften des Videos berücksichtigt [TGLZ02] [KTS+14] [TBF+15] [LZYN11]. Solche Analysen setzen oft eine größere Anzahl von Videodaten und Beschreibungen voraus. Datensets, die z.B. von online-basierten Bilderdatenbanken wie Flickr, YouTube oder Wikimedia zur Verfügung gestellt werden, können als Trainingsets für die neuronalen Netze benutzt werden. Neben den eigentlichen Bilddaten enthalten diese Sets optionale grafische Informationen (mit/ohne Klassen/Begriffe, mit/ohne Labels, mit/ohne BoundingBox (dt. Rahmen

mit X-,Y-Angaben), wie es u.a. der Fall bei dem Set von Visualgenome[66] ist. In [KTS+14] werden CNNs für die Klassifikation von Videos in 487 Kategorien auf Basis von 1 Millionen YouTube-Videos benutzt. Eine weitere Bilddatenbank bildet der Yahoo Flickr Creative Commons 100 Million YFCC100m-Sets[67], der über 100 Millionen Bilder und Videos beinhaltet. Diese können in Echtzeit aufgerufen werden und dienen als Grundlage für ein Training der neuronalen Netzwerke. Die daraus resultierende Kategorisierung der Bilder und Videos können zu einem späteren Zeitpunkt für eine Beschreibungs-basierte Suche dieser Medienobjekte dienen [KSDB15] [TSF+16]. Dieser Ansatz kann ebenso für die Analyse von Emotionen und Gefühlen innerhalb eines gegebenen Medienobjektes benutzt werden. Ein gutes Beispiel dafür bildet Deep Sentibank [CBDC14], eine Datenbank, die basierend auf 24 vorklassifizierten Emotionen, Videos mit einer „emotionalen" Beschreibung versieht und klassifiziert. Dank dieser können beschreibungsbasierte Suchen wie z.B. das Herausfinden von Bildern mit einem „schönen Himmel" oder von Personen mit „traurigen Augen" durchgeführt werden. Die Verwendung von Deep Learning-Verfahren auf Medienobjekte zeigt vielversprechende Ergebnisse, die jedoch sehr ausführlich beschriebene Trainingsdaten brauchen. Daher ist die Verwendung dieser Verfahren in Echtzeit im Kontext des Fernsehens möglicherweise eher für Fernsehsendungen, deren Inhalte bzw. Domänen schon vorher bekannt sind, wie z.B. Talkshows oder Fußball-Übertragungen sinnvoll. Dazu kommt, dass die Deep Learning-Ansätze derzeit noch sehr rechenintensiv sind und dedizierte Rechenarchitekturen und gut vorbereitete Trainingsdaten brauchen. Hierbei stellt sich die Frage, in wie fern Fernsehsender in solche Verfahren investieren möchten.

[66] `https://visualgenome.org`
[67] `http://webscope.sandbox.yahoo.com/catalog.php?datatype=i&did=67`

Werkzeuge für die automatische Erkennung und Extraktion von Semantik aus Bild und Video

Im folgenden Abschnitt werden Frameworks, Softwarekomponenten und APIs vorgestellt, die für einen schnellen Einsatz von Gesichts- und Objekterkennung in Softwarekomponenten geeignet sind. Bei Verfahren im Umfeld der Computer-Vision wie z.B. der Gesichtserkennung wird eine Vielfalt an komplexen Algorithmen (oft kombiniert mit Machine Learning-Verfahren) verwendet. Speziell entwickelte Bibliotheken ermöglichen eine schnellere und optimierte Verarbeitung.

OpenCV

Die wohl bekannteste und meist benutzte Software-Bibliothek im Bereich der Computer-Vision ist OpenCV. Diese Open Source-Bibliothek wurde ursprünglich 2005 von Willow Garage implementiert und wird seit 2009 von Intel weiterentwickelt. Dank einer sehr starken Community wird der Quellcode ständig gepflegt und aktualisiert [Hö11]. Außerdem wird die Bibliothek im Robotik-Bereich verwendet und sie wurde in das autonome Auto namens *Stanley* integriert, das 2005 den DARPA Grand Challenge-Preis gewann.
OpenCV beinhaltet über 500 Funktionen, um Aufgaben der visuellen Analyse durchführen zu können. Die Basisbibliotheken wurden ursprünglich in der Programmiersprache C entwickelt. Die neuere Version benutzt C++, um die verschiedensten Funktionen für Bild- und Video-Processing anbieten zu können. Damit Berechnungen schneller durchgeführt werden können, können Entwickler in OpenCV die integrierten OpenCL (OpenComputing Language)- und GPU- bzw. CUDA (Computed Unified Device Architecture)-Pakete benutzen. Damit ist es möglich, bisher zeitaufwendige Aufgaben verteilt und auf verschiede-

nen Prozessoren parallel auszuführen. Die Stärke von OpenCV besteht darin, dass die gängigsten Computer Vision Algorithmen (Viola Jones, Haar Cascade, Corner Detection Module, usw.) über Bibliotheken und Klassen direkt integriert sind.
OpenCV kann in folgenden Bereichen benutzt werden:

- Bildanalyse

  OpenCV kann Segmentierungen, Histogrammberechnungen und Umrisserkennungen bei geladenen Bilderdateien (JPEG, PNG, usw.) durchführen. Dafür findet eine Analyse auf Pixelebene statt. Die verschiedenen Algorithmen zur Erkennung von Merkmalen sind standardmäßig in OpenCV integriert und können über vereinfachte Aufrufe benutzt werden.

- Videoanalyse

  Ein besonderes Merkmal von OpenCV ist die sehr gute Unterstützung von videobasierten Algorithmen zur Erkennung von Objekten und Gesichtern. Da OpenCV verschiedene Eingabestreams unterstützt und diese einheitlich verwalten kann, können entweder Live-Kamera-Streams oder Videodateien als Eingabe für die Analyse benutzt werden. Klassische Aufgaben wie die Erkennung von bewegten Personen oder die automatische Erkennung von KFZ-Kennzeichen aus Live-Kamera-Streams werden in Form von vorgefertigten Bibliotheken innerhalb von OpenCV-Paketen angeboten.

- Machine-Learning-Ansätze

  Parallel zu den Video- und Bildanalysekomponenten wird eine Vielzahl von Bibliotheken für das maschinelle Lernen mitgeliefert, wie z.B. *k-Means, Adaboost, Bayes-Klassifikatoren, Support Vector*

Machine, Convolutional Neuronal Networks oder *Deep Learning*. Diese Algorithmen können nachträglich für die Gesichts- und Objekterkennung innerhalb von Videos oder Bildern angewandt werden.

Dank seiner flexiblen Architektur wird OpenCV nicht nur für Bildanalysezwecke benutzt, sondern findet Anwendungen im Bereich Skelett- und Personen-Tracking sowie beim Finger- und Handtracking auf Multitouch-Tischen [SBD+08] [DDSP08]. Neben der C/C++-basierten Version von OpenCV existieren sogenannte Wrapper-Versionen wie z.B. Emgu CV[68]. Dabei handelt es sich um .NET Wrapper für die OpenCV Platform, die es ermöglichen, über die Programmiersprache C#, OpenCV-Funktionen und Module direkt aufzurufen. Diese Wrapper können in Mono (eine Open Source Implementierung der Programmiersprache C#) kompiliert und für verschiedene Plattformen (u.a. Android oder auch iOS) integriert und benutzt werden. Abbildung 3.36 listet alle Kernkomponenten auf, die standardmäßig in OpenCV in Form von Klassen und Algorithmen integriert sind.

[68] `http://www.emgu.com`

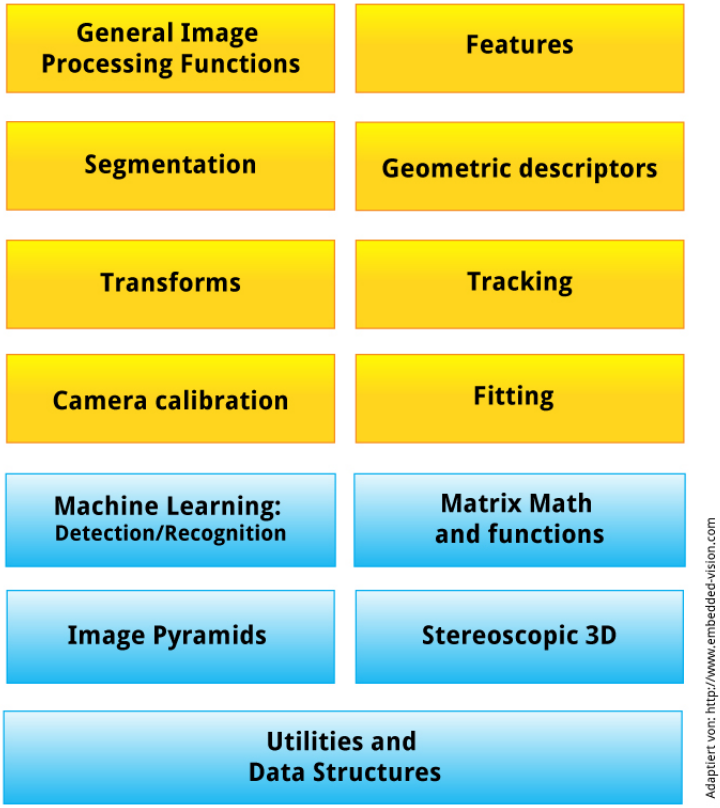


Abbildung 3.36: Komponenten von OpenCV. Grafisch adaptiert von embedded-vision.com

AForge

AForge.NET[69] ist ein Open Source Framework, das speziell für Softwareentwickler der Bereiche der Künstlichen Intelligenz und Robotik entwickelt wurde. Ähnlich wie bei OpenCV enthält diese Bibliothek Algorithmen und Funktionen für die Bildverarbeitung, die Suche über

[69] `http://www.aforgenet.com/aforge/framework`

neuronale Netze und maschinelles Lernen. Im Gegensatz zu OpenCV wurde AForge komplett in der .NET-Programmiersprache C# geschrieben. Eine Erweiterung zu AForge.Net bildet das *Glyph Recognition And Tracking Framework* (kurz *GRATF*), das eine Unterstützung zur Erkennung von Mustern wie z.B. QR-Codes ermöglicht. Dieses kann für OCR-Verfahren benutzt werden. Kombinierbar mit AForge.NET existiert die Software *Image Processing Lab*, die es ermöglicht, verschiedene Filter aber auch Merkmal-Erkennungsmethoden (wie z.B SUSAN oder Moravec) auf Bilder anzuwenden. Das Ergebnis der Bildanpassungen kann exportiert werden und als Eingabe für AForge.NET dienen.
Neben AForge und OpenCV existieren weitere Bibliotheken, u.a. SimpleCV[70] (für die Programmierung von Bilderkennungsverfahren mit Python), Accord.NET[71] (mit dem Schwerpunkt auf Signalverarbeitung und statistische Applikationen) oder VLFeat[72].

Optical Character Recognition

Prinzip

OCR ist die Abkürzung von *Optical Character Recognition*. Bei diesem Verfahren geht es um die Texterkennung von Zeichen in Dokumenten (z.B. Bilder, Texte oder Formulare), die durch ein optisches Eingabegerät wie z.B. einen Scanner oder eine Digitalkamera erzeugt worden sind. Das Bildformat dieser Geräte ist eine Rastergrafik (gebildet aus Pixeln), die für eine weitere elektronische Verarbeitung nicht benutzt werden kann, da sie keine für den Computer lesbaren Zeichen (z.B. in ASCII-Kodierung) enthält [BDG+00].

[70] http://simplecv.org
[71] http://accord-framework.net
[72] http://www.vlfeat.org/overview/tut.html

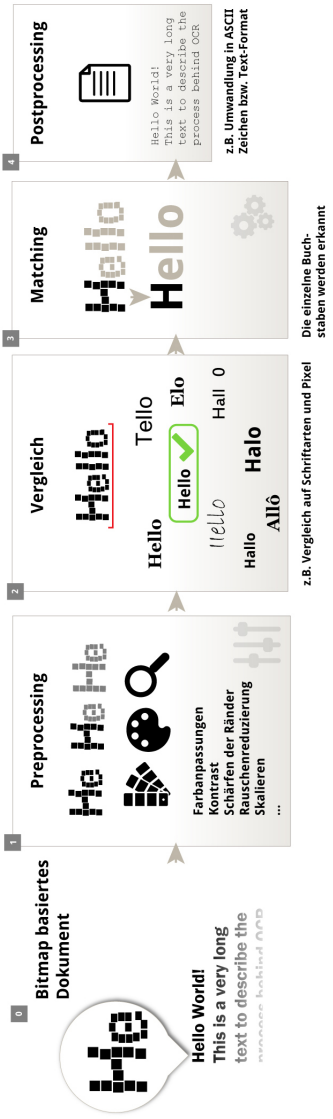


Abbildung 3.37: Prinzip der OCR-Erkennung

Die Aufgabe der OCR-Engine besteht darin, aus der Rastergrafik und deren Pixelinformationen die einzelnen Muster zu erkennen, zu extrahieren und sie mit einer Referenzdatenbank zu vergleichen (siehe Abbildung 3.37). Ähnlich wie bei der automatischen Bilderkennung werden die Eingabegrafiken erst automatisch normalisiert, d.h. Rauschen wird entfernt, optische Verzerrungen werden minimiert, der Schwarz-Weiß-Kontrast erhöht, die Farbinformationen angeglichen und eine Fehlerkorrektur auf Pixelebene (was bei Handschrifterkennung sehr oft nötig ist) vorgenommen. Damit kann die Erkennungsrate gesteigert werden. Dieses Verfahren nennt man *Preprocessing*.
In einem zweiten Schritt werden die besonderen Merkmale extrahiert. Hierbei können mehrere Verfahren benutzt werden, wie z.B. das *Template Matching*. Dabei wird ein Vergleich der Schrift mit vorher bekannten Schriftarten durchgeführt, indem eine strukturelle Analyse und Verteilung der einzelnen Pixelpunkte durchgeführt wird. Eine Erweiterung bildet die globale Merkmalerkennung. Diese erfasst die grafische Pixeleigenschaft (bzw. Struktur) von Wort und Buchstabe und verbindet diese mit einer globaleren Erkennung. Dieses Verfahren wird auch als *Intelligent Character Recognition* bezeichnet [Cha13].
Als Ergebnis dieses Schrittes werden die einzelnen erkannten Buchstaben oder Worte in Klassen eingeordnet. Diese Zuordnung kann mittels Bayesschen oder neuronalen Netzwerken durchgeführt werden.
Im letzten Schritt des OCR-Verfahrens (dieser Schritt wird auch *Postprocessing* genannt) werden die einzelnen Buchstaben und Worte, ausgehend von den ausgewählten Schriftarten, wieder zusammengefügt. Hierbei stützt man sich auf die Tatsache, dass der Raum zwischen einzelnen Worten immer grösser ist als der zwischen einzelnen Buchstaben. Das Ergebnisdokument kann anschließend mit einem Wörterbuch auf Plausibilität der erkannten Begriffe hin gegengeprüft

werden. Wenn all diese Schritte durchgeführt worden sind, ist ein für den Computer lesbareres ASCII-basiertes Dokument erzeugt worden. Diese einzelnen Bearbeitungsschritte und angewandten Verfahren werden detailliert in [Eik93] und [CKLS07] beschrieben.

Die Anwendungsdomänen von OCR sind sehr vielfältig. OCR wird z.B. bei der automatischen Digitalisierung von Bibliotheken und älteren Büchern (Projekt Gutenberg [Leb10]), der automatischen Auswertung von Autokennzeichen, der Erkennung von in Metall eingravierten Nummern im Industriekontext [ZCT14] oder für die Erkennung von Logos in Bildern [ZZL+12] benutzt.

Auch bei der Videoanalyse werden OCR-Verfahren verwendet. Bei [XT08] [TGLZ02] und [LVJ05] wird OCR für die Extraktion von Themen aus Nachrichtensendungen benutzt. Zunächst werden Videos auf Textpräsenz überprüft. Werden Texte in einem Videoframe erkannt (z.B. in einer Einblendung mit dem Namen des Moderators bzw. des angesprochenen Themas oder die Erkennung des Namens eines Sportlers anhand der Aufschriften seines Trikots), wird dieses Frame gespeichert und mit dem jeweiligen erkannten Text annotiert. Alternativ kann die grafische Position der Textzonen gespeichert werden. Nachträglich kann die Position des Einzelbildes und die darin beinhaltenden textuellen Annotationen als Referenz für eine videobasierte Suche genutzt werden. In den Arbeiten von Landais [LVJ05] wurde dieses Verfahren für die Annotation von Videos im Rahmen der Annotationsarbeiten der Videodokumentationsdatenbank des INA (Institut National de l'Audiovisuel[73]) evaluiert. Eine ähnliche Anwendung wurde bei [SBM+06] beschrieben. Hierbei wurde OCR als Vorannotationsvorgang bei der automatischen Annotation von Videoclips der Fernsehkanäle ABC und CNN benutzt. Das letzte Beispiel zeigt, dass OCR, als video-

[73] `http://www.ina.fr`

basierte Technologie für die Erkennung von Texten innerhalb eines Semantifizierungs-Workflows als geeignet betrachtet werden kann.

Werkzeuge

Die bekannteste und von Entwicklern meist eingesetzte OCR-Software bzw. API ist Tesseract. Tessaract ist eine Open Source OCR Engine (Apache Apache License 2.0) [Smi07], die ursprünglich von 1985 bis 1995 in den HP Labs entwickelt und seit 2006 von Google übernommen wurde. Seitdem wurde sie permanent weiterentwickelt und unterstützt mittlerweile über 60 Sprachen. Tesseract ist nur kommandozeilenbasiert und wird für verschiedene Betriebssysteme angeboten. Open Source Projekte wie z.B. Lector[74] oder SunnyPage[75] bieten eine grafische Benutzeroberfläche und Schnittstellen zu Tesseract.
Tesseract ist besonders im Bereich der Erkennung von gedruckten Texten mit bekannten Schriftarten performant. Laut der Tesseract-Webseite ist die Tesseract-Bibliothek für Handschrifterkennung nicht sehr gut geeignet. Hierzu kann eine andere Bibliothek Namens *Lipi Toolkit*[76] benutzt werden.
Neben Tesseract existieren weitere Open Source OCR-Lösungen wie z.B. GOCR[77] oder dem vom DFKI entwickeltem OCRopus[78,79], die jedoch der gleichen Philosophie wie Tesseract folgen: einerseits die rein kommandozeilenbasierte Bedienung und andererseits eine größere Modularität bei der Auswahl der Sprachen und der Anpassung der Erkennung. Andere kommerziell verfügbare Lösungen wie ABBYY

[74] https://code.google.com/p/lector/
[75] http://www.sunnypage.ge/en/
[76] http://lipitk.sourceforge.net/
[77] http://jocr.sourceforge.net
[78] http://www.dfki.de/web/forschung/projekte?pid=396
[79] http://www.ocropus.org

FineReader[80] oder Adobe Acrobat unterstützen die Erkennung von Text aus hundertneunzig unterschiedlichen Sprachen und bieten eine PDF-Exportfunktionalität an. Da es sich um integrierte Softwares handelt, die keine Schnittstelle über eine Kommandozeile anbieten, können diese bei von Entwicklern selbstgeschriebener Software nicht verwendet werden.

Neben desktop-basierten Versionen von OCR-Engine existieren sämtliche Cloud-basierte OCR-Werkzeuge und APIs bzw. Engines (z.B. OCRAPIService.com oder FreeOCR[81]), die es ermöglichen, nach dem Hochladen eines Bilddokuments ein Dokument in RTF-, PDF-, Word- oder Text-Format mit den extrahierten Texten generieren zu lassen. Bei diesen Lösungen ist es nicht immer klar, welche Verfahren serverseitig benutzt werden und wie der Datenschutz während des Analysevorganges gewährleistet wird.

Obwohl Cloud-basierte OCR-Lösungen gelegentlich von Nutzen sein können, offerieren sie nicht die gleiche Leistung in puncto Schnelligkeit wie eine lokal installierte Lösung. Zusätzlich sind diese OCR-Lösungen auf bestimmte Dokumentauflösungen beschränkt, sodass sie z.B. für umfangreichere Dokumente oder Bildmaterial nicht einsetzbar und geeignet sind.

Cloud-basierte Werkzeuge für die automatische Erkennung und Extraktion von Semantik aus Bild und Video

Die automatische Erkennung und Extraktion von Annotationen aus Bildern setzt voraus, dass große Rechenleistung verfügbar ist. Werden Ex-

[80] `http://www.abbyy.de`
[81] `http://www.free-ocr.com`

traktionen von Merkmalen aus größeren Datenbanken durchgeführt, besteht die Gefahr, dass die Performanz eines einzelnen Rechners alleine nicht mehr ausreicht, um die aufwendige Bildanalyse in Echtzeit zu realisieren. Um dieses Performanzproblem zu umgehen und verteilt eine Erkennung von Bildern innerhalb einer akzeptablen Zeit durchführen zu können, können Entwickler auf externe Cloud-basierte Dienste zurückgreifen.

LookThatUp

Der Dienst LookThatUp[82] bietet kostenpflichtig verschiedene Bilderkennungsfunktionen für mobile Dienste an. Die LookThatUp API ermöglicht entweder über eine Software oder eine Cloud-basierte Schnittstelle die Bildsignatur (*Image DNA*) zu extrahieren und diese mit vorgefertigten Bildersets zu vergleichen. Diese Sets können entweder vom Benutzer selbst definiert werden (über das Hochladen von Referenzbildern) oder aus dem Web stammen.
Die Programmierschnittstelle von LookThatUp ermöglicht es, eine bildspezifische Analyse durchzuführen, um z.B. nach Farbe, Inhalt oder nach kolorimetrischen und geometrischen Transformationen zu suchen. Die LookThatUp-API bietet Funktionalitäten, um z.B. Artefakte, wie das Weichzeichen von Bildern, die von mobilen Endgeräten aufgenommen wurden, zu reduzieren.

Tineye.com

Tineye[83] bietet mit seinem Konzept des *Reverse Image Search* eine sog. inverse Bildsuche an. Das bedeutet, dass für ein gegebenes Bild

[82] `mobile.ltutech.com`
[83] `https://www.tineye.com`

seine ursprüngliche Webseite wiedergefunden und ihm zugeordnet wird. Über eine API oder eine Webschnittstelle in Form eines Formulars kann der Benutzer sein Bild hochladen. Die Tineye-Suchmaschine kann die Quellen seines Bildes und die ursprüngliche Webseite zurückverfolgen. Nachträglich kann in Zusammenhang mit dem Text herausgefunden werden, welche Schlüsselwörter zum Bild passen. Tineye ermöglicht eine Suche mit mehreren Filteroptionen wie z.B. *Best Match* oder sogar *MostChanged*, um offenzulegen, ob ein Bild nachträglich verändert wurde. Diese Funktionalitäten sind primär zur Aufdeckung von Copyright- und Rechte-Verletzungen gedacht, damit sorglos kopierte und veränderte Medien im Web zurückverfolgt werden können.

Recognize.im

Das Online-Werkzeug Recognize.im[84] ermöglicht das Erstellen einer Fotodatenbank, die nachträglich als Grundlage für die automatische Erkennung von Objekten und Bildern benutzt werden kann. Dazu muss ein Entwickler ein erstes wiederzuerkennendes Bild auf den Server von Recognize.im hochladen und manuell annotieren. Die automatische Analyse wird gestartet, wenn ein Benutzer über eine App oder die dedizierte Webschnittstelle ein zweites Bild auf den Servern hoch lädt. Wenn beide Bilder 1:1 visuell übereinstimmen, kann eine vordefinierte Aktion wie z.B. der Aufruf einer Webseite ausgelöst werden. Diese Recognize.im Funktionalität wird in Kombination mit Markenlogoerkennung innerhalb von Werbe-Apps benutzt.

[84] `Recognize.im`

MoodStocks

Ähnlich wie bei Recognize.im bietet Moodstocks[85] eine Plattform zum Vergleich von Bildern auf Basis vorheriger hochgeladener Bilddateien. Der Benutzer kann Live-Videos auf die MoodStocks-Server übertragen (diese können über eine mobile App gestreamt werden) und diese mit seinen voreingestellten Bildern vergleichen lassen. Die Benutzung des Dienstes ist kostenpflichtig und von der Anzahl der voreingestellten Bilder abhängig. Maximal 100.000 Bilder können in der Datenbank abgelegt werden. Moodstocks bietet eine API für eine breite Palette von Programmier- und Geräteplattformen wie z.B. iOS oder Android an.

CloudCV

Der Dienst CloudCV[86] bietet eine Cloud-basierte Objekterkennung und Klassifikation von Bildern [ACC+13] an. Diese online-basierte Benutzung von Bilderkennungsdiensten und Algorithmen wird als CVaaS (engl. Computer Vision as a Service) [SEA+15] [THY15] bezeichnet. Dieser Dienst kann über Matlab- und Python-basierte APIs von Entwicklern genutzt werden. Die Erkennung benutzt u.a. Benchmark-Datensets von Pascal VOC oder ImageNet. Im Backend von CloudCV werden GraphLab[87] zur Parallelisierung der Analyseverfahren und OpenCV für die Bildanalyse benutzt.

[85] http://www.moodstocks.com
[86] http://www.cloudcv.org
[87] http://select.cs.cmu.edu/code/graphlab

SkyBiometry

SkyBiometry bietet zwei Cloud-basierte Analyseverfahren für Gesichter an. Die erste beruht auf einer Gesichtserkennung, die zweite auf einer Gesichtsanalyse. Bei der ersten werden die Merkmale von Gesichtern (z.B. Geschlecht, Gesichtsmimik, Öffnung der Augen, Präsenz von Brille, Emotionen wie „lachend", „traurig") inklusive der Neigung des Kopfes, die mittels der Position der Nase, des Mundes und der Augen berechnet wird, erkannt und ausgewertet. Mit dem zweiten Dienst können zwei Bilder miteinander verglichen werden, um letztendlich ein Ähnlichkeitswert zu bestimmen. Dieser prozentuale Wert gibt an, ob Personen vom ursprünglichen Bild in einem anderen Bild zu finden sind. Der Aufruf dieser Funktion wird über eine REST-API realisiert.

IBM Watson Services

Unter den Namen IBM Watson Services[88] vermarket IBM eine Reihe von REST-basierten Diensten in den Bereichen der Textanalyse, Konzeptextraktion und Emotionserkennung. Darunter der vorhin vorgestellte Alchemy Dienst. Darüber hinaus bieten IBM Watson Services online-basierte Sprachanalyse- und Transkriptionsmöglichkeiten. Ein besonderer Dienst bildet IBM Watson Personality Insights[89,90], der anhand von Textdaten aus den sozialen Netzwerken (z.B. die Twitter-Timeline eines Benutzers) versucht, psychologische Merkmale automatisch zu identifizieren, um Kaufentscheidungen besser zu bestimmen und potentielle Verhaltensmerkmale eines Benutzers vorherzusagen. Verschiedene Quellcodes und Beispiele helfen Entwicklern die REST-

[88] `http://www.ibm.com/cloud-computing/bluemix/watson`
[89] `https://personality-insights-livedemo.mybluemix.net`
[90] `https://www.ibm.com/smarterplanet/us/en/ibmwatson/developercloud/doc/personality-insights/sample.shtml`

APIs in Apps zu integrieren und können von der offiziellen IBM-Webseite entnommen werden.

Microsoft Cognitive Services

Microsoft Cognitive Services[91] bieten seit April 2016 über 21 unterschiedliche APIs in den Bereichen Textextraktion, Spracherkennung, videobasierte Entitäten-erkennungen und Suchalgorithmen auf Basis der Bing-Suchmaschine an. Diese APIs können alle über REST aufgerufen und somit in HTML5-basierte Apps eingebunden werden. Weitere Informationen über die umfangreichen Möglichkeiten dieses Dienstes können unter der Microsoft Cognitive Services Webseite nachgeschlagen werden.

Google Cloud Platform

Eine weitere Dienstplattform, die ermöglicht, Kontext- und Entitätenerkennung innerhalb eines Textes durchzuführen, bildet die Google Cloud Platform und insbesondere das Produkt Google Cloud Natural Language[92,93]. Letzteres ermöglicht den Zugriff auf Dienste, die zur Textanalyse aufgerufen werden können: die Sentimentanalyse, die Entitätenerkennung und die Syntaxanalyse bzw. die Analyse des Aufbaus eines Satzes.
Parallel dazu und ähnlich dem bereits vorgestellten Cloud CV-Ansatz bietet Google die Cloud Vision API[94] an. Mit dieser können Entwickler ihre Bilder über eine REST-Schnittstelle hochladen und über mehrere

[91] https://www.microsoft.com/cognitive-services/en-us/apis
[92] https://cloud.google.com/natural-language
[93] https://cloudplatform.googleblog.com/2016/07/the-latest-for-Cloud-customers-machine-learning-and-west-coast-expansion.html
[94] https://cloud.google.com/vision

unterschiedliche Verfahren wie z.B. Bildkategorie-, Logo-, Gebäude und Gesichtserkennung sowie OCR und Bildeigenschaften-Extraktion analysieren lassen.

Übersichtstabellen

Folgende zwei Tabellen (siehe Abbildungen 3.38 und 3.39) fassen die Dienste und Werkzeuge, die für eine automatische Erkennung und Extraktion von Semantik aus Multimediadaten im Rahmen dieser Arbeit betrachtet worden sind, zusammen. Zusätzlich, in Hinblick auf Swoozy und auf die Berechnung des opportunen Zeitpunktes der visuellen Einblendung von Informationen während der Videowiedergabe, wurde bei den Extraktionsmerkmalkriterien überprüft, ob diese in der Lage sind, Stimmung und Emotionen innerhalb eines Bildes zu erkennen.

			Kurzvorstellung
			Anwendungsbereiche & Hauptmerkmale
Online/Cloud-basiert	Textanalyse & Extraktion	**Alchemy Language**	Alchemy Language ist Teil einer Lösung zur Analyse von Kunden-verhalten und Webinhalten (Tweets, Emails, Seite).
		AYLIEN	Die AYLIEN Text Analysis API bietet 8 NLP Werkzeuge für den Bereich Semantic Advertising und Business Intelligence
		Standford CoreNLP	Stanford NER (Named Entity Recognition) Java-basiert Bietet verschiedene NLP Module (NER, Sentiment-Analyse, usw.)
		Semantria	Analyse von Texten, Emotionen und für Marektingzwecke aber auch für interne Unternehmenswissenspool
	Bild/ Tagging und Erkennung	**Alchemy Vision**	Alchemy Vision ist Teil einer Lösung zur Bildanalyse. Wird mit der Alchemy Language API angeboten
		AYLIEN	Die AYLIEN Text Analysis API bietet ein automatisches Taggen basierend auf Bildern
		Cloud CV	Cloud-basierte Lösung für die Erkennung von Objekten, Personen und Eigenschaften innerhalb von Bildern. API für Matlab und Python
		Google Cloud Platform	Cloud-basierte Lösung für Text-, Sprache, Bild und Videoanalyse
		IBM Watson	Cloud-basierte Lösung für Text-, Sprache und Videoanalyse
		KeyLemon	Integration von Gesichtserkennung in Internet Applikation für biometrische Authentifikation über Webkameras
		LookThatUp	LTU-Plattform für Bilderkennung, Farbmatching und Ähnlichkeits-suchen für Urheberrecht- und e-Commerce Applikationen

Abbildung 3.38: Auflistung der Dienste und Werkzeuge für eine semantische Extraktion 1/6

		Extraktionsmerkmale	
	Lizenz/ Preis	**Granularität**	**Emotionen/ Sentiment**
Alchemy Language	kostenlos (beschränkte API) /kommerziell	hohe	✔
AYLIEN	kostenlos / kommerziell	hohe	✔
Standford CoreNLP	GNU GPL	mittel (3 bis 7 Klassen)	✔
Semantria	kommerziell/ kostenlos	mittel	✔
Alchemy Vision	kostenlos (beschränkte API) /kommerziell	hoch bei Gesichts-erkennung	✘
AYLIEN	kostenlos	mittel bis hoch über Konfidenz-werte	✘
Cloud CV	kostenlos/ Open Source	hoch Deep Learning Ansatz	✘
Google Cloud Platform	kostenlos/ kommerziell	hohe	✔
IBM Watson	kostenlos/ kommerziell	hohe	✔
KeyLemon	kostenlos/ kommerziell	-	✘
LookThatUp	kostenlos/ kommerziell	anwendungs spezifisch	✘

Abbildung 3.39: Auflistung der Dienste und Werkzeuge für eine semantische Extraktion 2/6

	Verbindung zu Webdiensten		Weitere Merkmale
Alchemy Language	✓	u.a. mit DBPedia, Freebase MusicBrainz, OpenCyc	Entitätenextraktion Relationsextraktion Microformat Parsing
AYLIEN	✓	DBPedia Twitter Hashtag Vorschläge	Entitätenextraktion Relationsextraktion Microformat Parsing
Standford CoreNLP	✗	wird nicht von der API angeboten	erweiterbar durch Quellcode und Modell-erweiterungen
Semantria	✗	wird nicht angeboten	Analyse über Excel-Dokumente
Alchemy Vision	✓	Verbindung mit Knowledge Graph und ggfs. Webseiten	Analyse, Tagging und Bildähnlichkeitssuche
AYLIEN	✗	wird nicht angeboten	automatisches Taggen, Konfidenzwerte
Cloud CV	✗	wird nicht angeboten	Stichtching von Bildern, Objekterkennung und Klassifikation VIP-Modus
Google Cloud Platform	✓	Verbindung mit Wikipedia-Einträge	Bietet eine online-basierte OCR-Funktionalität
IBM Watson	✓	Dienstabhängig	Lösung bietet über 17 APIs an
KeyLemon	✗	wird nicht angeboten	Geschlechts-, Alter- und Gesichts-merkmalerkennung
LookThatUp	✗	wird nicht angeboten	Ähnlichkeitssuchen auf Farbe und Inhalts-basis

Abbildung 3.40: Auflistung der Dienste und Werkzeuge für eine semantische Extraktion 3/6

		Kurzvorstellung **Anwendungsbereiche & Hauptmerkmale**
Online/Cloud-basiert	Bild/ Tagging und Erkennung	**Microsoft Cognitive Services** — Cloud-basierte Lösung für Text-, Sprache und Videoanalyse
		MoodStocks — Bilderkennung für mobile Applikationen für Marketing, Print und Packaging. REST-basiert. API für iOS, Android und Google Glass
		Nametag.ws — Live Bild- und Gesichtserkennung. Erkennung von Personen über Google Glass. Unterstützt von Facialnetwork.com
		Recognize.im — Bild- und Objekterkennung API für mobile Endgeräte
		SkyBiometry — Cloud-basierte Lösung für Gesichterkennung Schwerpunkt auf biometrische Parameter
		Tineye.com — Inverse Bildsuchmaschine mit Fokus auf Ähnlichkeitssuche und Urheberrechtsverletzungs und Nachweise im Web
		Wirewax — Interaktive und automatische Objekt- und Gesichtserkennung aus online oder hochgeladenen Videos
Offline		**OpenCV** — Programmbibliothek für maschinelles Sehen u.a. Gesichts-erkennung, Module für Robotik, Gestenerkennung, 3D Tracking
		Tesseract — Optical Character Recognition Engine (OCR) zur Texterkennung in über 60 Sprachen.

Abbildung 3.41: Auflistung der Dienste und Werkzeuge für eine semantische Extraktion 4/6

			Extraktionsmerkmale	
			Lizenz/ Preis	Granularität	Emotionen/ Sentiment
Online/Cloud-basiert	Bild/ Tagging und Erkennung	Microsoft Cognitive Services	kostenlos/ kommerziell	hohe	✓
		MoodStocks	kostenlos/ kommerziell	Benutzer muss Bilder vorher hochladen	✗
		Nametag.ws	k.A	unbekannt	✗
		Recognize.im	kostenlos/ kommerziell	Benutzer muss Bilder vorher hochladen	✗
		SkyBiometry	kostenlos/ kommerziell	hohe präzise Merkmal-Extraktion	✓
		Tineye.com	Web: kostenlos/ API: kommerziell	abhängig vom Bildvergleich	✗
		Wirewax	kostenlos/ kommerziell	limitiert	✗
Offline		OpenCV	BSD Lizenz	einstellbar	möglich durch Anwendung der Klassen
		Tesseract	frei/ Apache License 2.0	variabel: hängt von der Qualität des Eingabebildes	✗ nicht zutreffend

Abbildung 3.42: Auflistung der Dienste und Werkzeuge für eine semantische Extraktion 5/6

				Entwicklung	
Verbindung zu Webdiensten		**Weitere Merkmale**	**API**	**Ergebnisformate & Strukturen**
✔	Verlinkung mit Bing und ggfs. Webseiten	Lösung bietet über 20 APIs an	✔	JSON
✘	wird nicht angeboten	eher für eine Benutzung in Rahmen mobiler Anwendungen	✔	JSON
✘	keine Informationen	aktueller Zustand ist unklar	k.A	-
✘	nicht zutreffend	eher für mobiles Marketing	✔	XML/JSON über SOAP/REST API
✘	nicht zutreffend	Geschlechts, Alter- und Gesichts-merkmalerkennung	✔	Ergebnisse als XML Struktur
✔	liefert die Quellen URL der zutreffenden Bildern	Vergleicht Bilder mit einer Datenbank von 10.000.000.000 Bildern	✔	Ergebnisse als XML Struktur
✔	URL zu Zusatzmedien	eher für online Videos automatische Erkennung zur Zeit noch limitiert	✘	keine Export-möglichkeit
✘	nicht zutreffend	sehr große Auswahl von Bibliotheken und Algorithmen	✔	kann vom Programmierer selbst bestimmt werden
✘	nicht zutreffend	sehr große Auswahl von Bibliotheken und Algorithmen	✔	kann vom Programmierer selbst bestimmt werden

Abbildung 3.43: Auflistung der Dienste und Werkzeuge für eine semantische Extraktion 6/6

Werkzeuge für die Annotation multimedialer Daten

Bei den Ergebnissen der Analyse und der Extraktion von Konzepten oder Entitäten aus Videos ist nicht auszuschließen, dass die Bilderkennungsalgorithmen sog. *False-Positives* liefern. Das bedeutet, dass die Erkennung zwar erfolgreich automatisch durchgeführt wurde, aber letzten Endes das Bild schlecht klassifiziert wurde und die resultierende Beschreibung zu ungenau oder gar falsch war. Ähnlich sieht es bei der automatischen Wissensextraktion aus Texten aus. Hier kann nicht immer gewährleistet werden, dass das extrahierte Wissen hundertprozentig passend ist. Das schlechte Erkennungsergebnis lässt sich auf zwei Problembereiche zurückführen: entweder sind die Eingangsdaten mit Rauschen versehen (z.B. ist das Bild nicht eindeutig genug oder es weist Probleme optischer Natur wie einen schlechten Kontrast, eine Verzerrung oder eine Unterbelichtung auf) oder es liegen zu wenige oder mangelnde Trainingsdaten beim Lernverfahren vor. Das Ziel, gleichzeitig gute Qualität und hohe Granularität während der Bilderkennung zu erzielen, kann insofern kontraproduktiv sein, als dass es bei einer Extraktion von Wissen aus nicht trainierten Domänen zu einer hohen Fehlerquote kommt.

Speziell bei Videos stellen schnelle Aktionen oder unbekannte, nicht trainierte Gegenstände eine besonders hohe Anforderung an die Rechenleistung und die Analysekomponente, die meistens im Live-Kontext nicht zur Verfügung steht. Dadurch besteht die Gefahr, dass unter Umständen automatische Extraktionsverfahren sehr spärliche oder gar keine sinnvollen semantischen Informationen liefern.

Um trotzdem eine sinnvolle Verknüpfung von Wissen mit Videos/Bildern zu realisieren, können parallel zur automatischen Erkennung,

Werkzeuge zum Einsatz kommen, die den Benutzern bzw. Medienerstellern die Möglichkeit geben, während der Produktion oder im Nachgang ein Video mit einer korrekten semantischen Beschreibung zu versehen. Dieser Schritt wird als manuelle Annotation bezeichnet. Die große Popularität von Videoplattformen wie YouTube oder Vimeo bringt es mit sich, dass manuelle Annotationen eine immer wichtigere Rolle spielen. Über diese können Verlinkungen zu ähnlichen Inhalten oder weitere Videos präzise und Frame-genau hinzugefügt werden.
Im professionellen Bereich sind Annotationsaufgaben sehr zeitaufwendig und stoßen sehr oft auf technische Probleme, die sich während des Produktionsworkflows (z.B. bei Montage und Schnitt) ergeben. Tatsächlich verursachen die Heterogenität der verschiedenen benutzten Videobearbeitungssoftwares und deren Metadaten-Ausgabeformate (siehe Kapitel 2) ein Problem mit der Kompatibilität der Ausgabeformate. Diese sind immer noch sehr heterogen. Die resultierenden Ergebnisformate lassen sich nur mit manuellen Anpassungen von einem Format ins andere übertragen. Bei diesen Anpassungen können Semantik, Entitäten und Konzeptinformationen verloren gehen. Zusätzlich fehlt bei diesem Schritt die Möglichkeit, das Material direkt mit einer semantischen Information, basierend auf eine Ontologie, zu verknüpfen. Um diese Probleme zu lösen, kann auf plattformunabhängige Videoannotationswerkzeuge zurückgegriffen werden. Diese Werkzeuge bieten entweder während des Produktions-Workflows oder in der Post-Produktion eine effiziente Möglichkeit, Videos zu annotieren und zu klassifizieren und somit mit Semantik zu versehen. Die Werkzeuge, die für diese Arbeit am relevantesten sind, wurden teilweise auf Basis von [DGL+11] selektiert und im Folgenden kompakt dargestellt.

Caliph

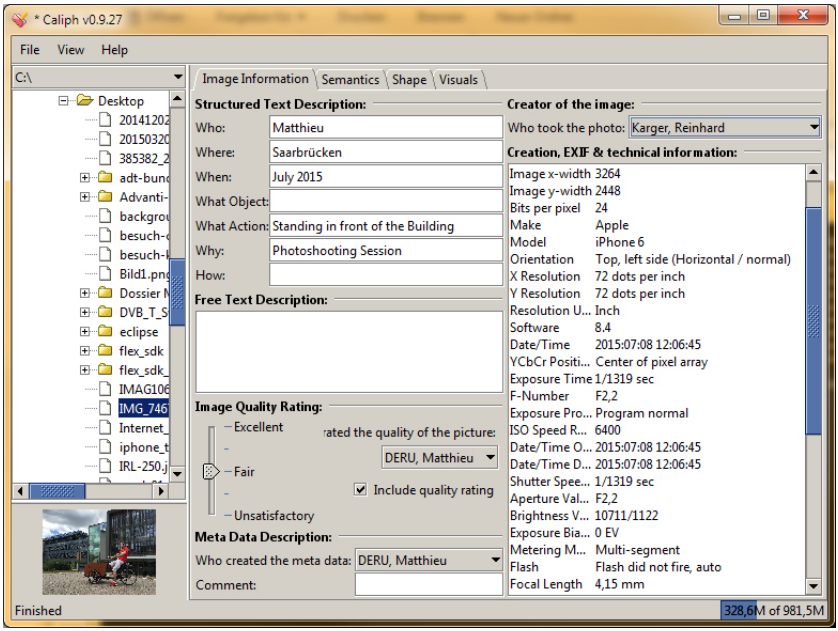


Abbildung 3.44: Bildschirmabzug der Software Caliph

Caliph[95] ist ein MPEG-7-basiertes Bildannotationswerkzeug, das Benutzer nicht nur digitale Bilder frei annotieren (u.a. über die Metadaten und ein Rating-system) lässt, sondern auch schon vorhandene Bilder-Metadaten (z.B. IPTC und EXIF - siehe Kapitel 2) in das MPEG-7-Format konvertiert. Zusätzlich werden Bildinformationen (*Descriptors*) wie z.B. das *ColorLayout* (die räumliche Verteilung der einzelnen Farben innerhalb eines Bildes), das *EdgeHistogramm* (die Bildkantenverteilung), die *DominantColor* (die im Bild dominanten Farben) und die *Scalable Color* (das Farbenhistogramm, das nach einer Haar-Transformation im Farbschema HSV enkodiert worden ist) aus den Mediendaten ex-

[95] `http://www.semanticmetadata.net/features`

trahiert und in das MPEG-7-Format transformiert [LC08]. Die Caliph-Bedienoberfläche ist so aufgebaut, dass der Benutzer ein Bild auswählt und eines von vier Tabs (*Image Information* in Abbildung 3.45) selektieren kann.

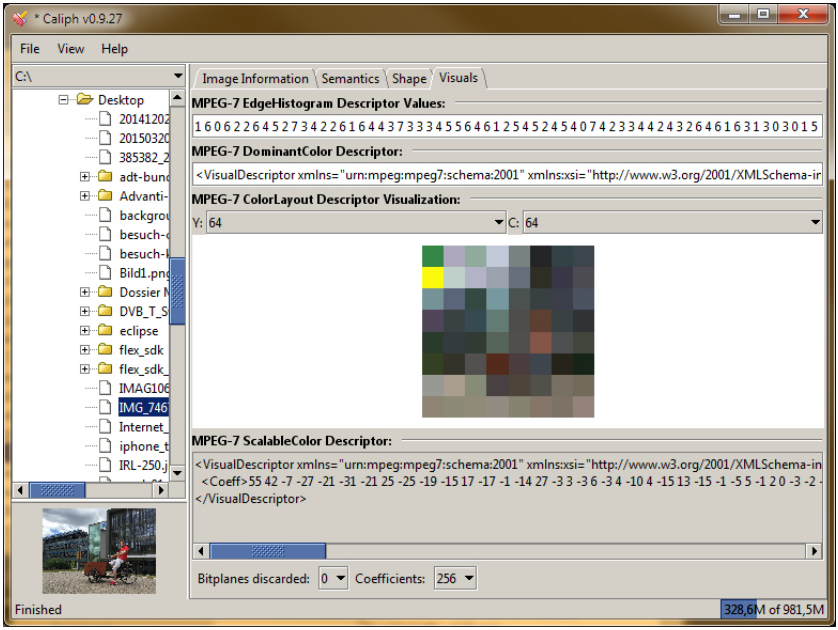


Abbildung 3.45: MPEG-7 Spezifikation in Caliph

Ab diesem Zeitpunkt kann der Benutzer eine strukturierte Textbeschreibung in Freitextfeldern angeben und erste Metadaten (Ersteller, Zeitpunkt) festlegen. Das Tab *Semantics* ermöglicht eine interaktivere Gestaltung der Annotation, indem per Drag'n'Drop semantische Konzepte (Personen, Orte, Zeitpunkte) angelehnt an die MPEG-7-Spezifikation (Agents, Events, Time, Place and Object), ausgewählt und diese mittels eines grafischen Editors verknüpft werden können. Im Hintergrund werden Relationen, die semantisch beschrieben wor-

den sind wie z.B. *locationOf, agentOf, accompanierOf* generiert. Die Relationstypen können nachträglich grafisch editiert werden und die MPEG-7-spezifischen Werte unter dem Tab *Visuals* angezeigt werden. Neben dieser Funktionalität bietet Caliph die Möglichkeit, Teile eines Bildes (über den *Shape*-Tab in Abbildung 3.45) zu markieren und seine relevantesten Zonen zu bestimmen. Diese Zonen lassen sich nachträglich mit einer Farbe versehen, die als Marker für das Wiederauffinden des Bildes oder des Teilbildes dienen soll. Somit wird die räumliche Platzierung eines Gegenstandes oder einer Person in einem Bild gespeichert.

Eine gute Ergänzung zu Caliph stellt die Software Emir[96] dar. Nachdem der Benutzer seine Metadaten gespeichert hat, besitzt er die Möglichkeit, seine Medien über eine Suchmaske wiederzufinden, indem er entweder einen der indizierten Begriffe benutzt oder über einen Grapheditor einen semantischen Graph zusammenstellt und indirekte Fragen (wie z.B. *Suche das Bild, das in Paris vorm Eiffel Turm aufgenommen wurde und auf dem Viktoria zu sehen ist)* stellen kann. Die erstellten Graphen lassen sich anschließend speichern und deren Ergebnisse in Form einer interaktiven Clustervisualisierung darstellen.

Emir stützt sich auf die Apache Lucene-Bibliothek[97] für die Textsuche. [GVC+08] [LKG04] [Lux09]. Die Ergebnisse von Caliph und Emir sind in die Java-Bibliothek LIRE[98] eingeflossen. Diese ermöglicht über Lucene einen Index von Bildermerkmalen zu generieren und Bilder basierend auf Farb- und Texturen wiederzufinden.

[96] https://github.com/dermotte/CaliphEmir
[97] http://lucene.apache.org
[98] http://www.lire-project.net

SVAS

Die Semantic Video Annotation Suite (SVAS)[99] [RBN+07] wurde am Joanneum Institut in Graz entwickelt und unterstützt die räumlich-zeitliche Annotation von Videos. Besondere Merkmale von SVAS sind u.a. die eingebauten Videoerkennungsalgorithmen zur Unterstützung des Annotationsvorgangs. Bevor die eigentliche Annotation beginnt, wird das Video von einer separaten Komponente (*Media Analyze Tool*) zuerst nach bestimmten Merkmalen durchsucht. Diese Merkmale beinhalten kameraspezifische Konfigurationen, technische Angaben zu verwendeten Kameraeinstellungen oder Szeneschnitten und Einblendungen. Alle diese Informationen werden in eine MPEG-7-Metadatei aus dem zu analysierenden Video extrahiert und separat gespeichert. Parallel zur Bedienung der Benutzeroberfläche von SVAS werden die einzelnen Bilder innerhalb eines Hintergrundprozesses analysiert. Letztere können zur besseren Navigation und zum Auffinden von bestimmten Szenen dienen (mittels des Zeitstempels).

In einem weiteren Schritt wird über ein Bilderkennungs-Plugin eine lokale Bildsuche durchgeführt. Diese Bildersuche unterstützt den Prozess der semi-automatischen Annotationsaufgaben, indem Objekte räumlich und farblich vormarkiert werden. Je nach Qualität der Erkennung können diese Objekte als *Region of Interest* bezeichnet und vom Benutzer validiert werden.

Die automatische Erkennung und Extraktion der Position der *Regions* oder *Points of Interest* kann sich in bestimmten Szenen als sehr schwierig erweisen, insbesondere bei veränderter Kameraposition oder aufgrund der Einblendegeschwindigkeit. Daher bietet SVAS die Möglichkeit, die Videos manuell mit Hilfe einer Bedienoberfläche zu

[99] `http://www.joanneum.at/en/digital/productssolutions/sematic-video-annotation.html`

annotieren. Über eine interaktive Timeline können verschiedene Begriffe zum richtigen Zeitpunkt bzw. Zeitstempel hinzugefügt werden. Schlüsselbilder (engl. Keyframes) einer jeweiligen Szene können einzeln extrahiert werden. Dadurch ist es möglich, innerhalb eines Videos ähnlich aussehende Frames zu finden und dadurch Annotationen zu übernehmen. Um dies zu realisieren, stützt sich die Suche auf die SIFT-Merkmale, die in jedem einzelnen Frame zu finden sind. Die gefundenen ähnlichen Frames werden als Liste vorgeschlagen. Ab diesem Zeitpunkt kann der SVAS-Benutzer die jeweilige Szene annotieren und die Vorschlagsliste ergänzt werden, indem in Freitextfeldern Angaben zu Objekten, Personen und Orten gemacht werden. Die SVAS-Software ermöglicht das Speichern von Kamera-Einstellungen wie z.B. die sog. *Shot-Size, Closeup Shot, Wide Angle* (Bildobjektgröße, Nahaufnahme, Weitwinkelaufnahme) oder den Typ der Aufnahme *Birds Eye Shot (Vogelperspektive), Panorama*.

SIVA

In [MMGK14] wird das Werkzeug *SIVA Producer*, das an der Universität Passau entwickelt wurde, vorgestellt, bei dem der Fokus auf einem einfach zu bedienenden Autorenwerkzeug liegt, um damit die Erstellung von nicht-linearen Videos zu realisieren. SIVA ist in der Lage, bei allen Schritten des Videoproduktionsworkflows zu assistieren und über eine einfach gestaltete Bedienoberfläche das Editieren von Videostruktur, die Szenen-Annotation und die Generierung eines Inhaltverzeichnisses, oder besser gesagt einer Szenenablaufübersicht, zu unterstützen. Der SIVA Producer bietet Funktionalitäten zur Anreicherung von linearen und nicht-linearen Videoinhalten (sog. Hypervideos), indem der Benutzer selbst bestimmen kann, welche Informationen zum Video

oder zu einer Szene hinzugefügt werden sollen. Der Videoeditor von SIVA bietet zwei Möglichkeiten ein Video zu editieren: die erste für lineare Videos (z.B. Filme) ist ein Timeline-basiertes Editieren, in dem der Benutzer die Annotationen manuell oder über eine automatische Szenenerkennung hinzufügen kann.

Die zweite Möglichkeit, die eher für nicht-lineare Videos in Frage kommt, besteht darin, einen Szenengraph zu definieren und diesen Szene für Szene mittels Annotationen und Reihenfolgeangaben nachträglich zu editieren. Dadurch können Szenen miteinander "verknüpft" werden, insbesondere, wenn diese ähnliche Konzepte beinhalten. Falls ähnliche Szenen während des Editier-Vorgangs auftauchen, werden diese automatisch dem Benutzer vorgeschlagen.

Damit der Editiervorgang leichter zu handhaben ist, werden beim Editieren des Szenengraphs drei sog. *Semantic Zoom Levels* angeboten. Unter diesem Begriff versteht man die Möglichkeit, entweder eine Szene nur mit einem *Thumbnail* und einer kleinen Beschreibung zu versehen oder bei jedem im Video auftretenden Konzept dieses ausführlich mittels einer grafischen Unterstützung (z.B. eines Icons) zu markieren. Zusätzlich kann über einen Medienmanager weiteres Bildmaterial importiert und über den Szenengraph verlinkt werden. So ist es möglich, insbesondere bei der Definition von geografischen Orten, zusätzliches interaktives Material einzubinden. Bei dem Editier- und Annotationsvorgang spielt die Zeitleiste (engl. Timeline) die zentrale Rolle. Annotationen werden pro Szene gesetzt und sind nur einmal gültig. Es sei denn, es werden sog. globale Annotationen eingesetzt, die, wie der Name schon verrät, für das komplette Video gültig sind. Lokale Annotationen werden zeitlich mittels Balken über die Zeitleiste festgelegt. Der Zeitpunkt, wann die Annotationen sichtbar sein sollen, kann manuell über Eingabefelder festgelegt werden.

Nachdem die Annotationen gesetzt worden sind, kann aus dem resultierenden Graph ein Inhaltsverzeichnis mit allen Konzepten erstellt werden. Dieses Inhaltsverzeichnis listet mit einem Zeitstempel jede Szene und jedes Konzept auf, die in dem jeweiligen Video annotiert worden sind.
Nachdem die Annotationsarbeiten abgeschlossen sind, wird eine XML-Datei generiert, die anschließend von den verschiedenen SIVA Player-Varianten (Mobile und PC-basierte Videoplayerversionen) verarbeitet und dargestellt werden kann.

ANVIL

ANVIL[100] ist ein am DFKI entwickeltes Werkzeug zur manuellen Annotation von Videos. Das Annotationsschema ist generisch und lässt sich sehr einfach erweitern, indem neue Metadaten-Attribute hinzugefügt werden können. Dank seiner großen Palette an Funktionalitäten wie z.B. der parallelen Anzeige von 3D Motion Capturing-Daten wird ANVIL in unterschiedlichen Forschungsprojekten auf den Gebieten der Linguistik, der Anthropologie und der Ozeanographie angewandt. Ähnlich zu anderen Video-Annotation-Tools werden die räumlichen Informationen mittels einer sequentiellen Zeitleiste benutzerfreundlich dargestellt. Bei ANVIL war der ursprüngliche Gedanke, Gesten, Sprache und deren semantische Ausprägung für die Analyse der multimodalen Interaktion zu annotieren.

[100] http://www.anvil-software.org

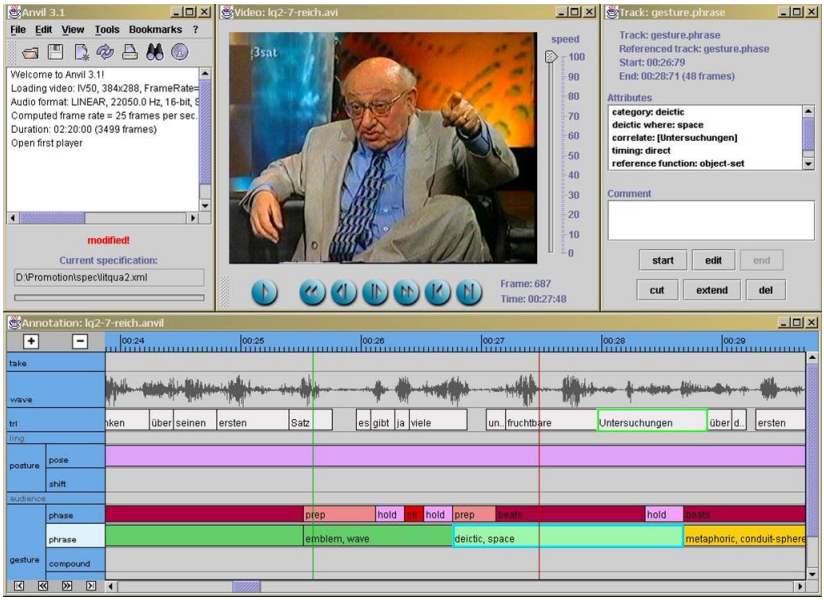


Abbildung 3.46: Benutzeroberfläche von ANVIL. Quelle: http://www.anvil-software.org/

Aus diesem Grund beinhaltet die ANVIL-Benutzeroberfläche parallele Spuren, ähnlich dem Prinzip einer Partitur (engl. timeline), die zeitgleich die Veränderung einer bestimmten Eigenschaft des Videos (z.B. Gesten, gesprochene Worte, Kopfbewegung einer dargestellten Person) grafisch darstellt. Im sog. *Annotation Board* kann manuell die Beschreibung der jeweiligen Frames eingetragen werden. Diese Annotationen erscheinen zum festgelegten Zeitpunkt und für eine vorgegebene Dauer innerhalb der Zeitleiste. ANVIL exportiert den Inhalt der Timeline samt Annotation in eine XML-basierte Datei. Für statistische Auswertungen werden die SPSS- und Statistica-Datenformate unterstützt [Kip07].

IBM VideoAnnEx

Einer der ersten MPEG-7-basierten Videoannotationseditoren war IBM VideoAnnEx [LTS03]. Obwohl das Projekt seit 2011 nicht mehr von IBM weitergeführt wird, kann man diese Software immer noch herunterladen und benutzen, um damit für Videos MPEG-7-kompatible Metadaten zu generieren.

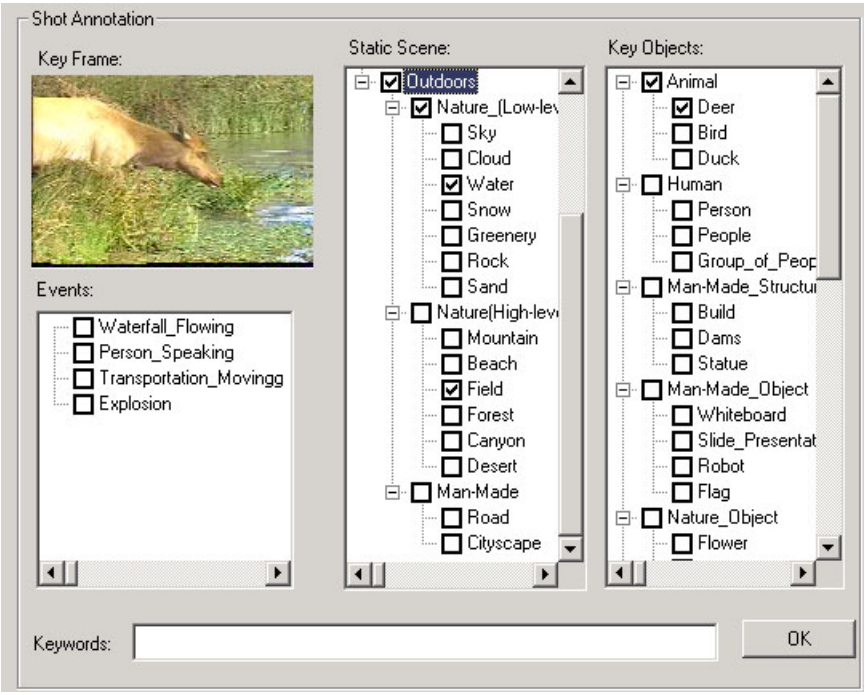


Abbildung 3.47: Bildschirmauszug der IBM VideoAnnEx Benutzerschnittstelle. Quelle: http://www.research.ibm.com/VideoAnnEx/usermanual.html

Ziel dieser Software ist es, das schnelle Editieren und Annotieren von Videosequenzen mittels MPEG-7-Metadaten zu erleichtern. Jede Videosequenz wird mit einem Set an Deskriptoren (diese werden auch als *Event-Beschreibung* bezeichnet) versehen. Alle Beschreibungen und

Annotationen werden mit der jeweiligen Sequenz verlinkt und im MPEG-7-Format gespeichert.
Die Metadaten und jeweiligen Annotationen können sich dabei entweder auf die gesamte Länge eines Videos, eine bestimmte zeitlich beschränkte Sequenz (engl. *Shots*) oder sogar auf einzelne Schlüsselbilder (engl. *Stills*) beziehen.
Informationen über Einblendungen, Kameraeinstellungen, Shots und Fade-In/Out-Video- Effekte können ebenfalls in IBM VideoAnnEx geladen werden, sowie sog. MPEG-7-kompatible *Shotfiles*, die von der Software *IBM CueVideo Shot Detection Toolkit* erzeugt wurden. Letztere ist nicht mehr erhältlich.
Die Annotation wird mittels eines vom Benutzer vorbereiteten bzw. gepflegten Lexikons durchgeführt (siehe Abbildung 3.47). Für die Hierarchisierung der Videoelemente und die Verwendung der Lexikonbegriffe werden drei Lexikon-Typen auf der grafischen Benutzerschnittstelle von IBM VideoAnnEx angeboten. Beim ersten werden sog. *Events*, wie z.B. *Person_Speaking* angelegt. Diese beschreiben den Inhalt der Aktion oder Szene innerhalb des ausgewählten Key-Frames. Im zweiten Fall können sie, falls die Szene statische Elemente (Objekte innerhalb des Videos, die sich nicht bewegen, wie z.B. der Hintergrund) beinhaltet, als *Static Scene* ausspezifiziert werden. In der Liste werden Komponenten der Szene (z.B. Sand, Wasser, Straßen, usw.) aufgelistet, die beim Annotationsvorgang vom Benutzer ausgewählt werden können. In der Liste *Key Objects* werden die eigentlich fokussierten Objekte oder Protagonisten (seien es Tiere oder Menschen) als Kategorien angeboten.
Auch wenn IBM VideoAnnEx nicht mehr weiter gepflegt wird, liefert die Software dennoch eine gute Übersicht über mögliche Annotationsfunktionalitäten, die mit dem MPEG-7-Format zu erzielen sind.

Advene

Das Advene-Projekt *Annotations de vidéo numérique échangées sur le Net* wird seit 2012 im LIRS-Labor des CNRS an der Universität Claude Bernard in Lyon in Frankreich durchgeführt[101].

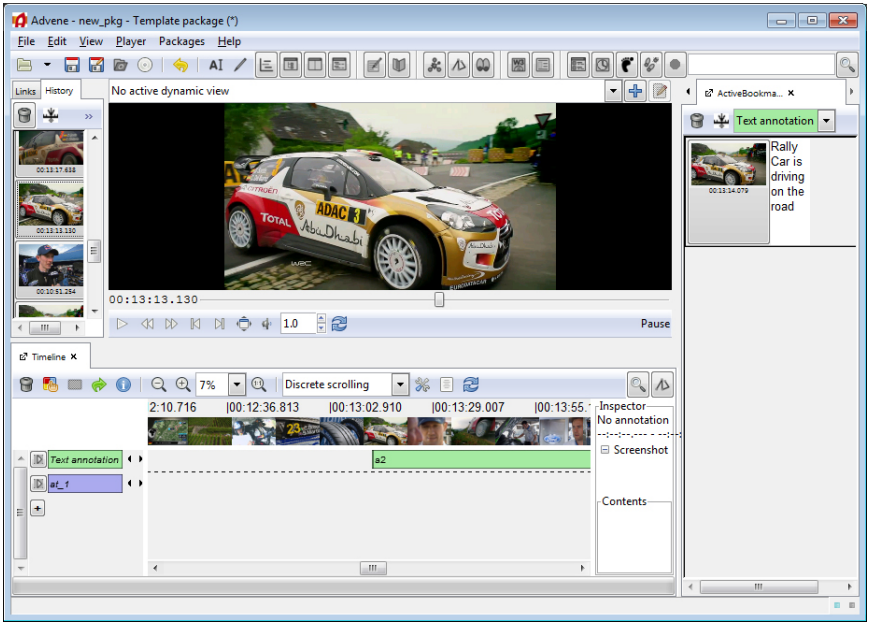


Abbildung 3.48: Benutzeroberfläche von Advene

Ziel dieses Projekts ist die Entwicklung von semantischen Modellen zur Videoinhaltsbeschreibung und einem einheitlichen Format, die es ermöglichen auf einfache Art und Weise digitale Dokumente wie z.B. Videos, Kurzfilme oder Konferenzbeiträge zu annotieren und damit verbundene Links zu passenden Webseiten und Artikeln zeitgleich anzubieten. Advene ermöglicht das Teilen von Kommentaren und das Editieren von Metadaten und Annotationen mittels verschiedener Vi-

[101] `http://liris.cnrs.fr/advene/index.html`

sualisierungen (Timeline, Baum-Visualisierung und eine Transkription). Als Ergebnis können diese Annotationen als SVG-Untertitel über ein abgespieltes Video integriert werden. Der eingebaute Player bietet die Möglichkeit, angereicherte Videos abzuspielen und zeitgleich die passenden Dokumente und Annotationen zu sichten. Die Annotationen, die zeitlich markiert sind, werden in eine ZIP-Datei gespeichert, die eine XML-basierte zeitliche Beschreibung der Szenen beinhaltet [AP05].

VCode und VData

Bei diesem Werkzeug handelt es sich um eine Software, die für die nachträgliche Annotation von Videomaterial (sog. Sessions), das aus wissenschaftlichen Experimenten stammt (z.B. medizinische Beobachtung von Patienten), benutzt werden kann. VCode[102] ist in der Lage, mehrere Videostreams abzuspielen und diese parallel mit Sensordaten zu verknüpfen. Somit können z.B. verschiedene Videos aus verschiedenen Kameraperspektiven abgespielt werden.
Die Darstellung erfolgt über eine Timeline und eine interaktive Anzeige von Sensordaten in Form von Kurven (siehe Abbildung 3.49). Die Annotationen sind wie bei den präsentierten Werkzeugen zeitbasiert und lassen sich in verschiedene Kategorien einordnen. Diese Kategorien werden vom Benutzer vor dem Start des Programms festgelegt und über die Bedienoberfläche angeboten.
Nach der Videoanalyse können die Daten über eine zusätzliche Software, VData[103], exportiert und für eine statistische Auswertung benutzt werden. Zusätzlich ermöglicht die Software das Nachladen von Sessions samt Annotationen.

[102] `http://social.cs.uiuc.edu/projects/VCode`
[103] `https://code.google.com/p/vcode`

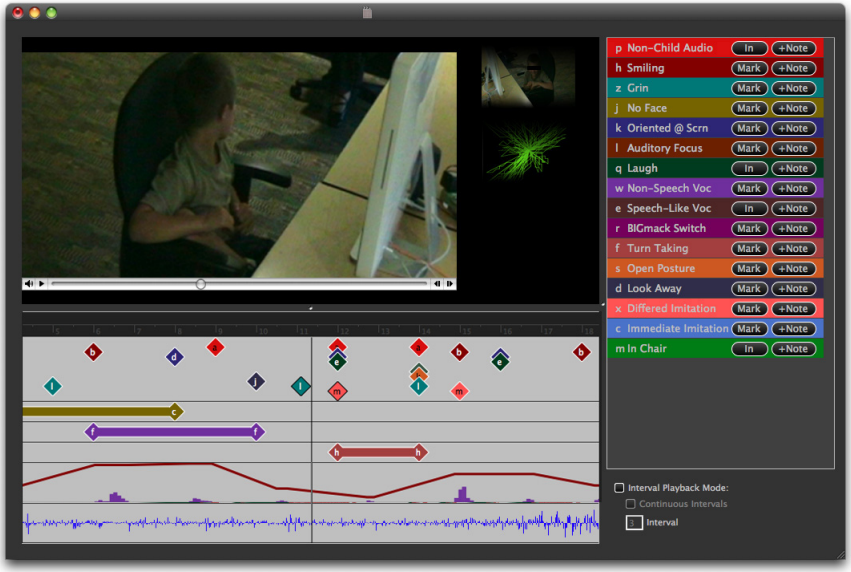


Abbildung 3.49: Benutzeroberfläche von VCode. Quelle: http://social.cs.uiuc.edu/projects/VCode

VCode besitzt eine Administrationsschnittelle, die speziell für den *Annotierer* bzw. die Person, die das Video annotiert und bewertet, entwickelt wurde, damit die benutzten Terminologien und Konzepte zwischen den Personen und während verschiedener Sessions gleich bleiben und keine Intersubjektivität entsteht.

In den meisten Fällen wird am Anfang des Video-Annotationsvorgangs ein Template generiert. Dieses enthält ausgewählte Begriffe und eine farbliche Nomenklatur der Ereignisse, die während des Annotierens möglicherweise auftauchen werden. Vorteil dieses Ansatzes ist, dass alle *Annotierer* eine einheitliche Beschreibung benutzen und somit Fehlinterpretationen vermieden werden. Die Software VCode wird unter dem Open Source-Lizenzmodell angeboten.

EXMARaLDA

EXMARaLDA (Akronym für Extensible Markup Language for Discourse Annotation)[104,105] ist ein vollständiges offline System für die manuelle computeruntergestützte Transkription und Annotation gesprochener Sprache, das am Hamburger Zentrum für Sprachkorpora (HZSK) entwickelt wurde [SW09] [Sch04] [Sch03b]. Dieses System kann u.a. für die Annotation von Videos benutzt werden. Die Editierfunktionalität wird durch das Partitur-Prinzip ermöglicht. Mehrere Audiopuren lassen sich so visualisieren und annotieren. Als Ergebnis der Transkription wird einerseits eine XML-basierte Datei generiert, die den kompletten annotierten Diskurs enthält (samt Zeitstempel) und anderseits können die Sprecher, den Ort an dem ein Interview geführt wurde, mitprotokollieren und als Korpora für eine weitere Benutzung speichern. Diese Korpora beinhalten Zusatzinformationen und speichern den Kontext und die Begriffe, die für eine objektive Annotation einer Audiospur benutzt werden können. Weitere Korpora können von der Webseite[106] heruntergeladen und mittels eines Korpus Managers erweitert werden. Um die Kompatibilität zu anderen Transkriptionssoftware zu gewährleisten, kann die in EXMARaLDA festgelegte Transkription in den Formaten EXB, EXS, EAF, FOLKER, PRAAT, TEI exportiert werden. Mehrere Forschungsprojekte benutzen diese Software für die Transkription von Interviewvideos oder Sendungen. Eins davon ist das Projekt LinkedTV[107], das später in diesem Buch präsentiert wird (siehe Kapitel 5). Die Abbildung 3.50 zeigt die Benutzerschnittstelle von EXMARaLDA und die Partitur-basierte Visualisierung der Annotationen.

[104] `http://www.exmaralda.org`
[105] `http://www1.ids-mannheim.de/fileadmin/prag/Programmbereich_Muendliche_Korpora/EXMARaLDA.pdf`
[106] `https://corpora.uni-hamburg.de`
[107] `http://www.linkedtv.eu`

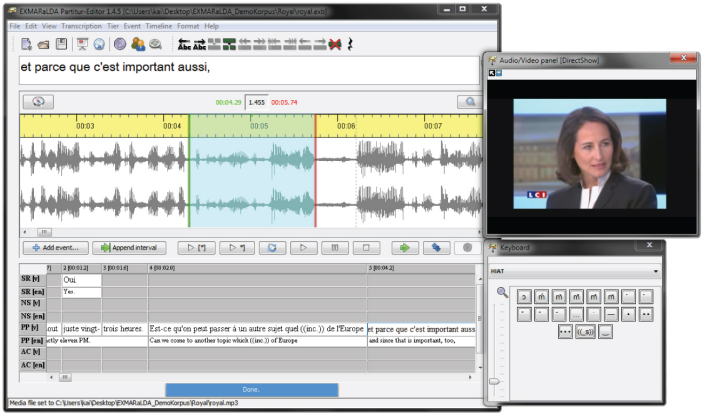


Abbildung 3.50: Annotation der französischen TV-Debatte zwischen Nicolas Sarkozy und Ségolène Royal mit EXMARaLDA. Quelle: EXMARaLDA Webseite.

VideoClix

VideoClix[108,109] war bis 2014 eine kostenpflichtige online-basierte Lösung, um sogenannte *Clickable Videos* für interaktive videogestützte Online-Werbung zu generieren. Sie war für Broadcaster und Online - Videoanbieter als hauptsächliche Kunden gedacht. Die Softwarelösung wurde 2015 von der Firma Brightcove[110], einer Onlinevideoplattform, akquiriert. Der Ansatz von VideoClix ist im Kontext der Realisierung von Swoozy dahingehend relevant, als dass dieser Dienst einen online-basierten Editor und ein Annotationswerkzeug für Online-Videos anbot. Nachdem der Benutzer ein Video auf die Videoclix-Plattform hochgeladen hatte, analysierte das System das Video und versuchte die interessantesten Merkmale dieses durch ein autonomes Trackingver-

[108] http://www.videoclix.tv
[109] https://www.youtube.com/watch?v=3VnUBXf8-Zg
[110] https://www.brightcove.com

fahren namens *VideoClix Smartrack* zu extrahieren. Diese Videoanalyse erfolgte auf den Servern von Videoclix und als Ergebnis wurden dem Benutzer Zonen vorgeschlagen, an die Zusatzinformation sinnvollerweise angehängt werden konnten. Diese Informationen konnten entweder Produktinformationen oder Weblinks sein. Die Zusatzinhalte wurden über die Server von Videoclix eingepflegt. VideoClix bot einen Onlineeditor zur Bestimmung und Festlegung der Zonen und der zu benutzenden Begriffe an, die mit Werbung verknüpft werden sollten (siehe Abbildung 3.51).

Abbildung 3.51: Web-basierte Benutzeroberfläche von VideoClix Quelle: Aufruf von Videoclix.com über Internet Archive

Die Zeitpunkte, wann ein Begriff eingeblendet werden soll, wurden in Form einer Liste dargestellt, die vom Benutzer nacheditiert werden konnte. Die Festlegung der Zonen und der Begriffe kann man gewissermaßen als *Annotation* bezeichnen, wobei keine Beschreibung der Szene erfolgte, sondern nur eine Produkt- und Frame-bezogene

Annotation. Hierbei wurden dem Benutzer sich auf die Werbekampagne beziehende Begriffe und Medien als Ergänzung vorgeschlagen. Nachdem der Benutzer ein Video annotiert hatte, konnte dieses Video mit dem dedizierten Flash-basierten VideoClix-Player abgespielt und in eine Webseite integriert werden.

WIREWAX

Ein weiterer interaktiver Dienst zur Annotation von Onlinevideos ist WIREWAX[111]. Um die Annotationen zu realisieren, wird den Benutzern die Möglichkeit gegeben, ein Video auf die Server von WIREWAX hochzuladen. Direkt danach beginnt die Analyse des Videos, indem Merkmale aus dem Video extrahiert werden. Mit Merkmalen sind hier Kamerabewegungen, Personen, Texte, Logos und transkribierte Sprache gemeint. Zusätzlich werden die Bewegungen der einzelnen Gegenstände oder Personen aus dem Video extrahiert und als *Suggested Face* (vom System automatisch vorgeschlagene Begriffe zu erkannten Personen) oder *Object* während des Editiervorgangs eingeblendet (siehe Abbildung 3.53). Mit dem Online-Editor können Schüsselwörter (Tags) hinzugefügt und mit Hyperlinks versehen werden. Der Editor bietet eine erste Klassifikation, da während des Editiervorgangs Vorschläge, z.B. zu Objekten oder Personen, automatisch eingeblendet werden. Dank einer Motion-Tracking-Unterstützung können einmal getaggte Personen in den später folgenden Einzelbildern wiedererkannt werden. WIREWAX positioniert sich mit seinem Produkt eher in den Bereichen Online-Marketing und Werbung und eignet sich primär für die direkte Verlinkung von online käuflichen Produkten, die in kurzen Clips auftauchen.

[111] `http://www.wirewax.com`

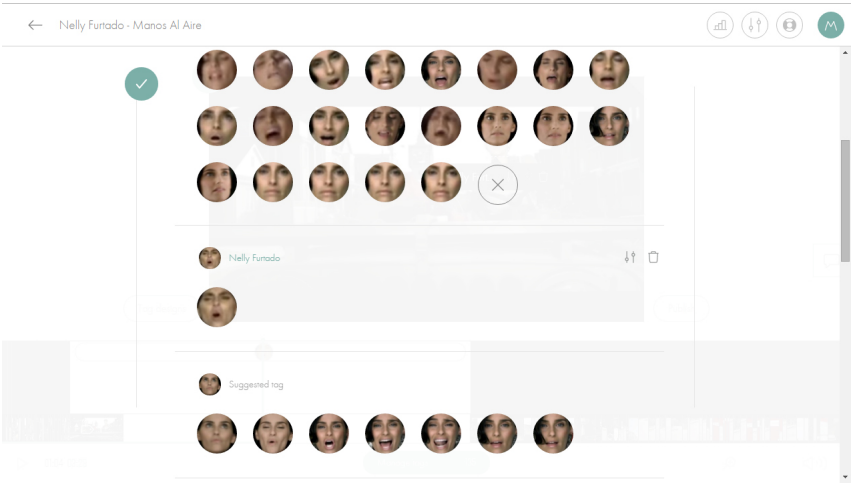


Abbildung 3.52: Web-basierte Benutzeroberfläche von WIREWAX.

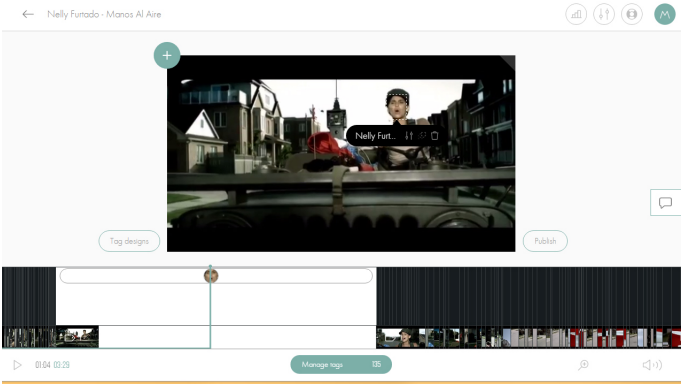


Abbildung 3.53: Beispiel von *Suggested Faces* in WIREWAX

Overlay.TV

Overlay.TV[112] ist ein Onlinedienst, der es ermöglicht, interaktiv in einem Video, Links zu Produkten hinzuzufügen. Registrierte Anwender müssen zuerst den genauen Zeitpunkt, bei dem das Produkt in einem Video eingeblendet werden soll, selektieren. Wenn diese Selektion durchgeführt wurde, kann das Produktbild innerhalb des Videos platziert werden.

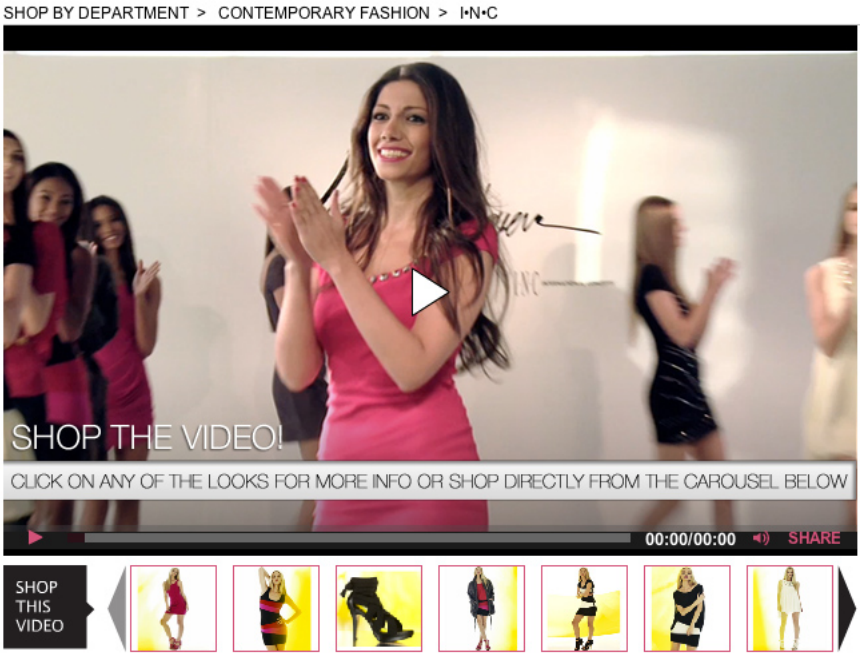


Abbildung 3.54: Integration der InStyle Boutique mit Video einer Modeshow. Das Video wird im dedizierten Overlay.TV Player wiedergegeben. Quelle: Webseite von http://www.overlay.tv

Als Ergebnis wird, parallel zu einem laufenden Video, ein Karussell eingeblendet, der die gerade im Bild präsentierten Objekte auflistet.

[112] http://www.overlay.tv

Das Video kann mittels eines dedizierten Players auf einer Webseite integriert und abgespielt werden (siehe Abbildung 3.54).
Der Konkurrent-Dienst Grabit Systems[113] bietet eine ähnliche Lösung an.

Adways Studio

Nach dem Hochladen eines Videos bietet der Dienst der Firma Adways-Studio[114] die Möglichkeit, bestehende Online-Videos, z.B. aus YouTube oder Dailymotion, mit Tags und interaktiven Zonen anzureichern. Der Fokus bei diesem Dienst ist das Videoerlebnis von Webseitenbesuchern zu erhöhen und Kontextinformationen anzubieten, ohne dass der Besucher die Webseite verlässt. Eine online basierte Objekt- und Personenerkennung sorgt dafür, dass die interaktiven Zonen das selektierte Objekt mitverfolgen. Zusätzlich zu den rechteckigen Bereichen können die Overlays interaktiv gestaltet werden, indem z.B. Quiz, Umfragen oder Facebook-Einträge integriert und über das Video eingeblendet werden.
Abbildung 3.55 zeigt die Integration und Informationsintegration innerhalb eines interaktiven Videos für die Sportmarke Sportway. Hierbei werden Zusatzinformationen über den Fallschirmsprung, das gerade im Video zu sehen ist, nach Klick auf einen Hotspot (Bereich, der anklickbar ist), gegeben.

[113] `http://grabitsystems.net`
[114] `http://adways-studio.com`

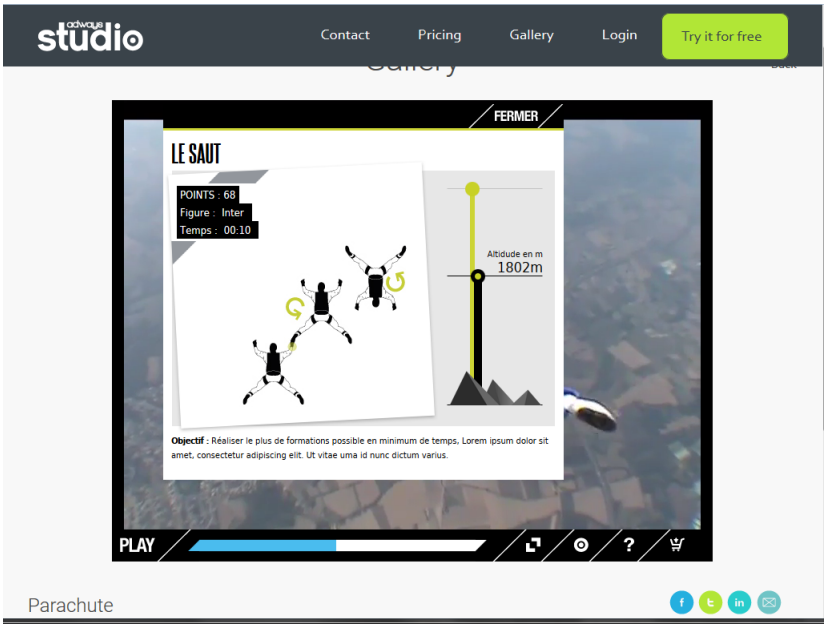


Abbildung 3.55: Beispiel eines interaktiven Videos von Adways. Quelle: Adways Studio

Folgende Abbildung 3.56 zeigt den Editor vom Adways. Hotspots und Popups können frei vom Benutzer angelegt werden. Eine Möglichkeit, Schriftarten oder Farben der Annotationen zu verändern, wird auch gegeben.

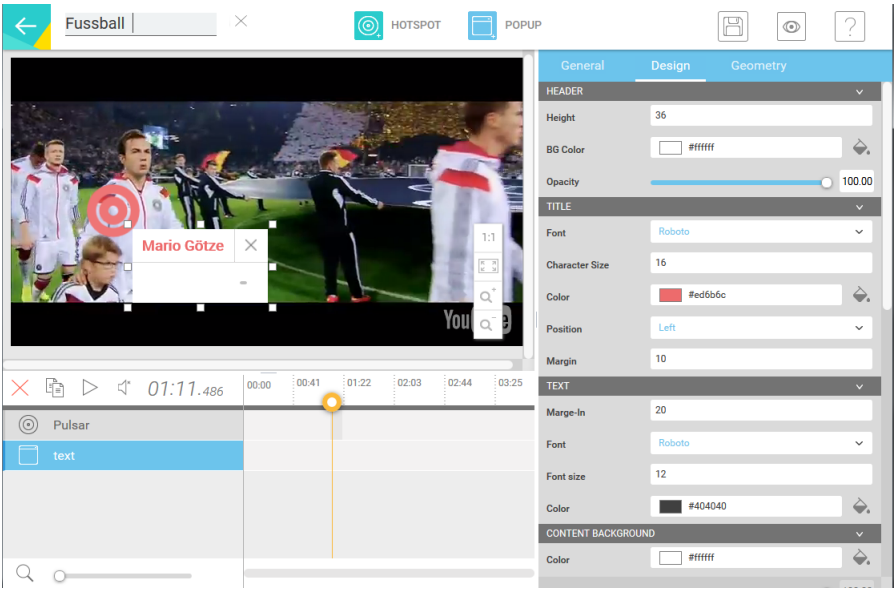


Abbildung 3.56: Annotation eines YouTube Videos mit Adways. Quelle: online Editor von Adways Studio

LinktoTV

Der LinkToTV[115]-Dienst ermöglicht das manuelle Platzieren von sog. rechteckigen Zonen (engl. Hotspots) über Videos der Online-Plattformen ooyala, Brightcove oder YouTube. Diese Hotspots werden mittels eines Online-Editors und einer Timeline gesetzt. Besonderes Merkmal bei LinktoTV ist die Möglichkeit, sog. Keypoints zu definieren. Diese bestimmen den zeitlichen Start- und Endpunkt und die Position eines Hotspots innerhalb des Videos. Die Zwischenschritte werden automatisch berechnet. Die kontinuierliche Anpassung der grafischen Position, z.B. einer Tasche innerhalb eines Modenschau-Videos, wird durch die Verwendung der Keypoints möglich und gibt somit den Eindruck, die

[115] http://www.linkto.tv

Hotspots würden automatisch die Tasche verfolgen. Zusätzlich werden den Kunden von LinktoTV verschiedene Statistiken wie z.B. die Anzahl der geklickten Hotspots oder die verbrachte Zeit über einen Hotspot über die Webseite angeboten.

YouTube Annotations

Mehrere der vorgestellten Dienste beruhen auf einer grafischen Überlagerung (engl. Overlay) von YouTube-Videos. YouTube selbst bietet eine manuelle Annotationsmöglichkeit (im YouTube-Jargon, werden Annotationen ins Deutsche als *Anmerkungen* übersetzt) in Form eines Online-Werkzeugs an. Mit den YouTube-Anmerkungen können fünf Annotationstypen ausgewählt werden: eine Sprechblase, ein Spotlight (Bereich, der bei einer MouseOver Interaktion hervorgehoben wird), ein Popup, ein Titel (Titeleinblendung über das Video) und ein Label (zeitliche Markierung eines Bereichs)[116,117]. Zusätzlich können Links zu anderen Videos oder sog. Infocards hinzugefügt werden. HTML-basierte Quiz, Umfrageformulare oder Werbungen können ebenso mittels Infocards in einem Video integrieren werden. Eine Analyse des eigentlichen Videoinhaltes wird, im Gegensatz zu WireWax, in YouTube (noch) nicht durchgeführt. Es wird jedoch YouTube-Videoproduzenten die Option überlassen, eine automatische Untertitelung über ein YouTube-internes Sprachanalysemodul durchzuführen. Als Ergebnis werden die erkannten Sätze als textuelle Einblendung über das abgespielte Video eingeblendet. Zusätzlich können Untertitelungsdateien von sog. genehmigten Anbietern importiert werden. Enthält ein Video eine Untertitelung, wird diese dem Benutzer mit einem Icon (CC-Icon) grafisch innerhalb des YouTube-Videoplayers signalisiert.

[116] `https://support.google.com/youtube/answer/92710?hl=de`
[117] `https://support.google.com/youtube/answer/2734796?hl=de`

LookAt

Der LookAt-Dienst[118] kann nicht direkt als Annotationswerkzeug für das breite Publikum oder soziale Medien bezeichnet werden, da es zunächst keine Webseitenintegration anbietet. LookAt zielt eher auf eine kollaborative Online-Plattform für Videoclip- und Filmemacher. Mit diesem Dienst können sie, in Echtzeit und gleichzeitig mit Kollegen, Annotation, Grafiken und kontextuelle Kommentare (engl. Contextual Commenting) zu einem Video hinzufügen. Zusätzlich können über 100 unterschiedliche Dateiformate hinzugefügt und miteinander ausgetauscht werden. Als einziger Dienst bietet LookAt über ein dediziertes Werkzeug (Adobe LookAt Extension)[119], eine sehr gute Integration mit den Software Adobe After Effects[120] (Software für Compositing und Post-Produktion von Filmmaterial) und Adobe Premiere, an. Annotationen, die über die Online-Plattform angelegt wurden, lassen sich nahtlos mit beiden Adobe Software importieren bzw. exportieren. Die importierten Annotationen erscheinen dann als Marker (Markierung) direkt in der Timeline der Adobe Software. Die Abbildung 3.57 zeigt ein Beispiel einer Annotation. Die rechte Leiste sammelt alle Kommentare, die innerhalb des Videos durchgeführt wurden, und listet diese nach Timecode (dt. Zeitstempel) zusammen. Einzelne Bereiche können manuell grafisch markiert werden (siehe blaue Linie in Abbildung 3.57) und von allen Teilnehmern einer Gruppe gesichtet werden.

[118] `http://www.lookat.io`
[119] `http://www.lookat.io/adobe-integration`
[120] `https://www.adobe.com/de/products/aftereffects.html`

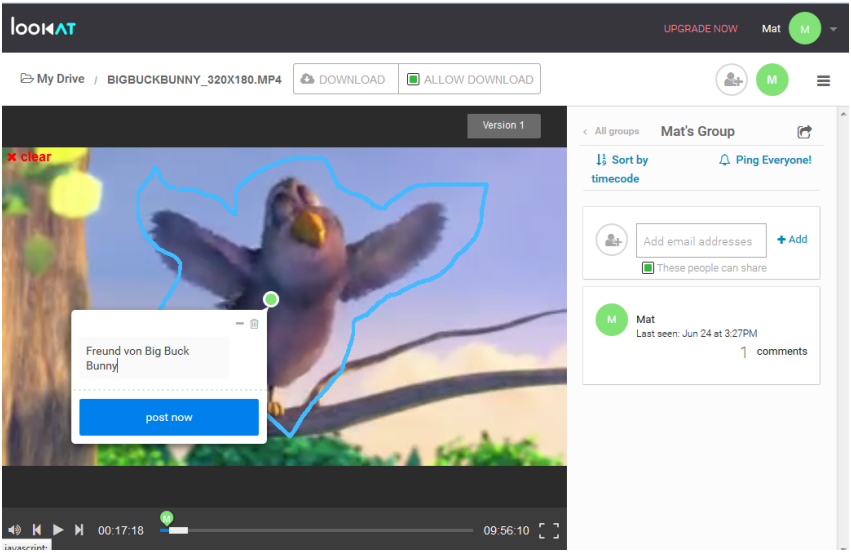


Abbildung 3.57: Kollaborative Annotation eines Videos mit LookAt

Adobe Premiere

Die professionelle Software Adobe Premiere[121] ermöglicht eine nachträgliche Analyse der Tonspur eines Videos und die Erkennung von Gesichtern. Die Erkennung extrahiert über eine *Spracherkennungs*-Komponente alle Wörter, die gesprochen worden sind, inklusive der Sprecherquelle. Zusätzlich wird das gleiche Material von einer Gesichtserkennungskomponente analysiert, die nach allen Gesichtern innerhalb eines Videos sucht.

Die Qualität der Erkennung wird über ein Einstellungsmenü bestimmt und von der dabei ausgewählten Analysegeschwindigkeit bestimmt. Um die Erkennungsrate der *Spracherkennung* zu steigern, können Adobe Story-Dateien (im SRT-Format) hinzugefügt werden. Diese Datei-

[121] `http://www.adobe.com/de/products/premiere.html`

typen beinhalten eine textbasierte, voll ausformulierte Version eines gesprochenen Dialogs oder der Narration eines Films. Sie werden z.B. von einem Regisseur in Form eines Skriptes für ein Storyboard (Drehbuch) angelegt und gespeichert.

Abbildung 3.58: Ergebnis einer Audioanalyse in Adobe Premiere

Nach der Analyse kann der Text innerhalb von Adobe Premiere weitereditiert und neue Metadaten hinzugefügt werden. Mittels des Timecodes wird die genaue Position eines Wortes innerhalb des Videos angezeigt. Als Endergebnis dieser Analyse erhält das Video eine vollständige XMP-basierte Beschreibung seines Inhaltes (siehe Kapitel 2), inklusive der räumlichen Position der erkannten Gesichter sowie den kompletten analysierten Text. Dieser und die Timecodes der einzel-

nen gesprochenen Wörter können in Form von Referenzzeitpunkten (engl. Cue Points) für die Verwendung mit anderer Adobe Software wie z.B. Adobe Flash[122] (Werkzeug zur Erstellung von Web-kompatiblen Animationen) oder After Effects exportiert werden.

Fazit: Werkzeuge zur semantischen Annotation

Die automatische Extraktion aus Bild- und Videomaterial kann aus verschiedenen Gründen nicht immer hundertprozentig korrekte Annotationen liefern. Die Qualität der Erkennung hängt sehr stark davon ab, inwieweit die Referenz- und Trainingsdaten mit denen der Erkenner initial arbeitet, vollständig oder für eine bestimmte Domäne verfügbar waren. Diese Referenzdaten bzw. Beschreibung eines Videoinhaltes können semi-automatisch von einem Redakteur oder einem „Annotierer" über eine Annotationssoftware realisiert werden. Diese Werkzeuge unterstützen das manuelle Annotieren von Konzepten innerhalb eines Mediums (Bild oder Video). Über grafische Werkzeuge kann der Annotierer (auch Redakteur) z.B. Zonen (mit X-, Y-Position, Höhen- und Breitenangabe) und Pixelbereiche definieren, in denen wichtige Gegenstände (Objekte, Häuser, usw.) oder Personen (z.B. Schauspieler oder Moderatoren) zu finden sind. Diese werden entweder mit einer vollständigen textuellen Beschreibung oder über semantische Verknüpfungen mit dem passenden DBpedia-Beitrag versehen. Das Prinzip von fast allen dargestellten Werkzeugen (z.B. ExMaRalda oder ANVIL) ruht auf einer Timeline-gestützten Editierfunktion. Damit lässt sich nicht nur Frame-genau die Position eines Videos einstellen, sondern darüber hinaus mehrere Annotationen oder Tags hinzufügen. Diese können entweder frei ausgewählt oder mit vor-

[122] https://www.adobe.com/de/products/flash.html

her angelegten semantischen Konzepten (z.B. aus einer Ontologie) verknüpft werden. Somit können komplexere Taxonomien und Hierarchien mitgespeichert werden. Werkzeuge wie z.B. Advene lassen zusätzlich eine Verknüpfung zu externen Links (Wikipedia, YouTube) und Medien zu. Diese Werkzeuge werden primär für wissenschaftliche Untersuchungen benutzt und fokussieren sich primär auf die Verwertung und Verknüpfung von Videoinhalten mit Ontologien und Konzepten.

Die größte Einschränkung bei allen präsentierten Annotationswerkzeugen sind die nicht standardisierten Ausgabeformate, die keine Interoperabilität ermöglichen. Das bedeutet, wenn ein Video mit einem Tool durchannotiert wurde, kann es meistens nur mit diesem benutzt und abgespielt werden. Die Annotationen können zwar exportiert, aber nur von der Software benutzt werden, die diese erstellt hat. Meistens stellen die Werkzeuge einen Videoplayer bereit, der in der Lage ist, diese Annotationen zu laden und zum korrekten Zeitpunkt parallel zum Videoabspielvorgang anzuzeigen. Bei Caliph wie bei der eingestellten Softwarekomponente VideoAnnEx von IBM, den beiden einzigen Annotationswerkzeugen, welche die MPEG-7-Spezifikation als Grundlage für den Export der Annotationen benutzen, werden zwar die Annotationen in einem standardisierten Format ausgegeben; es fehlen zum heutigen Zeitpunkt immer noch die Werkzeuge im Produktionsbereich, die dieses Format unterstützen [TTR03]. Diese Situation wird durch die MPEG-7-Formatkomplexität erschwert, insbesondere weil keines der aktuellen in Videoproduktionssystemen eingesetzten Tools, wie Adobe Premiere oder Final Cut Pro, eine Unterstützung für dieses Format anbieten. Bei Adobe Premiere können die Annotationen im XMP-Format exportiert werden und sind für andere Adobe-

oder XMP-kompatible Software wieder benutzbar. Bei der Betrachtung und Analyse der existierenden Annotationswerkzeuge wurde festgestellt, dass Dienste wie WIREWAX oder LookAt erste Cloud-basierte Ansätze zur Annotation von Onlinevideos bieten und zwar durch Verwendung des gleichen Timeline-basierten Prinzips. Bei WIREWAX sind die Annotationsmöglichkeiten ein wenig verfeinert worden. Nach dem Hochladen eines Videos und der Analyse werden die ersten Keyframes und Zonen extrahiert, in denen sich z.B. Gesichter oder Objekte befinden. Diese können später wieder manuell mit einem Tag oder einer textuellen Beschreibung ergänzt werden. Die dabei resultierenden Annotationen können weder exportiert noch importiert werden. Das Abspielen gelingt nur über spezielle Player, die entweder mit der Adobe Flash-Technologie implementiert worden sind oder mittels der HTML5-Videokomponente, welche die gespeicherten Annotationen mit dem abgespielten Video zeitlich verknüpfen. Eine weitere Verwendung der Annotationen in anderen Werkzeugen oder Cloud-basierten Diensten ist nicht möglich. Lediglich LookAt bietet eine Möglichkeit, Annotationen mit Adobe Premiere bzw. After Effects zu benutzen und zu exportieren.

Fazit

Über eine Zeitleiste (engl. Timeline) bzw. Partitur-basierte Oberfläche geben die Werkzeuge einem Redakteur oder Annotierer die Möglichkeit, auf sehr präzise Art und Weise Videomaterial zu annotieren. Die Annotation kann entweder frei gesetzt werden, z.B. in Form einer textuellen Beschreibung oder mit Konzepten, die von einer Ontologie oder Taxonomie übernommen worden sind. Lösungen wie WIREWAX und Adobe Premiere ermöglichen dank einer Bild- und Sprachvoranalyse

Zeitstempel und Positionen von Personen und Gesichtern innerhalb eines Videos automatisch zu extrahieren. Bei SVAS können eine OpenCV-basierte Videoanalyse zur Merkmalsextraktion im Vorfeld benutzt werden, um bestimmte Bereiche im Voraus zu markieren und den Annotationsvorgang zu vereinfachen.

Die Videoannotationen werden in nicht standardisierten, teils proprietären Formaten gespeichert, was die Interoperabilität und die Wiederverwendbarkeit erschwert. Auch onlinegestützte Annotationswerkzeuge bieten keine weitere Verwendungs- oder Exportmöglichkeit (außer bei LookAt). Meistens bieten die Annotationswerkzeuge einen speziellen Videoplayer, der nachträglich die Annotationen zum richtigen Zeitpunkt einblendet. Die Videoplayer funktionieren nur mit den Annotationen der jeweiligen Werkzeuge, die sie erstellt haben. Die Heterogenität der Ausgabeformate und die nicht standardisierten Annotationen bilden zurzeit die größten Hürden bei der Verwendung der vorgestellten Annotationswerkzeuge innerhalb eines klassischen Videoproduktionsworkflows.

Bemühungen seitens des W3C und der HTML-Community lassen hoffen, dass z.B. Formate wie TTML[123] oder WebVTT[124] in der Lage sein werden, neben den Untertiteln noch weitere, zeitgesteuerte, semantische Beschreibungen in HTML5-basierten Videoplayern zu unterstützen.

Die Tabelle [125] (siehe Abbildung 3.59) fasst die vorgestellten Werkzeuge zusammen, die zur Unterstützung des Annotationsprozesses eines Videos beitragen können. Leider erweist sich die Handhabung von

[123] `http://www.w3.org/TR/ttaf1-dfxp/`

[124] `http://dev.w3.org/html5/webvtt/`

[125] Der Leser kann sich eine hochauflösende Version folgender Tabelle samt zwölf weiteren Kriterien unter der Webseite des Autors `http://www.mat-d.com/books/swoozy/annotation_tools_comparison.pdf` herunterladen.

einem dieser Werkzeuge als zu kompliziert, da es vom Benutzer gute Kenntnisse im Bereich der Wissensmodellierung voraussetzt.
Drei Werkzeuge haben das MPEG-7-Format als Ausgabeformat benutzt. Das MPEG-7-Format ist zwar semantisch gut aufbereitet und standardisiert, trotzdem bildet es ein sehr komplexes Konstrukt, das durch seine geringe Flexibilität für einen Endbenutzer im journalistischen oder redaktionellen Bereich kaum zu gebrauchen ist. Nur Adobe Premiere benutzt das Format XMP als Ausgabestruktur. Zusätzlich stellt die Dateninteroperabilität bei allen Systemen ein Problem dar, denn, obwohl diese Werkzeuge Exportmöglichkeiten anbieten, sind die exportierten Dateiformate untereinander nicht kompatibel und bzgl. der generierten semantischen Beschreibung sehr unterschiedlich. Teils sind spezielle dedizierte Videoplayer nötig, damit die Annotationen abgespielt und eingeblendet werden können. Zugleich können diese nur von den Werkzeugen, welche sie produziert haben, benutzt und editiert werden. Dies stellt insbesondere eine weitere Hürde dar, wenn eine Integration in einen Produktionsworkflow erfolgen soll.
Fast alle Werkzeuge ermöglichen in einer mehr oder weniger komplexen Art und Weise das grafische Anlegen von Bildzonen/-bereichen. Diese Zonen können nachträglich über Tags erweitert werden. Ebenso wird das Anlegen von Freitexten und Zusatzbeschreibungen bei allen unterstützt. Bei der Lösung von WIREWAX werden automatisch Gesichter aus den einzelnen Bildern eines Videos erkannt und mitverfolgt. Dies wird über ein Cloud-basiertes Videoanalyseverfahren ermöglicht. Obwohl diese Vorgehensweise interessant ist, scheint die WIREWAX Motion-basierte Erkennung noch nicht ganz ausgereift zu sein, da eine manuelle Nachbearbeitung bzw. Platzierung, insbesondere bei schnell bewegten Bildern, trotzdem notwendig ist.

			Editier- und Annotierungsfunktionalitäten				
			Timeline-basiertes Editieren	**Einbindung von Ontologien**	**Editor**	**Annotation**	**Verbindung mit Web 2.0/3.0**
Annotationswerkzeuge	Offline	Bild	**Caliph**	✗ nicht zutreffend	✓	Drag'n'drop von Konzepten Zonenauswahl	✓	✗
		Video	**Adobe Premiere**	✓ indirekt	✗	✓	✓	✗
			Advene	✓	✗	✓	✓	✓ Ja über Hyperlinks
			ANVIL	✓	✗	✓	✓ Speech Transkription	✗
			SIVA	✓	✗	Szenengraph Semantic Zoom Levels	✓ semi/auto. Videoerkennung	✓
			SVAS	✓ Keyframe-basiert	✓	manuelles Anlegen von Region of Interest	Voranalyse des Video / SIFT Merkmale	✗
			VCode & VData	✓	✗	✓	Auswahl von vordefnierten Eventtypen	✗ nicht zutreffend
			VIA Tool	✓	✓ OWL	Frame by Frame und Still Zonenannotation	Vorschläge von Konzepten über Ontologie	✗ nicht zutreffend
			IBM VideoAnnEx	✓	✓	Shot und Regionen-Annotationen	Vorschläge von Konzepten über Ontologie	✗
			Xmeralda	✓	✗	Partitur-basiert	Unterstützung durch Korpora	✗
	Online		**Videoclix**[1]	✓	✗	Regionen-Annotationen	Vorschläge von Tags	✓ Ja über Hyperlinks
			Wirewax	✓	✗	Semi-automatische Annotationen	Vorschläge von Tags	✓ Ja über Hyperlinks
			LookAt	✓	✗	manuelle Annotationen	✗	✗
			LinktoTV	✓	✗	manuelle Regionen-Annotationen	✗	✓ Ja über Hyperlinks
			OverlayTV	✓	✗	manuelle Regionen-Annotationen	✗	✓ Ja über Hyperlinks
			YouTube Annotations	✓	✗	manuelle Regionen-Annotationen	✗	✓ Ja über Hyperlinks
			Adways Studio	✓	✗	manuelle Regionen-Annotationen	✗	✓ Ja über Hyperlinks

1 Seit 2015 nicht mehr verfügbar

Abbildung 3.59: Übersichtstabelle: Werkzeuge zur Video- und Bildannotation

Die vorgestellten Annotationssoftwares bilden durch ihre einfache Bedienung und Interaktionskonzepte (z.B. die parallel editierbare Spuren oder die interaktive Bildzonenmarkierungen) einen Ansatz für das Design und Konzipierung einer neuen Annotations- und Unterstützungskomponente namens *Swoozy-SKRPTR* (siehe Kapitel 7).

Zusammenfassung

Der folgende Abschnitt gibt einen kompakten Überblick über die gerade vorgestellten Technologien und präsentiert mögliche Anwendbarkeit weiterer Ansätze, die dazu beitragen könnten, die Extraktion von Semantik aus den unterschiedlichen Medienobjekten zu unterstützen.

Textuelle semantische Extraktion

Ansätze wie die Entitäten-Extraktion ermöglichen eine präzise semantische Analyse eines Textes. Als Ergebnis können die extrahierten Informationen mit den aus dem ausgestrahlten DVB-Stream enthaltenen Daten wie EPG oder HbbTV-Inhalten (oder besser gesagt der HbbTV-Webseite des Fernsehsenders) kombiniert werden, um die Domäne oder bestimmte Suchthematiken des laufenden Programmes besser einschränken zu können. Cloud-basierte Dienste wie Alchemy, AYLIEN, Microsoft Cognitive Services oder NEMEX liefern eine qualitativ schnelle und gute Analyse der übertragenen Textblöcke. Diese Analyse erfolgt über eine dedizierte REST-API, die entweder eine JSON- oder XML-Struktur als Ergebnis zurückliefert. Die Ergebnisse können mittels einer weiteren Verbindung über DBpedia oder Wikidata als Grundlage für eine erweiterte semantische Suche dienen. Bei den Lösungen von AYLIEN oder Alchemy werden Links zu verwandten Wikipedia-Seiten,

Videos oder Bildern und eine kurze Zusammenfassung entweder in textueller Form oder in Hashtag-Form zurückgeliefert. Somit können neue Zugriffsmöglichkeiten zu Medien angeboten werden.

Bildextraktion und Erkennung

Bei diesem Verfahren wurde gezeigt, dass viele Systeme in der Lage sind, aus einem Bild bestimmte Personen oder Gegenstände zu extrahieren und zu erkennen. Meistens stützen sich die Analysemodule auf die Bibliothek OpenCV. Diese integrierten Verfahren stammen aus dem Bereich des visuellen maschinellen Lernens und ermöglichen die Integration von Gesichts- und Objekterkennung in einem bewegten Bild. Im Falle der Bildextraktion können Cloud-basierte Dienste und Lösungen benutzt werden. Lösungen wie CloudCV, Tineye.com, Google Cloud Vision oder Microsoft Cognitives Services ermöglichen eine komplette Merkmalsextraktion aus Bildern und die partielle Erkennung von vorannotierten Objekten innerhalb eines Bildes über REST-APIs. Andere Lösungen sind auf Gesichtserkennung und die Extraktion der Gesichtsmerkmale spezialisiert und bieten eine Methode zum Gesichtervergleich und zur Klassifikation des Geschlechts, wie dies z.B. bei AlchemyVision oder SkyBiometry der Fall ist. Als Ergebnis der Erkennung werden der Name der Person, ihr Geschlecht, aber auch die Taxonomie innerhalb eines *Knowledge Graphs* angezeigt (z.B. /people/celebrities/nelly furtado), sowie *Linked Data* Referenzen und die dazugehörigen Konzepte (Person, Sänger) zurückgeliefert. Obwohl diese Lösungen eine gute Erkennungsrate vorweisen, kann nicht gewährleistet werden, dass die Erkennung im Livebetrieb schnell, einwandfrei und ohne Verzögerung funktioniert. Schnell bewegte Bilder oder ständige Veränderungen in Filmszenen können von diesen Werk-

zeugen noch nicht analysiert werden, da die Internetverbindungen und die Analyse auf dem Server längere Zeit (min. 5-6 Sekunden) in Anspruch nehmen. Diese technische Einschränkung bedeutet nicht, dass diese Dienste zukünftig für den Einsatz in Kombination mit Fernsehinhalten nicht geeignet sind. Ganz im Gegenteil: bei Sendungen wie z.B. Nachrichten oder Talkshows mit Prominenten oder bekannten Persönlichkeiten können diese als Überprüfungskomponente dienen, falls die lokal angesteuerte Videoanalyse keine eindeutigen Ergebnisse oder schlechtere Erkennungskonfidenzwerte bereitgestellt hat. So könnten diese Dienste als Zusatzkomponente und Unterstützung für die semantische Analyse und Extraktion benutzt werden.

Eine weitere Möglichkeit, den Inhalt eines Fernsehbildes zu erkennen und die Thematik des ausgestrahlten Programms besser zu erfassen, stellen OCR-Verfahren dar. Bei diesen Verfahren werden schriftliche Einblendungen erfasst, erkannt, analysiert und in maschinenverstehbare Begriffe transformiert und wieder zusammengestellt. Somit ist es möglich, eine generische Bilderkennungskomponente mit Zusatzinformationen zu füttern, um z.B. eine Domäne einzuschränken. Im Falle von Fernsehbildern bietet OCR eine gute Möglichkeit, parallel zum EPG, Zusatzinformationen aus dem aktuell ausgestrahlten Programm zu extrahieren. Dies wurde exemplarisch in [DC07] bewiesen, indem das Bildmaterial mittels OCR-Verfahren auf Nachrichtensendungen wie denen der ARD Tagesschau eingesetzt wurde. Zusätzlich wurde technisch belegt, dass es durchaus möglich ist, aus den innerhalb der Sendung angezeigten Texten, semantische und kontextbasierte Relationen zu extrahieren. Dank des eingeblendeten Themabildes im Hintergrund der Moderatorin und der geografischen Angaben im unteren Teil der Einblendung kann eine OCR-Erkennung durchgeführt werden.

Neue Relationen und semantische Beschreibungen können so entdeckt und aus dem Video bzw. Videobereich extrahiert werden. In eine ähnliche Richtung gehen die Arbeiten von [IDF+05], die versuchen, einzelne Bildregionen mit Konzepten zu versehen und diese zu klassifizieren.

Verfahren der Bildanalyse können eingesetzt werden, um zu erkennen, welche Personen und Gegenstände im Bild zu sehen sind. Um dies realisieren zu können, werden Bibliotheken wie OpenCV, die mit Gesichtserkennung und der Extraktion der Gesichtsmerkmale kombiniert werden können, verwendet. Diese Analyse beruht auf Trainingsdaten, die schon vorliegen und während des Erkennungsprozesses verwendet werden. Entsprechend umfangreich müssen diese für Sendungen oder Beiträge sein. Aktuell können Cloud-basierte Dienste für den Einsatz innerhalb einer Echtzeitanalyse nur sporadisch benutzt werden, da die Latenzzeiten der Server noch keine schnelle mit dem Live-Fernsehbild synchronisierte Extraktion ermöglichen

Videobasierte Extraktion

Ähnlich der Bildextraktion und Analyse können über spezifische Softwarebibliotheken bzw. APIs Videos analysiert und deren Inhalte extrahiert werden. Diese Analyse erfolgt meistens in zwei Schritten. Im ersten Schritt wird der eigentliche Inhalt des Videos extrahiert. Über Klassifikations- und Clustering-Verfahren wie etwa *Bag-of-Words* und *Deep Learning* lassen sich über vortrainierte Videosequenzen bestimmte Begriffe und Konzepte wiedererkennen. Dies bezeichnet man als die Lernphase des Erkenners. Bei diesem Schritt werden Merkmale wie Umrisserkennung, Farbhistogramme, Pixelpositionen mit den

jeweiligen Konzept-Clustern verbunden. Ähnlich funktioniert die Personenerkennung, bei der Merkmale nachträglich mit biometrischen Merkmalen (Position und Neigung des Kopfes, Position des Mundes, Position der Augen) eines vorher angelegten Datensets verglichen werden. Algorithmen wie Haar-Cascade helfen z.B. Gesichter, ihre räumliche Position im Bild und deren geometrische Merkmale zu extrahieren. Im zweiten Schritt erfolgt die Semantifizierung des Videomaterials. Dabei werden die erkannten Merkmale des Bildes mit Konzepten oder Namen verknüpft. Dieser Schritt liefert nur dann brauchbare Ergebnisse, wenn im Vorfeld gründlich angelegte und vorannotierte Videos als Trainingsdaten für den Erkenner vorliegen. Das bedeutet, dass bevor eine tatsächliche Erkennung z.B. von Personen erfolgen kann, schon Daten in Form von Video- oder Bildmaterial vorhanden und im besten Falle mit einer existierenden Annotation versehen werden müssen. OpenCV gehört zu der am häufigsten angewandten Lösung zur Merkmalextraktion, zum Vergleich und zur Analyse von Videos und ermöglicht Entwicklern über C++- und Open Source-basierten Bibliotheken kostengünstig und effizient Algorithmen des Bereichs *Computer Vision* einzusetzen. Darunter zählen u.a. Klassifikatoren wie Haar-Cascade, SIFT oder SURF oder Deep Learning-Verfahren wie die Benutzung von CNNs. Diese Klassifikatoren und Verfahren sind Bestandteile der OpenCV-Bibliothek und können zur Merkmal-und Videoanalyse per Code dank Benutzung der OpenCV APIs eingesetzt werden. Bei der Erkennung von Bild- und Videomaterial im Kontext vom Fernsehen müssen immer zwei getrennte Szenarien betrachtet werden:

- Wenn die Inhalte eines Videos zum Zeitpunkt der Analyse bekannt sind (Film oder Wiederholung einer Reportage), kann die

Erfassung schneller durchgeführt werden. Bestehen schon Annotationen oder gehören die Trainingsdaten zum selben Video, kann eine direkte Semantifizierung der Daten erfolgen.

- Im Livemodus können die Erfassung und die Analyse länger dauern, da die zu analysierenden Videoinhalte noch nicht bekannt sind und über keine vorher bekannten Annotationen verfügen. Es kann durchaus sein, dass keine Ergebnisse gefunden werden oder dass diese zu ungenau sind. Hier können Überprüfungsverfahren und Zusatzinformationen aus Quellen wie EPG, HbbTV, OCR oder der Audiospur benutzt werden, um die Erkennungsrate zu erhöhen und diese Teilinformationen zu ergänzen.

Die zur Erkennung benutzten Bibliotheken wie OpenCV oder Tesseract bieten zwar die Möglichkeit Videoinhalte zu identifizieren; diese Verfahren benötigen aber auch eine höhere Rechenleistung.
Bei der Verwendung von Deep Learning-Verfahren ist die Rechenleistung beim Training der Daten entscheidend. Das bedeutet, dass die Verwendung dieser in Set-Top-Boxen oder in Smart-TVs innerhalb von TV Apps — auch Clients genannt— aus Performanzgründen nicht möglich ist. Aus diesem Grund stellt eine serverorientierte Lösung für die Videoerkennung die geeignetste Lösung für die Video-Live-Analyse dar. Speziell für die Videoerkennung sollten dedizierte Server eingesetzt werden. Die Ergebnisse der Erkennung können in Form einer vereinfachten semantischen Struktur (z.B. in XML/RDF oder JSON) verteilt und gleichzeitig den Clients mitgeteilt werden.
Erste Cloud-basierte Lösungen wie z.B. WIREWAX, die teilweise Konzepte und Objekte automatisch erkennen, zeigen zwar, dass es technisch möglich ist, eine rein verteilte und online-basierte Videoanalyse durchzuführen, sie sind aufgrund der hohen Analysezeiten nicht für

einen Live-Betrieb geeignet, geschweige im Fernsehbereich einsetzbar. Eine neue Entwicklung im Bereich der Videoanalyse, die sich von der klassischen Videoanalyse differenziert, bildet die Anwendung von neuronalen Netzen bzw. Deep Learning-Verfahren zur Extraktion und Identifikation von Szenen und Konzepten innerhalb von Videos. Als Ergebnis dieser Verfahren wird eine detaillierte textuelle Beschreibung des Videoinhalts erzeugt. Die Benutzung neuronaler Netze wie LeNet [LBBH98], AlexNet [KSH12] oder GoogleLeNet [SLJ+14] zeigen, dass es schon heute möglich ist, Bilder und Videos sehr fein granular zu klassifizieren. Die Arbeiten von [VXD+14a] [RASL16] [RRH+15] [RTR+16] [Roh14] und [MRF16] demonstrieren, dass eine automatisch feingranulare Beschreibung von Filmen und Videobeiträgen durch die Anwendung von Deep Learning zu erzeugen, technisch möglich ist, wobei größere Datenmengen als Trainingsset verfügbar sein müssen. Eine davon, die Yahoo Flickr Creative Commons 100 Million (YFCC100M) [KSDB15] [NPB+15], wird z.B. für die Erkennung von Ereignissen (z.B. Geburtstagsfeier, Hochzeit) [BBE+15] innerhalb eines gegebenen Videos benutzt. Längerfristig denkbar wäre, dass mittels einer Aktivitätserkennung auf Basis einer Video- und Audioanalyse, wie in [Fre15] oder [RAAS12], [DBBF14] beschrieben, eine noch feingranularere und zeitbasierte Beschreibung eines Videos erzielt werden kann. Darüber hinaus zeigen Cloud-basierte kommerzielle APIs wie z.B. die von Microsoft Cognitive Services oder Google Cloud Vision, dass es bereits möglich ist, Gefühle und Emotionen aus einem Video über einen dedizierten Dienst zu extrahieren. Ein weiterer Ansatz bildet die kombinier-

te Extraktion von Semantik. Diese Kombination wird in den Arbeiten von [DWC01] und [NBSD08] [RKD+03] [BDNS08] skizziert. Die darin beschriebenen Ansätze benutzen unterschiedliche Quellen (Radio- und Fernsehbeiträge) in unterschiedlichen Formen (Spielerstatistiken, Aufstellungslisten, Spieletabelle in textueller Form) und in verschiedenen Sprachen, um Elemente der Fußballdomäne zu erfassen und zu erkennen. Bei diesem Ansatz war die Herausforderung, die zeitlichen Veränderungen der Entitäten zu erkennen, d.h. den Zeitpunkt eines Tores, oder wann und an welchen Spieler eine gelbe Karte gegeben wurde). Mittels Spracherkenner wurden die Audiospuren aus den Audiobeiträgen bzw. Fernsehbeiträgen analysiert, in Textform umgewandelt und mit Zeitstempeln versehen. Die kombinierte Benutzung von verschiedenen Kanälen (Video- und Audioanalyse) und Verfahren (OCR, Speech-To-Text) zur Extraktion semantischer Informationen und Konzepten aus Nachrichtensendungen wird in [MGK+10] detailliert vorgestellt.

Die Videoanalyse bildet die zentrale Komponente für eine automatische Extraktion von Informationen aus Live-Fernsehbildern. Dabei ist die Analysezeitdauer das wichtigste Kriterium; rein Cloud-basierte Lösungen können nicht „mithalten“, da die Erkennung und die Bereitstellung der Ergebnisse manchmal im Halb-Minuten-Bereich liegen. Aus diesem Grund sollte die Videoanalyse auf dedizierten Servern, die sich nur mit der Videoanalyse beschäftigen, durchgeführt werden. Falls eine journalistische oder redaktionelle Kuratierung notwendig ist (z.B. ein Journalist möchte bestimmte Begriffe hinzufügen, um den Kontext einer Reportage zu präzisieren), kann er dies über dedizierte Annotationswerkzeuge durchführen.

Nachdem die Werkzeuge und die unterschiedlichen Verfahren zur Semantifizierung von Videos näher betrachtet wurden, werden im nächsten Kapitel die Grundlagen zum digitalen und interaktiven Fernsehen eingeführt.

Technische Grundlagen zum digitalen interaktiven Fernsehen

Das digitale Fernsehen bietet den Zuschauern neue Möglichkeiten mit ihren Fernsehern zu interagieren. Die wichtigste Neuerung stellt die technische Möglichkeit dar, parallel zum Video- bzw. Fernsehsignal, Zusatzdienste zu übertragen, die von den Zuschauern interaktiv genutzt werden können. Dazu wurden verschiedene Technologien im Signalübertragungsbereich aber auch neue Standards etabliert, die Schritt für Schritt von den Fernsehherstellern entwickelt und implementiert wurden. Zugleich haben das Web und seine Verwendung in diesem Bereich immer mehr an Bedeutung gewonnen. Die Art und Weise, wie Fernsehsender und die Fernsehgeräte neue Hardware aber auch softwaretechnische Lösungen integrieren, hat einen direkten Einfluss auf die Interaktion und die Verwendung von Zusatzdiensten am Fernseher.

Die Technologien und Begriffe, die im folgenden Abschnitt vorgestellt werden, dienen dazu, sowohl den Fortschritt in puncto Interaktivität

bei der Benutzung des Fernsehers darzustellen als auch die grundlegenden Technologien der aktuellen Fernsehbetriebssysteme besser zu verstehen.

Da diese Technologien stets im Wandel sind, beruhen die präsentierten Technologien und Entwicklerwerkzeuge (SDKs und APIs) auf dem Stand von November 2016. Leser dieser Arbeit können sich mittels der angegebenen Links über Weiterentwicklungen der hier vorgestellten Technologien jederzeit informieren.

Um einen möglichst breiten und präzisen Überblick der eingesetzten Technologien zu liefern, beschränkt sich die Analyse nicht nur auf digitales Fernsehen, Smart TVs und ihrer Hardware, sondern es werden hybride Geräte wie Set-Top-Boxen (Zusatzgeräte, die den Videoeingang, meistens HDMI, eines Fernsehgeräts nutzen, um über dieses neue Dienste anzubieten) und auf TV-Sticks (Sticks, die in den HDMI-Videoeingang eines Fernsehgeräts angebracht werden, um, ähnlich den Set-Top-Boxen, Zusatzdienste oder Apps anzubieten) basierende Systeme betrachtet. Da deren Vielfalt zurzeit sehr groß ist und sie bzgl. ihrer technischen Unterschiede sehr heterogen sind, werden nur diejenigen berücksichtigt, die für diese Arbeit relevant sind. Bei der Betrachtung der technischen Aspekte des digitalen Fernsehens (mittels Satellit oder terrestrischer Übertragung) und des Grades der Verfügbarkeit interaktiver Dienste, werden nur die in Mitteleuropa (insbesondere Deutschland und Frankreich) angebotenen Technologien betrachtet. Technische Aspekte wie z.B. die Inhalte der Implementierung und Verwendung von digitalen Signalen und Protokollen werden für informative Zwecke vorgestellt. Diese sollen ein besseres Verständnis der Datenübertragungstechniken ermöglichen. Teile der vorgestellten Technologien wurden im Rahmen der Implementierung des Swoozy-Systems softwaretechnisch umgesetzt.

Das digitale Fernsehen (DVB)

Unter dem Begriff DVB (Digital Video Broadcasting) versteht man ein digitales Verfahren, das in der Lage ist, digitale Inhalte (TV und Radiosignale) und Zusatzdatendienste wie z.B. EPG (Electronic Programming Guide), MHP (Multimedia Home Platform) oder HbbTV (Hybrid Broadcast Broadband) d.h. zusätzliche TV-Inhalte, die durch digitale Technologien und Verfahren erzeugt wurden, zu übertragen. Diese Inhalte können nur von einem geeigneten DVB-Receiver abgespielt werden.
DVB ist ein standardisiertes Übertragungsverfahren, das für verschiedene Verbreitungsvarianten benutzt werden kann:

- DVB-S /S-2 : Satellitenempfang
- DVB-C : Übertragung über Kabelnetze
- DVB-T / T-2 : Terrestrischer Empfang von Fernsehsignalen

Die DVB-Übertragungsverfahren sind genormt und wurden vom ETSI (*European Telecommunications Standards Institute*[1]) standardisiert. Die Videoenkodierungsformate MPEG-2 werden für die DVB-T-Ausstrahlung mit SD-Qualität benutzt. Beim DVB-T2 wird das Format MPEG-4 (H264) für die Ausstrahlung von Programmen in HD-Qualität benutzt. Längerfristig soll das Videoenkodierungsstandard HEVC (High Efficiency Video Coding - H.265) beim neueren DVB-T2 HD eingesetzt werden, um Programme bis zu einer Qualität von 8K transportieren zu können. Der nächste Abschnitt erklärt im Detail die Grundprinzipien der DVB-Signalübertragung. Zum besseren Verständnis wird nicht die komplette Spezifikation der DVB-Übertragung erläutert, sondern nur die für eine spätere technische Ausarbeitung relevanten Aspekte.

[1] `http://www.etsi.org`

Technische Komponenten und Architektur

Bei DVB werden die Videodaten über einen sog. MPEG-Datenstream gesendet. Dieser wird als Transport-Stream (oder kurz MPEG-TS) übertragen. Die Abkürzung MPEG steht für Moving Picture Experts Group. Die Gruppe ist für verschiedene Standardisierungen im Audio- und Videobereich zuständig. Darunter fallen die Formate und die Datenkomprimierungsverfahren von Videos, die bei Mediaplayern hard- und softwaretechnisch implementiert sind.
Im Falle von DVB-S/2 und DVB-C beträgt die Übertragungsgeschwindigkeit etwa 40 Megabits/Sekunde (bei DVB-T nur 25 Megabit/Sekunde). Parallel zum eigentlichen DVB-Bildsignal enthält der MPEG-2 Stream weitere sog. *Substreams* (oder sog. *Elementary Streams*), die während der Ausstrahlung eines Videosignals parallel (jedoch für den Zuschauer unsichtbar) mitausgestrahlt werden.

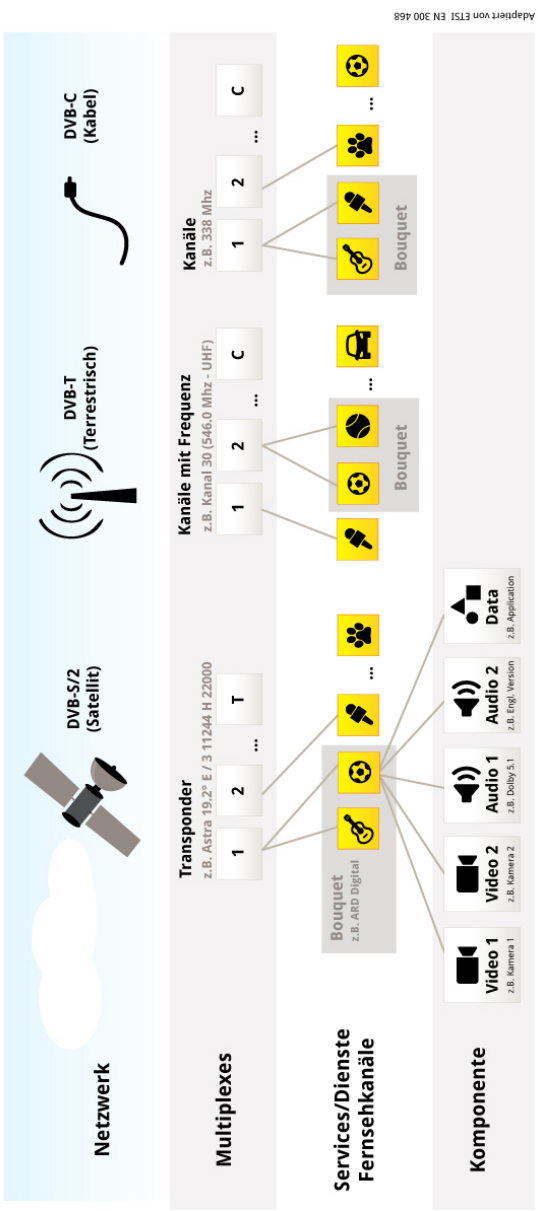


Abbildung 4.1: Aufbau eines Multiplexes. Grafisch adaptiert aus [ETSI EN 300 468]

Ein Transport-Stream kann mehrere separate Fernsehkanäle transportieren. Dies bezeichnet man als Multiplex. Unter einem Multiplex - manchmal *Bouquet* genannt - versteht man die Gruppierung von Fernsehsendern (oder Services) auf einem einzigen Kanal (oder einer Frequenz), sodass auf einem Stream mehrere Fernsehkanäle transportiert werden können. Im Saarland z.B. besitzen die Sender Das Erste, Phoenix und Arte beim DVB-T die gleiche Frequenz (642000 MHz). Erst der DVB-Receiver kann die Kanäle wieder „Demultiplexieren" (manchmal ugs. auch als „Demuxen" bezeichnet) und anhand der mitausgestrahlten Tabellen richtig anordnen, sodass Daten, wie z.B. EPG (Electronic Program Guide)-Daten, dem richtigen Kanal zugeordnet werden können (siehe Abbildung 4.1).
Ein Fernsehkanal wird als Service bezeichnet. Dieser Service enthält nach dem Demultiplexieren verschiedene Substreams, die während einer Sendung variieren können. Bei der Übertragung der Fußball WM 2014 konnte der Zuschauer verschiedene Kommentar- und Audiospuren auswählen. Ähnlich sieht es bei Sport- oder Musikkanälen aus, welche die Möglichkeit bieten, die Kameraperspektive zu wechseln. Der Aufbau jedes Streams erfolgt über Tabellen und Deskriptoren. Eine detailliertere Spezifikation dieser Tabellen ist u.a. in den Standards ISO IEC 13818-7 und ETSI EN 300 468 zu finden. Sie bilden die Grundbestandteile des Digitalen Fernsehens (DVB).

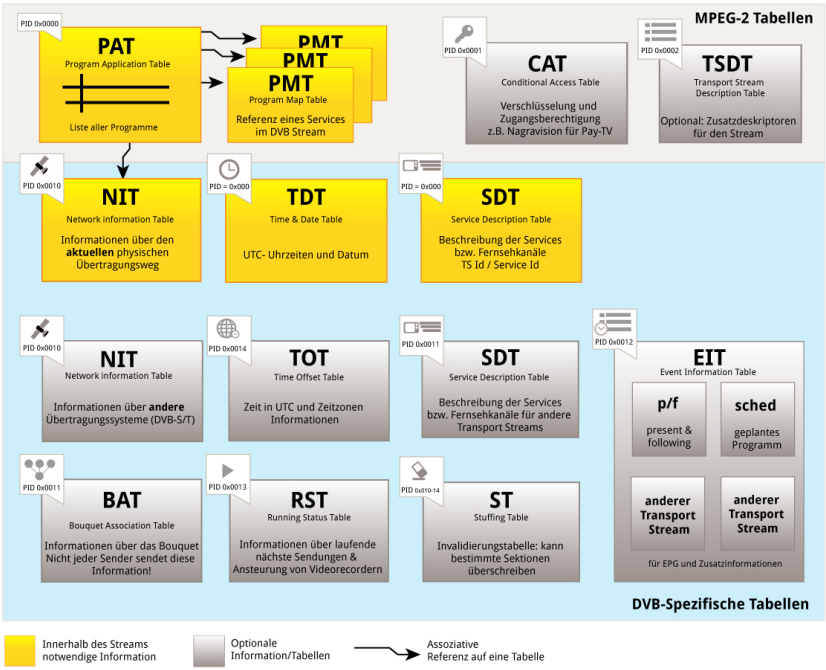


Abbildung 4.2: Informationstabellen im MPEG und DVB-Stream. Quelle: Grafisch adaptiert aus [ETSI EN 300 468]

Nachdem die Tabellen, die über MPEG-2-TS mitgesendet wurden, vom Receiver empfangen worden sind, werden die Bild- und Tonsignale dekodiert, die angehängten Zusatzdienste wie z.B. EPG (Electronic Program Guide) aufgerufen und je nach Speicherkapazität des DVB-Receivers in einem letzten Bearbeitungsschritt die mitausgestrahlten Daten (Bild und HTML-Dokumente) zwischengespeichert. Innerhalb des MPEG-Streams spielen sog. Tabellen, u.a. zur Konfiguration des Receivers, zur Einstellung des Multiplexers und zum Transport von EPG-Informationen eine bedeutende Rolle. Wechselt man den Multiplex oder das Bouquet (diese Aktion wird vom Benutzer über seinen

Receiver durchgeführt), dann müssen die Informationen innerhalb dieser Tabellen erneuert werden. Die Abbildung 4.2 listet grafisch diese Informationstabellen und Funktionen auf.

EIT: Event Information Table

Zu den wichtigsten dieser Tabellen zählt für den Benutzer bzw. Zuschauer die EIT (Event Information Table). Diese Tabelle beinhaltet die eigentlichen EPG-Informationen über das aktuell ausgestrahlte Programm. Sie werden als Event-Blöcke mitgeliefert. Eine kurze Zusammenfassung des aktuellen Programms findet man unter den *short_event_descriptor, extend*-Blöcken, die erweiterte Fassung wird im *extended_event_descriptor*-Block gespeichert. Zusätzlich kann die Tabelle auch bestimmte *Nibbles* (Blöcke) enthalten, welche die Kategorie der Sendung (z.B. News, Sport, usw.) bezeichnen. Die Bezeichner können je nach Spezifikation des DVB-Receivers entweder als kleines Ikon (engl. Icons) dargestellt werden (z.B. ein Ball für Sportsendungen, eine Zeitung für Nachrichtensendungen) oder als Programmkategorien in einem Untermenü erscheinen.
Eine weitere Tabelle, die direkt von HbbTV (Hybrid Broadcast Broadband TV) kompatiblen Receivern beim Empfang herausgefiltert und interpretiert wird, ist die sogenannte AIT (Application Information Table). Diese wird in Kombination mit einem Daten- oder Objektkarussell benutzt, um unter anderem an den Receiver den Befehl zu senden, HbbTV oder MHP (Multimedia Home Platform) Applikationen aus dem ausgestrahlten Signal „herunterzuladen“. Das Wort „herunterladen“ sollte dabei nicht im Web- oder HTTP-Kontext verstanden werden, sondern im Sinne eines monodirektionalen Empfangs via DVB-Signal (engl. Broadcast). Das Herunterladen von Daten kann aber auch via

Broadband-Verbindung, d.h. über Internetdienste geschehen. Diese Variante des Applikationsempfangs wird im Abschnitt über HbbTV näher skizziert.

AIT: Application Information Table

Die AIT-Tabelle wird als Service-Tabelle bezeichnet und enthält die Information für den sog. *Autostart*-Zeitpunkt der HbbTV-Applikation des jeweiligen Senders. Der Autostart-Punkt gibt dem Receiver den Befehl, die AIT-Tabelle zu interpretieren. Beim Auslesen der AIT-Tabelle durch den Receiver wird die URL des HbbTV-Angebots erfasst. Diese kann über den *Red Button* der Fernbedienung aufgerufen werden. Der Fernseher wechselt zu dem eingebauten Webbrowser, der diese URL öffnet.

DSM-CC: Daten- und Objekt-Karusselle

Zu den wichtigsten Konzepten, die bei DVB und MPEG-2-Stream zur Interaktivität und zum Herunterladen von Daten beitragen, gehören die sog. Daten- und Objektkarusselle. Die einfache Übertragung von Daten von einem TV-Sender (in diesem Kontext auch *Broadcaster* genannt) zu einem Receiver wird über ein DSM-CC-Daten-Karussell realisiert. DSM-CC wird im Standard [ETS14c] definiert und steht für *Digital Storage Media – Command and Control*. Darunter versteht man die technische Möglichkeit, über einen MPEG-2-Stream bestimmte Kommandos und Dateien an einen Receiver zu senden. Diese Kommandos können von diesem kompatiblen Receiver empfangen, analysiert und in entsprechende Funktionen umgewandelt werden.
Eigentlich wurde DSM-CC für eine Zwei-Wege-Kommunikation ausgelegt. Das bedeutet, dass ein Rückkanal (meistens über RPC (Remote

Procedure Call)) benutzt werden kann, um bestimmte Informationen (ähnlich eines URL/HTTP-Requests) oder Bestätigungskommandos zu dem Broadcaster (oder Server) zurückzusenden.
Heutzutage benutzen nur noch sehr wenige Broadcaster diese Möglichkeit (u.a. weil die Receiver diesen Rückkanal nicht mehr unterstützen). Die Smart-TV-Geräte und HbbTV dagegen tragen dazu bei, dass das Internet (Broadband) als primärer Übertragungsmodus benutzt wird. Um während dieser Transitionsphase trotzdem dem Benutzer die Möglichkeit zu geben, Daten aus dem TV-Signal zu empfangen, wird das Konzept des Karussells weiterhin benutzt und parallel zu den Smart-TV-Möglichkeiten wie z.B. durch die Benutzung von Fernseher-unabhängigen Apps eingesetzt. Das Karussellprinzip lässt sich am Beispiel des Teletextes gut erläutern (siehe Abbildung 4.3, die von [MSC05] grafisch adaptiert wurde). Jede Teletextseite besitzt eine eindeutige Seitennummer. Diese Seitennummer wird periodisch mit dem TV-Bildsignal gesendet. Das Versenden erfolgt dabei *rundenweise*. Das bedeutet, hat der Benutzer eine bestimmte Seitennummer mit seiner Fernbedienung angegeben, muss das Fernsehgerät zuerst abwarten, bis diese Nummer wieder im ausgestrahlten TV-Signal enthalten ist, damit die entsprechende Seite angezeigt werden kann. Dieses Prinzip wird als Karussell bezeichnet und kann zur Übertragung von Dateien und kompletten Applikationen benutzt werden.
Im Falle des DSM-CC-Objekt-Karussells werden nur 64-kByte große Module (Dateipakete) versendet. Diese werden vom Receiver meistens zwischengespeichert. Im Fall, dass größere Dateien verschickt werden müssen, werden diese in 64-kByte große Objektmodule aufgeteilt. Tritt ein Problem (z.B. während der Übertragung) auf, können die „verpassten“ Daten ab der nächsten Runde wieder abgerufen und heruntergeladen werden.

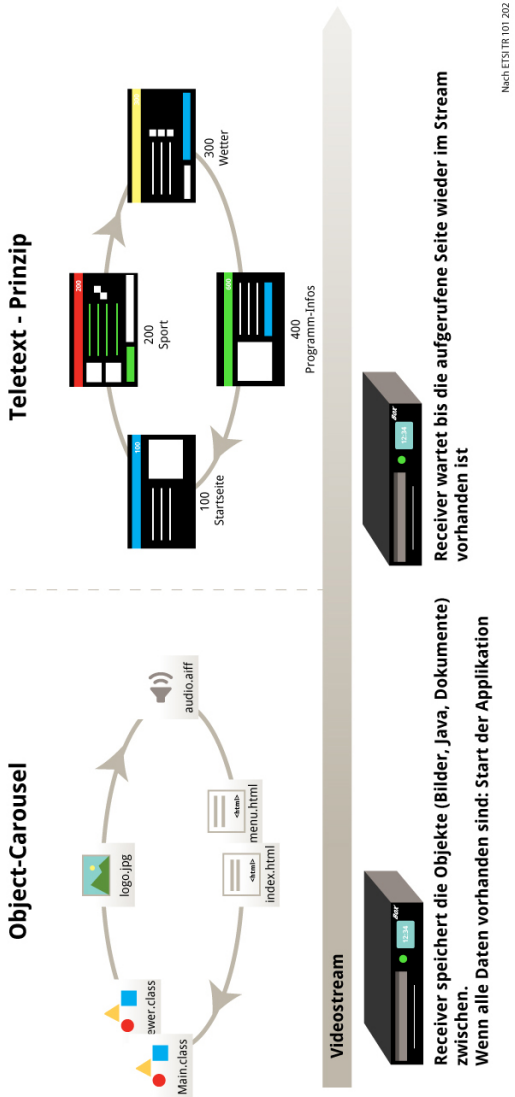


Abbildung 4.3: Analogie Object-Carousel und Teletex-Prinzip. Funktionsweise eines DSM-CC Karusell

Eine komplette Beschreibung der Tabellen und deren Deskriptoren (inklusive beispielhafter Inhalte) können in den Standards [ETS13] [ETS14a] und [ETS14c] nachgeschlagen werden. Im Swoozy-System (siehe Kapitel 6) werden diese Tabellen als Teilkomponenten, die bei der Gewinnung der semantischen Informationen aus einem laufenden Programm von Bedeutung sind, u.a. hinsichtlich ihrer technischen Details und der Methodik zur Informationsextraktion von EPG-Daten und semantischen Auswertung näher untersucht.

Die Extraktion der Daten wird in der Regel vom DVB-Receiver durchgeführt. Diese Stufen der Extraktion und die originalen MPEG-2-Streamdaten sind für den Endbenutzer nicht sichtbar. Entwickler (oder Rundfunktechniker) dagegen können zu Debbugging- und Signalanalysezwecken Software-Werkzeuge wie Kommandozeile-basierten Werkzeuge (z.B. DVBSnoop[2] unter Linux) oder graphisch gestützte Lösungen wie den DVB Inspector[3] nutzen.

Diese Werkzeuge sind nicht nur in der Lage, einzelne Tabellen zu extrahieren, sondern stellen alle Rohdaten aus einem Echtzeit- oder vorgespeicherten MPEG-Stream zur Verfügung.

Softwarelösungen für Computeranwender wie z.B. den VLC Media Player[4] sind in der Lage, MPEG-Streams aus einem DVB-Signal zu lesen und zu extrahieren. Dabei werden allerdings nicht alle Daten aus den Tabellen extrahiert. Meistens werden nur die Programmdaten und die EPG-Informationen gelesen und in der jeweiligen Software eingeblendet. Dagegen werden HbbTV-Applikationen nicht automatisch extrahiert und ausgeführt.

[2] http://dvbsnoop.sourceforge.net
[3] http://www.digitalekabeltelevisie.nl/dvb_inspector
[4] http://www.videolan.org

Electronic Program Guide

Der *Electronic Program Guide* kurz *EPG* wird aus den EIT-Tabellen nach dem ETSI EN 300 707- Standard generiert. Dank EPG kann sich der Zuschauer Zusatzinformationen über das laufende und kommende Fernsehprogramm direkt auf seinem Fernseher, meistens in Form einer Kurzbeschreibung, anzeigen lassen. Diese wird als Overlay (Einblendung) über das gerade ausgestrahlte Bild eingeblendet. Bei weiterer Betätigung der Fernbedienung erhält der Zuschauer eine längere Beschreibung des Programms, inklusive Bilder. Der Hersteller Technisat bietet eine EPG-Erweiterung mit der Option SiehFern[5]. Bei dieser werden die Daten über einen zusätzlichen Kanal (Data-Kanal – praktisch ein unbenutzter Fernsehkanal zum Datenaustausch) heruntergeladen und im Terminal bzw. Receiver gespeichert. Beim analogen Fernsehen wurde EPG mit dem Teletextsignal mitgesendet, beim Digitalen Fernsehen werden die Daten über eine dedizierte Tabelle mitausgestrahlt und vom Receiver interpretiert. Alle EPG-Daten werden letztendlich in die EIT-Tabelle gepackt.

Smart TV

Geschichte und Prinzipien

Die Idee, interaktive Inhalte auf einem Fernsehgerät anzubieten, ist nicht neu. Erste Versuche wurden in den 1970ern mit Systemen wie z.B. VideoText, Sceptre von AT&T oder PresTel und ViewData der britischen Post gemacht (siehe Abbildung 4.4). Diese Systeme benutzten den Fernseher als Hauptbildschirm, um Dienste über eine analoge

[5] `https://www.technisat.com/de_DE/SiehFern-INFO/352-496/`

Telefonleitung aufzurufen. Sie wurden über eine Tastatur bedient. In Deutschland wurden diese Dienste als BTX (Bildschirmtext) und in Frankreich als Minitel bekannt, wobei der Minitel in seiner Endversion nicht mit einem Fernsehgerät verbunden werden musste, da er selbst über einen Bildschirm verfügte. Die grafische Darstellung der Dienste geschah entweder in Form einer Kommandozeile oder ähnelte sehr stark der Teletextanzeige[6].

Abbildung 4.4: Geschichte und Evolution des Smart TV

Bei diesen Systemen war die Interaktivität sehr eingeschränkt und es mangelte aufgrund der damals verwendeten Hardware an der entsprechenden Geschwindigkeit. Obwohl die Anzeige über ein Fernsehgerät geschah, diente das Fernsehgerät nur als Ersatz für einen Computerbildschirm. Das Werbeargument „anschließbar an ein gewöhnliches Fernsehgerät" diente dazu, eine große Anzahl an Benutzern für diese neuen interaktiven Dienste zu begeistern. Computerterminals waren damals noch sehr groß, nicht transportabel und für das breite Publikum nicht verfügbar.

Das Konzept vom interaktiven Fernsehen, wie es heutzutage bekannt ist, wurde erstmals 1994 ausformuliert und innerhalb einer Patentanmeldung der französischen Firma FAST (FRANCE ADV SYS TECH SARL) [DJM94] wie folgt beschrieben. Der Text wurde aus dem Französischen

6 http://techchannel.att.com/play-video.cfm/2012/2/27/AT&T-Archives-Viewtron-Videotex-Sceptre

übersetzt[7]:

„Das Verarbeitungssystem umfasst eine Benutzeridentifikationseinrichtung (13), die eine einzelne oder bidirektionale Kopplung verwendet. Datenkompression und -Dekompression werden verwendet, um Übertragungszeiten zu reduzieren. Das System kann durch eine Fernsteuerung (16) in der Art einer herkömmlichen TV-Fernbedienung gesteuert werden. Der Benutzer kann Daten und Befehle über die Fernbedienung eingeben. Ein CD-ROM-Laufwerk und Bildplattenspieler (12) sind an das Netzwerk (24) gekoppelt".

Obwohl die beiden in der Patentbeschreibung genannten Technologien *Bildplattenspieler* und *CD-ROM-Laufwerk* im Zeitalter des Internets schon fast verschwunden sind, adressiert diese Definition bereits zwei Kernaspekte des Smart-TVs: Interaktivität und Datenkommunikation, letztere ist heutzutage ein Synonym für das Internet.

Eines der ersten Systeme, das Datenkommunikation benutzte (und somit das Web mit dem Fernsehen vereinigte) und für das breite Publikum käuflich zu erwerben war, stellte die 1997 von WebTV Networks entwickelte WebTV Set-Top-Box dar. Diese verfügte über ein Modem und die Möglichkeit, angepasste HTML-Inhalte direkt über den angeschlossenen Fernseher anzuzeigen. Einige Monate später wurde WebTV von Microsoft aufgekauft und 2001 unter dem Namen MSN TV (später als MSN TV2) vermarktet. Ende 2013 wurde MSN TV aus Strategiegründen von Microsoft eingestellt.

Parallel zu diesen Entwicklungen hatte 1996 auch Samsung[8] zusammen mit der Firma Diba Inc. das Produkt *Internet TV* auf den Markt in Südkorea, in den USA und in Japan gebracht. Die besondere Eigenschaft des internetfähigen Fernsehgeräts lag in der Integration eines 33.6-K-Modems, um das Gerät mit dem Web zu verbinden. Der Fern-

[7] http://patent.ipexl.com/FR/2726670-a1.html
[8] http://www.samsung.com/us/news/42

sehhersteller Philips brachte ebenso internetfähige Set-Top-Boxen auf den Markt. Exemplarisch seien die Philips Magnavox Box für WebTV mit eingebautem Modem oder der CD-i Player genannt.
Heutzutage wird der Begriff *Smart-TV* verwendet, um alle Fernsehgeräte zu bezeichnen, die mittels einer dedizierten Software mit dem Internet Zusatzdienste bieten. Beim Installieren von Smart-TV- oder TV-Apps werden Programme von Drittanbietern, ähnlich den Apps von mobilen Endgeräten, in das Fernseherbetriebssystem integriert. Diese benutzen hauptsächlich die Internetverbindung, um aktuelle Daten einzublenden. Die App-Palette reicht von Spielen über 3D-basierte Videostreaming-Plattformen bis zur direkten Verbindung mit sozialen Netzwerken wie Facebook oder Twitter. Diese Apps werden über dedizierte Stores des jeweiligen Fernsehherstellers heruntergeladen. Beim HbbTV-Angebot können Applikationen (diese werden allerdings nicht als Apps bezeichnet) über den MPEG-Stream vom Fernsehsender versendet werden. Dies bedeutet, dass der Benutzer sie nicht extra aus einem Store herunterladen muss. Da die Applikationen sehr viel Platz innerhalb des MPEG2-Streams einnehmen, wird diese Variante sehr selten von Fernsehsendern benutzt, da die Bildqualität und Bildrate des auszustrahlenden Videosignals entsprechend gesenkt werden muss. Aus diesem Grund wird bei HbbTV über die AIT-Tabelle eine URL mitgeliefert. Diese gibt dem Smart-TV die Anweisung, eine Internetseite aufzurufen oder ein kleines CE-HTML-basiertes Informationsbanner anzuzeigen. Diese Applikationen können vom Benutzer nicht gespeichert werden und sind nur solange verfügbar, wie der Zuschauer auf dem entsprechenden TV-Kanal oder der Sendung bleibt.

Smart TV Alliance

Vorstellung

Die Smart TV Alliance (kurz STA[9]) wurde im Juni 2012 von den Fernsehherstellern LG Electronic und TP Vision (Display-Hersteller von Philips) gegründet und hat sich zum Ziel gesetzt, Standards und Spezifikationen für Smart-TVs und deren Apps zu definieren.
Mittlerweile zählen zu dieser Allianz über fünfzehn Mitglieder, darunter LG, Philips, Toshiba, Panasonic, Vestel und IBM, die im Rahmen von regelmäßig publizierten Spezifikationen definieren, welche Technologien bei Hard- und Softwareweiterentwicklungen im Bereich Smart-TV benutzt werden sollen. Diese Spezifikationen müssen in der Folge von allen Mitgliedern respektiert und in Smart TV-Geräte der jeweiligen Hersteller implementiert werden.
Ein weiteres Ziel der STA ist es, einen einheitlichen App-Entwicklungs-Workflow für Smart-TV App-Entwickler anzubieten, wobei angestrebt wird, dass Apps, die für einen Hersteller der STA-Allianz geschrieben wurden, auch auf allen anderen Fernsehgeräten der Mitglieder lauffähig sein müssen. Um dies zu gewährleisten, publiziert die STA Spezifikationen und Werkzeuge, die diesen Prozess vereinfachen sollen. Zurzeit empfiehlt die Smart TV Alliance die Verwendung von HTML5-Technologien als grundlegende Basistechnologie für die Erstellung und Implementierung von Smart-TV Apps.

Werkzeuge

Die Smart TV Alliance bietet parallel zu den Standardisierungsaktivitäten Werkzeuge (basierend auf der Eclipse-Entwicklungsplattform[10])

[9] http://www.smarttv-alliance.org
[10] http://www.eclipse.org

zur Implementierung von Smart-TV-Apps. Dank der Herstellerunabhängigkeit und der plattformübergreifenden Spezifikationen brauchen diese Apps von Entwicklern nur einmal implementiert zu werden. Sie werden anschließend von der Smart TV Alliance überprüft, die sie zur individuellen Validierung an die Hersteller weiterreicht. Dieser Validierungsschritt gewährleistet, dass die App später auf allen Geräten der Hersteller installierbar und funktionsfähig ist. Es wird trotzdem empfohlen, die Apps mit den jeweiligen Werkzeugen der Hersteller zu testen. Spezielle Gerätefunktionalitäten, die über herstellerspezifische APIs aufgerufen werden können, sollten für diesen Validierungsschritt nicht benutzt werden[11].

Interaktivität beim digitalen Fernsehen

Im folgenden Abschnitt werden verschiedene interaktive Lösungen und Standards vorgestellt, die ermöglichen, parallel oder zeitversetzt zu einer laufenden ausgestrahlten Fernsehsendung, Zusatzinformationen einblenden zu lassen. Unter Zusatzinformationen versteht man Informationen über das Programm, das die Form einer Audiobeschreibung für Sehbehinderte oder vorbereitete Einblendungen (Untertitel) für Schwerhörige nimmt. Obwohl diese Informationen nicht immer sichtbar sind und erst nach einem expliziten Abruf des Zuschauers eingeschaltet werden, gehören diese Zusatzkanäle zu den kontinuierlich mitausgestrahlten MPEG-2-Datenströmen.

In Europa konkurrieren verschiedene auf DVB und MPEG-2 Stream basierende interaktive Systeme. Die Interaktivität dieser Systeme wird so umgesetzt, dass innerhalb der DVB-Signale Zusatzdienste hinzu-

[11] `https://developers.smarttv-alliance.org/forum/-/message_boards/message/1000000071`

gefügt und proaktiv von einem Receiver aufgerufen werden können. Neben YouView[12], einer von den britischen Fernsehsendern BBC (British Broadcasing Cooperation), ITV, Channel 4 und Channel 5 initiierten Internet-TV-Plattform und MHEG-5[13] (Multimedia and Hypermedia Experts Group), ein Standard für die Erstellung von interaktiven Diensten am Fernsehen, das in Kombination mit MHP (Multimedia Home Platform) in Großbritannien sehr verbreitet ist, existieren weitere Standards, die wie der aktuell in Europa verbreitete Standard HbbTV für interaktive Fernseher benutzt werden.

HbbTV

HbbTV steht für Hybrid Broadcast Broadband TV und ist seit 2010 ein fester Standard (ETSI TS 10 769 - Standard) zur Beschreibung und Definition von Zusatzinformationen und interaktiven Anwendungen, die von einem Programmanbieter angeboten und über ein Fernsehgerät angezeigt werden können. HbbTV wurde vom HbbTV-Konsortium ins Leben gerufen und war ursprünglich das Ergebnis einer bilateralen Initiative seitens Frankreichs mit dem *H4TV Projekt* und Deutschlands mit dem Projekt *German HTML Profil*. Mittlerweile gehören über dreißig Firmen und europaweit über fünfzig Fernsehkanäle, wie z.B. die France Television Gruppe, TF1, Canal+, die ARD, das ZDF, aber auch private Fernsehanstalten wie die RTL Group zur HbbTV-Initiative.
HbbTV ist in Europa besonders stark verbreitet und wird zurzeit in anderen Ländern wie z.B. Australien oder China getestet.
Die Aufgabe des HbbTV-Konsortiums ist es, eine standardisierte Schnitt-

[12] `http://www.youview.com`
[13] `http://www.mheg.org/users/mheg/archives/doc/MHEG-5_Profile_Issue_1.pdf`

stelle für die Verwendung von Internettechnologien (IP-TV, Streaming-Angebote, Webschnittstellen) mit Live-Fernsehprogrammen zu definieren und diese mit Fernsehhardware zu verknüpfen. Darüber hinaus werden Spezifikationen des DVB-Signals festgelegt, d.h. welche Events oder Deskriptoren innerhalb des DVB- Signals mitverschickt werden müssen, damit der Zuschauer bestimmte Informationen und interaktive Angebote aufrufen kann.

Prinzip

HbbTV funktioniert nur auf digitalen Empfängern, die in der Lage sind, HbbTV-spezifische Inhalte anzuzeigen. Dies ist der Fall bei allen Smart-TV-Geräten. Fälschlicherweise werden jedoch HbbTV und Smart-TV-Apps sehr oft verwechselt. Im Gegensatz zu Smart-TV-Apps, die eine dauerhafte Internetverbindung benötigen, funktioniert HbbTV in zwei distinkten und miteinander mischbaren Modi – daher auch der Name *Hybrid*:

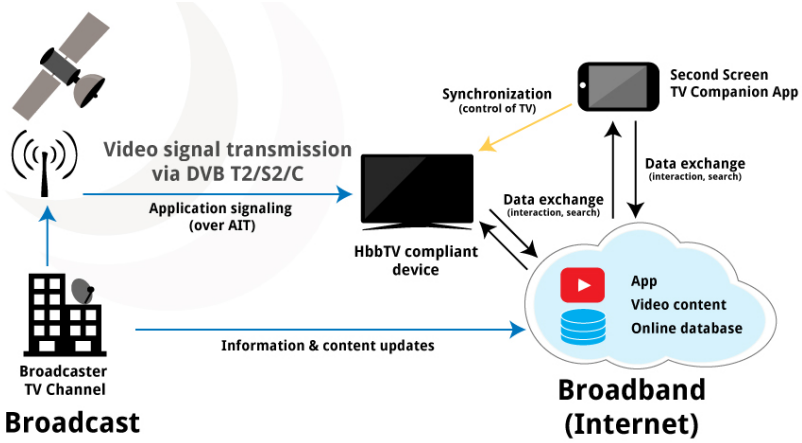


Abbildung 4.5: Architektur von HbbTV 2.0 (Stand 2016). Quelle: [Hbb16]

- **Broadcast**. Im Broadcast-Modus werden die Daten direkt vom Fernsehsender übertragen (verpackt in der AIT-Tabelle oder einem „Daten-Karussell“) und vom Receiver dekodiert. Komplette Applikationen können, parallel zum ausgestrahlten Videosignal, über das DVB-Signal ausgestrahlt werden. Um die Applikation zu starten, benötigt der Fernseher keinen direkten Internetanschluss. In den meisten Fällen wird die Präsenz solch einer Applikation dem Zuschauer mittels eines Banners signalisiert, welches die Aufforderung anzeigt, die rote Taste der Fernbedienung zu drücken. In diesem Zusammenhang wird sehr oft der Begriff *Red Button* benutzt, in Anlehnung an den BBC Red Button-Dienst, mit denen BBC-Zuschauer zusätzliche Inhalte schon seit 1999 aufrufen konnten. Bei Pro7 wird der Begriff *Red Button* auch als Marketingbegriff für das HbbTV-Angebot des Fernsehkanals benutzt. Vom Prinzip her ist der Broadcast-Modus dem etwas älteren Teletextkonzept ähnlich. Je nach Frequenz und Transponderbelegung verzichten Fernsehsender absichtlich auf solche komplexen HbbTV-Applikationen, da diese Applikationen (je nach Komplexität) die komplette Bandbreite in Anspruch nehmen und dadurch den Datenstromplatz zur optimalen Übertragung von Videos und Audiosignalen gänzlich reduzieren können. Aus diesem Grund wird meistens nur ein Link in Form einer URL zu der HbbTV-Seite des Fernsehsenders gesendet. Durch Betätigen der *OK-Taste* werden dem Zuschauer die gewünschten Dienste oder Webseiten zur Verfügung gestellt.

- Die zweite Variante - der **Broadband**-Modus - ermöglicht es, falls der Receiver über eine Internetverbindung verfügt, Zusatzinformationen zum ausgewählten Fernsehprogramm direkt aus dem

Internet aufzurufen. Hierzu wird eine CE-HTML/HTML5-basierte Webseite geöffnet (wobei in den meisten Fällen das aktuell ausgestrahlte Videosignal verkleinert dargestellt wird), die redaktionell vom jeweiligen Fernsehsender aufbereitet und auf einem Server im Internet hinterlegt wurde. Diese Webseite kann entweder Inhalte wie eine Programmübersicht und Hintergrundinformationen über die laufende Sendung weitergeben oder eine Verknüpfung zu Mediatheken beinhalten und weitere Videos und den Zugang zu interaktiven Erlebnissen vorschlagen. Der Broadband-Modus erfordert eine Internetverbindung. Falls diese nicht vorhanden ist, wird dem Zuschauer ein Informationsbanner angezeigt.

Die beiden Modi *Broadcast* und *Broadband* (siehe Abbildung 4.6, die aus [Hbb16] grafisch adaptiert wurde) stellen das Grundkonzept von HbbTV dar. Dabei behält der Fernsehsender die komplette Kontrolle über den Moment, in dem der Zuschauer Zusatzinformationen aufrufen kann, die Qualität der angezeigten Inhalte und die Verknüpfungen der sendungsbezogenen Inhalte. Aus diesem Grund kann HbbTV, im Gegensatz zu Smart-TV-Apps, die eine deutlich größere Vielfalt an Diensten und Informationszugängen vorschlagen können, zurzeit den Zuschauern nur ein begrenztes Inhaltsangebot anbieten.

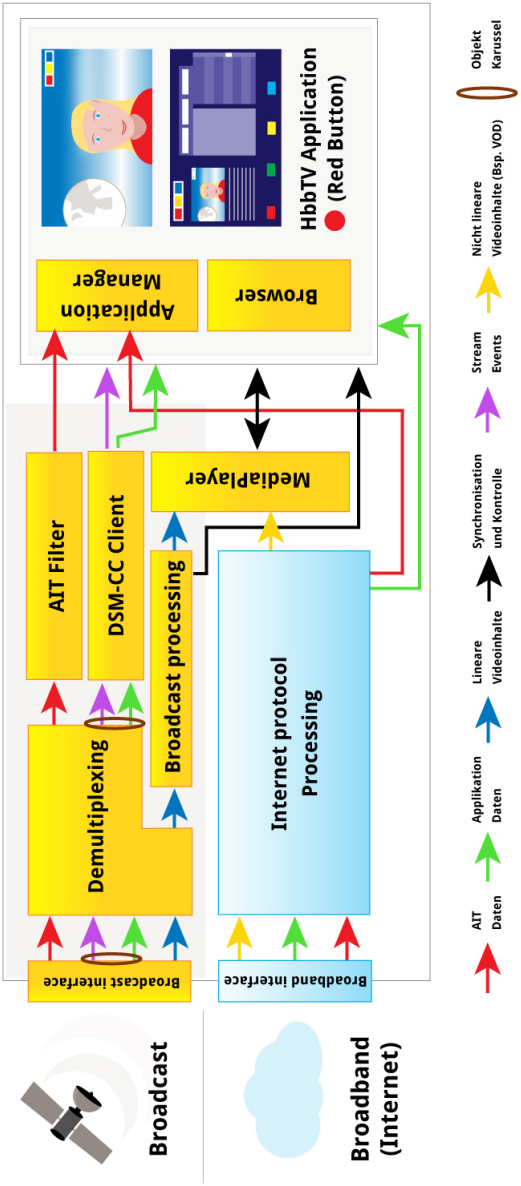


Abbildung 4.6: Broadcast/Broadband Architektur in HbbTV

Benutzerschnittstellen und Interaktionsformen

Im ersten Schritt muss die HbbTV-Applikation über die rote Taste (*„Red Button“*) der Fernbedienung aktiviert werden. Nachdem die HbbTV-Applikation gestartet wurde, erscheint eine HTML-basierte Oberfläche. Diese Bedienoberfläche enthält verschiedene Schaltelemente (eigentlich HTML-Links), die mittels der Pfeil- und OK-Tasten der Fernbedienung selektiert werden können. Anders als bei der normalen Navigation im Web können verschiedene Untermenüs über die vier farbigen Tasten (rot, grün, gelb und blau) der Fernbedienung aufgerufen werden. Meistens verbergen sich hinter diesen Optionsmenüs Einstellungen der HbbTV-App (z.B. Schriftart-Größe, Lautstärke der abgespielten Videos, usw.), rechtliche Informationen wie z.B. das Impressum oder die Kontaktdaten oder Links zu Extras oder Mediatheken.

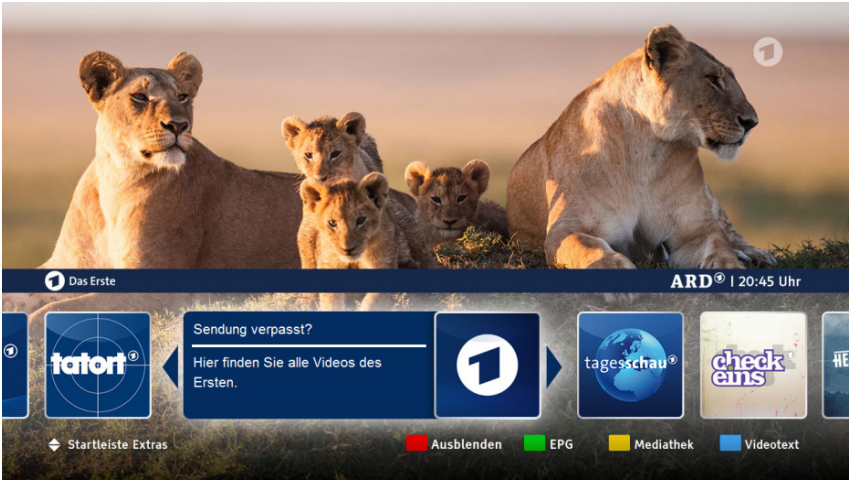


Abbildung 4.7: HbbTV-Angebot von ARD

Die Belegung und Bedeutung der Tasten ist bei jedem HbbTV-Senderangebot unterschiedlich. Dies ist der Fall bei den HbbTV-Benutzerschnittstellen

(siehe Abbildungen 4.7, 4.8 und 4.9), die sich von einem Fernsehsender zum anderen sehr unterschiedlich verhalten.

Die Probleme von HbbTV bzgl. der Interaktion und der Darstellung der Inhalte wurden erstmals im November 2014 von der Initiative Deutsche TV Plattform[14] untersucht und in Form einer Studie veröffentlicht [PDP14].

Die Vielfalt der HbbTV-Benutzerschnittstellen reicht bzgl. ihrer Variationen von einem sehr vereinfachten und schlichten Look'n'feel (siehe Abbildung 4.8) bis zur komplex-verspielten Benutzerschnittstelle, die ein fast identisches Aussehen zu Smart-TV Apps besitzt. Ein gutes Beispiel für die Nutzung von Broadcast- und Broadband-basierter Interaktivität bildet zurzeit die HbbTV-Plattform Multithek von MediaBroadcast[15] u.a. mit dem privaten Musiksender Ampya[16] (bis 2015 als PutPat.tv bekannt) (siehe Abbildungen 4.8 und 4.9).

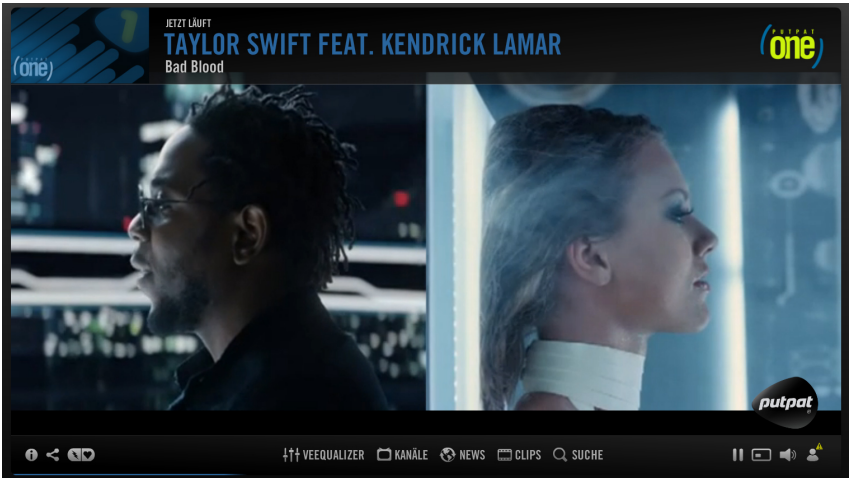


Abbildung 4.8: HbbTV-Angebot von Putpat.tv

[14] http://www.tv-plattform.de
[15] http://www.media-broadcast.com/en/startpage/
[16] https://ampya.com

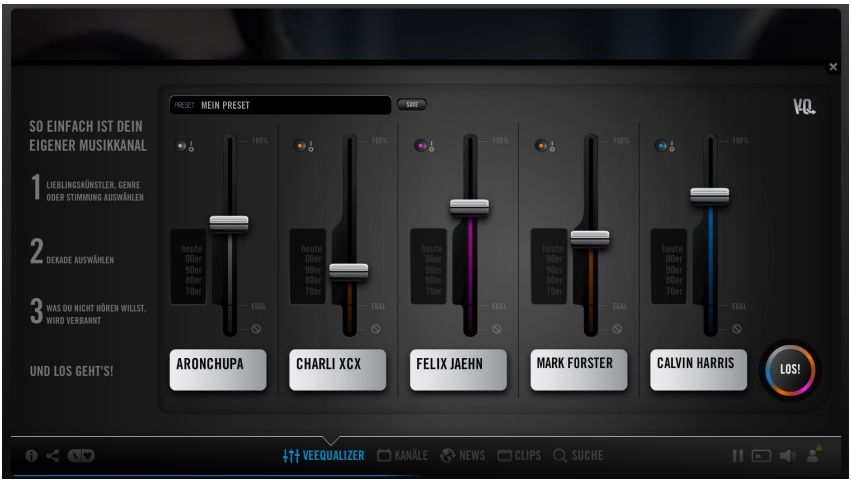


Abbildung 4.9: Beispiel einer erweiterten HbbTV-Interaktion bei Putpat.tv. Auswahl der Benutzerpräferenzen über die Fernbedienung.

Werkzeuge

Die Entwicklung und das Testen von HbbTV-kompatiblen Seiten oder Apps beruhen auf zwei Hauptvorgehensweisen: die eine besteht im Hinzufügen oder Injektieren der HbbTV Autostart-URL in die AIT-Tabelle eines DVB-Signals, die andere in der Entwicklung einer HTML-basierten Benutzerschnittstelle (HbbTV-Anwendung), die der Zuschauer parallel zur laufenden Sendung zu sehen bekommt und mit der er mittels seiner Fernbedienung interagieren kann.

Im ersten Fall existieren speziell für Fernsehsender (engl. Broadcaster) und Labs entwickelte Lösungen wie die von HTTV[17] oder Antlimited[18]. Diese Lösungen bieten in Form eines Playout Streamingservermoduls eine Infrastruktur zum Testen und Abspielen von HbbTV-Inhalten in einem Labor oder einem geschlossenen Übertragungsumfeld. Der

[17] http://www.httv.fr/head-end-products/hbbtv-starterkit
[18] http://www.antlimited.co.uk/cdk.asp?menu=153

Playout-Server fügt in ein vorhandenes DVB-Stream zusätzliche DVB-MPEG SI/PSI-Informationen hinzu, wie z.B. von Technikern oder Testern selbstdefinierte PAT-, PMT-, NIT-, BAT-, SDT-, TDT-Tabellen (siehe Abbildung 4.2). Dadurch ist es diesen möglich, ihre Infrastruktur zu testen und zu erkennen, wie verschiedene Receiver (auch *Headend* genannt) die Informationen verarbeiten und anzeigen. Zusätzlich bietet ein Streaming-Server die Funktionalität, Live-Daten direkt in den Videostream (über die oben erwähnten Tabellen) einzuspeisen und mögliche Veränderungen zu analysieren.
Die Abbildungen 4.10 und 4.11 aus der Webseite HTTV zeigen die Architektur der Lösung HbbTV-Starter Kit und den Workflow zur Bearbeitung des DVB-MPEG-Streams.
Im ersten Schritt wird das eigentliche Videosignal zum Server übertragen. Über eine grafische Administrationsschnittstelle können AIT- und PSI-Tabellen editiert werden. Befehle wie die Autostart-URL oder das DSM-CC-Karussell-Event können in das Signal eingespeist und getestet werden. Dieses nachträgliche Editieren ermöglicht das Live-Versenden der Autostart-URL innerhalb der AIT-Tabelle, um den Receiver zu veranlassen, die mitgelieferte URL auszuwerten und anzuzeigen.
Die HbbTV-Anwendungen, die von Smart-TVs gestartet werden, sind in den meisten Fällen entweder mit den Programmiersprachen HTML5 oder CE-HTML implementiert. Somit kann die im Fernsehsystem eingebaute Webbrowser-Engine die Applikation anzeigen. Die Interaktivität und Interaktion bei HbbTV-Anwendungen werden über die Programmiersprache Javascript realisiert und erfolgen über die Fernbedienung oder bei neueren Smart-TV-Geräten über einen eingeblendeten Cursor.
Über Emulatoren können Entwickler ständig das Verhalten der Seite und der Anwendungen auf verschiedenen virtuell emulierten TV-

Geräten überprüfen, inklusive der Simulation von Fernbedienungseingaben. Verschiedene frei verfügbare SDKs wie die von Opera[19] oder der Smart TV Alliance bieten Werkzeuge zur Realisierung von HbbTV-kompatiblen Applikationen. Eine kostengünstige Alternative zu den Playout-Servern und den professionellen HbbTV-Entwicklungslösungen bildet das Firefox-Plugin namens *Fire HbbTV*[20]. Dieses ermöglicht nicht nur das Debuggen und Testen von Inhalten, sondern simuliert im Mozilla Firefox-Browser vollständig das Verhalten eines Fernsehgerätes, inklusive der Fernbedienung. Eine besondere Fähigkeit von *Fire HbbTV* liegt darin, manuell DVB-Ereignisse (engl. Events) injektieren zu können (analog dem Verhalten eines Playout Servers). Somit können Live-Events getestet werden.

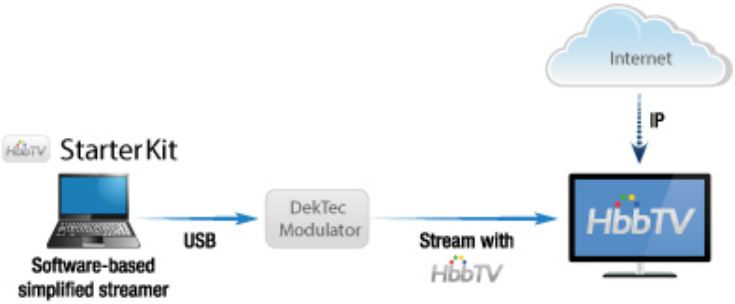


Abbildung 4.10: Architektur der Software-Lösung von httvStream. Quelle: httv.fr

Um das Live-Bild des virtuellen Fernsehers zu simulieren, kann über ein zusätzliches Plugin[21] des VLC-Players direkt das DVB-Signal, das über einen DVB-T USB-Stick eingespeist wird, abgegriffen und in die HTML Seite, die von Firefox geladen wurde, integriert werden. *Fire*

[19] http://www.operasoftware.com/products/tv/tv-store
[20] http://firehbbtv.iptv.aw.atos.net/
[21] http://www.webchimera.org/

HbbTV bietet durch seine Funktionalität eine kostenlose Möglichkeit, HbbTV-Apps und deren Verhalten bei verschiedenen Fernsehauflösungen innerhalb eines Webbrowsers zu testen.

Abbildung 4.11: Konfiguration von HbbTV-Testumgebung mit httvStream. Quelle: httv.fr

Plattformen und Werkzeuge für das interaktive Fernsehen

Zwischen 2013 und 2016 ist die Zahl der verkauften Smart-TV-Geräte stark gestiegen (siehe Statista.de[22]). Obwohl die Fernsehgeräte alle unter dem Namen *Smart-TV* verkauft werden, bestehen in dieser Geräteklasse bzgl. der technischen Implementierungsmöglichkeiten sehr große Diskrepanzen. Im folgenden Abschnitt werden, in Hinblick auf die Entwicklungen und Anforderungen des Swoozy-Systems, die

[22] `https://de.statista.com/statistik/daten/studie/325527/umfrage/anteil-der-tv-haushalte-in-deutschland-mit-smart-tv`

bedeutsamsten Betriebssysteme für Smart-TVs und Set-Top-Boxen genauer untersucht. Deren SDKs werden analysiert, um festlegen zu können, welche Systeme am ausgereiftesten sind und welche eine gute Unterstützung bei der Realisierung sog. TV-Apps bieten. Dazu werden nicht nur die zu Grunde liegenden Technologien präsentiert, sondern auch welche softwaregestützten Frameworks und Werkzeuge für die Realisierung dieser Apps von den jeweiligen Herstellern unterstützt werden. Zusätzlich werden die Interaktionsformen und Designaspekte der Darstellung und Anzeige berücksichtigt.

Samsung SDK – Smart-TV

Der Begriff *Smart-TV* wird oft mit der gleichnamigen Geräteklasse von Samsung verwechselt. Dies ist wahrscheinlich auf die Tatsache zurückzuführen, dass Samsung schon sehr früh damit begann, das vom iPhone her bekannte Konzept der Apps auf die Fernsehwelt zu übertragen. Im folgenden Abschnitt wird eine Übersicht der für Fernseher von Samsung angebotenen Möglichkeiten, Apps zu implementieren, aufgelistet. Aktuelle Samsung Fernseher (Stand November 2016) basieren auf dem neuerem Tizen OS,[23] wobei sich die vorherige Gerätegeneration bis 2015 auf einem Linux Betriebssystem (auch als Orsay OS bezeichnet) stützte. Samsung Fernseher unterstützen sämtliche Webtechnologien wie HTML5, Flash oder Streaming-Protokolle wie RTMP (Real Time Messaging Protocol).

[23] `https://www.tizen.org`

Samsung Web API

Die Samsung Web API SDK[24,25] ermöglicht es, mit einer einzigen Implementierung Apps für mehrere Samsung Zielgeräte (mobile, TV- und Web-App) zu erstellen. Dazu werden verschiedene APIs angeboten, die per HTML5 und Javascript aufrufbar sind. Obwohl diese SDK alle Zielgeräte der Marke unterstützen, sind bestimmte Funktionen nur für Fernsehgeräte verfügbar. Über bestimmte Javascript-Schnittstellen können Informationen zum aktuellen Programm, der Lautstärke oder sogar dem internen Tuner des Fernsehers über die Samsung Web API abgefragt werden. Zusätzlich erlaubt die Samsung Web API die Implementierung von Schnittstellen und Funktionen, die es ermöglichen, Second-Screen Applikationen direkt mit dem Samsung Fernseher zu verbinden.

Zugriff auf sog. Middleware-Funktionalitäten wie z.B. die Kontrolle von Video- und Audio- Abspielfunktionen des Fernsehgerätes oder die Benutzung von Gestensteuerung und Spracherkennung innerhalb einer App sind direkt mittels Javascript aufrufbar.

Eine Eclipse-basierte Umgebung, die Web App SDK, liefert grafische Werkzeuge zur Konzipierung von App-UIs und Animationen. Nachdem der Entwickler diese Apps implementiert hat, kann er seine Smart-TV-Apps zuerst virtuell in einem Desktop-basierten Simulator testen und nachträglich als TV-App „verpacken“ (engl. *packagen*) und sie anschließend auf einem realen Fernsehgerät installieren und testen.

[24] `http://www.samsungdforum.com`

[25] `http://img-developer.samsung.com/onlinedocs/Web_App_SDK_Developer_Guide/webapp_html/dtv/reference/browserspec_html5.html`

Samsung TV SDK

Bei der Samsung Smart TV SDK handelt es sich um eine vollständige Umgebung zur Implementierung und zum Debuggen von TV-Apps. Hierbei werden alle Webtechnologien unterstützt, inklusive HTML5 und Flash/Adobe AIR-gestützter Applikationen. Genau wie bei HTML5-basierten Applikationen können Entwickler über spezielle Javascript APIs (auch *Middleware APIs* genannt) Geräteinformationen abfragen. Eine frei verfügbare Eclipse-basierte Entwicklungsumgebung ermöglicht das Kompilieren der App für den Fernseher. Durch eine netzwerkbasierte Debugging-Funktionalität kann sich der Entwickler direkt die wichtigsten Debugger-Informationen des Fernsehers innerhalb der Entwicklungsumgebung anzeigen lassen. Zusätzlich kann ein sog. Systemabbild des Samsung-Betriebssystems über Emulatoren lauffähig gemacht werden, um das eigentliche Verhalten der App auf der PC/Mac-Ebene zu simulieren. Nebenbei bietet Samsung die Möglichkeit, über Unity und mittels der Programmiersprache C#, 3D-Applikationen zu implementieren.

Tizen TV Web SDK

Fernseher von Samsung, die nach Mitte 2014 produziert wurden, werden mit dem neuen Open Source-Betriebssystem Tizen OS ausgeliefert. Tizen OS ist ein offenes und flexibles, auf Linux-basiertes Betriebssystem, das von verschiedenen Herstellern unterstützt wird. Das Tizen-Betriebssystem existiert in mehreren Ausprägungen: Tizen Mobile (für die Entwicklung von mobilen Applikationen), Tizen Wearable (für intelligente Uhren), Tizen IVI (In-Vehicle Infotainment, für die Unterhaltungselektronik im Auto) und letztendlich Tizen TV für die Programmierung von TV-Geräten [JLK+14]. Im TV-Bereich wird Tizen

ausschließlich von Samsung angeboten und zwar unter dem Namen *Samsung Tizen TV*. Obwohl Tizen OS native Applikationen unterstützt, können mittels der Samsung Tizen SDK jedoch nur Web-basierte Applikationen entwickelt werden[26]. Aus diesem Grund werden die Tizen TV-basierten Applikationen mit Webtechnologien wie HTML5, CSS3 oder Javascript implementiert. Zusätzlich wird das JavaScript Framework namens *CAPH* Framework (innerhalb des Samsung Web UI Frameworks) angeboten, das unter anderem UI-Komponenten wie *Labels*, *Navigatoren*, *ListWidget* und andere integrierbare Komponenten anbietet, um die Entwicklung unter Tizen zu vereinfachen und zu beschleunigen. Bei dem *CAPH* Framework können HTML5-basierte Benutzerschnittstellen mittels der Javascript Frameworks jQuery oder Angular.js erstellt werden.
Parallel zu Tizen TV versucht Samsung mit der neuen Initiative TOAST[27], auf Basis der Webtechnologien HTML5 und des Framework Apache Cordova[28], ein einheitliches SDK zur Entwicklung von markenunabhängigen HTML5-basierten TV-Apps (darunter auch für webOS von LG) zu verbreiten.

Benutzerschnittstelle und Interaktionen

Die Benutzerschnittstelle des Samsung Smart-TV folgt dem *10 foot-Design*-Paradigma und besitzt mehrere grafische Komponenten. Dazu zählen ein verkleinertes Bild des aktuellen Programms und eine Art „Leiste“ mit den jeweiligen installierten Apps. Diese für Samsung spezifische Benutzeroberfläche wird als *Smart Hub* bezeichnet (siehe Abbildung 4.12). Wählt ein Benutzer eine bestimmte App aus, so wird diese

[26] `https://www.samsungdforum.com/TizenIntroduction`
[27] `https://www.samsungdforum.com/Features/TOAST`
[28] `https://cordova.apache.org`

mit voller Bildschirmauflösung eingeblendet. Je nach App-Typ können extern angeschlossene Geräte (Kamera, USB Stick, usw.) direkt über den Smart Hub bedient werden. Obwohl das Prinzip des *Smart Hub* über Jahre hinweg erhalten geblieben ist, verändern sich gleichwohl jährlich das Design, die Platzierung der Apps und die Menüstrukturen. Die Bedienung des *Smart Hub* kann entweder über Gestensteuerung, eine Gyroskop-basierte Fernbedienung oder Spracherkennung (ein Mikrofon ist in der Fernbedienung eingebaut) erfolgen. Der eingebaute Spracherkenner benutzt eine Online-basierte Erkennung der Firma Nuance.

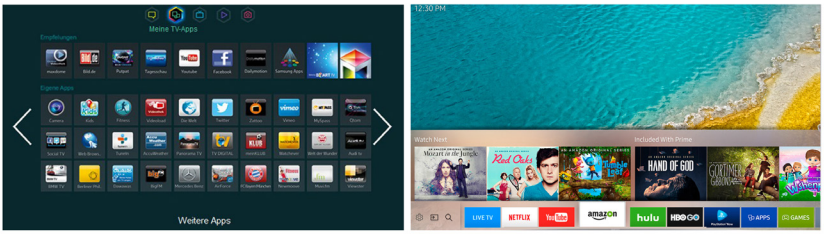


Abbildung 4.12: Samsung Smart Hub von 2015 (links) und 2016 (rechts). Quelle: Samsung.com

Die Gestensteuerung wird über eine im Fernsehgerät eingebaute RGB-Kamera realisiert (für die 2012er Fernsehgeräte) oder über eine externe per USB angeschlossene Zusatzkamera, wie es bei Fernsehgeräten, die ab 2014 produziert wurden, der Fall ist. Durch spezifische Handgesten kann ein Cursor auf dem Bildschirm bewegt und Apps oder Spiele kontrolliert werden.

Abbildung 4.13: Samsung Fernbedienungen von 2015 (links) und 2016 (rechts). Quelle: Samsung.de

Fazit

Die Technologien, die von Samsung angeboten werden, ermöglichen einen schnellen Einstieg in die Programmierung von TV-Apps. Aktuell existiert eine Vielzahl von Tools und SDKs, die zur Entwicklung der Smart-TV angeboten werden, die eher verwirren und zu einer gewissen Unübersichtlichkeit führen.

Außerdem scheint Samsung einer neuen Strategie in Richtung HTML5 und Tizen folgen zu wollen, was die Notwendigkeit sich technisch festzulegen seitens der Entwickler mit sich bringt. APIs, die heute noch benutzt werden können, sind eventuell morgen schon nicht mehr verfügbar. Diese sehr kurzfristigen Veränderungen werfen die Fragen des Wartungsaufwands für den Support einer TV-App innerhalb eines längeren Zeitraums auf. Es kann in manchen Fällen durchaus vorkommen, dass bestimmte vom Benutzer gerne benutzte Apps auf neueren Fernsehern nicht mehr funktionieren.

Bei der Samsung Plattform wechseln über die Jahre schnell die Eingabemöglichkeiten. Das bedeutet, dass z.B. die Gestensteuerung, die in 2012 standardmäßig unterstützt wurde und Hardware-technisch im Fernseher verbaut war, 2014 zu Gunsten einer fernbedienungsbasierten Variante mit einer 3D-Gyroskopsteuerung ersetzt wurde. Analog erging es bestimmten APIs, die nur noch für Samsung Partner zugänglich sind. Zugriffe auf Rohdaten des Kamerasignals, wie es z.B. bei der Skype-App der Fall ist, sind für nicht Samsung Partner-Entwickler nicht mehr möglich, ebenso wie bestimmte Overlay-Funktionalitäten, die per Default auf den für den europäischen Markt produzierten Geräten unterbunden wurden.

webOS / LG

LG-Fernseher stützen sich auf das Betriebssystem webOS und ermöglichen durch ein dediziertes webOS TV SDK (Software Development Kit) die Implementierung von TV-Apps.
Bei webOS handelt es sich um ein Betriebssystem, das ursprünglich vom Handheld-Hersteller Palm entwickelt wurde. Dieses Betriebssystem wurde speziell für den Palm Pre entwickelt und sollte, insbesondere dank der damals neuen Multitasking-Möglichkeit (mehrere Apps können gleichzeitig geöffnet werden und laufen im Hintergrund, sodass der Benutzter von einer App zur anderen „durchblättern" konnte), mit dem iPhone von Apple konkurrieren. Nachdem HP die Firma Palm aufgekauft hatte, wurde webOS für das Tablet HP Touchpad weiterentwickelt. Im Dezember 2011 kündigte HP die Veröffentlichung des Quellcodes unter der Open Source-Lizenz der webOS[29] an.

[29] `http://www.hpwebos.com`

Diese können von der Webseite OpenWebOsProject.org[30], samt Betriebssystem-Images und Emulatoren, aufgerufen und heruntergeladen werden. Im Januar 2013 wurde angekündigt, dass webOS von HP an LG Electronics lizensiert wurde und zwar mit uneingeschränktem Zugriff auf Quellcode und Dokumentation. LG hat im TV-Bereich webOS als Nachfolger von NetCast eingeführt und integriert dieses Betriebssystem in allen neuen LG-Fernsehern. Ein besonderes Merkmal von webOS besteht darin, dass die Apps komplett mit HTML5-Technologien zusammengestellt werden können und der Smart TV Alliance-Spezifikation folgen.

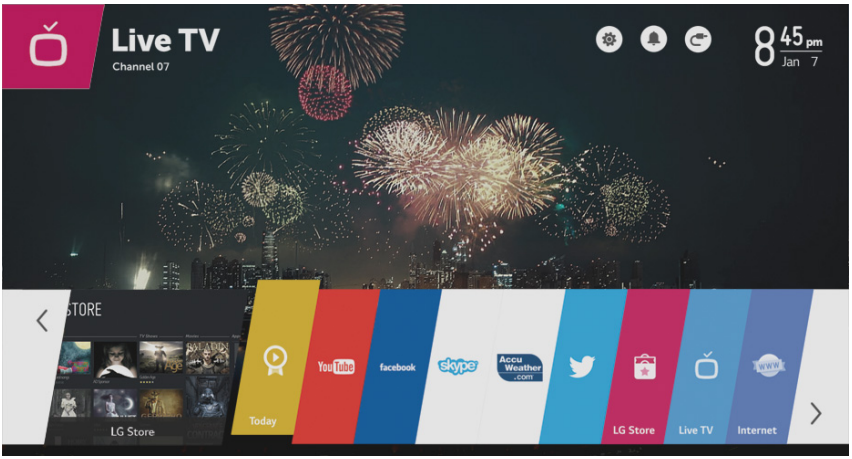


Abbildung 4.14: WebOS 2.0-Schnittstelle auf LG-Geräten. Quelle: LG.com

WebOS unterstützt offiziell (Stand August 2016) nur vier APIs, auch wenn diese komplett auf die HTML5-basierte Version der SDK die Benutzung von bekannten JavaScript Frameworks wie jQuery oder Angular.JS erlaubt.

[30] `http://www.openwebosproject.org`

Benutzerschnittstelle und Interaktionen

Die Benutzerschnittstelle von webOS baut auf dem 10-foot Design auf. Die TV-Apps werden immer in einer untenstehenden Leiste eingeblendet und als Logo mit buntem Hintergrund angezeigt. Diese Leiste kann jederzeit eingeblendet werden und erscheint über dem aktuellen Programm (siehe Abbildung 4.14). Die einzelnen Icons oder Logos dieser Leiste können vom Benutzer mittels der LG Magic Remote selektiert werden. Die LG Magic Remote ist technisch gesehen eine 3D-Gyroskop-basierte Fernbedienung. Sie ermöglicht eine genauere Positionierung und bessere Handhabung als die traditionelle Infrarot-basierte Fernbedienung. Ein Cursor symbolisiert die aktuelle Position der LG Magic Remote und der Benutzer kann alle UI-Elemente über die Fernbedienung selektieren.
Um diese Selektion grafisch deutlicher hervorzuheben, „hüpfen“ für einen kurzen Zeitpunkt die Icons auf und ab. Diese Animationen und Effekte sind charakteristisch für das LG-eigene webOS UI-Design namens Moonstone UI. Es wurde systematisch für alle Apps und Einstellungsmenüs des Fernsehers übernommen und ist fester Bestandteil des webOS SDK[31]. Alternativ können über die LG Magic Remote Spracheingaben zur Kontrolle der Fernsehfunktionen genutzt werden. Der Sprachkenner verwendet die Dragon Natural Speaking-Komponente der Firma Nuance.

Programmierschnittstellen

LG bietet zahlreiche APIs, um Apps, die auf webOS basieren, zu implementieren. Die offiziell unterstützten und vorgeschlagenen APIs für die Entwicklung von web OS-basierten Apps sind:

[31] `https://developer.lge.com/webOSTV/sdk/web-sdk`

- **Web API**

  Die Web API ermöglicht den Entwicklern, alle Javascript-basierten Frameworks mit der Unterstützung der HTML5- und CSS 3-Technologien und Elemente wie Canvas, Animationen, Websockets oder Videoplayer für die Implementierung von webOS-kompatiblen Applikationen einzusetzen.

- **Luna Service API**

  Luna-Dienste sind Programmschnittstellen, die Zugriff auf spezifische Funktionen des LG-Fernsehers (Lautstärke Controller, Kamera und Mikrofon-Zugriff) über eine API zur Verfügung stellen. Diese sind über Javascript aufrufbar.

- **Enyo API**

  Enyo[32] ist ein plattformübergreifendes Javascript Framework zur Erstellung von Apps und stammt ursprünglich vom Mojo-Framework[33] ab, das von Palm und HP unterstützt wurde. Ziel des Enyo Frameworks ist es, ein einheitliches Design und APIs für mobile Desktop- und TV-basierte Applikationen anzubieten. Dadurch lassen sich Enyo.js-basierte Apps für Android oder iOS-Endgeräte implementieren.

- **Connect SDK API**

  Die Connect SDK ist ein Open Source Framework, das eine nahtlose Verbindung zwischen Second Screen und TV-Apps ermöglicht. Die Connect SDK wird für viele Smart-TV-Hersteller angeboten, wie z.B. für Roku[34], Chromecast[35], Apple TV und Amazon Fire TV.

[32] `http://enyojs.com`
[33] `http://mojojs.com`
[34] `https://www.roku.com`
[35] `https://www.google.de/chrome/devices/chromecast`

Parallel zum Smart-TV-Bereich bietet LG über die Produktreihe (LG SuperSign und EZSign TV[36]) im Signage- und Hospitality-Bereich TV-Lösungen zur Anzeige von Informationen, die parallel zu einem abgespielten Video oder TV-Programm angezeigt werden können, an. Damit wird es möglich, neben dem Fernsehbild dynamische Bilder oder Informationen (z.B. im Hotelzimmer, in Cafés oder Restaurants) anzuzeigen. Diese Lösungen werden mit einer Editier-Software angeboten, die es erlaubt, Bilder und Animationen mittels bestimmter Templates einzufügen und auf den jeweiligen Fernsehern einzublenden.
Diese Systeme wurden der Vollständigkeit halber im Folgenden erwähnt, da man sie nur eingeschränkt unter der Gruppe der Smart-TV-Systeme einordnen kann. Die Anzeige der Informationen kann nur innerhalb eines geschlossenen Ökosystems bzw. Netzwerks (z.B. im Restaurant oder Hotelzimmer) gewährleistet werden. Die EZSign TV-Funktionalitäten bieten nur rudimentäre Bildeffekte und Textübergänge an. Es ist nicht möglich über eine Programmiersprache die Signage-Lösungen mit Zusatzfunktionalitäten oder Apps zu bestücken.

Werkzeuge

Das webOS SDK von LG bietet sowohl eine IDE (Eclipse-basierte Programmierumgebung) als auch einen Emulator zum virtuellen Testen der Apps auf PC-Basis. Zusätzlich können sog. Images vom LG-Betriebssystem heruntergeladen und in eine Virtualisierungssoftware wie z.B. VirtualBox[37] geladen werden. Damit sind Entwickler in der Lage, ihre Apps besser testen und debuggen zu können. Über diese Werkzeuge ist eine direkte Verbindung zwischen der IDE und dem Emulator mög-

[36] http://www.lg.com/us/commercial/commercial-tv
[37] https://www.virtualbox.org

lich. HTML5-basierte TV-Apps lassen sich mit jedem herkömmlichen Text-Editor oder via Eclipse-Umgebung implementieren.

Fazit

Mit dem webOS SDK hat sich LG auf die HTML5-basierte Programmierung der Applikationen konzentriert. Genauso wie bei der 2016er Version des *Smart Hub* von Samsung erscheinen die Apps in Form einer Leiste und es existiert keine strikte Trennung von TV-Bild und App-Auswahl. Eine nicht instrumentierte kamerabasierte Gestensteuerung gibt es in diesem Sinne nicht, dafür ist die Gyroskop-basierte Magic Remote in der Lage, Sprachbefehle über ein eingebautes Mikrofon umzusetzen. Analog dem Release-Zyklus von Samsung erscheint jedes Jahr eine neue Version von webOS für LG-Fernsehgeräte.

Apple TV

Apple TV ist eine Multimedia-Set-Top-Box, die seit 2007 von Apple entwickelt wird und das Streaming von Multimediainhalten wie Videos oder Musik von einem iOS-Gerät (iPad, iPhone) zu einem Fernseher ermöglicht. Apple TV-Geräte werden mit dem Betriebssystem tvOS[38], einer modifizierten Form des iOS-Betriebssystems, ausgestattet.

[38] `https://developer.apple.com/tvos`

Abbildung 4.15: Benutzerschnittstelle von tvOS. Quelle: Apple.com

Benutzerschnittstelle und Interaktionen

Die Benutzeroberfläche von Apple TV bzw. tvOS ist sehr stark App-zentriert und kann über zwei Kanäle bedient werden: einerseits über die mitgelieferte Fernbedienung oder andererseits über eine speziell für iOS-basierte mobile Endgeräte entwickelte App. Genau wie bei anderen Smart-TV-Herstellern verfügt die Fernbedienung über ein Mikrofon. Damit kann der Benutzer über Siri (die Sprachsteuerungskomponente von Apple) analog zu anderen iOS-Geräten (z.B. dem iPhone) die gleichen Befehle per Sprache eingeben.
Die Selektion von Menüelementen bzw. Apps erfolgt über die Fernbedienung. Eine Kachel-basierte Anzeige (siehe Abbildung 4.15) bietet eine Übersicht über die bereits installierten Apps. Ähnlich wie bei der LG-Benutzerschnittstelle werden die mit der Fernbedienung selektier-

ten Icons vergrößert dargestellt. Beim Apple TV sind die Apps oder VOD-Angebote innerhalb der Benutzerschnittstelle auf Reihen platziert. Mit der Fernbedienung kann der Benutzer die App-Reihen von links nach rechts und von oben nach unten bewegen und in seiner App-Bibliothek navigieren.

Werkzeuge und Programmierschnittstellen

Auf Apple TV-Geräten können alle iOS- bzw. tvOS-kompatiblen Applikationen über den Apple App Store heruntergeladen werden. Die Apps können mittels XCode[39] entweder in den Programmiersprachen Objective-C oder Swift implementiert werden. Die API-Zugriffe auf hardwarespezifische Funktionen sind äußerst beschränkt. Bei den vorinstallierten Apps handelt es sich meistens um von Apple ausgewählte IP-Content-Anbieter, die aus einem lizenzierten Medienkatalog Musik, Videos und Podcasts kostenlos anbieten, wobei sich das Angebot von Land zu Land verändern kann.

Die zweite Funktionalität des Apple TVs ermöglicht es, persönliche Medieninhalte wie Fotos oder Videos direkt und einfach auf das Fernsehgerät per WLAN zu übertragen und wiedergeben zu können. Dies geschieht mittels einer Streaming-Komponente, die eine 1:1-Kopie des Bildschirms eines iOS-Gerätes überträgt. Mit dieser Funktion namens *AirPlay* ist es möglich, alle im Apple App Store verfügbaren iOS Apps auf das Apple-TV-Gerät „zu spiegeln“ und in einer höheren Auflösung direkt auf den Fernseher anzeigen zu lassen.

[39] `https://developer.apple.com/xcode`

Fazit

Das Apple TV-Gerät ist eher eine limitierte Entertainment Set-Top-Box und in einer Nische angesiedelt als eine wirklich konkurrenzfähige Smart-TV-Lösung. Weder können DVB- oder Fernsehsignale eingespeist werden, noch hat der Benutzer die Möglichkeit, Webinhalte über einen Webbrowser zu laden. Der Zielmarkt von Apple-TV liegt also eher im Video On Demand-Bereich und erlaubt nur die Benutzung von Apps aus dem Apple App Store.

Android TV

Nach einem ersten missglückten Start von Google TV versucht Google mit Android TV ein überarbeitetes Konzept für interaktives Fernsehen anzubieten. Die erste Hardware, die Android TV unterstützte, war der Nexus Player– eine Set-Top-Box- die gemeinsam von Asus und Google entwickelt wurde. Android TV basiert auf Android 5.0 (genannt Lollipop) und weist ein einheitliches Bedienungskonzept über eine Fernbedienung (oder ein Gamepad) und eine Sprachsteuerung auf. Sehr oft werden die Begriffe Google TV, Android-basierte Set-Top-Box und Android TV vermischt und verwechselt, obwohl es sich dabei um völlig unterschiedliche Ausprägungen der Betriebssysteme und Hardware handelt.

Benutzerschnittstelle und Interaktionen

Bei Android TV werden die TV-Sendungen bzw. Inhalte, Videos und Apps mittels einer speziellen Oberfläche dargestellt. Menüstrukturen und Navigationselemente wurden nach dem *10 foot-Design*-Prinzip konzipiert und für eine Bedienung am Fernseher optimiert, was z.B.

bei Android-basierten Set-Top-Boxen nicht der Fall ist.
Parallel zu den Set-Top-Box Varianten von Android TV integrieren immer mehr TV-Gerätehersteller wie z.B. Sony das Android Betriebssystem. Mittlerweile ist Android TV sogar in den Set-Top-Boxen der französischen Internetprovider Free (in der Freebox Mini[40]) und Bouygues (in der BBox Miami[41]) integriert worden. Ähnlich der Benutzerschnittstelle von Apple TV werden installierte Apps als Kacheln angezeigt und sind mit der Fernbedienung per Click oder per Sprache bedienbar. Dieses Bedienungsprinzip wird in Abbildung 4.16 sichtbar.

Abbildung 4.16: Benutzerschnittstelle von Android TV. Quelle: http://www.google.com/nexus/player

40 http://www.free.fr/freebox/freebox-mini.html
41 https://www.bouyguestelecom.fr/blogs/bboxmiami/bbox-miami-sous-android-tv-le-beta-test-public-est-lance

Android-basierte Set-Top-Boxen (also kein Android TV) benutzen dagegen das „normale“ Betriebssystem Android. Dadurch ist die Benutzeroberfläche 1:1 mit der eines Android-Handys identisch. Spezielle Apps können nachinstalliert werden, um eine bessere Bedienbarkeit auf einem Fernsehbildschirm zu gewährleisten. Dadurch wird z.B. die Lesbarkeit der verwendeten Schriftarten erhöht.

Abbildung 4.17: Fernbedienung und Controller bei Nexus Player mit Android TV

Werkzeuge

Entwickler können mittels des Android TV SDK Applikationen für Android TV implementieren. Das Android TV SDK basiert auf Google Android 5.0 (kurz L), versehen mit Zusatzfunktionen und APIs für das Fernsehen und Android TV-Hardware. Android TV-Geräte werden zukünftig mit einer HDMI Through-Funktionalität versehen, sodass es möglich sein wird, über API-Aufrufe, z.B. die Lautstärke des Fernsehgerätes, einzustellen. Um dies zu ermöglichen, werden über das HDMI-Kabel die Signalpakete des eingespeisten Videosignals analysiert. Da Android TV relativ neu ist und auf eine neue Gerätegeneration setzt, kann im Moment nur schwer vorhergesagt werden, ob alle angekündigten Funktionalitäten direkt oder nur in einer limitierten Art und Weise aufrufbar sein werden.

Laut offizieller SDK-Dokumentation können so über spezielle APIs und Klassen (diese sind im Java-Package Namespace *android.tv*[42] zu finden) Informationen über das abgespielte Video- oder TV-Signal abgerufen und eine komplette Kontrolle des Signals für die Entwickler gewährleistet werden. Die Klassen des Java-Pakets *android.tv* weisen darauf hin, dass die Zugriffsmöglichkeiten noch sehr beschränkt sind. In der aktuellen Fassung (Stand: November 2016) des Android TV SDK kann nur der aktuell eingeschaltete Fernsehkanal limitierte Informationen zurückliefern oder beschränkte EPG-Informationen abfragen.
Zur Programmierung von Android TV Apps benötigt der Entwickler die neueste Version des Android Studios[43], eine Netbeans[44]-basierte intelligente Entwicklungsumgebung und das SDK mit Unterstützung von Android L API Level 21. Über den Emulator lässt sich ein Image des Android TV starten und somit können Entwickler, ohne ein physisches Android TV-Gerät zu besitzen, die Apps auf Stabilität und Bedienbarkeit testen.

Amazon Fire TV – Set-Top-Box und Stick

Benutzerschnittstelle

Ein neuer Akteur in der Welt der Android-basierten Set-Top-Boxen ist die im Jahre 2014 erschienene Amazon Fire TV-Lösung. Diese beruht auf dem Fire OS 5.1.1-Betriebssystem, das sich auf Android Lollipop (5.0) stützt (Stand November 2016). Das Betriebssystem Amazon Fire TV darf aus rechtlichen Gründen keinen Google Play Store oder Google Software beinhalten. Trotzdem bleiben alle Grundfunktionalitäten vom

[42] `https://developer.android.com/training/tv/tif/index.html`
[43] `https://developer.android.com/sdk/index.html`
[44] `https://netbeans.org`

Android Betriebssystem erhalten.

Abbildung 4.18: Benutzeroberfläche von Amazon Fire TV. Quelle: http://www.amazon.de

Die Benutzerschnittstelle von Amazon Fire TV unterscheidet sich durch eine andere grafische Aufteilung und Struktur der Bedienoberfläche sehr stark von Android TV. Beim Amazon Fire TV sind die Angebote sehr Amazon-zentrisch, d.h. es werden erst die Amazon-bezogenen Angebote und Dienste auf der Benutzeroberfläche angezeigt.
Die Bedienung von Fire TV kann über drei unterschiedliche Arten von Fernbedienungen erfolgen: Die Amazon Fire TV-Fernbedienung, den Amazon Game Controller und die Amazon Fire TV Voice Remote. Letztere ermöglicht Spracheingaben wie z.B. die Suche nach bestimmten Musik- oder Filmstücken per Sprachbefehl. Amazon Fire TV unterstützt keine natürlichsprachigen Befehle oder längeren Sätze. Dieses Manko wird jedoch mittels des Sprachassistenten Alexa beseitigt. Der Dienst

Alexa, der in Zusammenhang mit der Amazon-Hardware Echo[45] (die u.a. mit 7 Mikrophonen ausgestattet ist) funktioniert, ermöglicht erweiterte Suchmöglichkeiten per Spracheingaben und die Anzeige der Suchergebnisse direkt auf die Benutzerschnittstelle von Amazon Fire TV.

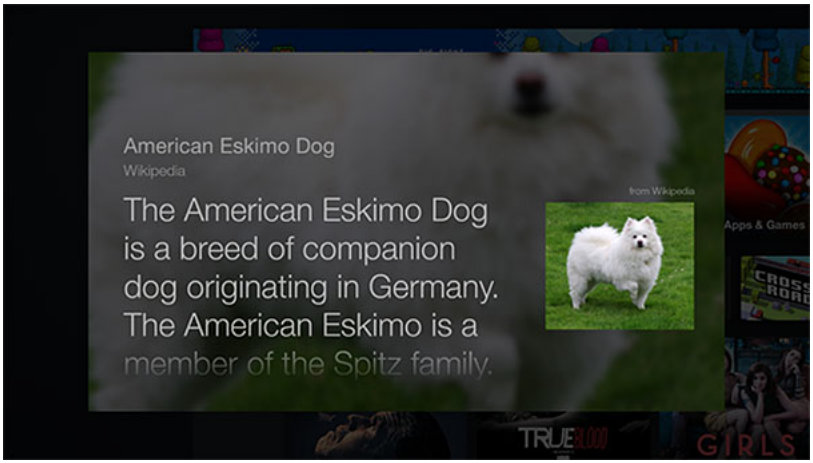


Abbildung 4.19: Beispiel einer Wikipedia-Anfrage über Echo und grafischer Darstellung des Ergebnisses. Quelle: http://www.amazon.com

Eine weitere Einschränkung für Entwickler stellt die Tatsache dar, dass die 3D-Bewegungen der Fernbedienung weder über die Android API noch über die MotionEvent-Klasse abgefragt werden können. Diese Einschränkungen werden auf der Amazon-Seite[46] detailliert beschrieben.

[45] `https://www.amazon.com/Amazon-Echo-Bluetooth-Speaker-with-WiFi-Alexa/dp/B00X4WHP5E`

[46] `http://www.amazon.com/gp/help/customer/display.html?nodeId=201497650`

Werkzeuge

Da Amazon Fire TV auf dem Android-Betriebssystem basiert, können alle Android-Werkzeuge und IDEs (z.B. Android Studio oder Eclipse) für die Entwicklung von Amazon Fire TV-basierten Apps benutzt werden. Der Programmierer muss das Amazon Fire TV SDK zusätzlich zu seinem Android SDK herunterladen.
App-Entwickler sind aufgrund der Rechtsstreitigkeiten zwischen Amazon und Google bei der Programmierung verpflichtet, keine Google APIs in Fire TV Apps zu verwenden, sondern nur die APIs von Amazon. Gleiches gilt bei der Bereitstellung der Apps, die nur über den Amazon eigenen App Store realisiert wird. Die zu benutzende Programmiersprache für die Implementierung von Fire TV Apps muss, bedingt durch die hinterliegende Android-Betriebssystem-Schicht, hauptsächlich Java sein. HTML5 in Kombination mit Javascript kann ebenfalls für die Programmierung von Apps benutzt werden. Dafür muss die Amazon API-spezifische Webview-Komponente (einen Anzeige-Container für Webinhalte) verwendet werden. Diese Webview-Komponente stellt dann dem Benutzer die HTML5-basierte Applikation dar.
Über bestimmte Amazon Fire TV-spezifische Funktionalitäten wie dem X-Ray-Dienst können Zusatzinformationen über den gestreamten Film ermittelt werden. Diese Informationen erscheinen wahlweise entweder direkt innerhalb der Fire TV-Benutzerschnittstelle oder können über ein mobiles Android oder Fire TV-basiertes Endgerät gesichtet werden. Diese Dienste sind nur für Fire TV-Entwickler verfügbar. Fire TV kompilierte Apps können für den Kindle Fire vorbereitet werden, ohne dabei den gesamten Quellcode verändern zu müssen.

Fazit

Das Amazon Fire TV-Gerät bietet interessante Interaktionsmöglichkeiten, um z.B. per Sprache Zusatzinformationen über abgespielte Medien zu erhalten. Die Anbindung an Amazon-bezogenen Dienste beschränkt ein wenig die Möglichkeiten: Benutzer bleiben immer im Amazon-Mikrokosmos und können nicht erschöpfend von den Möglichkeiten der Android-Plattform profitieren. Dies erschwert die Aufgabe der Entwickler, die nicht die Möglichkeit haben, offizielle Android APIs wie z.B. Google Maps zu benutzen, und somit auf Amazon-spezifische APIs(z.B. die dedizierte Amazon Map API) zurückgreifen müssen.
Obwohl Amazon Fire TV auf Android basiert und viele Ähnlichkeiten mit Android TV bzgl. der Interaktionskonzepte (z.B. mit der Spracherkennung über die Fernbedienung) teilt, bleibt das Amazon Fire TV ein eigenständiges Produkt im Amazon Mikrokosmos.

Firefox OS für Fernseher

Firefox OS gehört zu den neuesten Betriebssystemen im Smart-TV-Segment und wurde im Sommer 2015 erstmalig in die VIERA Geräteserie der Firma Panasonic[47] implementiert.

Benutzerschnittstelle

Firefox OS ist ein Betriebssystem, welches genauso für mobile Endgeräte wie auch für andere Geräte geeignet ist. Die Benutzerschnittstelle wird *Gaia* genannt und ist App-zentriert. Beim Start werden die Optionen *Live TV*, *Applications*, *Dashboard*, *Devices* und *Browser* als große

[47] `http://mzl.la/1EUtBMh`

Icons eingeblendet. Im Hintergrund der Benutzerschnittstelle wird ein Videosignal (z.B. vom aktuell abgespielten Video) eingeblendet. Wird eine Option oder App ausgewählt, wird das Hintergrundvideo überlagert (siehe Abbildung 4.20) und die entsprechenden Schaltelemente oder Apps angezeigt. Diese Metapher wird als *Cards*- und *Decks*-Prinzip[48] bezeichnet.

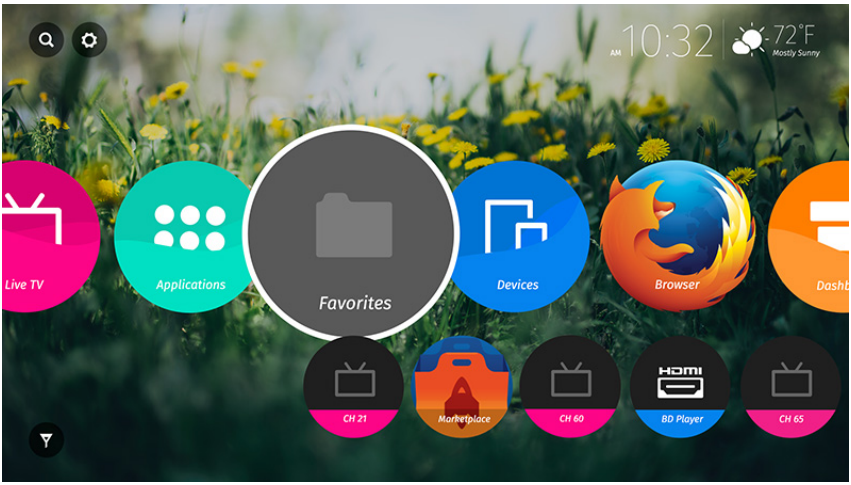


Abbildung 4.20: Benutzerschnittstelle von Firefox OS Quelle: https://www.mozilla.org/de/firefox/os/devices/tv/

Werkzeuge

Firefox OS Apps können nur mittels der HTML5-Programmiersprache entwickelt werden. Dabei kann das Mozilla eigene Werkzeug *Web IDE*[49] benutzt werden. Web IDE ist eine kostenlose Desktop-Software, welche die Programmierung von Web Apps erleichtert und eine Unterstützung beim Debugging anbietet, ohne dabei ein physikalisches Gerät

[48] https://developer.mozilla.org/en-US/Apps/Design/Firefox_OS_TV_UX

[49] https://developer.mozilla.org/en-US/docs/Tools/WebIDE

benutzen zu müssen. Die Web IDE ermöglicht es, unterschiedliche Firefox OS-basierte Geräte zu simulieren, darunter Full-HD und 4K-Fernseher.

Fazit

Firefox OS wurde bisher nur in der VIERA-Geräteserie von Panasonic[50] integriert. Firefox OS folgt dem HTML5-Weg und bietet die Möglichkeit über verschiedene API-Aufrufe, Funktionen des Fernsehgeräts anzusteuern. Da Firefox OS im Fernsehsegment ein relativ junges Betriebssystem ist, ist es unklar, in welcher Form sich dieses Betriebssystem im TV-Bereich in Zukunft durchsetzen wird und ob andere Hersteller dieses Betriebssystem integrieren werden. Der verfolgte Ansatz einer kombinierten Verwendung von HTML5 und Javascript als Technologie zur standardisierten Entwicklung von TV-Apps wie z.B. bei webOS oder Tizen OS lässt hoffen, dass Firefox OS für Fernsehgeräte weiterhin unterstützt und vorangetrieben wird.

Werkzeuge, Vergleich und Analyse der TV-Hardware

Die folgende Tabelle vergleicht die oben vorgestellten Smart-TV-Lösungen auf der Basis von drei Kernaspekten: erstens bzgl. der Programmierung und der API- Zugriffe, welche die Hersteller erlauben, um Apps oder Zusatzprogramme zu integrieren, zweitens bzgl. der Interaktionsformen, die zur Verfügung gestellt werden und drittens bzgl. der Anbindung von Diensten und des Zugangs zu Informationen. Unter HTML5 ist nicht nur die reine Markierungssprache gemeint, son-

[50] `https://firefoxosdevices.org/en/#type:tvs|coming-devices:yes`

dern die Kombination dieser mit der Programmiersprache Javascript und der Stylesheet Sprache CSS3. Die Möglichkeiten der einzelnen Plattformen per API auf Untertitel, d.h. den Text der vom Fernsehsender parallel zu einer Sendung gesendet wird, zuzugreifen und die TV-Apps per Second Screen anzusteuern werden ebenso berücksichtigt. Die Spalte „Spezifische intelligente Dienste“ listet auf, welche TV-Plattformen Zusatzdienste anbieten. Letztere sind meistens schon in der Plattform fest integriert, wie z.B. die Personalisierungskomponente bei Fernsehern von Samsung oder das X-Ray bzw. Echo beim Amazon Fire TV. Diese Dienste sind oft proprietär und lassen sich teilweise über APIs aufrufen. Die Informationen der Tabelle sind teils aus den Herstellerspezifikationen entnommen und durch eigenständige selbst durchgeführte Tests konsolidiert worden. Informationen, die weder von den Tests noch von den Spezifikationen ermittelt werden konnten, wurden mit der Abkürzung k.A. (keine Angabe[n]) markiert.

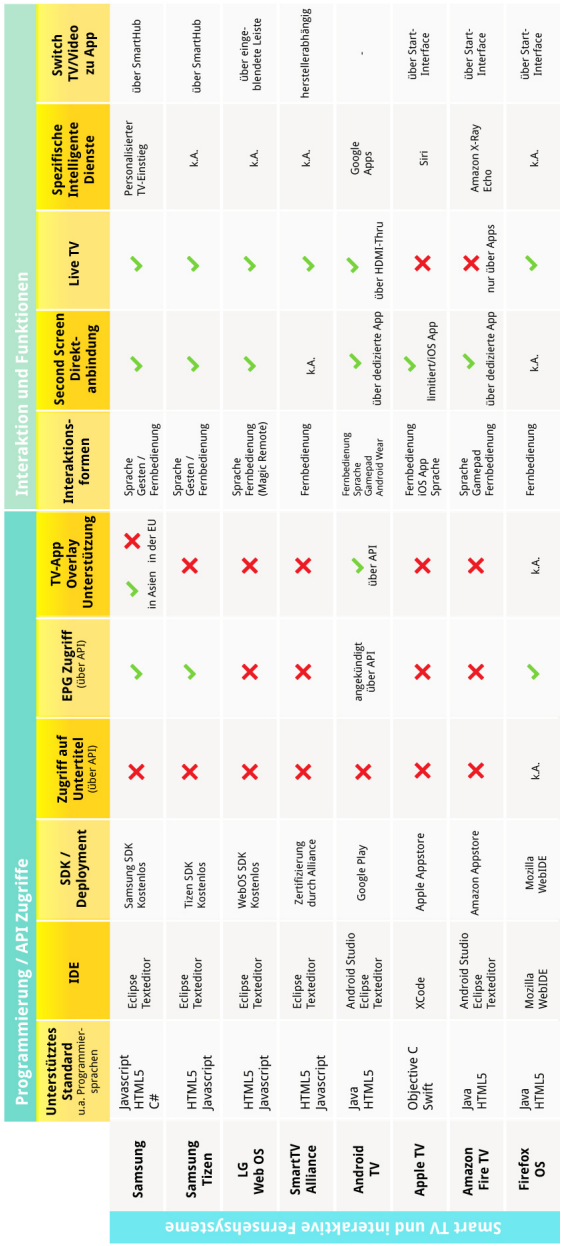


Abbildung 4.21: Vergleichstabelle von Smart-TV Systemen

Schlussfolgerung und Bewertung

Nach der technischen Betrachtung der verschiedenen Fernsehbetriebssysteme werden im folgenden Abschnitt die Vor- und Nachteile jedes Ansatzes aufgelistet und seine potentielle Verwendbarkeit beim zu Grunde liegenden Vorhaben dieser Arbeit, der Erstellung eines intelligenten semantisch-basierten Fernsehsystems, überprüft.

Interaktionen und Benutzerschnittstellen

Die vorgestellten Systeme lehnen sich an Leitlinien an, die das *10 foot-Design*-Paradigma als Designansatz verfolgen. Die Benutzerschnittstellen, inklusive der Schaltelemente, wurden für eine Bedienung auf einem größeren Bildschirm konzipiert. Dieser Aspekt ist bei der Gestaltung der Steuerungselemente und der Größe der benutzten Schriftarten wichtig. Dadurch wird eine bessere Lesbarkeit der UI-Elemente auf dem TV-Bildschirm erreicht.
Bei allen Systemen wird die Auswahl der Apps klar vom abgespielten Video oder dem Live-Fernsehsignal getrennt. Entweder wird das aktuelle Fernsehbild verkleinert und auf einer speziellen Oberfläche samt Apps angezeigt oder über eine Fußleiste, die über das Fernsehbild eingeblendet wird, wie es bei LG- oder Android-TV-Geräten der Fall ist. Die Einblendungstechniken und Auswahlbildschirme von Apps können sich von Version zu Version ändern, daher müssen Entwickler diese Veränderung bei der Konzipierung von Apps oder Auswahlschaltelementen berücksichtigen.
Beim Starten einer App ereignet sich bei allen Systemen ein Interaktionsbruch. Dieser liegt zwischen dem Moment der Auswahl der App und dem Zeitpunkt, an dem diese tatsächlich gestartet wird. Bei allen

Systemen wird das Fernsehbild komplett abgeschaltet. Eine parallele Verwendung von Apps und dem Live-Fernsehsignal ist zum heutigen Zeitpunkt nicht möglich, auch wenn manche herstellerspezifischen Menüpunkte und Einblendungen dies grafisch realisieren, wie es die nächste Abbildung 4.22 zeigt. App-Entwickler bekommen jedoch keine Möglichkeiten, solche Einblendungseffekte innerhalb von Apps zu integrieren.

Bei der Interaktionsform bleibt die Fernbedienung das Hauptsteuerungselement, auch wenn Hersteller wie Samsung über eine kameragestützte Gestensteuerung versucht haben, eine neue Form der Interaktivität ins Spiel zu bringen. Der erwartete Durchbruch dieser Interaktionsform blieb aus. Dies führte dazu, dass die kamerabasierte Gestensteuerung sehr schnell wieder obsolet geworden ist, weil die benutzte Technologie zur Erkennung der Hand rudimentär und nur auf einer RGB-Pixelwerteanalyse beruhte, im Gegensatz zu Tiefenbildkamera-basierten Lösungen.

Ein Konsens scheint sich bzgl. der Verwendung geeigneter Fernbedienungstypen gebildet zu haben. Die Benutzung einer 3D-Gyroskop-basierten Fernbedienung, die alle Interaktionsparadigmen der Computermaus in etwas abgewandelter Form übernimmt, scheint sich als Standard bei der TV-Hardware durchgesetzt zu haben. Mittlerweile wird dieser Typ von Fernbedienung bei allen Herstellern standardmäßig verwendet.

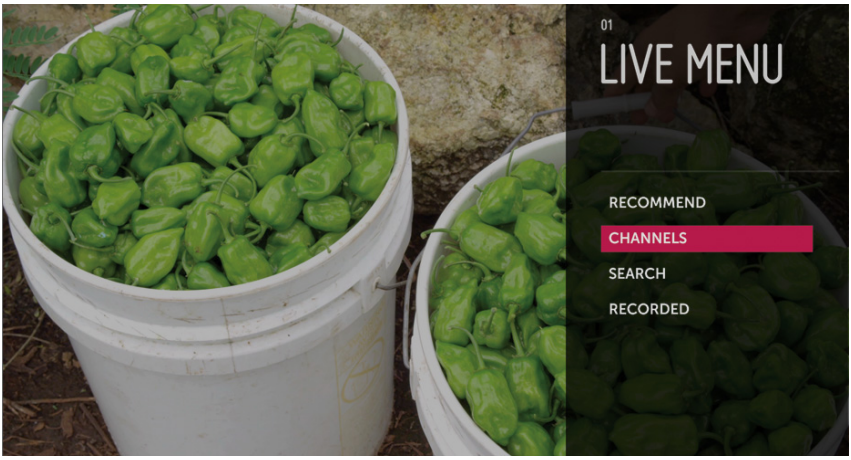


Abbildung 4.22: Beispiel einer Überblendung von Schaltelementen über Live-TV Signal. Quelle: LG

Ein neuer Trend ist die serienmäßige Ausstattung der Fernbedienungen mit einem Mikrofon, damit Zuschauer per Sprache mit dem Fernseher oder dem Set-Top-Box-System interagieren können. Die Erkennung ist Cloud-basiert und wird meistens von der Firma Nuance angeboten, wie dies bei Fernsehern der Marke LG und Samsung der Fall ist. Parallel dazu erlauben diese Systeme eine Ansteuerung des Fernsehers über dedizierte Second Screen-Apps. Diese Apps sind miteinander nicht kompatibel und jeder Hersteller benutzt eine spezifische Implementierung, um seine Fernsehgeräte via Tablett anzusteuern, auch wenn Initiativen existieren, diese Schnittstelle softwaretechnisch zu vereinheitlichen, wie z.B. durch Connect SDK[51].

[51] `http://connectsdk.com/apis/`

Entwicklungssprachen und Werkzeuge

Alle Systeme bieten die Möglichkeit, sog. Smart TV-Apps zu entwickeln und diese später über einen dedizierten Store anzubieten. Die Frage, welche Programmiersprache verwendet werden sollte, scheint dagegen mittlerweile beantwortet zu sein. Die Smart TV Alliance (STA) und das HbbTV-Konsortium bevorzugen eindeutig HTML5 als Standard zur Erstellung von Smart TV Apps. Die Tendenz diesem Trend zu folgen, lässt sich bei Fernsehherstellern, wie LG und dessen Verwendung von webOS, welches das ältere Netcast abgelöst hat oder beim Strategiewechsel von Samsung hin zum Betriebssystem Tizen, das nur noch HTML5-basierte Apps unterstützt, ausmachen.
Die Implementierung dieser HTML5-basierten Apps erfolgt über kostenlose herunterladbare Entwicklungsumgebungen, die entweder auf Eclipse basieren oder mit Open Source-Projekten realisiert wurden.
Als einziger Hersteller unterstützt Samsung zusätzlich die Flash- und AIR TV Laufzeitumgebungs-Technologie. (AIR TV steht für: Adobe Integrated Runtime für interaktive Fernsehapplikationen). Folglich können Applikationen mit der Programmiersprache Actionscript und mit Flash oder Flex implementiert werden und die daraus resultierende kompilierte Datei als TV-App gepackt werden. Parallel dazu erlaubt Samsung die Implementierung in C# und das Abspielen von 3D-Spielen mittels der Unity-Engine[52].
Bei Amazon Fire TV und Android TV kann die Implementierung der App über das Android SDK realisiert werden. Hierbei wird die Programmiersprache Java benutzt, um die Apps zu programmieren. Obwohl beide Systeme auf Android basieren, existieren kleinere Unterschiede,

[52] `http://docs.unity3d.com/Manual/samsungtv-gettingstarted.html`

wie z.B. bei der Verwendung der APIs. Ein besonderes Merkmal der vorgestellten SDKs ist es, dass diese sogenannten Emulatoren über mitgelieferte Images der Fernsehbetriebssysteme anbieten. Somit ist es Entwicklern möglich, ihre Apps zu testen, ohne das entsprechende Fernsehgerät zu besitzen. Leider ist die Funktionalität zwar für in sich geschlossene Apps sinnvoll, jedoch bieten die Emulatoren keine Möglichkeit, komplexe Interaktionen, wie Sprach- oder 3D-Gesten-Eingaben der Fernbedienung, zu simulieren.

Einschränkungen und Problematiken

Schnittstellen

Obwohl alle APIs und SDK der vorgestellten Systeme sehr offen scheinen und viele Möglichkeiten bieten, gibt es zahlreiche Funktionalitäten des Fernsehers, die nur beschränkt verfügbar sind. Dies gilt besonders für den Zugriff auf EPG-und MPEG-Daten aus dem Demultiplexer. Keines der Systeme erlaubt es, über APIs an solche Informationen zu gelangen. Bei der mittlerweile von LG eingestellten NetCast API war es möglich festzustellen, welchen Fernsehkanal der Benutzer ausgewählt hatte und so auf die EPG-Information des Kanals zuzugreifen. Dies ist in der aktuellen Version von webOS nicht mehr möglich. Komplexere Aufgaben wie z.B. eine Bildanalyse (via OCR oder Gesichtserkennung) des Live-TV-Bildes sind nicht möglich, da keiner der APIs einen Zugriff auf das „Fernsehbild“ als Bitmap-basierte Pixelstruktur anbietet (d.h. das Fernsehbild lässt sich als Bildinformation nicht extrahieren). Zugleich bedeutet dies, dass keine Analyse z.B. über OpenCV oder OCR-Verfahren direkt auf dem Gerät möglich ist. Die von den Herstellern verfolgte HTML5-Strategie trennt immer mehr das eigentliche Betriebssystem von der App-Interaktionsschicht. Alle Kernfunktionen

des Fernsehers können nicht mehr direkt angesteuert und aufgerufen werden. Dies stellt aus Entwicklersicht eine „Bremse“ und einen wesentlichen Rückschritt bei der Implementierung von TV-Apps dar.
Eine weitere Erkenntnis aus der durchgeführten technischen Analyse ist, dass spezifische Funktionen von Fernsehgeräten nur für bestimmte ausgewählte und dafür zahlende Entwickler oder Firmen zur Verfügung gestellt werden. Samsung bietet z.B. im Rahmen von Kooperationen mit Kunden Zugriffe auf sog. private APIs, die offiziell nicht dokumentiert sind und die auf Hardware- und Softwarekomponenten des Gerätes beruhen. Dies ist z.B. der Fall bei der Skype-App, die Zugriff auf den Kamera-Stream erlaubt, obwohl es keine dokumentierte API dafür gibt.

Overlay-Problematik

Ein weiteres Problem in Hinblick auf die Implementierung von Swoozy ist, dass insbesondere bei den Set-Top-Boxen, die auf dem Markt erhältliche Hardware zurzeit Funktionen, wie das Einspeisen von externen Fernsehsignalen über die HDMI-Through-Funktion (das Videosignal, wird ohne Veränderung über die Set-Top-Box weitergeleitet und in der Benutzerschnittstelle des Systems angezeigt) nicht mehr unterstützt wird. Weder beim Amazon Fire TV noch bei neueren Android TV Set-Top-Boxen wird diese Funktion angeboten, obwohl diese beim eingestellten Google TV noch vorhanden war und in der aktuellen Version der Android TV API beschrieben wird. Es ist für Entwickler unklar, ob eine Möglichkeit besteht, ein DVB-Videosignal 1:1 in eine App einzuspeisen und dieses zu überlagern, auch wenn die HDMI-Spezifikation und Android TV APIs dies technisch unterstützen würden.
Ein weiteres technisches Problem stellt die Implementierung von *Over-*

lays bzw. die grafische Einbettung von Einblendungen über das ausgestrahlte Videobild dar. Unter *Overlay* versteht man die Möglichkeit, das aktuell laufende Fernsehbild mit einer halbtransparenten Grafik zu überlagern. Dies wird manchmal als *Ticker-Applikation*[53] bezeichnet. Bei Samsung ist es z.B. nicht möglich, diese Funktionalitäten auf EU-Fernsehmodelle zu implementieren, hingegen ist die Benutzung dieser Funktionalität auf Geräten, die auf dem amerikanischen oder asiatischen Markt verkauft werden, erlaubt. Die Erklärung für diese Unterschiede liegt in der Existenz von rechtlichen Problemen bei der Verwendung des Overlay-Prinzips in der EU. DVB-Signale inklusive Videos dürfen ohne eine explizite Vereinbarung zwischen Fernsehsender und App-Entwickler nicht mit zusätzlichen Grafiken oder Inhalten versehen werden. Wird eine solche Vereinbarung dem Fernsehhersteller nicht vorgelegt, wird die App vom Store des jeweiligen Fernsehherstellers nicht akzeptiert[54].

Diese „Overlay-Problematik" wird in einem von der EU-publizierten Dokument *Grünbuch der EU-Kommission über die Vorbereitung auf die vollständige Konvergenz der audiovisuellen Welt: Wachstum, Schöpfung und Werte*[55] beschrieben. Dieses Dokument analysiert den Wandel der audiovisuellen Medienlandschaft und identifiziert Potenziale der aktuellen Medientechnologien wie z.B. OTT (engl. Over-The-Top)-TV (siehe Kapitel 1) oder der Set-Top-Boxen. Darin teilte die EU Kommission ihre „*Bedenken hinsichtlich kommerzieller Einblendungen (Overlays) [...], die innerhalb der linearen Dienste der Rundfunkveranstalter angezeigt werden*" mit. In einer Stellungnahme der ARD-Gremienvorsitzenden Konferenz

[53] https://www.youtube.com/watch?v=N3TKOMrYaJk

[54] http://www.dbuschke.de/blog/beitrag-ueber-ein-misslungenes-projekt-der-hausautomation/

[55] http://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:52013DC0231

(GVK)[56] wurde diese Frage der Einblendung aufgegriffen und wie folgt beantwortet: *„Kommerzielle Ein- und Überblendungen sollten nur mit Zustimmung des Inhalteanbieters erfolgen können. Zudem sollten nutzerinitiierte Einblendungen z.B. von Social-Media-Diensten möglich sein.“*
Versucht man trotzdem per Programmierung über das aktuell abgespielte Fernsehbild etwas einzublenden, so erscheint z.B. bei Samsung Geräten lediglich ein schwarzes Fenster und nur die Audiospur des ausgewählten Kanals ist zu hören.
Bei anderen Systemen wird das Overlay-Prinzip innerhalb von selbst geschriebenen Apps nicht unterstützt, wie z.B. bei LG. Dort ist es im webOS System nicht möglich über eine API das aktuelle Fernsehbild (auch nicht mit reduzierter Größe) in einer App zu benutzen[57] [58].

Fazit und Analyse

Auch wenn die Smart TV Alliance und viele Hersteller sich bemühen eine einheitliche technologische Spezifikation bereitzustellen, bestehen auf Hardware- und Software-Ebene noch sehr viele Diskrepanzen. Die permanenten Spezifikationsveränderungen der Hersteller verhindern, dass Entwicklern eine einheitliche und feste Grundlage für die Erstellung von TV-Apps zur Verfügung steht.
Die Unterschiede innerhalb von Produktlinien des gleichen Herstellers lassen keine langfristige Entwicklungsperspektive zu, weder für den Support noch für die Erweiterung von TV-Apps. Bei Samsung wird zukünftig Tizen mit HTML5 vorangetrieben. Bei Philips, obwohl Mitglied

[56] http://www.ard.de/download/378296/index.pdf
[57] http://developer.lge.com/community/forums/RetrieveForumContent.dev?detailContsId=FC22134225
[58] http://developer.lge.com/community/forums/RetrieveForumContent.dev?detailContsId=FC02171035

der Smart TV Alliance, setzt man hingegen verstärkt das Android-Betriebssystem auf der neuen Hardware ein. Aus Entwicklersicht ist es unklar, welcher Strategie gefolgt wird und ob letztendlich nur HTML5-basierte oder doch hybride Android-basierte Apps unterstützt werden. Unklar ist, ob diese Apps immer noch aus einem dedizierten Store des jeweiligen Fernsehherstellers heruntergeladen werden müssen oder direkt über den Android Play Store zur Verfügung gestellt werden.
Die benutzten, sehr heterogenen Technologien zeigen, dass der Smart-TV-Markt ziemlich segmentiert und App-getrieben ist. Selbst einfache Funktionalitäten, wie die Anzeige eines TV-Bildes innerhalb einer App oder die Extraktion von EPG-Informationen, werden von den meisten Herstellern nicht mehr angeboten. Diese Situation, auch wenn sie wahrscheinlich nur vorübergehender Natur ist, stellt für Entwickler ein echtes Hindernis bei der Smart TV App-Entwicklung dar.
Selbst wenn den Entwicklern Lösungen, wie frei verfügbare Testumgebungen, Frameworks und Emulatoren zur Verfügung gestellt und somit die Entwicklungszeiten reduziert werden können, besteht die Gefahr, dass aufgrund der von den Herstellern vorgenommenen andauernden Spezifikationsveränderungen die Möglichkeiten der Smart-TV-Apps sehr eingeschränkt bleiben werden. Neue Interaktionsparadigmen können nur schwer innerhalb von solchen Apps realisiert werden. Genau wie im PC-Bereich ändern sich im Smart-TV-Bereich ständig die Hard- und Software-Spezifikationen. Bestehende und bereits benutzte Betriebssysteme und Benutzerschnittstellenkonzepte können zu Ungunsten der Entwickler unangekündigt eingestellt und ersetzt werden.
Die Bedienoberfläche spielt bei einem Fernsehsystem eine zentrale Rolle. Die detaillierte Analyse der verschiedenen Systeme deutet darauf hin, dass die Industrie zukünftig verstärkt auf HTML5-basierte

Visualisierungstechniken setzten wird. Hierbei besteht ein Konsens bzgl. der Verwendung des 10-foot Designs als wesentliche Grundlage für die Gestaltung und Konzipierung von Bedienoberflächen. Dies wird insbesondere bei den angebotenen Programmierungsumgebungen und SDKs der verschiedenen Hersteller sichtbar. Diese SDKs helfen Entwicklern Oberflächen korrekt zu gestalten. Vorgefertigte und durchdesignte Steuerungselemente wie Listen, Buttons oder Menü-Komponenten können 1:1 übernommen und z.B. über CSS (Cascading Style Sheets[59]) nachträglich farblich angepasst werden. Somit wird gewährleistet, dass einerseits das Look'n'Feel der realisierten App mit dem der Zielplattform harmoniert, und dass andererseits die Leitlinien des 10-foot Design eingehalten werden. Bei LG, Samsung oder der Smart TV Alliance gehören diese Aspekte zu den Grundempfehlungen bei der Konzipierung von TV-Apps.

Durch die durchgehenden HTML5-Unterstützungen können bewährte Visualisierungs- und Javascript-Bibliotheken wie D3.js, jQuery und Angular JS benutzt und integriert werden. Als einziger Hersteller unterstützt Samsung die Adobe AIR-Plattform. Somit können Spiele, die mittels der Programmiersprache Actionscript implementiert wurden, auf Fernseher portiert werden. Heutige Smart-TV Systeme besitzen Einschränkungen bezüglich der APIs und bieten teilweise keinen kompletten Zugriff auf alle Funktionen des Fernsehers. Wegen dieser Einschränkungen können keine Bild- oder DVB-Datenströme direkt von der Hardware des Fernsehers als Bilddatei oder Videosequenz extrahiert werden. Ähnlich sieht es beim Zugriff auf Kamerabilddaten aus, die nur für ausgewählte Firmen wie z.B. Skype zur Verfügung gestellt werden. Auch wenn Sprachinteraktionen als Eingabemöglichkeit von

[59] `http://www.w3.org/Style/CSS`

den Herstellern angeboten werden, bedeutet dies nicht, dass Entwickler diese innerhalb einer App frei benutzen und z.B. eigene Kommandos programmieren oder Grammatiken angeben können. Diese Situation geht so weit, dass viele Funktionen und Verarbeitungsschritte, die teilweise direkt auf der Hardware des Fernsehers realisierbar wären (wie die Extraktion von EPG-Daten, die sowieso automatisch erfolgen muss), nur über Umwege realisiert werden können. Eine weitere Hürde bei der Realisierung einer komplett integrierten App mit Fernseh- bzw. Video-Livebild stellen das Overlay-Problem und das Problem des Interaktionsbruchs dar. Die gängigsten Fernsehbetriebssysteme, inklusive der von Set-Top-Boxen, sind so implementiert, dass der Benutzer bei der Auswahl einer App immer beim Hauptvideo starten muss. Auch wenn Zwischenlösungen existieren, wie z.B. die App Auswahlleiste bei webOS von LG oder bei Android TV mit den Notifications, die partiell das Hauptvideo überdecken, ist es über frei zugängliche APIs nicht möglich, grafische selbstdefinierte Elemente über das Videobild zu legen. Diese Situation beweist, dass es heute immer noch einen klaren Interaktionsbruch zwischen den Aktivitäten *„Video anschauen“* und *„App benutzen“* am Fernseher gibt. Das App-Paradigma hat sich bei den Betriebssystemen der Fernseher durchgesetzt. Durch die Initiativen von webOS oder der Smart TV Alliance, die Programmiersprache HTML5 als Standardprogrammiersprache für TV-Apps zu etablieren, wird Entwicklern die Möglichkeit gegeben, einheitliche und homogene Apps zu implementieren. Dabei können Javascript-basierte Frameworks wie z.B. jQuery oder Enyo.js mühelos integriert werden. Komplexere und visuell anspruchsvollere Apps (wie z.B. Spiele) können mittels anderer Programmiersprachen wie Flex, Flash oder mit Unity in C# programmiert werden, wie dies bei Samsung-Geräten der Fall ist. Auch wenn Hersteller sehr viele APIs und

Entwicklungsumgebungen für ihre Plattformen anbieten, bleibt der Zugriff auf die Kernfunktionen der Fernsehbetriebssysteme verwehrt. Dieser ist nur nach Abschluss einer kostenpflichtigen Lizensierungsvereinbarung zwischen Hersteller und App-Entwickler möglich. Diese Situation schränkt die technische Möglichkeit auf bestimmte Hard- und Softwarekomponenten des TV-Geräts zuzugreifen ein und bremst die Entwicklung von Systemen, die versuchen, eine parallele Benutzung eines Live-Fernsehbilds mit einer grafischen Überblendung zu ermöglichen.

Zusammenfassend muss festgehalten werden, dass die angestrebte Swoozy-Lösung zur Realisierung eines interaktiven semantischen Fernsehsystems mit den aktuell verfügbaren Smart-TV-Technologien nur begrenzt und mit sehr starken Einschränkungen in Bezug auf Interaktion und grafischer Benutzerschnittstelle umzusetzen wäre. Aus diesem Grund wurde, wie es in Kapitel 6 skizziert wird, zur Überbrückung bis die aktuellen Smart-TV-Systeme alle grundlegenden Funktionalitäten über offizielle APIs anbieten, eine hybride und verteilte Architektur ausgewählt, die den DVB-T-Empfang, die Analyse der Streams und die parallele Anzeige von Zusatzinformationen in sich vereint.

Erste Schritte in Richtung semantisches Fernsehen

Das Wort „Fernsehen“ ist eine Zusammenstellung aus den Begriffen „fern“ und „sehen“ und wurde erstmals im Jahre 1891 von Raphael Eduard Liesegang in seiner Veröffentlichung *„Das Phototel: Beiträge zum Problem des elektrischen Fernsehens“* benutzt [Mie39]. Der eigentliche Begriff „Television“ wurde das erste Mal im Jahre 1900 von Constantin Perskyi während eines internationalen Kongresses über Strom (franz. Congrès international d’électricité) im Rahmen der Weltausstellung in Paris[1] verwendet. Dieser Begriff ersetzte rasch andere, wie z.B. den Téléctroscope[2], den Téléphote[3] oder die Bildtelegraphie bzw. die Radio Vision [Abr87] [Mie39] [Pro07]. Folgende Abbildung 5.1 aus der Webseite eines Postkartensammlers[4] zeigt eine Vision des Fernsehens aus

[1] http://histv2.free.fr/19/perskyi.htm
[2] http://www.archivespasdecalais.fr/Anniversaires/9-octobre-1842-naissance-de-Constantin-Senlecq-pionnier-de-la-television
[3] http://ilyaunsiecle.blog.lemonde.fr/2010/02/08/8-fevrier-1910-voulez-vous-regarder-le-telephote-ou-la-television/
[4] http://www.tvhistory.tv/1890s\%20Victorian\%20Trade\%20Card.htm

dem Jahre 1900. Diese suggerierte eine baldige technische Möglichkeit, in Echtzeit und multimodal mit einem entfernten Ereignis interagieren zu können.

Abbildung 5.1: En l'an 2000 - Postkarte

Die Gesellschaft für Interdisziplinäre Bildwissenschaft der Universität Tübingen[5] gibt eine weitere Definition von Fernsehen bzw. der Benutzung des Fernsehers: *„Eine weitere wesentliche Differenz zum Kino ist die Möglichkeit der Live-Übertragung über das Fernsehen, das eine aktuelle Teilhabe an einem entfernten Ereignis ermöglicht [...] Das Programm bildet somit die Schnittstelle der Mensch-Maschine-Anordnung und unterscheidet das Dispositiv Fernsehen von anderen Bildmedien, wie z.B. dem Computer. Durch die implizite Möglichkeit der Programmwahl durch den Rezipienten wird der Zuschauer als Subjekt - anders als im Kino - ein mitbestimmender Faktor innerhalb des Dispositivs“*.

[5] `http://www.gib.uni-tuebingen.de/netzwerk/glossar/index.php?title=Fernsehen`

Diese Beispiele aus der Geschichte des Fernsehens und die letzte Definition verdeutlichen, dass das Fernsehen per se die Schnittstelle zwischen einem entfernten Geschehnis und einem Teilnehmer ist, wobei der Zuschauer durch seine Interaktion und Vorlieben, diese Geschehnisse auswählen kann und in der Lage ist, wie in den letzten Kapiteln beschrieben, über verschiedene Interaktionsmuster und Modalitäten, Zusatzinformationen zu diesen Ereignissen aufzurufen.
Mit der Entwicklung des Webs und den Informationstechniken sind in den letzten Jahren zahlreiche innovative Technologien und Verfahren entwickelt worden, z.B. die Bild-, Video- oder Textanalyse, die nun eine semantische Verarbeitung und Extraktion von Medienobjekten erlauben und den Weg zum semantischen Fernsehen öffnen.
Im letzten Kapitel wurden diese Verfahren und die verschiedenen interaktiven Fernseherkonfigurationen vorgestellt, die dem Benutzer mittels einer Fernbedienung ermöglichen, erweiterte Eingabemöglichkeiten zu nutzen, um eine Suche im Web über das Fernsehgerät durchzuführen. Diese detaillierte Analyse zeigt, dass die Komponenten der semantischen Analyse, der Annotation der Daten, der Anzeige der Suchergebnisse und der Implementierung der Fernsehbenutzerschnittstellen eigenständige Softwaremodule und theoretisch über geeignete Programmierschnittstellen (API) oder REST-Schnittstellen nahtlos kombinierbar und aufrufbar sind.
Die Idee, das Fernsehen mit Inhalten aus dem Web bzw. Semantic Web zu verknüpfen, wurde in mehreren Forschungsprojekten erforscht und prototypisch realisiert.
Der folgende Abschnitt gibt einen Überblick über diese Projekte, die einen Bezug zum Fernsehen und der parallelen Benutzung des Semantic Web besitzen. Dafür werden nicht nur die angewandten Technologien, sondern auch die Interaktionsparadigmen, die Informationsex-

traktionsverfahren und die Verbindung von Video mit dem Web näher vorgestellt. Obwohl all diese Projekte den Fernseher als Zielplattformen betrachten, muss zwischen der tatsächlichen Echtzeitinteraktion mit den Live-ausgestrahlten Sendungen und der Interaktion mit Videos aus einem Archiv wie z.B. Video on Demand unterschieden werden.

Forschungsprojekte mit Bezug zum Fernsehen und der semantischen Verarbeitung von TV-Programmen

In diesem Abschnitt werden abgeschlossene Forschungsprojekte präsentiert, die einen direkten Bezug zum Fernsehen besitzen, und Teilkomponenten, die zur Realisierung eines semantischen Fernsehens beitragen. Diese verschiedenen Projekte werden chronologisch dargestellt. Fokussierter wird das THESEUS-Projekt beschrieben, da es als technische und wissenschaftliche Grundlage für die Realisierung von Swoozy dient. Der letzte Abschnitt gibt in Form einer grafischen Zusammenfassung, einen Überblick über die vorgestellten Projekte und stellt somit die Abgrenzung zu Swoozy dar.

EMBASSI

Das EMBASSI (Elektronische Multimediale Bedien- und Service-Assistenz)-Projekt[6] [HKS01] [HK02a], das vom Bundesministerium für Forschung und Technologie von 1999 bis 2003 gefördert wurde, hatte als Ziel, den Nutzer bei der Bedienung von Alltagstechnologien zu unterstützen. Das Projekt wurde von mehreren Industriepartnern unterstützt,

[6] https://web.archive.org/web/20041228182912/http://www.embassi.de

u.a. durch die Grundig Fernseh-Video-Produkte und Systeme GmbH, Loewe Opta GmbH und weitere Firmen der Unterhaltungsbranche wie Sony International [HKS01] [HS00] [HK02b] [RV08b] [Zei11]. Im Rahmen des Projektes wurden Demonstratoren für drei Anwendungsfälle realisiert: eine Fahrerassistenz für die Steuerung von Infotainmentsystemen im Auto, einen Bedienassistenten für die Interaktion mit Bank- und Verkaufsautomaten und eine multimodale Steuerung von vernetzten Multimedia- und Infotainment-Systemen, u.a. eines Fernsehgeräts. Damit der Benutzer mit diesen Systemen intuitiv interagieren konnte, wurden als Interaktionsmöglichkeit multimodale Verfahren wie z.B. Sprach- und Gestenerkennung (Kamera-basiert und Stift-basiert) eingesetzt. Die Verwendung von multimodalen Verfahren war dadurch motiviert, dass den Benutzern mit physischen Einschränkungen die Möglichkeiten gegeben werden sollten, u.a. durch verschiedene technische Hilfen (z.B. ein Braille-Pad oder ein Joystick), mit komplexeren Systemen, wie z.B. öffentlichen Terminals, zu interagieren.

Speziell im Home Entertainment Bereich zeigte EMBASSI seine Stärke. Ein an das Fernsehgerät angebrachtes Mikrophone-Array ermöglichte die Erfassung der Spracheingabe des Zuschauers. Der Zuschauer konnte Sätze wie *„Ich möchte heute Abend einen Krimi aufnehmen!"* oder " *ZDF einschalten!"* formulieren oder mit einer kombinierten Interaktion aus Touch- und Sprach-Interaktion, z.B. mit dem Sprachkommando „einschalten!" und dem Klick auf das ZDF-Logo, mit dem System interagieren. Nach einer Spracherkennung wurde die Eingabe von einem Dialogmanager analysiert und verarbeitet. Je nach Kommando und nach einer semantischen Analyse des gesprochenen Satzes wurden die passenden Dienste (z.B. EPG) aufgerufen.

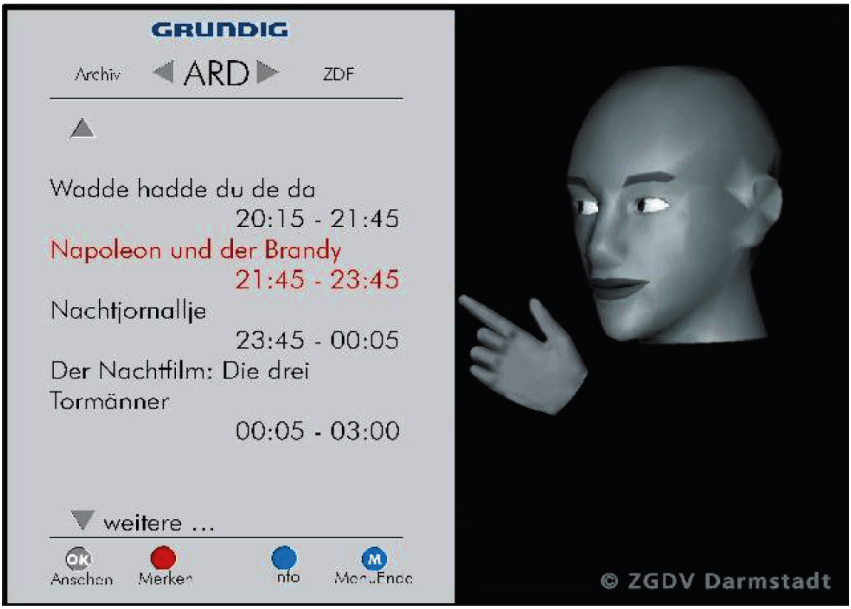


Abbildung 5.2: Anzeige von EPG und dem Avatar innerhalb der Fernsehbenutzerschnittstelle

Die Antworten und das Feedback des Systems wurden innerhalb der TV-Benutzerschnittstelle mittels eines virtuellen 3D-Avatars (siehe Abbildung 5.2) realisiert. Dieser teilte dem Zuschauer die Ergebnisse seiner Suche durch eine synthetische Sprachausgabe mit. Die Abbildung 5.2 zeigt die 3D-Avatar-basierte grafische Ausgabe nach einer EPG-Abfrage. Eine Auswahl des Sendungs-Genres durch den Benutzer wurde dadurch realisiert, dass die unterschiedlichen Themen aus dem EPG-Text extrahiert und als Liste innerhalb der grafischen Benutzerschnittstelle dargestellt wurden. Zusätzlich gab der 3D-Avatar dem Benutzer ein Feedback und konnte dem Zuschauer zusätzliche Fragen stellen. Wenn die Lieblingssendung des Zuschauers auf einem anderen Fernsehkanal startete, fragte der Avatar nach, ob das Fernsehgerät

automatisch zu diesem Kanal wechseln sollte.
Die Abbildung 5.3 verdeutlicht die Architektur des EMBASSI Haushaltszenarios und die zentrale Rolle des Fernsehens als grafische Aus- und Eingabeschnittstelle, die über verschiedene Kommunikationskanäle angesteuert werden konnte.
Ein Dialog-Manager verwaltete die verschiedenen Funktionsgruppen wie z.B. die Gestenerkennung, das Lippenlesen und die Animationen des Avatars.
Parallel zu dem Fernsehszenario konnte der EMBASSI-Benutzer per Gestik und Sprachkommandos die Raumbeleuchtung und weitere Geräte der Haustechnik wie z.B. einen Videorekorder interaktiv ansteuern. Kontextinformationen wie die Lichthelligkeit oder Vorlieben des Zuschauers wurden während der Interaktion berücksichtigt. Auf Basis der entwickelten EMBASSI-Softwarekomponenten wurde ein TV-basierter Shopping-Assistent entwickelt, der den aktiven Benutzer bzw. Einkäufer vorm Fernseher über eine an die Fernbedienung angebrachte Fingererkennung identifizierte [JSB01]. Parallel dazu wurden dem Benutzer verschiedene Medieninhalte, passend zu seinen Vorlieben, über Internet-Dienste eingeblendet.

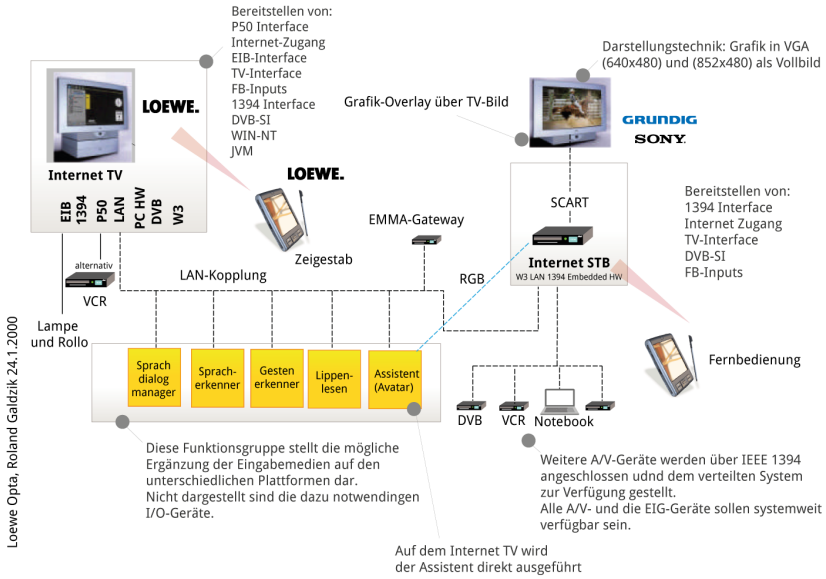


Abbildung 5.3: Demonstrator-Architektur „Privathaushalt“ Architekturbild des EMBASSI. Quelle: Grafisch adaptiert vom Dokument „EMBASSI - White Paper V.1.1“

Die im Rahmen des Projekts entwickelten Softwarekomponenten flossen in die offene Linux-basierte Plattform OpenEMBASSI[7] ein. Diese ermöglichte Linux-Anwendern, einzelne Softwarekomponenten von EMBASSI herunterzuladen, zu testen und zu erweitern.

Das EMBASSI-System ist insofern für das semantische Fernsehen interessant, als es eines der ersten Projekte war, das multimodale Eingaben in Kombination mit dem Fernsehen eingesetzt hat und durch benutzerzentrierte Interaktion in der Lage war, Zusatzdienste aus dem Web aufzurufen. Aus heutiger Sicht waren die entwickelten Schnittstellen zwischen dem Fernseher und der Fernbedienung samt Zeigestab eine

[7] http://www.pro-linux.de/news/1/5305/openembassi-multimediale-assistenz-unter-linux.html

Vorstufe der Second Screens. Die Extraktion von EPG-Informationen und die grafische Darstellung derselben innerhalb der ersten kommerziell verfügbaren Internet-TVs haben gezeigt, dass eine Interaktion mit Daten aus dem Web und EPG durchaus mit der Fernsehaktivität benutzerfreundlich und technisch realisierbar war.

SmartKom

Ziel des im Jahre 2003 abgeschlossenen Forschungsprojekts SmartKom war es, sprachlich dialogische Kommunikation mit intuitiven grafischen Benutzerschnittstellen zu kombinieren und im Rahmen von Szenarien in den Bereichen Informationskiosk (SmartKom Public), Haushalt (SmartKom Home) und mobiler Anwendungen (SmartKom Mobile) zu demonstrieren. Ähnlich EMBASSI wurde das SmartKom-Projekt durch Industriepartner unterstützt, u.a. durch die Philips GmbH oder die Sony International (Europe) GmbH. SmartKom befasste sich innerhalb der drei vorgestellten Szenarien mit der Verarbeitung von Diskurs- und Dialogmodellierung [ABP06], der multimodalen Fission [EP06] und der grafischen Ausgabenrepräsentation durch einen Präsentationsmanager [PT06]. Bei dem ersten Szenario (SmartKom Public) wurde ein Informationskiosk entwickelt, der über Kameras den Zustand des Benutzers und seiner Hand erfasste. Eine horizontale Projektionsfläche zeigte die Benutzerschnittstelle, die per Hand- und Gesteninteraktion vom Benutzer angesteuert werden konnte an (siehe Abbildung 5.4).

Abbildung 5.4: SmartKom Public - Demonstrator

Die Ansteuerung per Gesten wurde über Bildverarbeitungsalgorithmen durchgeführt, welche die Position der Hand ermittelt und die Interaktion dieser in Eingabebefehle für das multimodale Dialogsystem umgewandelt haben [RLR06] [SAB+06]. Eine weitere Eingabemodalität war die Benutzung von Mimik, die nach einer Analyse des Gesichts in der Lage war, den Zustand des Benutzers zu bestimmen, um ihm, falls er unzufrieden ausgesehen hatte, kontextuelle Hilfe anzubieten [Wah02]. Dem Benutzer wurde während der Interaktion mit einem der drei SmartKom-Systeme, eine Unterstützung in Form eines virtuellen Assistenten namens *Smartakus* angeboten. Letzterer konnte im kontinuierlichen sprachbasierten Dialog mit dem Benutzer, Rückfragen stellen und komplexere Fragen beantworten [PGE+03] [PGE+06] [Wah06b].

Abbildung 5.5: Benutzung von SmartKom-Home

Innerhalb des Szenarios SmartKom-Home wurde gezeigt, inwiefern Benutzer dank intelligenter multimodaler Benutzerschnittstellen und Dialogsystemen in der Lage waren, verschiedene heterogene Haushaltsgeräte wie z.B. den Videorekorder oder das Fernsehgerät zu bedienen. Der Zuschauer konnte per Sprachbefehl seinen Fernseher einschalten und über ein Tablet-PC mit dem virtuellen Assistentent Smartakus interagieren (siehe Abbildungen 5.5 und 5.6). Die Tablet-PC basierte Applikation war in der Lage, Sprach- und Gesteninteraktionen des Benutzers zu interpretieren und diese mit den aktuellen EPG-Daten der Fernsehsender zu verknüpfen. Bei der Extraktion von EPG-Daten wurde eine Generierung eines dynamischen Lexikons durchgeführt, sodass Sprachdialoge wie z.B. *„Was kommt heute Abend im Fernsehen?“,“Gibt es Sportsendungen? [. . .] oder einen Film mit Nicole Kid-*

man?“,“*Schalte zu ARD um!*“ oder „*Gibt mir Information dazu!*“ kombiniert mit einer Eingabe per Touch-Stift möglich waren. Ein Planungsmodul war für den Wechsel von einer Applikation zur anderen auf dem Tablet-PC zuständig. Der TV-Bildschirm von SmartKom wurde so aufgeteilt, dass Informationen über den aktuellen ausgewählten Kanal und den Status des Videorekorders im unteren Bereich der Benutzerschnittstelle angezeigt (siehe Abbildung 5.6) und die Hauptinteraktionsfläche mit den Informationen zum aktuellen Fernsehprogramm zentral positioniert wurden [PGE+06]. Darüber hinaus konnte die Programmierung eines Videorecorders in Kombination mit dem EPG, dem Kalender und einer Genregesteuerten Auswahl durchgeführt werden.

Abbildung 5.6: Anzeige von EPG-Informationen und des Avatars Smartakus innerhalb der Fernsehbenutzerschnittstelle vom SmartKom-System

SmartKom verschmolz die Vorteile der sprachlich dialogischen Kommunikation mit einer benutzerzentrierten graphischen Bedienoberfläche für Kiosk- und Fernsehgetriebene Systeme. Die Benutzung eines Zusatzgerätes während des Fernsehens bereitete - aus heutiger Sicht – auf die Second Screens vor, die parallel zum Fernsehen, erweiterte Informationsinhalte anzeigten. Die Verwendung eines Avatars, der zusätzlich per Sprache oder Touch-Eingabe auf einem Tablet-PC angesprochen werden konnte, bildete eine Interaktionsform, die in dieser Form bis heute noch nicht im Fernsehbereich eingesetzt wurde. SmartKom gehörte zu den ersten Systemen, die basierend auf einer RGB-Kamera, die Gesteninteraktionen eines Benutzers interpretierte und mit zusätzlichen Modalitäten wie Sprache und Mimik kombinierte. SmartKom-Home bewies, dass eine erweiterte Interaktion mit EPG-Daten, kombiniert mit multimodalen Benutzereingaben, im Fernsehkontext realisierbar war.

AVATAR

AVATAR (Akronym für Advanced Telematic Search of Audiovisual Contents by Semantic Reasoning) [FPAN+06] [BPG+05] hatte als Ziel eine Unterstützung von Empfehlung und Personalisierung im Kontext des Fernsehens anzubieten. AVATAR führte eine automatische Erkennung der Vorlieben der Benutzer durch, um eine ebenso automatische inhaltsbasierte Empfehlung für Sendungen zu generieren. Dies wurde dadurch erzielt, dass Informationen aus dem Semantic Web und Ontologien benutzt wurden. Eine speziell für AVATAR erstellte Ontologie ermöglichte die Angleichung von TV-Anytime[8] Metadaten mit vordefinierten Konzepten. Somit konnte das Empfehlungssystem ständig

[8] `https://tech.ebu.ch/tvanytime`

erweitert und neue Relationen zwischen Schauspielern, Künstlern und Fernsehsendungen gefunden werden.

AVATAR wurde auf einer Set-Top-Box mittels der Technologien MHP und DVB-J implementiert. MHP ist die Abkürzung von *Multimedia Home Platform* und ermöglicht beim digitalen Fernsehen, Zusatzanwendungen zum TV-Programm wie multimediale interaktive Angebote anzubieten. Diese zusätzlichen Funktionalitäten und Anwendungen werden über den MPEG-2 Stream ausgestrahlt und als MHP-Anwendungen oder *Xlets* (Java Applet für Java TV) bezeichnet. In [BBC02] werden die Grundfunktionalitäten und der Architekturaufbau von MHP skizziert. MHP-basierte Applikationen können im Receiver mittels zweier unterschiedlicher Programmiersprachen entwickelt werden: DVB-HTML und DVB-J. Dies ermöglicht Java-basierte Applikationen, die innerhalb des DVB-Signals oder besser gesagt der MPEG-Pakete mitausgestrahlt werden.

Der Ansatz von AVATAR ist insofern interessant, als neue Relationen mit Bezug auf ältere ausgestrahlte Programme oder Sendungen, die gerade von anderen Zuschauern angeschaut wurden, dank der semantischen Verknüpfungen gefunden werden konnten. Reasoningverfahren über neugefundene Relationen und die Beschreibung der ausgewählten Programme ermöglichen dank der Korrelationswerte die Berechnung des *Degree Of Interest* (Akzeptanz- und Interessewertes des Zuschauers). Die Rückkopplung der Zuschauerprogrammauswahl, gebunden mit einer aktiven Benutzung von Semantik, trägt aus Zuschauersicht zur Anreicherung des Fernseherlebnisses bei.

DICIT

Das in den Jahren 2006 bis 2009 durchgeführte europäische Projekt DICIT (Akronym für Distant-talking Interfaces for Control of Interactive TV) hatte als Fokus, die interaktive Kontrolle von Smart-Home-Umgebungen bzw. Fernseher per Sprache zu ermöglichen [MOM+08] [BEPS08] [SAC10] [MABE08]. Dazu wurden u.a. neue Technologien und Anwendungen in Kombinationen mit Sprache- und Audioanalyse implementiert. Diese Analyse wurde mit mehreren Mikrophonen (engl. Microphone Array), die in verschiedenen Positionen im Raum platziert waren, durchgeführt. Die Herausforderung war dabei, parallel zur Benutzung der Fernbedienung, die Benutzer räumlich zu trennen, den Sprecher zu identifizieren und ihm die Möglichkeit zu geben, den Fernseher oder das Aufnahmegerät mit Sprachkommandos (auf Englisch, Italienisch und Deutsch) anzusteuern. Die Abbildung 5.7 aus der DICIT-Broschüre[9] erklärt das realisierte Szenario.

Diese Sprachkommandos wurden in einem weiteren Schritt von einem Modul detektiert und verarbeitet. Dieses Modul benutzte Verfahren der Echoannullierung (engl. echo cancellation) und hatte als Aufgabe, den Originalton des Fernsehers und die Geräuschkulisse (z.B. andere Töne im Raum) herauszufiltern, damit diese schließlich von einem Sprachkenner (engl. Automatic Speech Recognizer, kurz ASR) analysiert und erkannt werden konnten. Im letzten Schritt dieser Verarbeitung wandelte ein Dialogmanager diese Sprachbefehle, dank einer Grammatik, in Kommandos für die angeschlossene Set-Top-Box [MSM+09] [Dag07] um. Im DICIT-Szenario konnte der Benutzer Sprachbefehle wie z.B.: *„Was läuft am Donnerstag auf Eurosport?“ oder „DICIT, stoppe das Programm!“* äußern.

[9] `http://dicit.fbk.eu/DICIT_final_brochure_NEW.pdf`

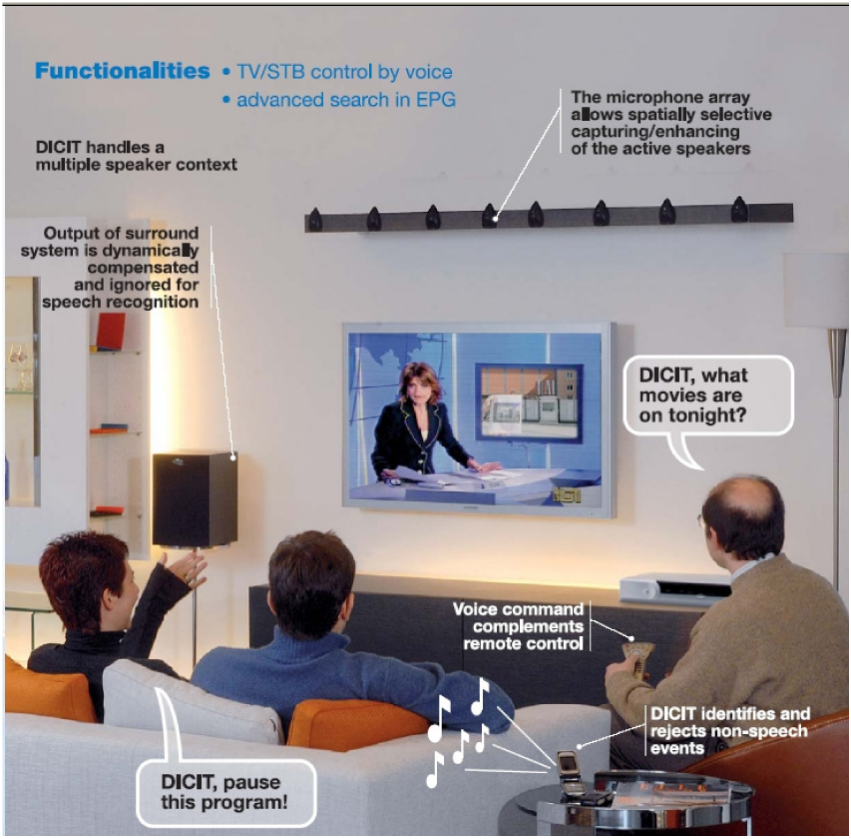


Abbildung 5.7: Erklärung des DICIT Szenarios. Quelle: DICIT Broschüre

Als Ergebnis dieser Interaktion wurden die EPG-Informationen der Set-Top-Box extrahiert und ein visuelles Feedback wurde dem Benutzer über eine Dialogbox über das Fernsehbild und per Sprachsynthese wiedergegeben (siehe Abbildung 5.8 oder das Demonstrationsvideo[10]).

[10] https://youtu.be/xx-lLgW5VXY

Abbildung 5.8: Benutzerschnittstelle des DICIT-Systems. Quelle: DICIT Final Activity Report

THESEUS

Im Rahmen des vom Bundesministerium für Wirtschaft und Energie (BMWi) initiierten und geförderten Leuchtturmprojekts THESEUS[11,12] [WGW+14] wurden zwischen 2007 und 2012 zahlreiche Demonstratoren und Anwendungen, die u.a. die Notwendigkeit der Forschung auf dem Gebiet der semantikgestützten Interaktion innerhalb von Multimediasystemen und sog. Terminal- oder Kiosksystemen hervorho-

[11] http://theseus.pt-dlr.de/
[12] http://www.bmwi.de/DE/Themen/Digitale-Welt/Digitale-Technologien/internet-der-dienste.html

ben, realisiert. Diese Systeme, *CoMET*[13], *Calisto*[14] und *Cirius* [BLS+14] [SDB09], werden in diesem Abschnitt detailliert vorgestellt. Obwohl diese Systeme auf den ersten Blick keinen direkten Bezug zum Thema Fernsehen besitzen, ermöglichen ihre zugrundeliegenden Technologien, einen (mehr-)benutzerzentrierten Zugriff auf multimediales Archivmaterial bzw. Videosequenzen, die semantisch vorannotiert worden sind und über ein multimodales Dialogsystem über unterschiedliche Modalitäten aufrufbar sind.

Semantische Interaktionen mit Multimedia-Interaktionen

Ein Paradigma für diese semantische Interaktion mit Multimedia- und Medienobjekten wurde von mir im Rahmen der Entwicklung der Kiosksysteme *CoMET*, *Calisto* und *Cirius* mitentwickelt und softwaretechnisch umgesetzt. Dieses Paradigma ist das Spotlet-Paradigma [SDB09]. Dieses Konzept versucht sich komplett von einer herkömmlichen App-zentrierten Interaktion zu lösen, indem über einfache und intuitiv gestaltete Bedienelemente dem Benutzer die Möglichkeit gegeben wird, auf allen Arten von Bedienoberflächen (Mobile, Kiosksystem oder Couch-Tisch) und mit möglichst vielen Interaktionsformen (z.B. per Sprache oder Multi-Touch-Gesteneingaben) einen generischen Zugang zu semantischen Medienobjekten und Websucheergebnissen anzubieten. In diesem Abschnitt wird die Funktionsweise der *Spots* und *Spotlets* anhand von Beispielen genauer vorgestellt.

[13] https://www.youtube.com/watch?v=hAAwKxeoCrk
[14] https://www.youtube.com/watch?v=jPJhFqflSRA

Motivation

Das *Spotlet*-Konzept ist die Grundlage für die hier präsentierten Arbeiten und baut auf folgenden Beobachtungen auf. Möchte man (mobile oder TV-basierte) Apps – wie in den vorherigen Abschnitten beschrieben – im Kontext einer semantischen Suche z.B. von Multimediadaten benutzen, begegnet man folgenden Herausforderungen:

- Apps sind nur linear zu benutzen. Der Benutzer startet eine Suche und erhält nach wenigen Sekunden Ergebnisse, die immer grafisch fest definiert worden sind (Ergebnisse mit einem bestimmten Darstellungsformat oder in Listenform). Diese Ergebnisse lassen sich nicht mehr für eine erneute Suche nutzen. Aus diesem Grund muss jedes Mal eine neue Suche initiiert werden.

- In den meisten Fällen können Apps nur bestimmte festgelegte Eingabetypen wie Zeichenkettenfolgen, die eingetippt oder per Sprache eingegeben wurden, benutzen und keine Medienobjekte wie z.B. Bilder, Karten oder Textdokumente. Hier fehlt eine semantische Verbindung zwischen Benutzereingaben und tatsächlicher Sucheingabe.
  Möchte der Benutzer z.B. Bilder vom Brandenburger Tor erhalten, muss er das komplette Wort in ein Suchfeld eintippen. Dabei hätte er schneller in seiner Bildergalerie ein schon vorhandenes Bild vom Brandenburger Tor selektieren können und dieses, als semantisch beschriebenes Medienobjekt, zur Eingabe für eine erweiterte Suche verwenden können.

- Apps besitzen meistens nur eine integrierte Informationsquelle, die meistens per REST-Aufruf abgefragt wird und Ergebnisse liefert. Es gibt sehr wenige Apps, die kombinierte Aufrufe ermögli-

chen. Dies ist kein technisches Problem, sondern weitestgehend auf die kommerzielle Ausprägung der Apps zurückzuführen. Eine App ist meistens nur auf einen Informationsprovider oder eine Informationsquelle zugeschnitten und liefert nur für eine spezifische eingeschränkte Domäne kontextorientierte Ergebnisse. Diese Situation begrenzt enorm den Zugriff auf Wissensdaten, wie es leicht provokativ Tim O'Reilly in einem Interview „The Web Is Dead? A Debate"[15] aus dem Jahr 2010 darlegte.

- Die dargestellten Ergebnisse eines Suchvorganges sind statisch und können nicht weiter referenziert oder per Sprachinteraktion angesprochen werden.
- Möchte der Benutzer verschiedene Ergebnisse aus verschiedenen Apps oder Informationsquellen aggregieren, muss er mühselig von einer App zur anderen wechseln und es bleibt ihm nichts anderes übrig, als die gesuchten Begriffe per *Copy 'n' Paste* in die zweite App einzugeben. Durch diesen Vorgang gehen wichtige Informationen komplett verloren, wie z.B. von welchem Typ die ursprünglichen Ergebnisse waren, welche Datenquellen für die Suche benutzt wurden, welche Medientypen gefunden wurden und wie sie miteinander kompatibel waren.
- Das Wechseln von einer App zur anderen, nur um die Informationsquelle zu wechseln, ist nicht intuitiv und verursacht einen Interaktionsbruch.
- Die Einbindung der Apps innerhalb eines Dialogvorganges ist nicht möglich. Das bedeutet, dass der Benutzer zwar die Möglichkeit hat eine globale Websuche, z.B. per Sprache, durchzuführen,

[15] `http://www.wired.com/2010/08/ff_webrip_debate`

aber nicht eine App mittels Sprachbefehlen komplett anzusteuern oder nur bestimmte Ergebnisse aus einer Liste auszuwählen. Komplexere Interaktionen wie z.B. die kombinierte Eingabe von deiktischen Gesten- und Sprachbefehlen werden zur Zeit von den Apps nicht unterstützt.

Prinzip

Diese im App-Konzept fehlenden Elemente zeigen, dass eine semantische Beschreibung innerhalb eines Interaktionssystems sinnvoll und notwendig ist. Aus diesem Grund wurde das Spotlet-Prinzip entwickelt, um u.a. schnelle Zugriffe auf die Darstellungen von Ergebnissen der Webdienste via Interaktionen wie Sprache oder Multitouch anzubieten. Die Spotlets sind in zwei Komponenten unterteilt (siehe Abbildung 5.9):

- Das graphische Hauptelement ist das Spotlet. Die graphische Benutzeroberfläche des Spotlets ist generisch über seine Kernfunktionalität definiert und lässt sich, z.B. um sich an bestimmte grafischen Anforderungen oder Designs anpassen zu können, leicht erweitern. Diese Anpassbarkeit lässt sich wie die zu Grunde liegenden verwendeten Webdienste und Wissensdatenbanken generisch definieren. Spotlets besitzen eine festgelegte Funktionalität wie z.B. eine Textsuche, eine Videowiedergabe oder eine interaktive Websuche und werden explizit über Benutzerinteraktionen angesteuert. Diese Interaktionen können entweder per Sprache (*„Gib mir Bilder und Informationen über Berlin“*, *„Spiele ein Lied von Michael Jackson“*) oder per Multitouch-Gesten erfolgen. Ergebnisse werden in visueller Form in einem definierten Bereich neben den Spotlets generiert und angezeigt. Die Ergeb-

nisse erscheinen in grafischer Form und können vom Benutzer als weitere Eingabe benutzt und in einen der Spots per Multi-Touch gezogen werden (siehe Abbildung 5.13).

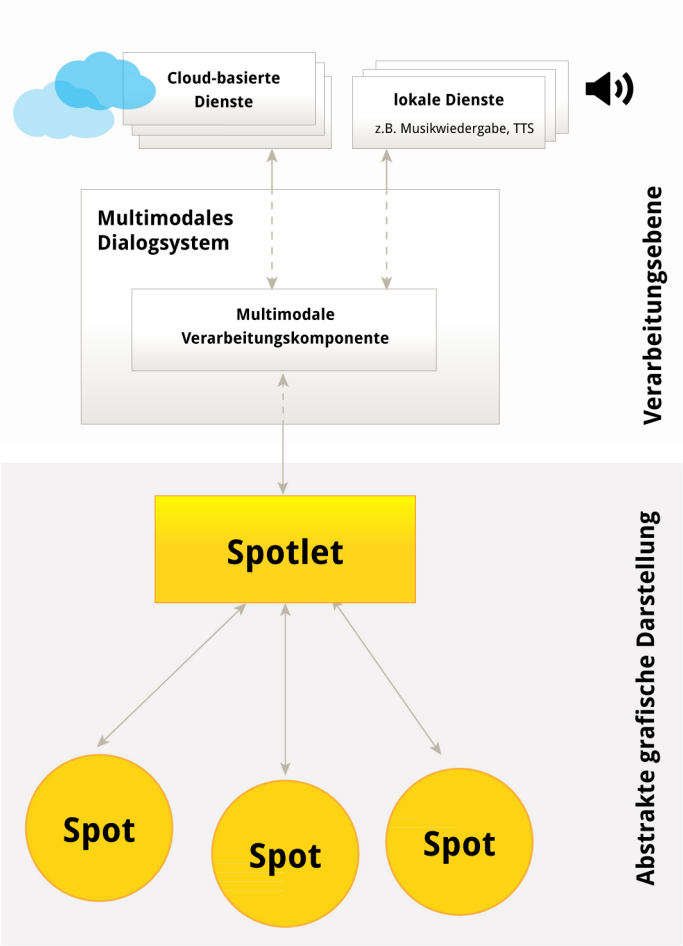


Abbildung 5.9: Abstrahierte Funktionsweise eines Spotlets

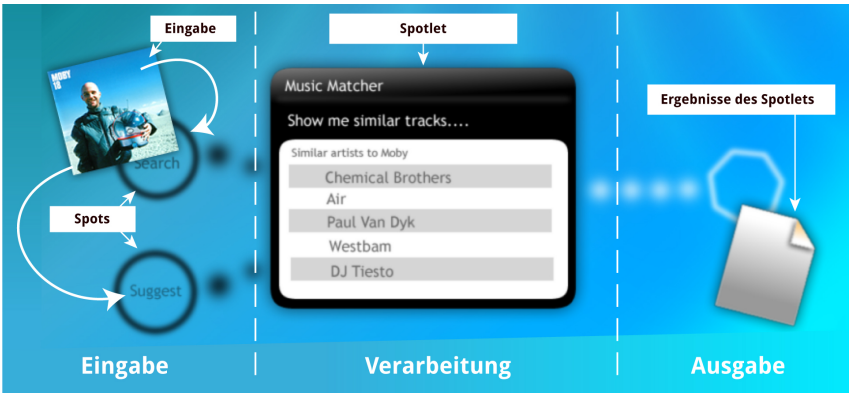


Abbildung 5.10: Aufbau und Prinzip eines Spotlets

- Ein Spot (siehe Abbildung 5.11) ist die grafische Darstellung eines interaktiven Bereichs, zu dem ein Suchelement oder Ergebnis hinzugefügt werden kann. Nach dem Hinzufügen kann das zugeordnete Spotlet seine Aufgabe erfüllen. Möchte der Benutzer z.B. eine Musiksuche starten und neue Musikstücke in einer Mediathek oder bei einem Online-Musikdienst finden, kann er ein Foto eines Künstlers oder das Cover eines Albums per Touch-Geste in den Spot „hineindroppen" (siehe Abbildung 5.14). Diese Interaktion wird als Suche interpretiert und die Eingabe ausgewertet. Eine Interaktion mit dem CoMET-System ist auf YouTube zu sehen[16].

  Da die Eingabe (es sei in diesem Fall angenommen, dass es sich um ein Albumcover eines Künstlers handelt) schon semantisch referenziert wurde, sind der Name und die Referenz des gesuchten Künstlers automatisch dem multimodalen Dialogsystem bekannt und intern für eine weitere Interaktion und Suche referenzierbar. Ab diesem Zeitpunkt übernehmen das Spotlet und

[16] `https://www.youtube.com/watch?v=hAAwKxeoCrk`

die semantische Verarbeitung die Interpretation der Suche und fragen beim entsprechend angebundenen Dienst an. Die Eingabe eines Spotlets erfolgt immer über eine „Drop"- Interaktion. Ein Spotlet kann mehrere Spots verwalten, die für unterschiedliche Eingabetypen oder Funktionen (z.B. Suche bei YouTube oder Aufruf eines DBpedia Artikels) und deren Verarbeitung bestimmt sind.

Abbildung 5.11: Prinzip und Verarbeitungsebenen eines Spotlets

Abbildung 5.12: Beispiele von Spotlets zur Musiksuche, Austausch und Wiedergabe innerhalb des CoMET-Systems

Durch den Spotlet-Ansatz und seine semantischen Beschreibungen innerhalb eines Multimediasystems ergibt sich die folgende Liste von Mehrwerten:

- Eingaben sowie Ausgaben werden als semantische Datensätze oder Objekte betrachtet. Somit können entweder Bilder, Videos oder Textdokumente als Sucheingaben benutzt werden.
- Ergebnisse können von einem anderen Spotlet stammen und als Eingabe für ein weiteres benutzt werden. Als Beispiel kann eine Suche nach einem Album eines Künstlers MP3-Objekte liefern, die als grafische Albumcovers innerhalb der grafischen Benutzerschnittstelle angezeigt werden. Diese können als Eingabe für eine Videosuche benutzt werden. Die generische Beschreibung

der grafischen Elemente ermöglicht eine Interoperabilität auf Interaktionsebene.

- Die Einbindung des Spotlet-Konzepts innerhalb eines Sprachdialogsystems oder multimodalen Dialogsystems wird über die Referenzierbarkeit der Elemente realisiert. Jedes Ergebniselement, das von einem Spotlet generiert wurde, besitzt seitens der Bearbeitungskomponente bzw. des Dialogsystems eine eindeutige semantisch repräsentierte Referenz, die es nachträglich ermöglicht, entweder per Sprache oder per Kombination von Sprache und Touch-Geste auf das Ergebniselement zurückzugreifen.
- Spotlets werden generisch beschrieben. Dank dieser generischen Beschreibung können die Kernfunktionen, die hinterlegenden Dienste und Designs, schnell ausgetauscht werden. Das eigentliche Design und die grafische Darstellung von Spotlets sind von ihrer Kernfunktionalität getrennt (siehe Abbildungen 5.16, 5.18, 5.19). Die grafische Ausprägung kann verändert werden, ohne dass eine direkte Auswirkung auf Interaktion, Suchmechanismen oder interne Funktion des Spotlets stattfindet. Ein generisches semantisches Modell bzw. eine generische semantische Beschreibung der Spotlets ermöglicht dies.

Grafische Ausprägungen

Das Spotlet-Konzept zeigt, dass durch die Verwendung einer generischen Beschreibung der Funktionen, aber auch durch die Benutzung intuitiver Ein- und Ausgabe-Modalitäten, die Interaktionsmöglichkeiten innerhalb eines Multimediasystems deutlich erhöht werden können und weit fortgeschrittener als beim herkömmlichen App-Konzept sind.

Abbildung 5.13: Touch-basierte Interaktion mit den Spotlet beim CoMET-System

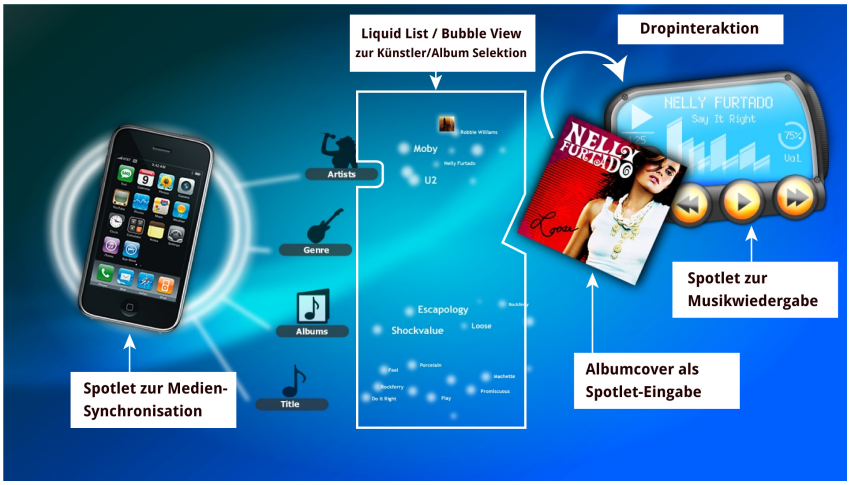


Abbildung 5.14: Liquid List bzw. Bubble List und Spotlets zur Musiksuche, Austausch und Wiedergabe innerhalb des CoMET-Systems

Der Spotlet-Ansatz und dessen Benutzung bilden eine Grundlage für

die technische Realisierung des semantischen Fernsehens. Das im nächsten Kapitel präsentierte Konzept der *Semantic Terms* bzw. *Grabbables* ist eine direkte Weiterentwicklung der Spotlets.

Semantische Verarbeitung

Im Falle der Systeme *CoMET*, *Calisto* und *Cirius* werden die Interaktions-, Suchanfrage- und Ergebnisgenerierungsprozesse von einer im Hintergrund arbeitenden Komponente bearbeitet und verwaltet. Diese Komponente benutzt verschiedene Softwaremodule und Austauschformate. Darunter werden die reinen softwaretechnischen Verarbeitungsmodule wie u.a. der Präsentationsplaner, die Joint Service Engine (JSE) [Ber14b] [Ber14a], der Spracherkenner (innerhalb eines Interaktionmanagers), Regelprozessoren (diese definieren über Regelsätze das Systemverhalten) aber auch Wissensquellen, Ontologien und intermodulare Austauschformate wie PreML (Presentation Markup Language) [SDB09] [BLS+14] oder eTFS [Car05] eingruppiert. Abbildung 5.15 präsentiert abstrakt die Architektur eines multimodalen Dialogsystems, wie dieses initial in [MW98] vorgestellt wurde und später mit den Systemen SmartKom [Wah06a], SmartWeb [Wah04] [RBE+05] [SEH+07] und THESEUS [WGW+14] weiterentwickelt und innerhalb von diversen Szenarien adaptiert wurde, um ein nahtloses benutzerzentriertes Erlebnis während einer multimodalen Interaktion gewährleisten zu können. Im nächsten Kapitel werden sie detaillierter vorgestellt, da sie für die semantische Bearbeitung innerhalb von Swoozy in ihrer erweiterten Ausprägung verwendet wurden. Die abstrahierte Abbildung 5.15 zeigt einen Prozessablauf einer typischen semantischen Bearbeitung durch ein multimodales Dialogsystem.

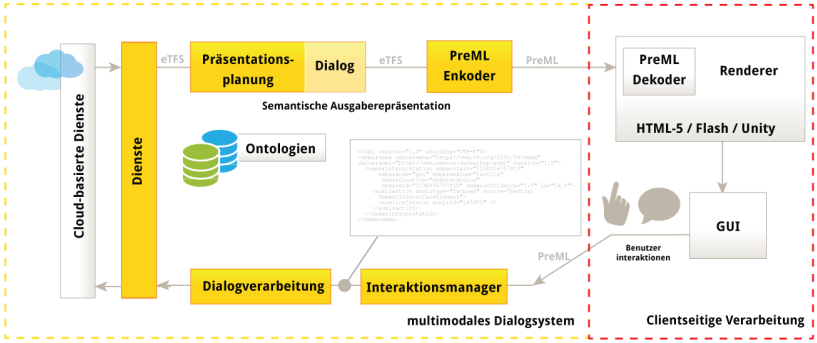


Abbildung 5.15: Vereinfachter Workflow einer semantischen Verarbeitung innerhalb eines Dialogsystems

Auf der rechten Seite der Abbildung 5.15 stellt die abgebildete Benutzerschnittstelle die zentrale Interaktionskomponente zum multimodalen Dialogsystem dar. Diese Schnittstelle kann in mehreren Ausprägungen vorhanden sein (z.B. HTML5 Desktop-Applikation, Flash- oder Flex-Applikation) und visuell in unterschiedlichster Form z.B. entweder als App, Multitouch-Anwendung, Kiosksystem oder integriert in ein Fahrzeug-Infotainmentsystem [DN15] realisiert werden. Die Darstellungstechnologien und die Zielplattformen spielen - abstrakt betrachtet - keine wesentliche Rolle, lediglich eine „Rückreferenzierung“ und eine generische Beschreibung z.B. in Form eines Modells oder einer Ontologie müssen vorhanden sein. Unter Rückreferenzieren versteht man die Möglichkeit, seitens eines Dialogsystems jederzeit heraus finden zu können, an welcher Stelle (grafische X-,Y-Position innerhalb einer Benutzerschnittstelle) welche Funktionalitäten und welche Interaktionsformen (Sprache, Multitouch, Click) zu einem bestimmten Zeitpunkt von einem Benutzer ausgelöst wurden [NF14]. Um all diese Informationen zum Dialogsystem weiterzuleiten, wurde in die Kiosk-Systeme *Cirius* und *Comet* ein Interaktionsmanager

integriert. Dieser interpretiert die Interaktionen und vergleicht sie mit vorher generisch angelegten Regeln. Diese Regeln werden angesteuert und aktiviert, wenn bestimmte Bedingungen (z.B. drückt ein Benutzer gleichzeitig auf ein Spotlet und gibt einen sprachlichen Befehl nach verwandten Medien zu suchen) erfüllt sind. Die Benutzerintention wird über die Regeln bestimmt und interpretiert. Erfolgen mehrere Sprachinteraktionen nacheinander oder stellt der Benutzer Rückfragen (z.B. [Benutzer]: *„Zeige mir Lieder von Nelly Furtado!"* [System]: *„Hier sind die gesuchten Lieder"* [Benutzer]: *„… und von Madonna!"*) werden diese von einer Dialogkomponente verarbeitet, die den Zeitpunkt, wann eine Interaktion und ein Dialog stattgefunden haben, speichert. Dies bezeichnet man als Diskursgedächtnis, da die Referenz einer vorherigen gestellten Frage vom Dialogsystem zu einem späteren Zeitpunkt immer noch aufgelöst und dialogisch beantworten werden kann. Dieser Ansatz erhöht die Effizienz des Dialoges und ermöglicht, dank der vom System erzeugten Rückfragen, eine Präzisierung der ursprünglichen Benutzerspracheingabe (z.B. [Benutzer]: *„Spiele mir Bad von Michael Jackson!"* [System]: *„Meinen Sie das Lied oder das Album?„* [Benutzer]: *„Das Album!"* [System]: *„Das Album Bad von Michael Jackson wird abgespielt"*).

Ontologien

In den THESEUS-Systemen spielen Ontologien innerhalb der Dialogsysteme eine wichtige Rolle, denn sie versehen alle Konzepte und Objekte mit Semantik. Diese kann während einer Interaktion oder eines Dialoges benutzt werden, um z.B. eine Referenzauflösung von Ellipsen auf Grund eines Diskursgedächtnisses durchzuführen. Innerhalb eines Dialogsystems kommen nicht nur eine einzige Ontologie

zum Einsatz, sondern mehrere wie z.B. eine Präsentationsontologie, eine Dialog- und Interaktions-Ontologie (auch *Dialogshell Ontologie* genannt) oder eine domänenspezifische Ontologie. All diese Ontologien übernehmen bei jedem Schritt der Verarbeitung eine spezifische Beschreibungsrolle und können somit von Regeln, z.B. bei einem Dialog, benutzt werden. Eine Präsentationsontologie beschreibt z.B. generische Konzepte der Spotlets. Auf der Bedienoberfläche spielen diese die Rolle des semantischen Informationsträgers. Somit lassen sich z.B. Listen oder listenartige Darstellungen wie die *Liquid List* oder *Bubble View*, (siehe Abbildung 5.13) besser spezifizieren. Die Präsentationsontologie ist derart generisch spezifiziert, dass neue Spotlet-Typen nahtlos hinzugefügt werden können, ohne dass es zu komplexeren Anpassungen auf GUI- und auf Dialogsystemebene kommen muss. Dies hat den Vorteil, dass die Beschreibungen der grafischen Ausgaben seitens des Dialogsystems immer generisch gehalten werden können. Die Dialog- und Interaktionsontologien beschreiben die Interaktionsformen (z.B. Touch, Sprache oder Gesten). Im Falle einer kombinierten deiktischen Geste mit einer Spracheingabe werden der genaue Zeitpunkt und die damit referenzierten Elemente zusammengefasst. Dieses Verfahren nennt man Fusion [Neß16] [EP06] [Pfl07]. Unter referenziertem Element versteht man alle grafischen und nicht-grafischen Elemente, die in strukturierter Form vom System ausgegeben und in einer weiteren Interaktion adressiert werden können. Wenn der Benutzer z.B. eine Liste seiner Lieblingslieder vom System erhalten möchte, kann er dies per Spracheingabe mit dem Kommando *„Zeige mir meine Favoriten-Liste an“* realisieren. Das System kann entweder als Antwort eine geordnete Liste anzeigen oder eine Sprachausgabe starten und über eine Text-to-Speech-Komponente die komplette Liste „durchsagen“. Die Antwortstruktur beinhaltet eine eindeutige

Referenznummer jedes Eintrages (z.B. eines Lieds), die auf eine interne, im Dialogsystem abgebildete Struktur, zurückweist. Möchte der Benutzer eines der Lieder auswählen, hat er zwei Möglichkeiten. Entweder per Touch-Interaktion mit der Liste oder per Sprache mit Befehlen wie: *„Spiele das Lied ab!“* oder *„Spiele das fünfte Lied der Liste ab!“*.

Abbildung 5.16: Benutzeroberfläche des Calisto-Systems und Anzeige von Video- und Bildergebnissen

Um das gewünschte Lied in der grafischen Darstellung der Musikstückeliste korrekt anordnen zu können, wird das Dialogsystem dessen Referenz auflösen müssen und mittels dieser eindeutigen Referenz bestimmen, welches Lied abgespielt werden muss. Die Domänenontologie beschreibt Terminologien (Konzepte, die innerhalb einer Thematik oder Wissensdomäne vorkommen), die während einer Interaktion oder eines Dialogaktes benutzt werden können. Beim System *Comet* wurde die Musikdomäne verwendet, bei *Calisto* die spezifische Domäne „Das Leben von Konrad Zuse“ (siehe Abbildung 5.16) und bei *Cirius*

(siehe Abbildung 5.18) die geografische Domäne. Die Bestimmung einer Domäne und die Spezifikationen der Terminologien können entweder von einer neuen oder von einer existierenden Ontologie ausgehend manuell erweitert werden [PDB+14]. Das Calisto-System ermöglichte nicht nur eine Multitouch-basierte Interaktion, sondern erlaubte, dass Benutzer über eine dedizierte mobile App und durch eine simulierte Frisbee Geste[17] , Multimediaobjekte, die in ihren mobilen Endgeräten gespeichert waren, in Richtung des Terminals zu „schleudern“ [BDP10].

Abbildung 5.17: Schritte der Frisbee-Interaktion im Calisto-System

In Abbildung 5.17 wird diese neue Interaktionsform verdeutlicht: zuerst muss der Benutzer ein Bild, das auf seinem Handy gespeichert ist, selektieren (Schritt 1). Nach diesem Schritt wird eine virtuelle Frisbee-

[17] https://www.youtube.com/watch?v=jPJhFqflSRA

Geste durchgeführt, indem das Handy in Richtung des Terminals virtuell geworfen wird (Schritt 2). Durch diesen nachgeahmten Frisbee-Wurf wird eine Synchronisation mit dem Calisto-Terminal gestartet und kurz danach erscheint das „geschleuderte“ Bild innerhalb der grafischen Benutzerschnittstelle des Calisto-Terminals (Schritt 3).

Abbildung 5.18: Benutzeroberfläche des Cirius-Systems

Somit entstand eine neue Metapher für die Datensynchronisation unterschiedlicher Systeme. Des Weiteren konnten Spracheingaben, die an das Calisto-Kiosksystem gerichtet waren, statt über das im Kiosk eingebaute Mikrofon, über das persönliche mobile Endgerät des Benutzers formuliert werden [BDP10].

Abbildung 5.19: HTML5-Ausprägung von Spotlets

Semantische Suche

Nachdem ein Dialogbeitrag interpretiert wurde, können Funktionen oder Dienste innerhalb des Dialogsystems aufgerufen werden. Bei den vorgestellten Systemen *Calisto* und *Cirius* liegt der Fokus der Interaktion auf der Suche im semantischen Web (Web 3.0). Die Systeme müssen die Ergebnisse dieser semantischen Dienste, die teils über heterogene Ergebnis- und Abfrageformate wie XML, JSON, RDFs oder SPARQL gebildet werden, vereinheitlichen.

Dies bedeutet, die Formate auf einer einheitlichen homogenen Struktur- und Konzeptebene abzubilden, um diese mit der passenden Domänenontologie in Einklang zu bringen. Um dies zu realisieren, werden die Antwortstrukturen dieser Webdienste aggregiert und auf den entsprechenden Domänenkonzepten abgebildet. Natürlich muss im Vorfeld gewährleistet werden, dass je nach Suchkriterien die passenden

Dienste, APIs und Schnittstellen mit dem korrekten Anfrageformat aufgerufen werden, damit nachträglich die Ergebnisstrukturen zusammengefügt werden können. Diese Operation kann mittels eines generischen Moduls realisiert werden, wie z.B. der Joint Service Engine (JSE), die ausführlicher in [Ber14b] und [Ber14a] beschrieben wird.
Innerhalb der semantischen Bearbeitung und der Präsentationsausgabe muss entschieden werden, z.B. nachdem Multimediadaten (Bilder, Videos) als Ergebnis einer Web 3.0 Suche zurückgeliefert wurden, mit welcher grafischen Darstellung diese Ergebnisse auf der Benutzeroberfläche der verschiedenen Client-Typen (Kiosksystem, Desktop PC, mobile Endgeräte, usw.) erscheinen sollen.
Diese dynamischen Anpassungen werden innerhalb des Präsentationsplaners durchgeführt, der einerseits indirekt die semantisch beschriebenen Ergebnisstrukturen der JSE auswertet (z.B. im extended Typed Feature Structure (eTFS)-Format [Pfl07]) und andererseits diese über ein Regelsystem gemäß eines UI-Modells in eine für die Clients kompatible Ausgabestruktur (PreML, XML oder JSON) umwandelt. Diese generische Struktur wird zum Client weitergereicht und entsprechend grafisch visualisiert.

Präsentationsplanung

Die Präsentationsplanung, die innerhalb eines Dialogsystems erfolgt, hat zur Aufgabe, die semantischen Strukturen soweit generisch aufzubauen, dass diese in Ausgabeformaten (z.B. PreML, XML oder JSON) für die grafische Benutzerschnittstelle enkodiert und nachträglich von Visualisierungs- bzw. Renderingtechnologien (HTML5, Flash oder Flex) geparst und grafisch umgesetzt werden können.
Anhand des generischen Modells der grafischen Benutzerschnittstelle

und eines Abgleichs mit der Präsentationsontologie wird ein vereinfachtes Regelsystem angewandt. Dieses lädt Vorlagen (auch Transformationsregeln genannt), die als Basis für die Umwandlung der internen semantischen Strukturen (z.B. eTFS) in PreML oder XML dienen.

Dieser Schritt kann als adaptive semantische Ausgabe bezeichnet werden, da strukturierte, semantisch beschriebene Daten in eine kompaktere Form umgewandelt und für eine Ausgabe, sei es eine grafische oder akustische, vorbereitet werden.

Listing 5.1: Beispiel einer PreML-Ausgabe zur Listendarstellung von Ergebnissen (siehe Abbildung 5.9)

```
<presentation target="KnowledgeSpotlet">
<query>what are the neighboring countries of france</query>
<list type="country">
<item id="49180ede-536c-4b08-886e-744241e3b1ee">
<label>Germany</label>
<image src="http://..."/>
</item>
[...]
</list>
</presentation>
```

Ein anschauliches konkretes Beispiel, wie der Vorgang während einer Interaktion mit dem *Cirius*-System aussieht, liefert folgende Situation. Nach einer Suche können verschiedene Typen wie z.B. *dm#Country* oder *dm#River* als Ergebnis zurückgeliefert werden. Die grafische Ausprägung z.B. des semantischen Konzepts *Land* (dm#Country) wird durch die Vorlage als eine grafische Zusammenfassung, die aus einer Karte, einer Flagge und einer Kurzbeschreibung besteht, definiert. Dagegen kann das Konzept *Fluss* (dm#River) mittels eines Videos und einer längeren Audiobeschreibung dargestellt und ausgegeben wer-

den.

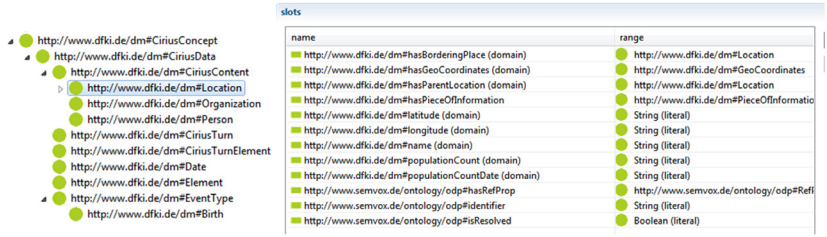


Abbildung 5.20: Auszug der Cirius-Ontologie mit den Klassen der Basisontologie mit dem Namensraum #dm

Falls Videos oder längere Texte als Ergebnis zurückgeliefert werden, wird über die Vorlage und die Transformationsregeln entschieden, wie diese Medienobjekte darzustellen sind. Der Präsentationsplaner spielt eine Kernrolle innerhalb des Dialogsystems, indem er die semantischen Ergebnisse über ein Regelsystem vergleicht und in Ausgabeformate wie PreML oder JSON transformiert und kontextorientiert an eine interactive Darstellungskomponente weiterreicht.

Grafische Darstellung und Visualisierungen

Der letzte Schritt der Verarbeitung besteht in der Ausgabe und der Generierung der Visualisierung. Bei diesem Schritt werden dem Benutzer die Ergebnisse grafisch angezeigt und er kann, je nach Interaktivitätsgrad, mit diesen interagieren. Der Präsentationsplaner verarbeitet die bereitgestellten JSON- oder PreML-Strukturen, verändert die grafische Benutzerschnittstelle bzw. die Visualisierung und ergänzt diese mit den gefundenen Ergebnisstrukturen. Eine Darstellungskomponente (oft als Visualisierung bezeichnet) ist streng betrachtet eine generische Komponente, die als Platzhalter für die Ergebnisstrukturen dient und

dynamisch erweitert werden kann. Die grafische Ausprägung dieser (z.B. als Spotlet oder Liste mit Bildlaufleiste) und die Visualisierungsformen (z.B. eine *Bubble View*) können von einem Benutzerschnittstellendesigner gestaltet werden. Ähnlich dem Model View Control-Prinzip (MVC) werden die Daten bzw. Ergebnisstrukturen getrennt von ihrer eigentlichen Visualisierungsart betrachtet.

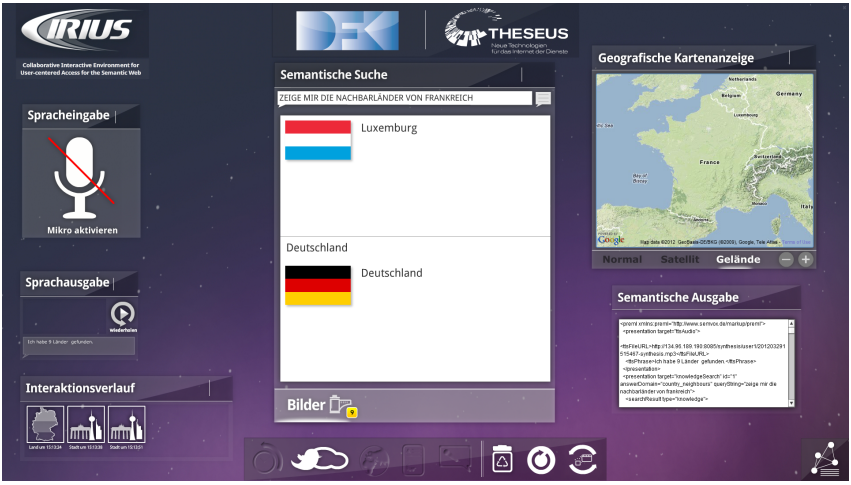


Abbildung 5.21: Grafische Ausgabe der Listendarstellung nach Bearbeitung der PreML-Nachricht von Listing 5.1 innerhalb des Systems Cirius

Die dahinterliegende Technologie für die grafische Ausgabe hängt stark von der Zielplattform (mobiles Endgerät, Kiosksystem, usw.) und vom Anwendungsszenario ab. Im Fall der Kiosksysteme *Calisto* oder *CoMET* wurde Adobe AIR in Kombination mit der Adobe Flash-Technologie benutzt. Parallel dazu wurde eine identische grafische Darstellung der Benutzeroberfläche mittels HTML5 über CSS3 und Javascript realisiert (siehe Abbildung 5.19). Auch andere Technologien wie Objective-C (für mobile iOS- Anwendungen) konnten erfolgreich in Kombination mit einem Dialogsystem eingesetzt werden [PSN09].

Erfolgt eine Veränderung oder Interaktion auf Seite der Benutzeroberfläche, wird diese über einen Socket- oder REST-Schnittstellenbasierten Kanal zum Dialogsystem weitergeleitet und dort analysiert und verarbeitet. Ähnlich sieht es bei sogenannten *UI-Updates* (automatische neue Gestaltung der Benutzerschnittstelle während des Programmablaufs) aus. Diese können proaktiv und dynamisch vom Dialogsystem angesteuert werden, z.B. durch Kommunikationsmechanismen wie Polling (mehrfache und kontinuierliche Abfrage einer REST-Schnittstelle) in Kombination mit AJAX-Aufrufen.

Austauschformate

Damit die Kommunikation zwischen den internen Dialog- und Softwaremodulen erfolgreich durchgeführt werden kann, muss ein generisches aber auch dynamisch erweiterbares Austauschformat ausgewählt werden. Innerhalb des multimodalen Dialogsystems namens *ODP* (Ontology-based Dialog Platform), das u.a. bei den Systemen *CoMET*, *Calisto*, *Cirius*, *BabbleTunes* [SPPS08] oder in [SZK+11] benutzt wurde und heute kommerziell von der Firma SemVox[18] angeboten wird, bilden die eTFS-Strukturen [Pfl07] die Grundlage für das Austauschformat zwischen den internen Softwaremodulen und den Interpretationskomponenten.
Auf grafischer Benutzerschnittstellenebene werden generische Formate wie JSON- oder XML-Formate in Zusammenarbeit mit dem Dialogsystem angewandt, um u.a. die verschiedenen Zustände der grafischen Oberfläche abzufragen und diese dynamisch zu verändern. Somit werden eine modulare, generische und technologieunabhängige Verarbeitungslogik und ein Austausch zwischen der Visualisie-

[18] `http://www.semvox.de`

rungskomponente bzw. Benutzerschnittstelle und dem Dialogsystem gewährleistet.

LinkedTV

Einer der umfangreichsten und ambitioniertesten europäischen Initiativen im Bereich *Fernsehen der Zukunft* fand von 2011 bis 2015 im Rahmen des LinkedTV-Projekts[19] [20] statt [Nix13]. In diesem Projekt waren u.a. Industrie- und Forschungspartner wie z.B. Fraunhofer IAIS[21], Condat AG[22], Eurecom[23] und Noterik[24] involviert. Zusätzlich trug inhaltlich der Fernsehsender Rundfunk Berlin-Brandenburg (RBB) als Medienpartner Videomaterial und technische Unterstützung in verschiedenen Szenarien bei. Ziel von LinkedTV war das Fernsehen bzw. Fernsehinhalte mit Wissensdatenbanken aus dem Web zu verbinden, um Zuschauern ein angereichertes Fernseherlebnis anzubieten. Als Ergebnis des Projekts wurde die LinkedTV Plattform als *Platform as a Service (PaaS)* angeboten. Innerhalb von LinkedTV wurden drei Szenarien durchgeführt: Interactive News[25] (Anreicherung von Nachrichtensendungen), Hyperlinked Documentary (Second Screen-basierte Applikation zur Darstellung von Informationen mit Auswahlmöglichkeiten von verschiedenen Kapiteln und Entitäten) und Media Arts (Kunstinstallation in Form eines Kiosksystems in einem Museum mit multimodalem Zugriff auf Videoinhalte). Parallel dazu wurden zahlreiche Aspekte der Audio- und Videoanalyse sowie Werkzeuge für die Annotation

[19] `http://http://www.linkedtv.eu`
[20] `http://linkedtv.project.cwi.nl/`
[21] `http://www.iais.fraunhofer.de`
[22] `http://www.condat.de`
[23] `http://www.eurecom.fr`
[24] `http://www.noterik.nl`
[25] `http://linkedtv.project.cwi.nl/rbb`

durch Fernsehsender entwickelt. Jeder einzelne Projektpartner war für die Entwicklung von Werkzeugen und Komponenten zur Anreicherung von Videos bzw. Fernsehen zuständig. Diese Werkzeuge und Ansätze sind sehr vielfältig und reichen von der Objekt- und Szenenwechselerkennung, der Videoinhaltsanalyse auf spatio-temporaler Basis [IACM15], der Audioextraktion, der kombinierten Erkennung von Konzepten und Entitäten (z.B. mit NERD, einer spezifischen für LinkedTV entwickelten REST-basierten Extraktionskomponente, die parallel zwölf Named Entity Erkenner (kurz. NER)-Module benutzt), bis zu einem Wikifier (eine Komponente, die automatisch extrahierte Konzepte oder Entitäten mit Wikipedia verbindet), oder einem Editor[26], der basierend auf den von der LinkedTV Plattform extrahierten Annotationen, dem Fernsehsender die Möglichkeit gibt, diese Annotationen mit Zusatzinformationen aus dem Netz zu verbinden. Zusätzlich wurde eine spezifische LinkedTV-Ontologie aufgesetzt, die Konzepte und das zulässige Vokabular der semantischen Annotation definiert. Die Ontologie baut auf zahlreichen bestehenden Formaten und Standards für die Beschreibung von Multimediainhalten auf, um somit die LinkedTV Anforderungen und die Interoperabilität zwischen den vorhin aufgelisteten Werkzeugen und Ansätzen zu maximieren. Die Personalisierung der angebotenen LinkedTV-Inhalte oder die Kontextualisierung einer Benutzerinteraktion werden durch eine Ontologie modelliert. Diese Ontologie (LUMO) basiert auf mehreren bestehenden Ontologien und Vokabularen wie z.B. den DBpedia-, GUMO-, YAGO-Ontologien, den Vokabularen von Schema.org oder den IPTC Newscodes[27]. Um eine passende Zuordnung von LUMO auf diese externen Ontologien und Vokabulare zu ermöglichen, wurden die 524 Klassen von LUMO auf

[26] `http://editortool.linkedtv.eu`
[27] `https://iptc.org/standards/newscodes`

785 externe Klassen zugeordnet und angeglichen [TM14].
Die LinkedTV-Ontologie wurde in Kombination mit dem Werkzeug TV2RDF[28] benutzt. Letzteres analysiert unterschiedliche Metadatenformate aus der Fernsehproduktion wie z.B. TVAnytime[29] [SB13] und wandelt diese in RDF-Dokumente, um basierend auf den vordefinierten Konzepten der LinkedTV-Ontologie.

Abbildung 5.22: Bildschirmauszug der Second Screen App. Quelle: http://linkedtv.project.cwi.nl/rbb

Die Transkripte bzw. Untertitel der Beiträge, die im Produktionsvorgang als .srt-Dateiformat vorliegen und die daraus resultierenden Media Fragments[30] kombiniert mit Entitäten, die von NER-Komponenten extrahiert wurden, werden ebenso von diesem Werkzeug berücksichtigt.
Neben den Werkzeugen, die dediziert für die Produktion, Annotati-

[28] https://github.com/jluisred/TVRDFizator
[29] https://tech.ebu.ch/tvanytime
[30] https://www.w3.org/TR/media-frags

on und Verwaltung der Medien auf Sender (engl. Broadcaster)-Seite erstellt wurden, wurden im LinkedTV-Projekt Second Screens Applikationen entwickelt. Die Entwicklung dieser dedizierten Benutzerschnittstelle wurde auf Basis von HTML5 durchgeführt und benutzte ein sog. *Multiscreen Toolkit* des Projektpartners Noterik[31]. Dieses ermöglichte den schnellen Einsatz eines Basisgerüsts (engl. Template) für die Realisierung der HTML5-gestützten Second Screen Applikation.

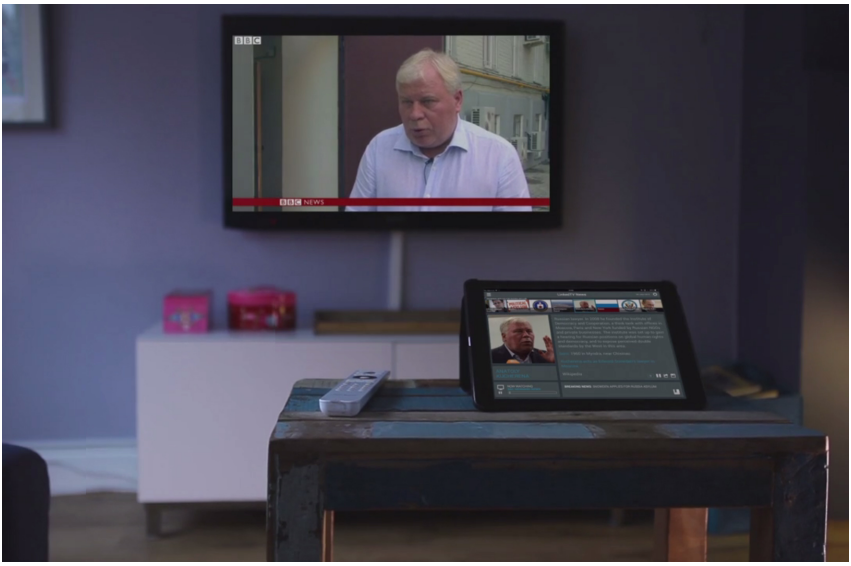


Abbildung 5.23: Synchronisation zwischen Fernsehinhalt und Second Screen
Quelle: Bildschirmauszug aus LinkedTV–Projekt Video
(https://vimeo.com/112383692)

Ein weiterer interessanter Ansatz bildet die grafische Benutzerschnittstelle des Second Screens Demonstrators *International News Companion* vom „Interactive News"-Szenario [NBRH14] [RGHRT14]. Hier spielt die mobile Anwendung eine zentrale Rolle. Synchron zum Fernsehbild

[31] `https://github.com/noterik`

werden Informationen aus dem Web extrahiert und auf dem Second Screen dargestellt (siehe Abbildung 5.23). Startend von diesen Informationen, kann der Benutzer weitere einblenden oder diese für einen späteren Zugriff als Lesezeichen speichern. Dieser Demonstrator stützte sich auf annotierte Videoinhalte vom RBB und dem holländischen Fernsehprogramm über Kunst, Tussen Kunst & Kitsch[32]. Die Abbildung 5.22 zeigt die Benutzerschnittstelle des Second Screens und die Einblendung der Zusatzinformationen. Parallel zum Second Screen und in Kooperation mit dem RBB wurde eine HbbTV Applikation entwickelt, die es ermöglicht während einer Videowiedergabe weitere Kapitel und dazugehörige Annotationen einzublenden (siehe Abbildung 5.24).

Abbildung 5.24: LinkedTV RBB-HbbTV Demonstrator. Quelle: Bildschirmauszug Linked TV - IFA 2014 Präsentationsvideo (https://vimeo.com/106264077)

[32] http://web.avrotros.nl/tussenkunstenkitsch

Zusammenfassend kann behauptet werden, dass LinkedTV ein sehr ambitioniertes Projekt war, das viele Aspekte der Kombination von Fernsehen mit dem Web dargestellt hat. Die Verlinkung zu Wissensdatenbanken mit Inhalten bildet einen zentralen Aspekt des semantischen Fernsehens und findet sich in dem Projekt wieder. Zusätzlich wurden Werkzeuge für die Produktion entwickelt, welche eine semantische Beschreibung von Sendungen und Videobeiträgen erlauben und die von einer Ontologie unterstützt werden. Obwohl LinkedTV eine Integration von REST-basierten Diensten zur Video- und Audioanalyse anbietet und diese zusammen in einer Plattform kombiniert, ist es nicht ersichtlich, inwieweit sich diese Kombinierbarkeit mit einer Liveanalyse des Fernsehbildes ausprägt.

NoTube

NoTube [SBB+10] [ANM11] [AND09] [ACD+09] war ein dreijährig EU-gefördertes Forschungsprojekt, das im Jahre 2012 abgeschlossen wurde und als Ziel hatte, die Zukunft des Fernsehens zu erforschen und das Fernseherlebnis mit dem Web, dem Social Web und Personalisierungskomponenten zu verbinden. Das Projekt wurde von verschiedenen Partnern durchgeführt, u.a. durch die Vrije Universiteit in Amsterdam, die BBC, das Institut für Rundfunktechnik, den italienischen Fernsehersendern RAI, STI und Thomson Video Networks.
Im Showcase „Personalized News“ [BDPM+09] wurde ein Fernsehprogramm der RAI mit den Präferenzen der Benutzer zusammengeknüpft und direkt auf der grafischen Fernsehschnittstelle angezeigt (siehe Abbildung 5.25). Zusätzlich wurde eine animierte Begriffswolke über das Fernsehbild eingeblendet, die Konzepte, die innerhalb des Nachrichtenvideos vorkommen, gruppiert darstellt und diese mit

DBpedia-Einträgen grafisch verbindet. Durch einen Klick konnte der Zuschauer eine Zusammenfassung des DBpedia-Eintrags einblenden lassen (siehe Abbildung 5.26).

Abbildung 5.25: Benutzerschnittstelle von NoTube. Quelle: Internet-Seite von NoTube.tv

Um dies zu realisieren, wurden u.a. Verfahren der Forschungsfelder Entitätenerkennung, Linked Data, Empfehlungen und Personalisierung technisch umgesetzt und in Form von neuen Protokollen und APIs innerhalb der NoTube-Demonstratoren integriert. Einer davon, der NoTube BeanCounter, ermöglichte die Erfassung der Lieblingsprogramme des Zuschauers und kategorisierte bzw. verknüpfte diese mit Metadaten. Dies wurde dadurch ermöglicht, dass die EPG-Texte analysiert und auf Entitäten überprüft wurden. Angereichert wurde diese Analyse durch einen direkten Vergleich zwischen Empfehlungen, heterogenen Metadaten, die vom EPG extrahiert worden sind, und dem sozialen Web. Kontextinformationen wurden über eine zusätzliche App namens *iZapper* mitprotokoliert. Alle angesammelten

Informationen flossen letztendlich in die Suchangaben für die Twitter- und YouTube-Anfragen des Benutzers ein.

Abbildung 5.26: Benutzerschnittstelle von NoTube. Quelle: Internet-Seite von NoTube.tv

Beim BeanCounter lag der Fokus auf Social TV und der Semantifizierung der Empfehlung durch eine Verlinkung mit semantischen Diensten wie z.B. Linked Data, MusicBrainz, der BBC Ontologie oder DBpedia. Mithilfe dieser Zusatzinformation konnte der BeanCounter eine benutzerzentrierte Empfehlung generieren [VAAR+09].
Eine weitere Entwicklung von NoTube, die in Zusammenarbeit mit der BBC durchgeführt wurde, war die Realisierung einer Second Screen Applikation namens *N-Screen* (siehe Abbildung 5.27), die basierend auf HTML5-Technologien eine Integration der Personalisierungs- und Empfehlungskomponenten auf einem Tablet, parallel zu einem Fernsehvideo, vorwies. Um die technische Synchronisation zwischen Tablet und Fernseher zu realisieren, wurde als Austausch- und Kommunikati-

onsprotokoll das Protokoll XMPP (Extensible Messaging and Presence Protocol)[33], das mit der MythTV[34]-Software (diese verwaltet die Anzeige und Wiedergabe von Videos auf dem Fernseher) kommunizierte, benutzt.

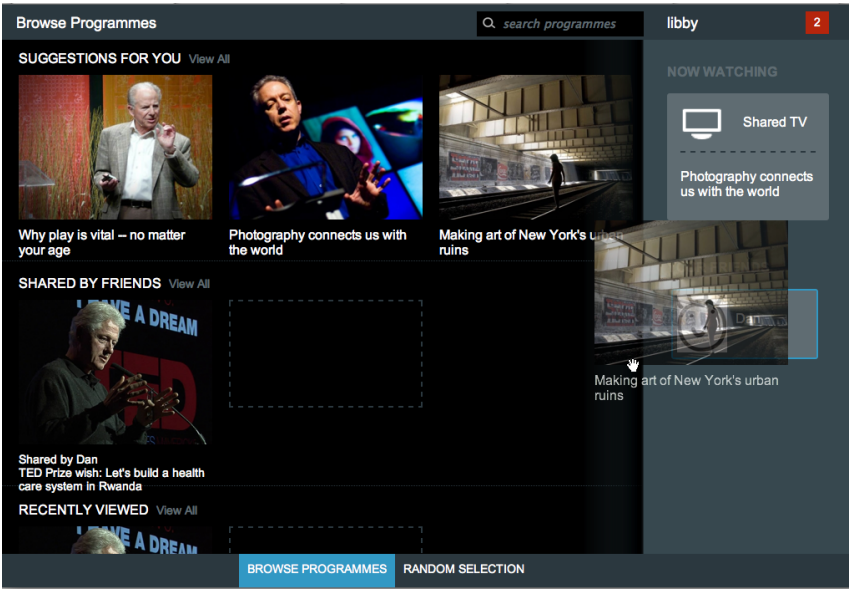


Abbildung 5.27: Benutzerschnittstelle von N-Screen. Quelle: http://www.notube.tv

Bei dem Projekt NoTube war der Fokus primär auf das Social Web und die Anreicherung der TV-Angebote durch Empfehlungen und Personalisierungsmodule gesetzt worden. Die präsentierten Ansätze gingen sehr stark in die Richtung einer Analyse von EPG-Informationen, die in einem weiteren Schritt u.a. durch die Verwendung von Wissensdatenbanken wie DBpedia oder MusicBrainz angereichert wurden. Die Ansätze der Second Screen Applikation, N-Screen, zeigen, dass die

[33] https://tools.ietf.org/html/rfc3920
[34] https://www.mythtv.org

Ergebnisse der Personalisierungsmodule außerhalb einer Fernseh-basierten Benutzerschnittstelle technisch durchgeführt werden konnten.

iFanzy

iFanzy[35,36] war ein Prototyp, der im Zuge des Projekts NoTube entstanden ist und das Konzept des *Personalized EPGs* präsentiert [BAH+07] [BVDSAH08] [BVKK09] [AAB06].

Abbildung 5.28: Benutzerschnittstelle von iFanzy innerhalb der dedizierten Set-Top-Box. Quelle: [BVKK09]

Die Idee hinter iFanzy war, eine personalisierte Fernseh- und mobile

35 http://www.ifanzy.nl

36 http://ercim-news.ercim.eu/en72/special/distributed-personalization-bridging-digital-islands-in-museum-and-interactive-tv

Applikation zu realisieren, um Zuschauern ein personalisiertes TV-Programm anzubieten. Der Ansatz ist insofern relevant, als iFanzy die im EPG befundenen Metadaten analysiert und diese mit dem Semantic Web bzw. Ontologien wie z.B. WordNet[37], GeoNames[38], OWL-Time[39] kombiniert, um die Qualität der Vorschläge zu verbessern [AAB06].

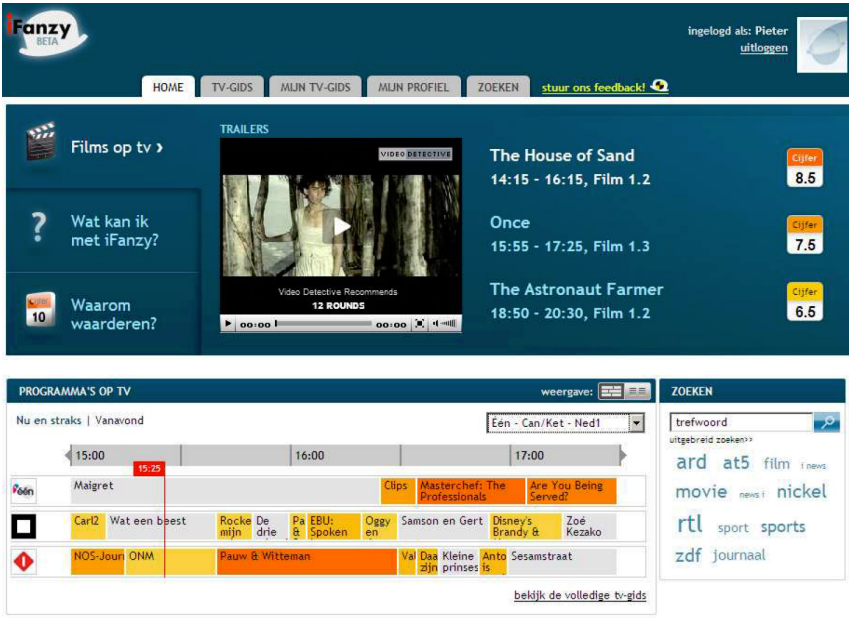


Abbildung 5.29: Benutzerschnittstelle von iFanzy als WebTV Applikation. Quelle: [BVKK09]

Diese Zusatzdaten wurden vom SenSee-Framework geliefert [ABB+07]. Dieses integrierte unterschiedliche Datenquellen von Dritten wie z.B. die von BBC Backstage, XML-TV und IMDB[40], und führte diese zusammen auf, damit iFanzy diese Daten weiterverarbeiten

[37] https://wordnet.princeton.edu
[38] http://www.geonames.org
[39] http://www.isi.edu/~hobbs/owl-time.html
[40] http://www.imdb.com

konnte.

Innerhalb der Benutzerschnittstelle wurden verschiedene Empfehlungen in Form einer Begriffsliste angezeigt (siehe Abbildungen 5.28 und 5.29). Optional bekam der Benutzer die Möglichkeit nach eigenen Begriffen (z.B. Fußball) zu suchen. Wird einer dieser Begriffe selektiert, werden nicht nur alle Programme, die einen direkten Bezug zu diesem Thema umfassen, angezeigt, sondern auch erweiterte Begriffe wie z.B. „Soccer" oder „Football", die miteinander semantisch in Relation stehen, angeboten.

Die Eingaben der Benutzer wurden serverseitig analysiert und konnten in das Personalisierungsmodul einfließen. Zu Zeiten des Prototyps gab es drei Versionen von iFanzy: eine mobile, eine WebTV-basierte und eine Set-Top-Box-basierte Applikation [41] [BVKK09]. Aktuell ist iFanzy als mobile Applikation für Android und iOS erhältlich[42,43].

ViSTA-TV

ViSTA-TV[44,45,46] war ebenfalls ein von der europäischen Union gefördertes Projekt, das von 2012 bis 2014 lief und als Ziel hatte, Verhalten und Interesse der Zuschauer beim Fernsehen in Echtzeit besser zu erfassen und zu verstehen [SKB12] [VGJ13] [Mac12]. Dieses Projekt wurde u.a. von den Unternehmen BBC[47] und Zattoo[48] unterstützt, da

[41] https://www.youtube.com/watch?v=ICl9pT9sMvI
[42] https://itunes.apple.com/nl/app/ifanzy-tv-gids-voor-iphone/id838118096?\&mt=8
[43] https://play.google.com/store/apps/details?id=com.stoneroos.ifanzy\&hl=nl
[44] http://vista-tv.eu
[45] http://www.bbc.co.uk/rd/projects/vista-tv
[46] https://vistatv.files.wordpress.com/2012/05/factsheet.pdf
[47] http://www.bbc.co.uk/blogs/researchanddevelopment/2012/08/vistatv-linked-open-data-stati.shtml
[48] http://zattoo.com

der Fokus auf der Erfassung der IP-TV-Benutzung durch Zuschauer lag. Die Hauptaufgabe der ViSTA-TV Partner bestand darin, ein Framework zur Analyse von videobasierten Strömen, um z.B. den Zeitpunkt eines Szenenwechsels innerhalb des Videos und der EPG-Informationen zu erfassen, zu erstellen. Diese EPG-Informationen wurden nachträglich mit Linked Open Data Set (kurz: LOD) angereichert und in einem RDF-Store gespeichert. Durch diese Anreicherung konnte die ausgestrahlte Sendung mit passenden Konzepten versehen und, über die Zeit, ein verfeinertes Profil vom Zuschauer gewonnen werden. Das Erstellen des Zuschauerprofils geschieht über eine anonyme Datenerfassung. Selektiert der Zuschauer unterhalb einer festgelegten Zeitspanne die Senderwechsel-Taste, so interpretiert das System, dass der Zuschauer für die gerade dargestellte Sendungsthematik kein oder wenig Interesse hat. Über weitere statistische Verfahren und Model-basiertes Lernen (engl. Model Based Learning) konnten weitere Rückschlüsse auf die der Vorlieben der Zuschauer ermittelt werden.

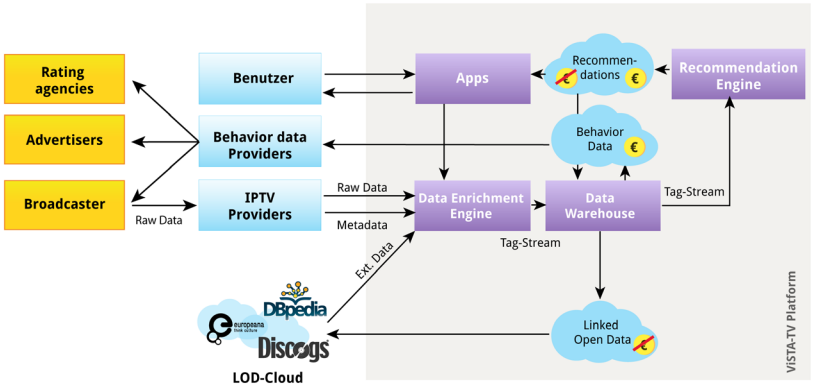


Abbildung 5.30: Gesamtarchitektur der ViSTA-TV-Plattform. Grafisch adaptiert aus [SKB12]

Diese mit der Zeit über den Zuschauer gewonnenen Daten flossen in ein Empfehlungssystem ein, das ein genaues Profil der Zuschauervorlieben generierte. Als Ergebnis wurden den Fernsehsendern nicht nur Zuschauerquoten angeboten, sondern auch detaillierte Statistiken darüber welche Interaktionen bzw. Verhalten (engl. behaviour Data) der Zuschauer während einer Sendung vorwies. Diese Statistiken konnten für Werbezwecke oder von Marktforschungsinstituten weiterverwertet werden. Die Abbildung 5.30 verdeutlicht die Architektur und die verschiedenen Komponenten der ViSTA-TV-Plattform. Das ViSTA-TV-Projekt adressierte zuerst den Wunsch der Fernsehsender ihren Zuschauern besser kennenzulernen und dessen Vorlieben zu erfassen. Dies wurde auf Daten- und Interaktionsebene durchgeführt, indem in Echtzeit Informationen über IP-TV und das Verhalten der Zuschauer anonym erfasst wurden.

Der Ansatz der Anreicherung der EPG-Daten mit Zusatzinhalten aus dem LOD bildet einen ersten Schritt in Richtung semantisches Fernsehen, wobei der Fokus auf der Datenanreicherung liegt. Von dieser wird letztlich nur der Fernsehsender bzw. Marktforschungsinstitute profitieren. Der Bezug zum semantischen Fernsehen ist minimal und kann nur auf Datenebene angesiedelt werden.

Der Aspekt der Echtzeitextraktion und Anreicherung der EPG-Daten bildet jedoch einen interessanten Ansatz, der im Rahmen der Entwicklung des semantischen Fernsehens eine zentrale Rolle spielt.

Smart Video Buddy

Das Projekt MOONVID[49], das vom Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI) bzw. der Gruppe *Multimedia Analysis and Data Mining Lab* und der Technischen Universität von Kaiserslautern geführt wurde, mündete in mehrere Systeme und Technologiedemonstratoren im Bereich der Videoanalyse. Eins davon, InViRe – (Akronym für *Intelligent Video Retrieval*)[50], ermöglichte mittels einer Analyse der Farben, Texturen und Bewegung eines Videos, ähnliche Szenen bzw. Beiträge innerhalb einer Videodatenbank von vorannotierten Fernsehsendungen wiederzufinden. Ein automatisches Lernen von Konzepten durch die Verwendung von statistischen und KI-basierten Verfahren basierend auf Medien von Flickr oder YouTube fand innerhalb der InVire-Komponente statt. Dabei wurden automatisch jedem Video, Schlagwörter (engl. Tags) wie z.B. *Talkshow, Nachrichten, Interview* zugeordnet, die nachträglich für die semantische basierte Videosuche benutzt wurden [SP12] [BKU11]. Zusätzlich bietet der Ansatz, Medieninhalte aus Flickr oder YouTube als Training Set zu benutzen, den Vorteil, urheberrechtlich geschütztes Videomaterial leichter auffinden zu können. Im Technologiedemonstrator Navidgator[51] [VMD09] [BSUB08] wurde diese Ähnlichkeitssuche-Funktionalität für die visuelle Suche innerhalb von Nachrichtensendungen benutzt.

Parallel zur Entwicklung von InViRe wurde das System Smart Video Buddy[52] realisiert. Letzteres ist eine Empfehlungskomponente, die dem Benutzer parallel zu einem abgespielten Video, weiterführende Informationen aus Webseiten (darunter Videoclips und Texte) inner-

[49] http://madm.dfki.de/moonvid
[50] http://www.dfki.de/web/research/km/systems-and-prototypes/InViRe-InFiRe-20101119-englisch.pdf
[51] http://madm.dfki.de/navidgator-howto/navidgator-howto.html
[52] http://madm.dfki.de/_media/tagging-flyer.pdf

halb einer interaktiven grafischen Oberfläche anbietet. Eine automatische Erkennung und Kategorisierung der Videoinhalte ermöglichte zusätzlich die Einblendung produktbezogener Werbung. Die Abbildung 5.31 zeigt die Benutzerschnittstelle von Smart Video Buddy.

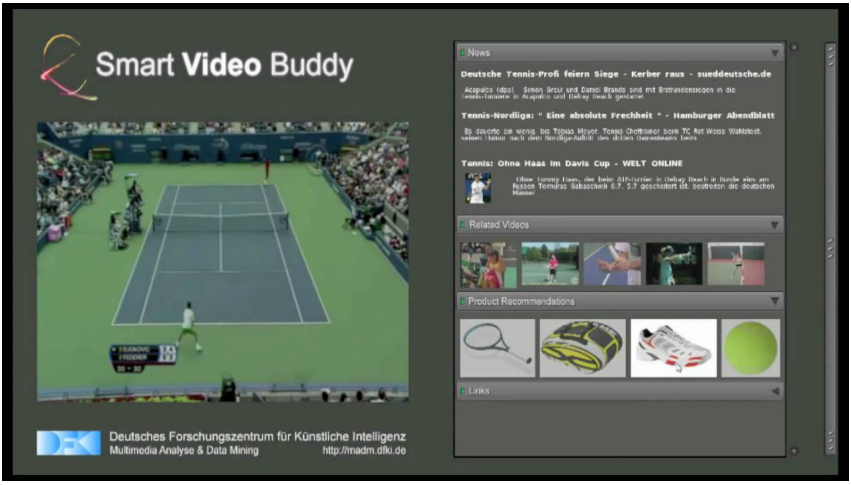


Abbildung 5.31: Benutzerschnittstelle von Smart Video Buddy

Auf der linken Seite der Benutzerschnittstelle wird ein Video abgespielt. Währenddessen berechnet das System in Echtzeit die Wahrscheinlichkeit, dass das Video zu einer der vorgesehen Kategorien (z.B. Fußball, Darts, Schwimmen, usw.) gehört. Wird eine Kategorie gefunden, erfolgt im Hintergrund eine automatische Suche im Web. Als Ergebnis bekommt der Benutzer, auf der rechten Seite der Benutzerschnittstellen, Neuigkeiten, Links und passende Produkte bzw. Werbung über die erkannte Kategorie eingeblendet. Obwohl die Entwicklung von Smart Video Buddy abgeschlossen ist, fokussieren sich die Aktivitäten der Entwickler-Gruppe weiter auf die Echtzeit-Extraktion von Konzepten innerhalb von Videos u.a. mit Verfahren aus dem Gebiet Deep Learning

und der Emotionsanalyse (z.B. mit DeepSentibank) [CBDC14]) [Bor14]. Der Technologie-Demonstrator Smart Video Buddy weist einen direkten Bezug zum semantischen Fernsehen, denn durch das automatische Lernen von Konzepten anhand eines Videos wurde bewiesen, dass eine Verlinkung von Konzepten und Videoinhalten realisierbar ist und mit personalisierten Inhalten und Webseiten verknüpft werden kann.

xLiMe.eu

Das europäische Projekt xLiMe (crossLingual crossMedia knowledge extraction)[53] [ZFT+15] [BKM+15], das von 2013 bis 2016 durchgeführt wurde, hatte als Ziel, Wissen aus verschiedenen Medienkanälen und Modalitäten (Social Web, Blogs, online Video, Tweets, Audio- und Videosignal von einer Fernsehsendung) in unterschiedlichen europäischen Sprachen mit Hilfe semantischer Technologien zu extrahieren.

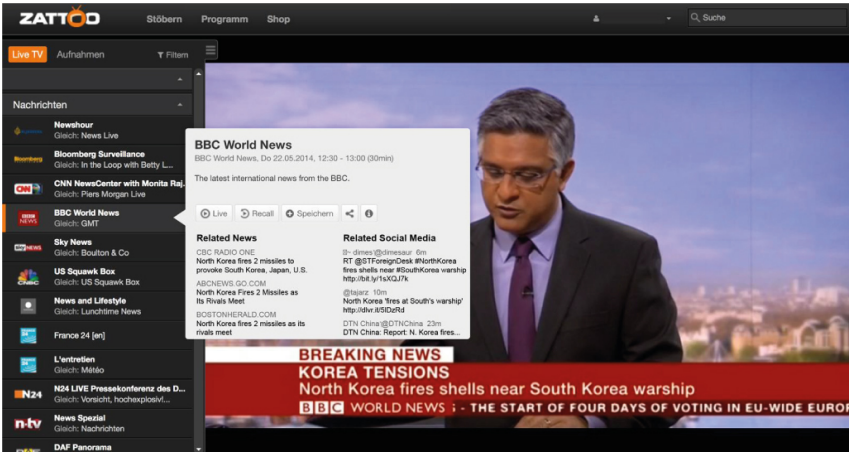


Abbildung 5.32: Screenshot der Zattoo App. Quelle: xLiMe-Poster.

[53] `http://xlime.eu`

Fazit

Durch die Realisierung verschiedener Demonstratoren konnte das Projekt erste Szenarien entwickeln, auch wenn diese nicht direkt auf die Interaktionen mit dem Fernseher zielten. Eines dieser Szenarien war das Zattoo Live TV Recommendations. Die Firma Zattoo, die über ihre eigene App, IP-TV anbietet, konnte somit ihre auf Second Screen angebotenen Sendungen mit angereicherten semantischen Inhalten aus Nachrichtenfeeds und sozialen Medien versehen (siehe Abbildung 5.32, die aus dem xLiMe-Poster[54] stammt).

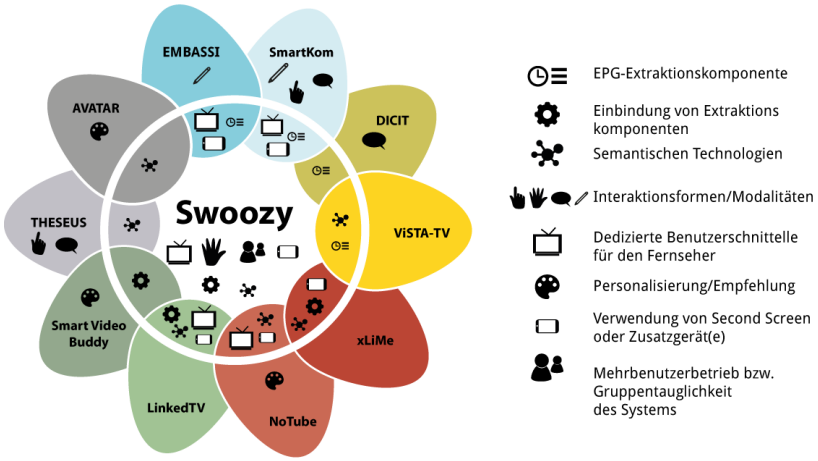


Abbildung 5.33: Thematische Übersicht der Projekte mit einem Bezug zum Fernsehen

Die Abbildung 5.33 fasst alle vorgestellten Projekte zusammen und stellt thematische Überlappungen bzw. Gemeinsamkeiten mit dem Swoozy-System dar. Unter Gemeinsamkeit versteht man, dass die

[54] http://xlime.eu/images/resources_press_material/xlime_poster_YINDAY2015.pdf

Ansätze ähnlich oder annähernd sind. Konzeptuell können diese die Form unterschiedlicher Module, Programmierschnittstellen oder Verfahren nehmen, die trotz ihrer internen Unterschiede zu einem identischen Ergebnis führen. Als Grundlage für die Abbildung 5.33 wurden folgende Kriterien innerhalb der Systeme berücksichtigt:

- Die Benutzung von Extraktionsmechanismen, die EPG-Informationen als Hauptquelle benutzen.
- Die Einbindung von video- bzw audiobasierten Extraktions- und Analysekomponenten.
- Der Einsatz von semantischen Technologien (Anwendung von Ontologien, RDF-Stores bzw. die Benutzung vom Semantic Web und von Wissensdatenbanken wie z.B. DBpedia oder Wikidata) zur Unterstützung einer Suche oder zur Beschreibung der internen benutzten Konzepte.
- Die Integration dedizierter Interaktionsformen und Eingabemodalitäten (Gesten-, Touch- und Sprachinteraktionen) in Zusammenhang mit einer Benutzerschnittstelle.
- Die Implementierung und Verwendung einer dedizierten Benutzerschnittelle für den Fernseher. Hier wird dargestellt, ob im Rahmen des vorgestellten Systems eine Benutzerschnittstelle entwickelt wurde. Die Benutzung von Designleitlinien spielt bei diesem ebenso eine Rolle.
- Der Einsatz von Personalisierungs- und Empfehlungsmodulen, die dazu beitragen, dass dem Benutzer selektierte und individualisierte Informationen bzw. Inhalte, nach einer Analyse seines Verhaltens angeboten werden.

- Die Verwendung von Second Screen als Zusatzgerät für die Interaktion mit Medienobjekten und den Funktionalitäten des Fernsehgerätes.
- Die Mehrbenutzerbetrieb- bzw. Gruppentauglichkeit des Systems. Hier stellt sich die Frage, ob das System von mehreren Personen gleichzeitig benutzt werden kann, sei es mittels eines Second Screens oder direkt am Fernseher durch eine dedizierte Benutzerschnittstelle.

Die farblichen Überlappungen symbolisieren die Themen und technischen Aspekte, die in den jeweiligen Projekten berücksichtigt worden sind und Gemeinsamkeiten mit dem Swoozy-System vorweisen. Auch wenn manche Projekte wie NoTube, xLiMe oder LinkedTV sehr viele Aspekte der semantischen Extraktion und Verarbeitung berücksichtigen, muss immer genauer betrachtet werden, ob es sich um einzelne separate Softwaremodule oder komplette Lösungen handelt.
Es ist nicht bei jedem Projekt ersichtlich, ob die implementierten Module tatsächlich in einem der technischen Demonstratoren zum Einsatz gekommen sind. Unklar ist bei jedem der vorgestellten Systeme, inwiefern diese in der Lage sind, in Echtzeit zu arbeiten bzw. eine Echtzeitanalyse und Extraktion der verschiedenen Audio- und Videoinformationen durchzuführen.

Zusammenfassung: Bausteine für die semantische Interaktion

Die vorangegangene Analyse hat gezeigt, dass die technischen Grundlagen und Konzepte für das semantische Fernsehen zwar schon teilweise vorhanden, aber nur in Teilkomponenten und in sehr heterogener Form benutzt und eingesetzt worden sind. Größtes Manko bei den bestehenden Ansätzen ist die fehlende semantische Interaktion. Unter semantischer Interaktion versteht man einerseits eine klare semantische Beschreibung der zu bedienenden Elemente und andererseits, dass jede Aktion des Benutzers eine interne eindeutige Repräsentation aufweist. Das bedeutet im ersten Schritt, dass jedes grafische Element über eine Ontologie oder Taxonomie beschrieben wird und dass diese als Modell für die komplette Benutzerschnittstelle dienen soll. Ein Menüpunkt oder Schaltelement erhält eine eindeutige Identifikationsnummer, die es später ermöglicht, alle Aktionen dieses Elements nachträglich zu referenzieren oder über eine erweiterte Modalität (Zeigegeste oder Sprachkommando) auf dieses zuzugreifen. Bei diesem Referenzieren bilden die Art des Auslösens von Funktionen (d.h. welche Funktion wird durch die Benutzung des Elements getriggert) und der Zeitpunkt (z.B. über einen Zeitstempel), wann die Benutzerinteraktion stattfindet, die wesentlichen Merkmale der Interaktionsbeschreibung. All diese definierten Eigenschaften werden innerhalb eines Modells, das entweder die Form einer hierarchischen Struktur oder einer Ontologie besitzt, beschrieben. Mit diesem generischen Ansatz lassen sich die Ergebnisse einer Websuche beschreiben und korrekt anordnen bzw. referenzieren und grafisch kontextorientiert darstellen. Wenn man von einer semantischen Suche

im Kontext eines Multimediasystems spricht, stammen die Ergebnisse aus unterschiedlichen Wissensquellen und müssen, bevor diese grafisch dargestellt werden können, erst zusammengefügt und mit einem einheitlichen und generischen Vokabular versehen werden. Dieses Vokabular beschreibt, *was* abgebildet ist, d.h. den Typ (Bilder, Text, Audioinhalt) des Ergebnisses, und *wann* und *in welcher Quelle* (Webdienst) es gefunden wurde. Diese Informationen können als interne Referenz mitgespeichert werden. Anschließend muss die Präsentationskomponente diese Ergebnisse analysieren und korrekt darstellen. Dafür stützt sie sich auf deren semantische Beschreibung. Über eine Referenznummer kann der Benutzer die Ergebniselemente adressieren. Unter Adressierung versteht man, dass jedes Element über seine interne Referenz jederzeit aufgerufen und benutzt werden kann, sei es mittels Sprache, Geste oder Augenbewegung. Dieses Konstrukt bildet die Grundlage eines semantischen Interaktionssystems und der damit verbundenen semantischen Interaktion.

Die letzten Abschnitte und die Beschreibung der Projekte haben gezeigt, dass alle technischen Voraussetzungen für die Konzeption eines Grundgerüsts für ein semantisches videogestütztes System vorhanden sind. Die technische Integration und Kombination dieser meist heterogenen Komponenten stellen die größte Herausforderung dar. Zusätzlich muss die eigentliche Semantik (d.h. was kann repräsentiert und rückreferenziert werden) der dargestellten grafischen Komponenten festgelegt werden. Dies kann über eine einheitliche generische Struktur, die in das System eingebunden wird, realisiert werden. Des Weiteren zeigt sich, dass semantische Webdienste wie DBpedia, Cloudbasierte Dienste wie AYLIEN und intelligente Verarbeitungskomponenten wie z.B. NEMEX in der Lage sind, in Echtzeit einen direkten Zugriff auf multimediale Informationen zu ermöglichen und diese intelligent

miteinander zu verknüpfen. Die heutigen beliebten APIs und REST-basierten Programmierschnittstellen vereinfachen den Zugriff auf das Semantic Web und auf *„Semantifizierungs"*-Komponenten, insbesondere durch die Benutzung von online-basierter Entitätsextraktion.

Zu den bestehenden aktuellen Ansätzen zeigen die gewonnenen Erkenntnisse aus der Analyse, dass die heutigen Smart-TVs noch sehr stark App-zentrisch arbeiten und wegen technischer, aber auch kommerzieller Überlegungen, kaum Spielraum für ein breiteres Spektrum an Interaktionskonzepten anbieten. Dies führt sehr schnell zu Interaktionsbrüchen und nicht-intuitiven Bedienkonzepten. Die Systeme *CoMET*, *Calisto* und *Cirius* haben gezeigt, dass die Anwendung einer semantischen Beschreibung innerhalb eines Multimediasystems deutliche Vorteile in Bezug auf Interaktion, Dialogfähigkeit und Multimodalität mit sich bringt. Diese „mehrschichtige semantische Dimension" fehlt immer noch bei den heutigen Smart-TVs und bei den vorgestellten Projekten.

Alle Erkenntnisse, die durch die Analyse der verwandten Arbeiten inklusive Forschungsprojekten, der Funktionsweise des digitalen Fernsehens und der Ansätze zur Erstellung semantisch-gestützter Systeme gewonnen wurden, sind in die Konzeption und Erstellung des im nächsten Kapitel vorgestellten Systems *Swoozy* eingeflossen.

6 Konzept und Realisierung von Swoozy. Das semantische Fernsehen

Dieses Kapitel befasst sich mit der Realisierung des *Swoozy*-Systems, das erste technische Ansätze in Richtung des semantischen Fernsehens aufweist und in Form eines integrierten Multimediasystems zeigt, wie das Zusammenspiel zwischen aktueller Fernsehinfrastruktur, Fernsehgeräten und dem Semantic Web (Web 3.0) mit intuitiven Interaktionsformen realisiert werden kann. Dem Zuschauer bieten sich dadurch neue Möglichkeiten mit dem Fernsehen zu interagieren. Im ersten Schritt werden das zu Grunde liegende Systemkonzept und der Weg zur Systemrealisierung beschrieben. Hierbei wird das Konzept des semantischen Fernsehens näher erläutert, insbesondere wie das Zusammenspiel mit der aktuellen DVB-Infrastruktur konkretisiert werden kann.

Die komplette Systemarchitektur von Swoozy, von der textuellen Wissensextraktion aus TV-Programmen, d.h. aus dem EPG oder aus dem Live-Videomaterial, bis zur Bearbeitung der Ergebnisse der aufgerufe-

nen Webdienste, wird ebenfalls vorgestellt. Dafür werden die Verfahren präsentiert, die es ermöglichen, ein beliebig ausgestrahltes Video mit Inhalten aus dem Web anzureichern und dadurch Zusatzinformationen zu gewinnen. Hierzu werden insbesondere die OCR-Verfahren und die semantische Extraktion aus Texten näher erklärt. Diese Aufgaben werden verteilt vom Swoozy-Server ausgeführt. Seitens des Zuschauers spielt der Swoozy-Client die wesentliche Rolle.

Der Swoozy-Client hat nicht nur die Aufgabe, die grafische Oberfläche zu generieren und die Suchergebnisse grafisch aufbereitet darzustellen, sondern muss die Interaktionen des Benutzers korrekt interpretieren. Dabei spielt die grafische Aufbereitung der Benutzeroberfläche und der benutzerzentrierten Anzeige der Ergebnisse eine zentrale Rolle. Im letzten Teil des Kapitels wird die Zusammenarbeit von SwoozyML (Swoozy-Markup Language) mit der in Echtzeit ablaufenden semantischen Verarbeitung dargestellt, um die enge Verbindung zwischen Diensten, Ergebnissen, Semantik und Fernsehbildmaterial besser zu beschreiben. Als letzter Punkt werden die Strategien und Herausforderungen der automatischen Extraktion von Semantik aus Live-Fernsehbildmaterial näher betrachtet und ihre Semantifizierung thematisiert. Zusätzlich werden die Fragen der Granularität und des Zeitpunktes der Anzeige von Annotationen unter einem gestalterischen Aspekt betrachtet, damit ein akzeptables Gleichgewicht zwischen „zu viel“ und „zu wenig“ Live-Informationen innerhalb der grafischen Benutzerschnittstelle von Swoozy erreicht werden kann.

Konzept von Semantic TV

Einleitung

Wie bereits in den vorherigen Kapiteln dargestellt, ist das Ziel die Realisierung eines semantischen Systems, das parallel zu einem laufenden Video, wie z.B. VOD, Live-Fernsehen über DVB, relevante Medienobjekte aus dem Semantic Web dank Annotationen findet und diese innerhalb einer dedizierten Benutzerschnittstelle am Fernseher anzeigt. Da das semantische Fernsehen Live-Fernsehbildmaterial als Eingabe bearbeitet und auf der aktuellen Infrastruktur des DVB-Ausstrahlungs- bzw. Empfangsstandards und des Internetzugangs beruht, mussten neue Strategien entwickelt werden, die trotz noch nicht vorhandenen semantischen Beschreibungen, Annotationen oder, im minimalen Fall, Kontextinformationen innerhalb ausgestrahlter Videos oder Filmen, in der Lage sind, diese Zusatzinformationen aus einer Live-Sendung zu extrahieren.

Das Grabbable-Konzept

Das Kernkonzept des vorgestellten Swoozy-Systems wird das *Grabbable* bilden. Ein Grabbable ist die grafische Ausprägung einer erzeugten Annotation, die aus einer Echtzeitanalyse extrahiert wurde. Diese Annotation (auch intern als *Semantic Term* bezeichnet) beinhaltet eine textuelle und semantische Beschreibung der erkannten Entitäten wie z.B. eine Person, ein Objekt, ein Ort oder ein Produkt. Grafisch werden die *Grabbables* als Kästchen dargestellt, die über die komplette Benutzerschnittstelle „per Geste mitgezogen" und als generische Eingabe für die Suche benutzt werden können.

Das Design der grafischen Form der *Grabbables* war dadurch motiviert, dass erstens das Konzept (im semantischen Sinne) und der Name erscheinen sollten, damit dem Benutzer die Information über eine bestimmte Szene oder präsentierte Gegenstände schnell bereitgestellt wird. Aus diesem Grund wird immer neben dem Namen ein Icon eingeblendet, welches das Konzept (z.B. Objekt, Person, Ort) angibt. Die puzzleartige Form des *Grabbables* stellt visuell sicher, dass dieses nur in Kombination mit einer *Dropzone* (grafischer Eingabepunkt für eine gezielte und typisierte Suchanfrage ähnlich den *Spotlets*-Mechanismus verwendet werden kann. Die Schnelligkeit der Interaktion war ein Hauptkriterium, welches der Konzeption der Benutzerschnittstelle zu Grunde lag. Diese Schnelligkeit wird dadurch gewährleistet, dass die *Grabbables* durch eine einzige Gesteninteraktion vom Benutzer „gefasst" und mittels einer Wischgeste in eine der fünf *Dropzonen* „fallen gelassen" werden können.

Abbildung 6.1: Grabbables für das Gebäude „Kolosseum" und die Stadt „Rom" innerhalb der Benutzeroberfläche von Swoozy

Dadurch wird die Zeit zwischen der benutzerinitiierten Selektion und der Anfang des Suchvorganges (Drop in die *Dropzone*) stark reduziert. Das Konzept der *Dropzone* und die Kombination auf Interaktionsebene zwischen einer Dropzone und einem Grabbable werden in Abbildung 6.21 expliziter dargestellt. Die *Grabbables* können wahlweise proaktiv vom Benutzer über eine Geste generiert werden. Sie erscheinen als grafische Überlagerung (engl. Overlay) über dem Video oder innerhalb einer grafischen Leiste. Diese Leiste befindet sich im unterem Bereich und blendet entweder die herkömmliche allgemeine Information über das laufende TV-Programm (als EPG-Text) ein oder zeigt die zeitgesteuerten *Grabbable* an. Der Benutzer kann wieder per Gesteninteraktion innerhalb der Leiste einen dieser semantischen Begriffe selektieren und per Grab'n'Drop-Interaktion als Suchelement für eine webbasierte Suche benutzen.

Jedes *Grabbable*, das in eine der *Dropzones* eingefügt (oder besser gesagt, „gedroppt") wird, ist mit einer Annotation versehen, d.h. es besitzt eine eindeutige semantische Repräsentation. Das bedeutet u.a., dass eine Suche über eine Person eine andere interne Bedeutung hat als z.B. eine Suche über ein Objekt. Auch wenn visuell nur der Name der gesuchten Person als Eingabeparameter benutzt wird, kann die semantische Suche im Hintergrund mit anderen Zusatzinformationen angereichert (z.B. Name, Vorname, Geschlecht und Beruf der Person) werden. Diese semantischen Zusatzinformationen, die dem Benutzer aus Einfachheits- und Übersichtlichkeitsgründen nicht angezeigt werden können, werden von der semantischen Suche benutzt. Dieser Ansatz folgt dem „*No presentation without representation*"-Paradigma. An jedem grafisch dargestellten *Grabbable* hängt immer eine für den Benutzer unsichtbare semantische Struktur.

Herausforderungen

Aus obiger Definition des semantischen Fernsehens und der Vorstellung der Interaktion mit Grabbables ergeben sich für eine erfolgreiche technische Umsetzung des Konzepts verschiedene Herausforderungen konzeptueller und technischer Art, die gemeistert werden müssen. Ausgehend von dieser Betrachtung ergeben sich folgende konkrete Fragen, die innerhalb des Kapitels beantwortet werden:

- **Analyse und Wissensextraktion aus TV-Programmen und Videosignalen**

  Im DVB-Datenstrom sind Informationen zu EPG, HbbTV oder zu Programmkategorien eingebettet. Zusätzlich ermöglichen Komponenten wie die Video- und Audioanalyse die Live-Extraktion von Zusatzinformationen.

  Wie können die grundliegenden DVB-Signalspuren benutzt werden, um angereicherte Informationen aus einem TV-Programm zu gewinnen?

- **Einbeziehung von Zusatzinformationen**, die vom Fernsehsender ausgestrahlt werden (z.B. Audiodeskription, Telextext, HbbTV) und über Zusatzkanäle angeboten werden

  Fernsehsender bieten über verschiedene Nebenkanäle wie Mediatheken oder über HbbTV-Angebote die Möglichkeit, gezielt an nicht semantifizierte, aber für den Zuschauer dennoch interessante Zusatzinformationen, zu gelangen. Wie können diese Informationen in den Analyse-Workflow einfließen und eingebunden werden?

- **Semantifizierung und Extraktion** bilden die Kernkomponenten des Systems. Hier wird das Zusammenspiel zwischen Fern-

sehsendung und Live-Extraktion von Wissen und deren semantischer Abbildung skizziert.

- **Grafische Benutzerschnittstelle und Einblendung von Annotationen**

  Wie kann der Zuschauer auf einfache Art und Weise innerhalb einer Sendung die Präsenz von Grabbables wahrnehmen und diese für eine Suche benutzen?

- **Suchmechanismen zum Semantic Web**

  Wie kann der Zuschauer auf intuitive Weise eine semantische Suche von seinem Fernseher starten? Wie können die verschiedenen heterogenen Webdienste wie Twitter, Google Knowledge Graph oder DBpedia aufgerufen und eingebunden werden? Wie werden diese Ergebnisse einheitlich, homogen und strukturiert aufbereitet und bereitgestellt?

- **Visualisierung der Ergebnisse**

  Wie lassen sich verschiedene Ergebnisarten (z.B. Videos, Texte, Bilder, Tweets oder Shopping-Empfehlungen) benutzerfreundlich darstellen?

- **Verwaltung der Eingabemodalitäten**

  Welche Möglichkeiten, abgesehen von der normalen Fernbedienung, besitzt der Benutzer, um eine Selektion oder eine Websuche vorm Fernsehschirm sitzend zu initiieren?

- **Ausgabestrategien der Suchergebnisse für unterschiedliche Clients** (TV oder Tablets)

  Welche Anforderungen an die grafische Darstellung müssen respektiert werden, damit die „User Experience" für den Benutzer optimal

ist? Und welche Informationen dürfen/müssen auf dem sog. Second Screen *(d.h. auf einem zusätzlichen Medium wie ein Tablet oder Smartphone) angezeigt werden?*

- **Technische Realisierung und Architektur des Swoozy-Frameworks**

  Welche Soft- und Hardware-Komponenten werden für das Framework benutzt? Wie lassen sich diese in die heutige Fernsehsenderinfrastruktur einbinden?

- **Generalisierung des Swoozy-Ansatzes über SwoozyML**

  Hier wird SwoozyML als ein leichtgewichtiges Annotations-, Präsentations- und Austausch-Format für das semantische Fernsehen vorgestellt, das Fernsehredakteuren helfen soll, zu einer Sendung semantische Informationen hinzuzufügen.

Die oben vorgestellten Fragen beziehen sich primär auf die *zuschauerseitige* Variante des semantischen Fernsehens. Unter *zuschauerseitig* versteht man alle Prozesse, die ab dem Empfang des DVB-Signals beim Zuschauer bzw. auf seiner Set-Top-Box laufen, inklusive Interaktion und Anzeige der grafischen Benutzeroberfläche. Als Bezeichnung für diese Komponente wird im Folgenden der Begriff *Swoozy-Client* benutzt.

Neben den Clients spielt der *Swoozy-Server* eine wichtige Rolle innerhalb des Systems. Dieser verwaltet und analysiert die Fernsehbildinformation in Echtzeit und stellt die daraus extrahierte Information den Clients zur Verfügung. Hier spielen die Fragen, wie semantische Daten zum Live-Fernsehbild automatisch hinzugefügt und in einem nächsten Schritt den Clients angeboten werden können, eine entscheidende Rolle. Dieser Schritt muss oft redaktionell unterstützt werden und wirft folgende Fragen auf:

- **Automatische Erkennung der Entitäten**, d.h. von Personen, Gegenständen, geographischen Orten oder fiktiven Charakteren. *Wie lassen sich Informationen z.B. aus einem Fußballspiel oder Filmtrailer extrahieren, dass Schauspieler oder Protagonisten erkannt und in die Form einer semantischen Repräsentation umgewandelt werden können?*
- **Verfahren zur Echtzeitextraktion** *Welche Verfahren sind am geeignetsten, um Informationen zu extrahieren?*
- **Serverseitige Erzeugung von Informationen** *Welche technischen Verfahren sind nötig, damit ein zeitgesteuertes Signal geschickt werden kann, um das Erscheinen der Suchbegriffe synchron und kohärent zu einer Live-Sendung oder einem Video (d.h. passend zum angezeigten Videoeinzelbild) gewährleisten zu können?*

Terminologien

Zum besseren Verständnis des vorgestellten Swoozy-Systems werden im Folgenden Begriffe und Terminologien definiert, die anschließend in den einzelnen Abschnitten dieses Kapitels Verwendung finden. Die folgende Liste verhilft dem Leser zu einem besseren Verständnis der Grundaspekte des Systems, bevor sie in den jeweiligen Abschnitten ausführlicher präsentiert werden.

- Ein *Semantic Term* ist eine kompakte, intern benutzte, semantische Repräsentation einer im Videobild erkannten Entität. Der Begriff setzt sich aus dem Konzept (im ontologischen Sinne) und einer internen Referenz (z.B. auf weitere ontologische Begriffe) bzw. einer externen Verlinkung (z.B. auf DBpedia oder Wikidata)

zusammen. Die grafische Ausprägung, die für den Benutzer sichtbar ist, bildet das Grabbable (siehe Abbildung 6.1). Die Begriffe *Semantic Term*, *Grabbable* und *Annotation* sind im Falle einer vom Benutzer initiierten Interaktion Synonyme.

- Die Begriffe *Swoozy-Framework* und *Swoozy-System* sind Synonyme. Darunter verstehen wir die komplette technische Infrastruktur, welche die Funktionalitäten von Swoozy verwaltet und technisch unterstützt. Wenn nicht explizit erwähnt, ist immer der *Swoozy-Server* (oder die Serverkomponente) und nicht der Client gemeint. Das System ist insoweit flexibel ausgelegt, als dass neue Zusatzfunktionalitäten und Dienste nahtlos und mit minimalen Anpassungen integriert werden können.

- Unter *Benutzerschnittstelle*, *TV/Fernsehoberfläche*, *Swoozy-Interaktionsfläche* und *Swoozy-UI* verstehen wir die grafische Umsetzung und die kombinierte Anzeige von Fernsehvideosignalen, dem Swoozy-Design und den überlappenden Interaktionselementen (z.B. Menüs und Ergebnisliste). Diese Begriffe, die primär das bezeichnen, was Zuschauer bzw. Benutzer sehen, können mit dem Wort *Swoozy-Client* bzw. *UI-Client* bezeichnet werden. Die grafische Ausgabe und Anzeige der Ergebnisse werden auf dem Fernsehbild realisiert. Als *Client* bezeichnet man nicht nur die grafische Darstellung, sondern auch die dahinterstehenden dafür zuständigen Präsentationsmodule, die dazu beitragen eine benutzerzentrierte grafische Ausgabe auf dem Fernseher einzublenden.

  Als Pendant zum *Client* realisiert der *Swoozy-Server* remote und vernetzt die Rolle der Bildanalyse, Wissensextraktion und der semantischen Suche. Der Server bildet die Kernkomponente des

Swoozy-Systems, indem er die Aufgabe der Verarbeitung der Suchergebnisse übernimmt und diese in generischer Form den angebundenen Clients weiterreicht. Eine weitere Aufgabe des *Swoozy-Servers* ist es, den Clients über SwoozyML bestimmte Befehle wie z.B. der Zeitpunkt der Anzeige der *Grabbables* auf die Benutzeroberfläche im Falle einer Live-Sendung zu senden. Im Falle einer integrierten Hardwarelösung befinden sich Client- und Swoozy-Server auf der gleichen physischen Hardware, z.B. einer Set-Top-Box. Dies lässt sich zwar technisch realisieren, möchte man aber einem distribuierten Ansatz folgen, müssen beide Komponenten getrennt voneinander laufen und sich über Netzwerke austauschen. Daher wird im Rahmen dieses Buch auch gerne der Begriff *Cloud-basiertes Fernsehen bzw. Cloud-basierte Lösung* benutzt.

Architekturübersicht

Das Swoozy-System wurde modular aufgesetzt und benutzt generische Softwarekomponenten zur Analyse und Verarbeitung der Signale und Ergebnisse, die jederzeit durch andere ersetzt werden können. Dieser Philosophie wurde sowohl auf Server- als auch auf Client(s)-Ebene gefolgt, sodass sich das System erweitern lässt, ohne dabei die grundlegende Architektur ändern zu müssen. Die Abbildung 6.3 zeigt im Detail den internen Aufbau und Workflow der Module seitens des Swoozy-Clients und Swoozy-Servers. Ein anderes wichtiges Merkmal ist die komplette Unabhängigkeit des implementierten Systems von der Art des DVB-Systems oder des Videosignals, das als Signaleingangsquelle benutzt wird. Die Komponenten wie z.B. der *De-*

muxer oder der *Dekoder* (Schritt 2 in Abbildung 6.3) müssen dazu nur leicht angepasst werden. Diese Veränderungen haben keinen direkten Einfluss auf Komponenten wie die Ergebnisvisualisierung oder die semantische Analyse. Aus dem Grund wurde die Entscheidung getroffen, für den Swoozy-Prototypen das Videosignal eines DVB-T-Empfängers zu benutzen, da DVB-T aus Sicht der Anschaffungskosten für die Hardware eine günstigere Empfangsmethode (ca. 9 Euro für einen DVB-T Stick) (1) gegenüber DVB-S oder DVB-C (ca. 49 Euro exkl. Anschluss) darstellt. Eine spätere Variante von Swoozy (siehe Kapitel 8) benutzt einen Sat-To-IP-Empfänger, um das Fernsehsignal einzubinden.

Das Swoozy-System ist in zwei distinkte Komponenten unterteilt: den Swoozy-Server und die Swoozy-Clients. Der Swoozy-Server verwaltet zwei Aufgaben. Die erste davon besteht in der Echtzeitanalyse von DVB-T-Signalen und der semantischen Extraktion von Informationen aus dem Live-Videomaterial durch Verfahren wie OCR oder videobasierte Analyseverfahren (3,4 und 5).
Die zweite Aufgabe des Swoozy-Servers konzentriert sich auf die Erzeugung der *Grabbables* aus den extrahierten Inhalten der Echtzeitanalyse (6). Die extrahierten Annotationen beinhalten eine textuelle und semantische Beschreibung der erkannten Entitäten (z.B. Person, Objekt, Ort, usw.) mit ihren dazugehörigen Namen. Die *Semantic Terms* bzw. *Grabbables* können mittels einer leichtgewichtigen selbst-definierten Ontologie oder Taxonomie (9,10) angeglichen werden und es können, falls es notwendig ist, zusätzliche Begriffe über externe Werkzeuge wie Swoozy-Livana oder Swoozy-SKRPTR (siehe Kapitel 7) in Echtzeit redaktionell hinzugefügt werden. Im letzten Schritt wird das Endergebnis der Analyse durch den Swoozy-Server über eine dedizierte REST-Schnittstelle als JSON-Struktur oder SwoozyML den Clients zur

Verfügung gestellt (12). Auf Swoozy-Client-Seite wird das Fernsehbild samt Swoozy-Benutzeroberfläche angezeigt (12). Dieses Fernsehbild kann aus verschiedenen Quellen stammen. Im Falle der Set-Top-Box Variante von Swoozy kommt dieses Signal direkt von der integrierten DVB-T-Karte, optional von einem IP-basierten Videostream oder einer HDMI-Videograbberkarte.

Bei den ersten Versionen von Swoozy wurde das Signal von einer Raspberry-PI basierten Streamingkomponente bearbeitet. In den neueren Versionen stammt das Signal von einem IP-basierten Videostream, der über einen zwischengeschalteten Sat-To-IP-Empfänger erzeugt wird. Abbildung 6.2 zeigt die Raspberry-PI-Hardware, die für den verteilten stream-basierten Empfang benutzt wurde.

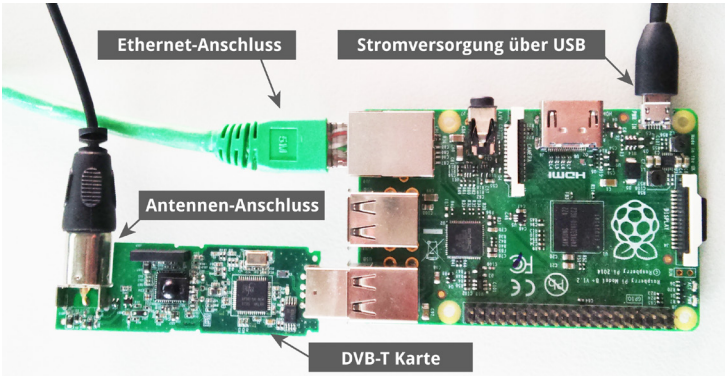


Abbildung 6.2: Raspberry-PI mit DVB-T Empfänger

Die grafische Benutzeroberfläche des Clients ermöglicht dem Benutzer mit dem Swoozy-System zu interagieren und per Gesten nach den eingeblendeten *Grabbables* im Web zu suchen. Die Gesten und Eingabemodalitäten werden unabhängig von der Hardware verwaltet. Im Fall der bereits realisierten Demonstratoren können diese miteinander kombiniert und gleichzeitig die Microsoft Kinect, die Leap Motion

und eine Gyroskop-basierte Fernbedienung für die Interaktion mittels Gesten benutzt werden.

Die Verwaltung der Hardwareschicht (z.B. Auswahl und Verwaltung der Eingabesignale der Kinect und der Leap Motion) wird autonom durchgeführt und ist für den Benutzer nicht sichtbar, sodass er sich rein auf die Interaktion konzentrieren kann. Die Verarbeitung und Erkennung der verschiedenen Komponenten wird über die sog. Hardwareschicht des Clients durchgeführt.

Stellt ein Benutzer eine Suchabfrage, so wird diese zum Swoozy-Server über eine REST-Schnittstelle (7) weitergereicht und dort nach einer Analyse des Abfragetyps (z.B. eine Bild- oder Videosuche)- zu den jeweiligen Cloud-basierten Diensten (8) weitergeleitet. Die Ergebnisse dieser Dienste werden mittels der leichtgewichtigen Swoozy-Ontologie angeglichen (10) und ggfs. überprüft und mit Zusatzannotationen angereichert (z.B. mit einer Wikidata-Referenz). Nach diesem letzten Verarbeitungsschritt (11) werden die Ergebnisse entweder in SwoozyML oder in Form einer JSON-Struktur zum Client weitergereicht. Dieser bearbeitet die erhaltenen Ergebnisstrukturen, die meistens aus Multimediadaten (Bild, Text, Videos) bestehen. Außerdem verwaltet er ihre Anzeige auf der Benutzeroberfläche des Clients (12).

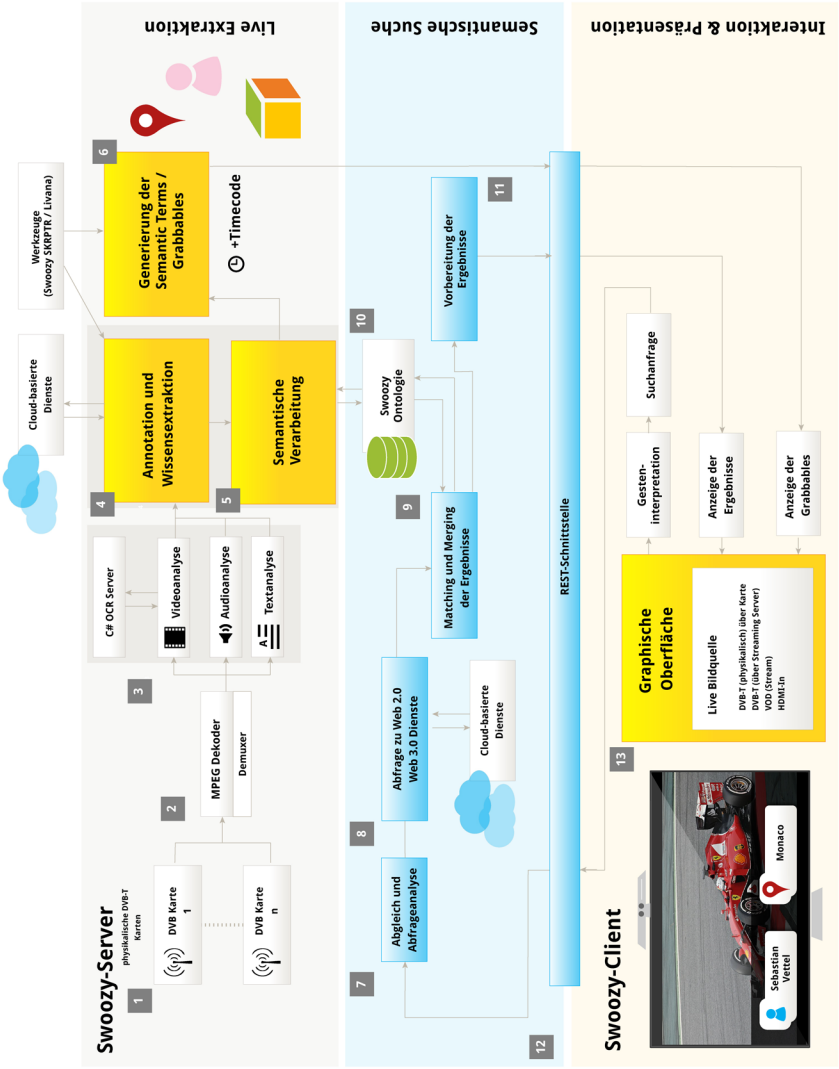


Abbildung 6.3: Architekturbild des Swoozy-Systems

Swoozy-Server

Einleitung

Die zentrale Komponente der Architektur ist der Swoozy-Server. Der Swoozy-Server besitzt verschiedene Rollen und Ebenen, u.a. eine TV-Programmbezogene Analyse durch den intelligenten Aufruf der unterschiedlichen Module. Im folgenden Abschnitt werden die Komponenten und Funktionsweisen innerhalb des Swoozy-Servers näher beschrieben und die Zusammenhänge der Module untereinander aus einem technischen Gesichtspunkt präsentiert.

Technische Architektur

Der Swoozy-Server setzt komplett auf der node.js[1] Plattform auf. Node.js ermöglicht ein unabhängiges Aufsetzen von Webanwendungen und Webservern mittels der Programmiersprache Javascript und deren Interpretern, der Google V8 Javascript Engine. Node.js ist Event-getrieben und folgt dem Ansatz der asynchronen, nicht blockierenden Verarbeitung von Daten. Daher ist sie für leichtgewichtige datenintensive Webapplikationen besonders gut geeignet. Eine asynchrone, nicht blockierende Verarbeitung bedeutet, im Gegensatz zu einer synchronen, dass die Funktionen, die viele Ressourcen brauchen, den „Aufrufer“ (hier wäre es der Client) nicht blockieren. Somit können zeitintensive Funktionen parallel ausgeführt werden. Zusätzlich ermöglicht node.js dank seiner Architektur den Betrieb von Echtzeitapplikationen und die direkte und unkomplizierte Unterstützung von (Daten-)Streaming. Node.js ist plattformunabhängig, sodass die implementierte Serverlösung auf sog. Embedded Plattformen lauffä-

[1] https://nodejs.org/en/

hig installiert werden kann. Aus all diesen Gründen wurde node.js als Kernservertechnologie für die Swoozy-Plattform ausgewählt.

Umsetzung

Bei der implementierten Swoozy-Architektur wurde node.js als Servertechnologie benutzt, da einerseits verschiedene (semantische) Webdienste parallel aufgerufen und deren Ergebnisse zusammengefügt werden müssen und andererseits die Hauptaufgabe des Servers in einer schnellen Verarbeitung der Ergebnisse und der Verbindung mit einer semantischen Repräsentation besteht. Zusätzlich können neue Ergebnisse und Beschreibungen manuell über Werkzeuge wie Swoozy-Livana und SKRPTR, hinzugefügt werden. Alle diese Aufgaben werden parallel vom Swoozy-Server erledigt. Damit Befehle, z.B. vom Server bis zum Client und umgekehrt, versendet und verarbeitet werden können, bietet der Webserver eine REST-Schnittstelle (Punkt 13 in der Abbildung 6.3). Diese ermöglicht das Versenden von Kommandos, die z.B. vom TV-Client oder der Swoozy-Benutzeroberfläche initiiert worden sind. Der verteilte Ansatz ermöglicht dem Swoozy-Server über eine DVB-T-Karte mehrere Kanäle eines Multiplexes gleichzeitig zu empfangen und zu analysieren. Verfolgt und entwickelt man diesen Ansatz weiter, kann jeder Fernsehkanal bzw. Multiplex mit einem oder mehreren anderen Kanälen gleichzeitig analysiert werden. Die eigentliche Verarbeitung, die Bildanalyse und die textuelle Analyse erfolgen verteilt und auf nur für diese Aufgabe dedizierten Rechnern. Diese Verteilung der Verarbeitung von Bild- bzw. Videoanalyse und der Aufbau des Swoozy-Servers ermöglichen ein modulares Aufsetzen aber auch ein Ersetzen der verschiedenen Komponenten, ohne dass es zu Integrationsproblemen führt. Durch diesen generischen Ansatz ist es

möglich, problemlos die Textanalysekomponente durch eine andere zu ersetzen bzw. mit anderen Diensten aus der Cloud zu kombinieren, ohne dass der komplette Analyseworkflow davon betroffen ist.

Analyse und Wissensextraktion aus TV-Programmen

Konzept und technische Architektur

Die Extraktion von Zusatzinformationen aus dem eigentlichen DVB-Signal stellt eine der Kernaufgaben des Swoozy-Systems oder, besser gesagt, der Swoozy-Server-Komponente dar. Mit *Analyse- und Wissensextraktionskomponenten* sind mehrere Softwaremodule gemeint, die in der Lage sind, aus dem Video und dem darunterliegenden DVB-Datenstrom die einzelnen Datenpakete zu dekodieren, in textuelle Form umzuwandeln, und anschließend in ein maschinenlesbares Format zu konvertieren, um sie von weiteren Komponenten verarbeiten lassen zu können. Abbildung 6.4 zeigt als Übersicht die grundlegende Funktionsweise der Analyse eines DVB-Videostreams innerhalb von Swoozy. Da das System verteilt funktioniert, können dank dieser Architektur mehrere Fernsehkanäle (oder Multiplexe) gleichzeitig analysiert werden, da jede Analysekomponente über ihre eigene DVB-Karte verfügt. Wäre nur eine „zentrale" DVB-Karte im System integriert worden, wäre nur eine beschränkte Anzahl von Fernsehkanälen gleichzeitig analysierbar. Innerhalb der ersten Komponente (Punkt 1 der Abbildung 6.4) wird der DVB-Videostream als ungefilterter „Rohdaten"-Stream behandelt und direkt aus einer (oder mehreren) DVB-T-Karten extrahiert. In diesem Stream sind nicht nur Textinformationen enthalten, sondern

auch Videoinformationen. Der textuelle Inhalt des DVB-Streams wird mittels der Linux-Komponente *DVBsnoop*[2] extrahiert. Parallel dazu werden sog. DVB-Parser (ähnlich der Software *TV-headend*[3]) benutzt, um EPG-Informationen aus der EIT-Tabelle per Software zu extrahieren. Wenn der Empfang der Sendung und ihrer beiliegenden Informationen auf textueller und visueller Ebene vorhanden sind, können die weiteren drei Komponenten d.h. Audio-Analyse (2), Videoanalyse (3) und Textanalyse (4) initiiert werden. In der Komponente (2) wird eine Speech-To-Text Komponente benutzt.

Diese wurde so implementiert, dass sowohl die Audiodeskription als auch der Originalton analysiert werden. Zusätzlich wird eine Vormarkierung der Dialogstellen mittels *Timecode* durchgeführt, sodass anschließend eine direkte Verknüpfung von Zeitstempel und Audiosignal möglich ist. Als Ergebnis der Audioanalyse werden Textblöcke bzw. Sätze erzeugt.

Ähnliches geschieht bei der Textanalyse (4). Hier werden alle Texte, die sich im DVB-T-Signal befinden, – inklusive der Tabellen – extrahiert. Zusätzlich wird die HbbTV-URL extrahiert und für einen späteren Zugriff zwischengespeichert. Die HbbTV-URL, die innerhalb der AIT-Tabelle zu finden ist, ermöglicht den Zugang zu weiteren Informationen einer Sendung. Diese weiterführenden Informationen dienen als Kontextinformationen, die für eine Anreicherung der aktuell ausgestrahlten Sendung benutzt werden (siehe Kapitel 4). Genauso sieht es mit den sog. Nibbles oder EPG-Einträgen aus, die als Zusatzinformationen dienen können. Die Zusammenstellung der Texte ergibt eine logische Ansammlung von Wörtern und Sätzen, die innerhalb der Komponente (7) extrahiert werden.

---

[2] `http://dvbsnoop.sourceforge.net`

[3] `https://github.com/tvheadend/tvheadend`

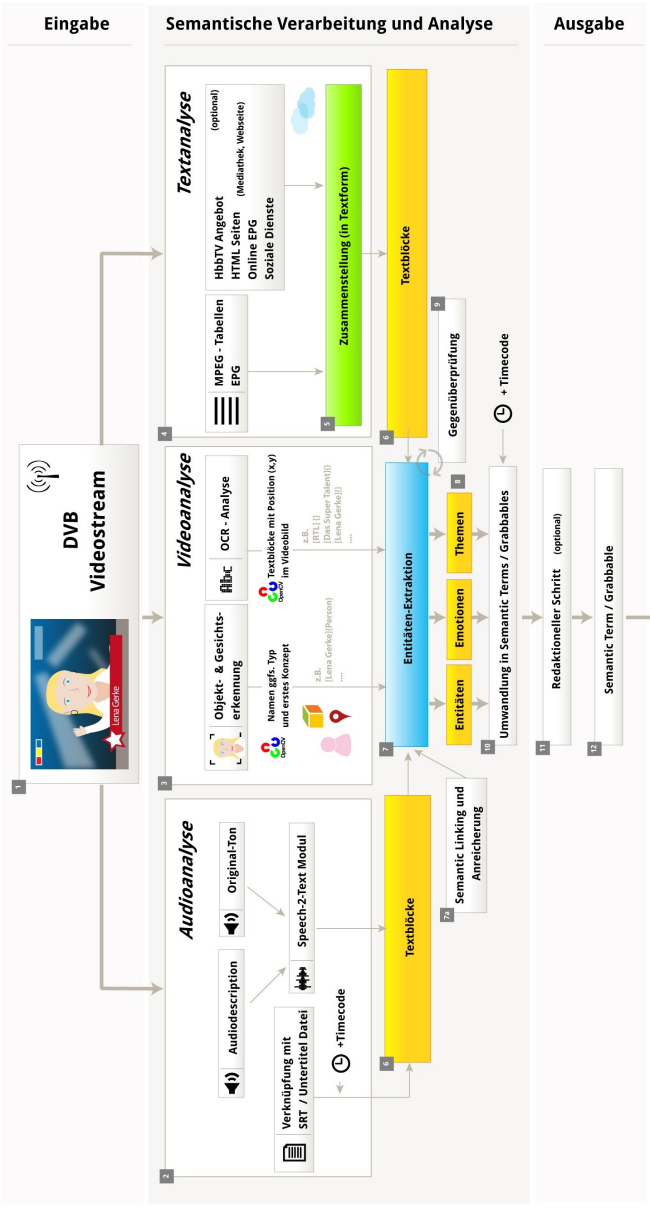


Abbildung 6.4: Übersicht der Extraktion und der semantischen Analyse

Bei der *Videoanalyse* (3) dagegen werden zwei unterschiedliche Ansätze angewandt. Der erste Ansatz besteht darin, Objekte bzw. Gesichter zu erkennen und zu klassifizieren. Dabei werden über die Videoanalyse-Bibliothek OpenCV die Positionen der zu erkennenden Elemente innerhalb eines Videos bestimmt. Das Matching mit einer bestehenden Datenbank erfolgt in einem weiteren Schritt. Dabei können nicht nur Konzepte sondern auch Namen der Personen erkannt werden. Mit einem ähnlichen Ansatz wird die OCR-Analyse durchgeführt. Diese ist auf die Präsenz von Textblöcken in Bildern trainiert. Das bedeutet, wenn eine textuell-grafische Einblendung im Bild (z.B. der Name des Reporters) zu sehen ist, wird versucht, diese zu analysieren und die sich in der Einblendung befindenden Texte werden entnommen.
Ähnlich der Komponenten 2 und 4 werden bei der Komponente *Videoanalyse* (3) die Ergebnisse in Form maschinell-lesbarer Textblöcke (6) generiert. Diese können entweder wohlgeformte Sätze, falls diese aus einem gesprochenen Dialog extrahiert werden, aber auch sehr kurze, qualitativ unterschiedliche unstrukturierte Texte sein wie z.B. EPG-Beschreibungen.
Obwohl Informationen aus dem Bild, den Tabellen und Audiosignalen des DVB-T-Datensignals extrahiert worden sind, liegt die Information nur in textueller Form vor. Das System soll dann aus diesen Informationen einen Bezug zu einer Domäne und Relationen zu erkannten Konzepten erstellen. Dieser Schritt wird von der Komponente *Entitäten-Extraktion* (7) geleistet. Innerhalb dieser Komponente werden alle Textblöcke auf Entitäten, Stimmung und Themen überprüft (8). Während dieser Extraktion findet eine kontinuierliche Gegenüberprüfung (6) statt. Das bedeutet, dass erkannte Entitäten aus den drei Komponenten miteinander verglichen werden. Tauchen nach einer Analyse bei jeder der Komponenten die gleichen Begriffe auf, ist die

Wahrscheinlichkeit hoch, dass diese Begriffe im aktuellen Kontext als „valide“ gekennzeichnet werden können.
Die Komponente (10) transformiert über ein dediziertes Format die erkannten Begriffe in *Grabbables*. Parallel dazu wird ein Zeitstempel hinzugefügt. Dieser dient dazu, den zeitlichen Moment der Einblendung zu erfassen und zu einem späteren Zeitpunkt (z.B. während einer Wiederholung der gleichen Sendung) die *Grabbables* automatisch anzeigen zu lassen, ohne dass eine komplette Neuanalyse des Videosignals nötig ist.
Durch die Kombination der *Grabbables* und des *Timecodes* wird eine Beschreibungsdatei (mittels des SwoozyML-Formats) generiert, die von einem Redakteur wie z.B. während einer späteren Ausstrahlung der gleichen Sendung geladen werden kann. Bevor diese Begriffe über eine dedizierte REST-Schnittstelle dem Swoozy-Client zur Verfügung gestellt werden, kann ein optionaler redaktioneller Schritt durchgeführt werden (11). Dieser Schritt wird von den Werkzeugen Swoozy-Livana und SKRPTR unterstützt (siehe Kapitel 7). Dabei kann ein Redakteur entscheiden, welche Begriffe (*Grabbables*) tatsächlich bei den Swoozy-Clients eingeblendet werden. Im folgenden Abschnitt werden die technischen Komponenten beschrieben, die dafür sorgen, dass die oben beschriebenen Schritte erfolgreich durchgeführt werden können.

Textuelle Wissensextraktion

Die textuelle Wissensextraktion bezieht sich auf die Komponente Nummer 4 der allgemeinen Architektur (siehe Abbildung 6.4). Unter *textueller Wissensextraktion* werden alle Prozesse und Arbeitsschritte definiert, die beginnend von einer heterogenen nicht ausspezifizierten Textmenge in der Lage sind, daraus Begriffe und Relationen herzuleiten und

mit Wissen zu verknüpfen. Zudem versteht man unter *„Wissen“*, für den Menschen als auch für die Maschine, verstehbare Inhalte, die eine Verknüpfung mit einer Wissensdomäne bilden. Die besondere Herausforderung im Kontext „Fernsehen“ ist, dass es keine im Vorfeld festgelegte Domäne gibt, bei der ein fester Satz an Begriffen oder Terminologien vorhanden wäre. Ganz im Gegenteil gehören die im Fernsehkontext benutzten Begriffe zu der sog. *Open Domain* (dt. offene Domäne), die je nach Sendungsarten oder Themen permanent variieren kann, sodass es schwierig ist, sie zu identifizieren. Diese Herausforderung stellt sich bei den Ansätzen zur textuellen Wissensextraktion. Im folgenden Abschnitt werden zuerst die möglichen Textquellen innerhalb eines DVB-T-Streams dargestellt. In einem weiteren Schritt werden die Analyseverfahren der Textquellen, die im Rahmen der Implementierung von Swoozy benutzt wurden, detailliert vorgestellt.

Extraktion aus DVB-Tabellen und EPG-Informationen

In Kapitel 4 wurde die Funktionsweise des digitalen Fernsehens vorgestellt. Dabei wurden u.a. die verschiedenen Tabellen (u.a. AIT, EIT, PMT) [ETS14a] [ETS14b] vorgestellt, die einen Zugriff auf den EPG oder die erweiterten Programmdaten ermöglichen. Die hier in Swoozy entwickelte Dekodierungskomponente bildet den ersten Schritt der Analyse und des Workflows der Semantifizierung, die innerhalb des Systems benötigt wird, um anschließend eine semantische Analyse der Inhalte durchführen zu können.

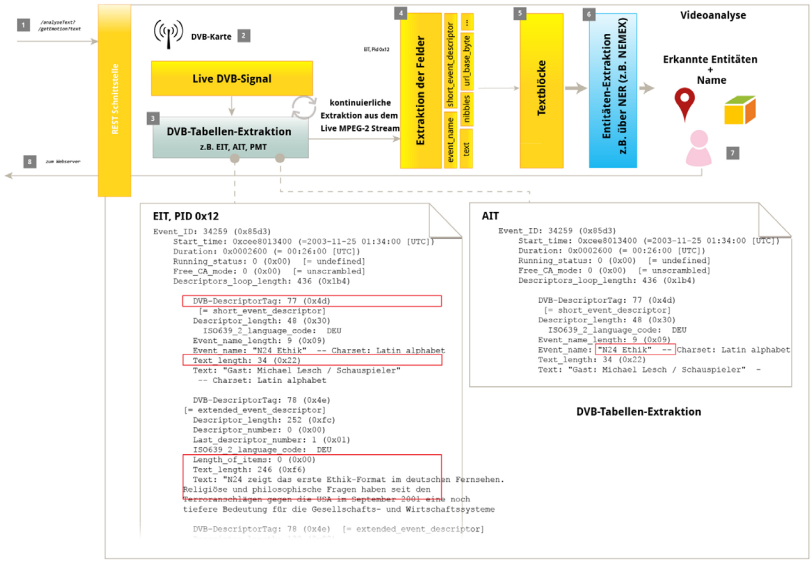


Abbildung 6.5: Semantische Extraktion aus EPG (EIT-Tabellen)

Dies wird über eine spezielle Komponente, die nur bestimmte vorausgewählte Werte von Tabellen und Feldern extrahiert, realisiert. Somit ist eine gezieltere Extraktion von Informationen (z.B. Programmname, Ausstrahlungszeiten, Teletext, usw.) möglich.

Die Abbildung 6.5 zeigt den Ablauf des Extraktionsvorganges innerhalb der ersten Version von Swoozy. Innerhalb der EIT-Tabelle befinden sich alle EPG-Informationen. Diese listet alle Programme im aktuellen Multiplex auf, die eine erweiterte Beschreibung besitzen. Das bedeutet, dass innerhalb dieser Tabelle nicht nur EPG-Programmdaten von einem bestimmten Fernsehkanal zu finden sind, sondern auch von allen anderen Kanälen innerhalb des Multiplexes. Eine Filterung wird durchgeführt, damit zum jeweiligen Fernsehkanal die passenden Programminformationen zugeordnet werden können.

Die daraus extrahierten EPG-Informationen werden, falls die darin beinhaltenden Einträge wenige Informationen liefern (d.h. nur die Uhrzeit und den Namen des Programmes), von der online-basierten EPG-Komponente von Rovi[4] (siehe Abbildung 6.6) angereichert. In den ersten implementierten Versionen von Swoozy-Live (siehe Kapitel 8) wurden die EPG-Daten über die REST-Schnittstelle von TV-Headend, eine auf Linux-basierte Komponente für den Empfang von DVB-Streams mittels DVB-Karten, bezogen. Bei den neuen Versionen von Swoozy werden diese EPG-Daten direkt vom Rovi-Dienst abgerufen. Diese Schnittstelle wird über eine DVB-T-Karte, die an einem Raspberry PI 2-Rechner angeschlossen ist, abgefragt (siehe Abbildung 6.2).

Parallel zur Extraktion der EIT-Tabelle wird die „Application Information Table“ (AIT) [ETS14a] ausgelesen, die von HbbTV-kompatiblen Fernsehern benutzt werden kann. Wenn eine solche Tabelle identifiziert wurde, wird die eingebettete URL aufgerufen und deren Textinhalt analysiert. Der Zeitpunkt zu dem das HbbTV-Angebot bzw. die HbbTV-Anzeige verfügbar gemacht wird, wird durch den „Autostart“-Befehl, der sich innerhalb der AIT-Tabelle befindet, getriggert [Mer11].
Eine HbbTV-Seite ist im Grunde genommen eine HTML-Webseite, die normalerweise nur für Smart-TV und HbbTV-kompatible Receiver sichtbar ist (siehe Kapitel 4). Desktop-basierte Webbrowser können diese Webseiten zwar öffnen und darstellen, es fehlen aber bestimmte Funktionalitäten wie Menüauswahl oder Navigationsleisten. Die HbbTV-Seite ist designtechnisch speziell nur für den Browser des Fernsehers aufbereitet und optimiert worden. Dabei werden die Dimensionen der Anzeigefläche und die Auswahl der Menüpunkte und Schaltelemente über die Fernbedienung (statt über die Maus) berücksichtigt.

[4] `http://www.rovicorp.com/`

Bestimmte Funktionen wie die parallele Einblendung des aktuellen DVB-Videosignals werden mittels speziellen Javascript-Bibliotheken realisiert und vom jeweiligen Smart-TV-Gerät umgesetzt und ausgewertet. Es kann nicht immer garantiert werden, dass die HbbTV-Webseite des Fernsehsenders zeitnah Informationen zu der ausgestrahlten Sendung liefert.

Extraktion über Untertitel

Eine weitere Quelle für Zusatzinformationen, die zur Vorbereitung der Semantifizierung des Live-Bildes von Nutzen ist und die im DVB-Stream zu finden sind, sind Teletext und Untertitel (für Gehörlose und Schwerhörige) (engl. *Closed captioning* oder kurz *CC*). Diese werden innerhalb des EBU-Teletexts[5] mitausgestrahlt.
Im Fall von Untertiteln werden Szenen oder Dialoge der Schauspieler in Form eines Textes transkribiert und über das Video grafisch in textueller Form eingeblendet. Diese Einblendung wird vom DVB-Receiver angesteuert und kann beliebig ein- und ausgeblendet werden. Hierbei muss zwischen zwei Empfangs- und Darstellungsverfahren unterschieden werden: eines dieser Verfahren ist rein textbasiert und folgt der ETSI TS 300 472-Spezifikation. Dieses Verfahren wurde vom analogen Fernsehen verwendet. Bei diesem Verfahren werden die Textinhalte mit dem Teletextstream innerhalb des MPEG-Streams (PES - Packetized Elementary Streams) mitgeschickt. Möchte der Zuschauer die Untertitel sehen, muss er die entsprechende Teletextseite (z.B. 888 oder 777) aufrufen. Über einen Transparenzmodus werden die dekodierten Textblöcke der jeweiligen Teletextseite durchsichtig über das ausgestrahlte Bild eingeblendet. Dieses Verfahren wird heute in Europa, aufgrund

[5] `http://www.daserste.de/specials/service/barrierefreiheit-im-ersten-index-zeilen100.html`

der Abschaltung des Analogsignals, nicht mehr benutzt.
Beim zweiten, neueren Verfahren (gemäß ETSI TS 300 743 Spezifikation) können die Untertitel in Form einer Pixeltabelle (Colour Look-Up Table kurz CLUT) ausgestrahlt werden. Dabei werden die Pixelposition, die grafische Kodierung der Pixel und der Zeitpunkt der Anzeige bestimmt. Die Tatsache, dass Grafiken statt Texte (im Sinne von ASCII-kodierten Zeichen) innerhalb des MPEG-Streams transportiert werden, erschwert deutlich die nachträgliche Erkennung. Für den Fernsehsender bietet diese Technik, insbesondere hinsichtlich der benutzten Farben, der Schriftartzeichenauswahl und Position des Textes im laufenden Bild, eine größere Darstellungsflexibilität. Um trotzdem eine Analyse der Untertitel durchzuführen, werden zwei alternative Methoden angewandt. Eine davon besteht in der reinen Dekodierung der PES-Pakete, d.h. ihrer Umwandlung und Speicherung dieser einzelnen Datenpakete in Bildern. Diese Aufgabe wird meist direkt von der DVB-Hardware und den Software-gestützten Receivern durchgeführt. In einem weiteren Schritt wird die Echtzeitdekodierung und Umwandlung der Datenpakete in Bildern realisiert. Diese dekodierten Bilder müssen anschließend in einen zeichenbasierten Text umgewandelt werden, der anschließend analysiert werden kann. Innerhalb von Swoozy wurde diese Echtzeitdekodierung der Untertitel mittels OCR- und Bildanalyseverfahren gelöst. Dadurch ist eine Dekodierung der speziellen Pixeltabellen, welche die einzelnen Fernsehsender bei Untertiteln verwenden, nicht mehr nötig.
Die farbliche Enkodierung der Untertitel liefert Kontextinformationen zum gesprochenen Satz[6]. Ein Satz, der z.B. mit weißer Farbe angezeigt wird, wurde vom sichtbaren Protagonist formuliert. In gelber Farbe

[6] `http://nicolas.anquetil1.free.fr/docs/page007/sous-titrage_television_histoire.pdf`

erscheinen Sätze, die außerhalb des Sichtfeldes der Kamera zu hören sind. Eine grüne Farbe markiert eine gesprochene Fremdsprache und die Magenta Farbe, eine Hintergrundmusik[7].
Dadurch ist es möglich, die Stimmung und sogar die Handlung einer jeweiligen Szene zu erfassen und mit zu protokolieren. Diese Information kann für die Entscheidung, welche *Semantic Terms* zu welchem Zeitpunkt angezeigt werden sollen, von Nutzen sein.
Ähnliches gilt für die Ausstrahlung eines schon bekannten und vorannotierten Filmes. Hier sind die Texte oder Dialoge der verschiedenen Szenen vorher bekannt. Damit lässt sich parallel zum textuellen Inhalt, z.B. eine Sentiment-Analyse, durchführen. Weitere Hintergrund- und technischen Informationen zur Auswertung von Audiodeskription in textueller Form liefert der Abschnitt über die Audioanalyse.
Vollständigkeitshalber muss erwähnt werden, dass die EBU (European Broadcasting Union)[8] neue Formate für die Untertitelung von Livesendungen ins Leben gerufen hat, um das ältere STL-Format zu ersetzen. Darunter befindet sich u.a. EBU-TT[9], welches sich an das W3C Timed Text Markup Language (TTML)[10]-Format anlehnt. Bei diesem Format werden die Untertitel in eine XML-Datei, samt Information über die Schriftarten, die Zeitpunkte der Einblendung, die Dauer der Einblendung und die Textpositionen gespeichert und können entweder in das DVB-Signal eingespeist werden oder im Falle von VOD/IP-TV von einem webbasierten Server heruntergeladen werden. Durch die Verwendung eines solchen Formats wird die textuelle Extraktion deutlich vereinfacht. Auch die HTML5-Spezifikation folgt einem ähnlichen An-

[7] http://www.medias-soustitres.com/television/En-savoir-plus/Sous-Titrage-Teletexte-STT
[8] http://www3.ebu.ch/home
[9] https://tech.ebu.ch/docs/tech/tech3370.pdf
[10] http://www.w3.org/TR/ttaf1-dfxp/

satz, dem Web Video Text Tracks-Format (kurz WebVTT[11]), das sich auf TTML stützt und sich an das SRT-Format anlehnt. Letzteres ist ein beliebtes Format für die Untertitelung von „gerippten" (den Inhalt einer DVD in ein Videoformat konvertieren) Filmen.

Extraktion über EPG-Nibbles

Die EIT-Tabelle beinhaltet neben den EPG-Einträgen weitere Daten oder sog. *Content Descriptors* „Inhaltsbeschreiber", die das aktuelle Programm beschreiben oder zumindest ihre Kategorie angeben. Technisch werden diese Zusatzinformationen als *nibbles* bezeichnet. Diese sind auf einen maximalen Informationsgehalt von 4 Bit kodiert und geben an, zu welcher Kategorie (z.b. *movie, drama, news, sport*) ein ausgestrahltes Programm gehört. Diese Identifikation wird im Normalfall von den DVB-T-Receivern benutzt, um die Kategorie eines Programmes, parallel zu den EPG-Informationen, anzuzeigen und ein dazu passendes Icon einzublenden. Diese Kategorisierung kann vom Swoozy-System benutzt werden, um u.a. den Kontext einer Sendung oder eines Programmes in Zusammenhang mit den dargestellten Personen oder Protagonisten zu extrahieren. Je nach Sendungsarten kann so die Auswahl der sog. Domäne verbessert werden. Unter Domäne versteht man hauptsächlich die Hauptthematik (auch Kontext genannt) einer Sendung (z.B. Fußball, Formel Eins, Freizeit, Unterhaltungsserie, usw.).

Ein Videointerview mit einem bekannten Sportler während einer Filmpremiere hat z.B. nicht den gleichen Kontext wie das Auftauchen des gleichen Sportlers mit seiner Mannschaft während eines Fußballspieles. Im ersten Falle wird der Fokus auf seiner filmischen Karriere und

[11] `http://www.w3.org/TR/2011/WD-html5-20110113/video.html#text-track`

dem Film liegen, im zweiten Fall werden seine Spielerstatistiken innerhalb seines Vereines in den Vordergrund gestellt.

Extraktion über Teletextseiten

Genau wie beim EPG, sind Teletextseiten aus dem Teletext (auch manchmal als EBU-Teletext bezeichnet) Bestandteile des DVB-Datenstroms und beinhalten Zusatzinformationen zum aktuellen Programm und weitere sendungsbezogene Informationen. Diese werden in Form von Teletextseiten mitausgestrahlt, die von einem DVB-Receiver dekodiert und angezeigt werden.
Bei Teletext können die Seiten so extrahiert werden, dass diese anschließend analysiert und in Zeichenketten umgewandelt werden. Ähnlich wie beim EPG, werden die Informationen aus den DVB-Tabellen (genauer gesagt aus der PES-Tabelle) extrahiert, wie im folgenden Tabellenausschnitt der Software DVBSnoop[12] zu sehen ist.

Listing 6.1: Ausschnitt von einer PES-Tabelle, die mit DVBSnoop extrahiert wurde.

```
[...]
Packet_start_code_prefix: 000001
Stream_id: 189 (0xbd)  [= private_stream_1]
PES_packet_length: 1282 (0x0502)
    reserved1: 2 (0x02)
    PES_scrambling_control: 0 (0x00)  [= not scrambled]
    PES_priority: 0 (0x00)
    data_alignment_indicator: 1 (0x01)
    copyright: 0 (0x00)
    original_or_copy: 0 (0x00)
    PTS_DTS_flags: 2 (0x02)
    ES_rate_flag: 0 (0x00)
```

[12] http://dvbsnoop.sourceforge.net/examples/example-pes-teletext.html

```
    additional_copy_info_flag: 0 (0x00)
    PES_CRC_flag: 0 (0x00)
    PES_extension_flag: 0 (0x00)
    PES_header_data_length: 36 (0x24)
    PTS:
        Fixed: 2 (0x02)
        PTS:
            bit[32..30]: 2 (0x02)
            marker_bit: 1 (0x01)
            bit[29..15]: 29905 (0x74d1)
            marker_bit: 1 (0x01)
            bit[14..0]: 17704 (0x4528)
            marker_bit: 1 (0x01)
             ==> PTS: 3127428392 (0xba68c528)  [= 90 kHz-Timestamp
                 : 9:39:09.204]
    stuffing bytes:
[...]
    PES_data (private_stream_1):
        EBU data:
            data_identifier: 16 (0x10)  [= EBU data EN 300 472 (
                teletext)]
            data_unit_id: 2 (0x02)  [= EBU Teletext non-subtitle
                data]
            data_unit_length: 44 (0x2c)
            Teletext data:
               reserved: 3 (0x03)
               field_parity: 1 (0x01)
               line_offset: 7 (0x07)
               frame_coding: 228 (0xe4)  [= EBU Teletext]
                => decoded:
                  magazine number (X): 3 (0x03)
                  packet number (Y): 13 (0x000d)  [= normal packet
                       intended for direct display]
                  packet data (parity stripped):
                       0000:  03 1d 01 20 20 20 20 20 20 20 75 6e
                            73 65 72 65
```

```
                0000:   .  .  .                       u  n
                     s  e  r  e
                0010:  72 20 57 65 6c 74 20 28 38 29 20 20
                    20 20 20 20
                0010:   r     W  e  l  t     (  8  )
                0020:  20 20 3e 54 49 50 50 20
                0020:         >  T  I  P  P
data_unit_id: 2 (0x02)  [= EBU Teletext non-subtitle
    data]
data_unit_length: 44 (0x2c)
Teletext data:
   reserved: 3 (0x03)
   field_parity: 1 (0x01)
   line_offset: 8 (0x08)
   frame_coding: 228 (0xe4)  [= EBU Teletext]
    => decoded:
      magazine number (X): 3 (0x03)
      packet number (Y): 14 (0x000e)  [= normal packet
           intended for direct display]
      packet data (parity stripped):
           0000:  03 1d 01 20 20 20 20 20 20 20 54 68
               75 72 6e 20
           0000:   .  .  .                       T  h
                u  r  n
           0010:  26 20 54 61 78 69 73 20 20 20 20 20
               20 20 20 20
           0010:   &     T  a  x  i  s
           0020:  20 20 20 20 20 20 20 20
           0020:
```

Seitens der Fernsehsender gibt es keine einheitlichen Strukturangaben für Teletextseiten. Meistens sind die Struktur, die Nummerierung der Seiten und die dazu passenden Kategorien (z.B. Sport, Nachrichten, usw.) innerhalb des Teletextangebots bei jedem Fernsehsender unterschiedlich.

Der Inhalt dieser Seiten sowie die Seitennummer können ohne Vorwarnung vom Teletextredakteur verändert werden. Diese Situation führt dazu, dass die DVB-T-basierte Teletextextraktion nicht als primäre Quelle für die semantische Extraktion innerhalb von Swoozy benutzt werden kann. Stattdessen wird optional auf die webbasierten Teletextseiten-Angebote zurückgegriffen. Dabei werden die identischen textuellen Teletextinformationen, die über DVB ausgestrahlt werden, in Form einer Webseite angeboten, wie es z.B. bei ARD[13], RTL[14] und ZDF[15] der Fall ist.

Bei der ARD wird sogar eine mobile Version mit einem festen URL-Schema bereitgestellt[16]. Diese Seiten können für die textuelle Extraktion benutzt werden. Da jedoch allmählich die Teletext-Angebote über HbbTV ausgestrahlt werden, wird innerhalb des Swoozy-Systems folgende Variante bevorzugt. Die HbbTV-Seite des jeweiligen Fernsehsenders wird, genau wie eine herkömmliche Webseite, geparst und analysiert. Dies ist vom Aufwand deutlich handhabbarer als eine teletextbasierte Analyse und bietet zugleich eine übersichtlichere Aufbereitung der Grunddaten. Somit ist es nicht mehr notwendig, jedes Mal für jeden Fernsehsender die passende Teletextseite herauszusuchen. Wenn trotzdem keine HbbTV-URL in der AIT-Tabelle verfügbar oder auffindbar ist, kann im letzten Schritt auf kommerzielle oder Community-basierte EPG-Dienste wie z.B. die von Rovi[17] oder Kazeer[18] über eine REST-API online zurückgegriffen werden. Diese Dienste bieten eine REST-Schnittstelle an, die Informationen über die abgespielte Sendung eines bestimmten Fernsehsenders in Form von JSON-Strukturen

[13] http://www.ard-text.de
[14] http://www.rtl.de/cms/service/service_navigation/teletext.html
[15] http://module.zdf.de/teletext/master.html
[16] http://www.ard-text.de/mobil/100
[17] http://www.rovicorp.com/
[18] http://kazer.org/

oder XML-TV-Formaten zurückgibt. Der Workflow der Teletextanalyse innerhalb von Swoozy wird in Abbildung 6.6 grafisch dargestellt.

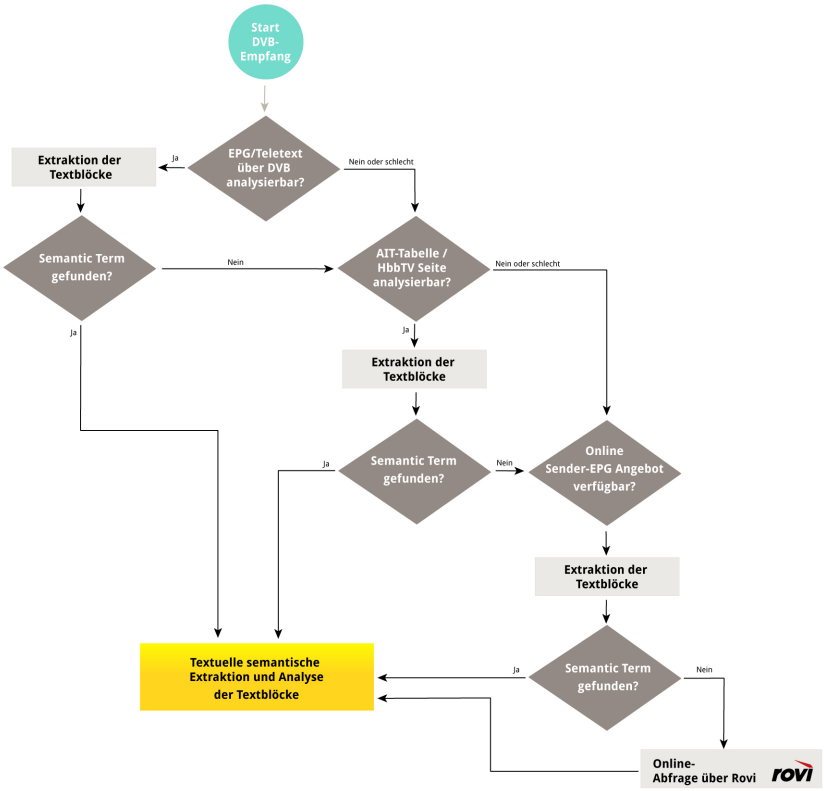


Abbildung 6.6: Ablauf der Teletext-Extraktion

Wenn alle oben genannten Informationsquellen abgefragt worden sind, werden die daraus resultierenden Texte an eine Extraktionskomponente übergeben.

Damit werden eine semantische Annotation und später die grafischen *Semantic Terms* erzeugt. Aggregiert man Teletext, HbbTV und externe

Online-Teletextdienste, erreicht man eine präzisere Beschreibung der aktuellen Sendung.

Verfahren

Die textuelle Analyse der extrahierten Texte und die Entitätenextraktion werden von einer dedizierten Komponente durchgeführt. Diese in Javascript implementierte Komponente wird über eine REST-API angesteuert und hat als Aufgabe, verschiedene Dienste abzufragen, wie in Abbildung 6.7 skizziert. Im ersten Schritt werden die REST-Schnittstellen zur Verfügung gestellt und über die jeweiligen URLs aufgerufen (Schritt 1 und 2 von Abbildung 6.7). Der Aufruf kann einzeln durchgeführt werden, indem der Client mittels eines Parameters bei jedem Aufruf den zu verwendenten Dienst angibt.

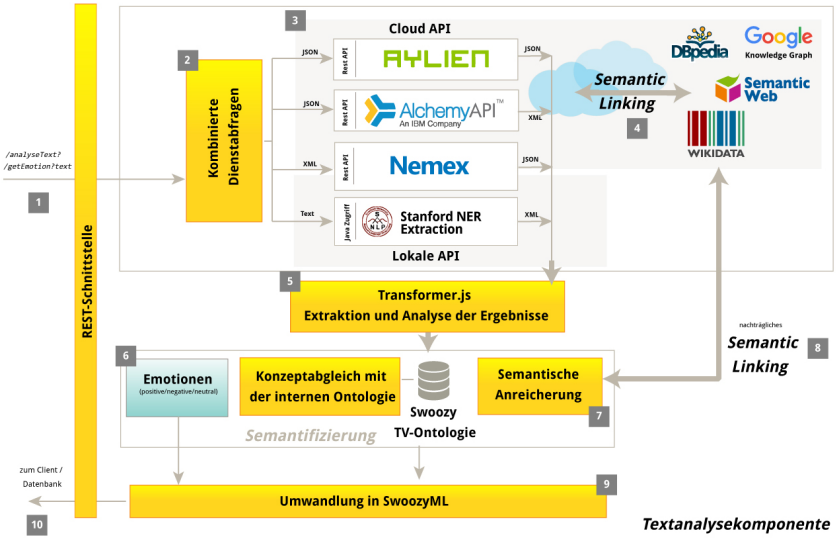


Abbildung 6.7: Named Entity Extraktion über Cloud-basierte Dienste

Eine Kombination von Webdiensten kann durch die Parameterangabe durchgeführt werden. Diese wird die einzelnen URL-Aufrufe und REST-Endpoints-URLs des jeweiligen Dienstes mit den entsprechenden Parametern aufrufen. Dieser Ansatz hat den Vorteil, dass die APIs der jeweiligen Cloud-Dienste innerhalb des Swoozy-Servers über eine einzige generisch definierte Schnittstelle aufgerufen werden können und letztendlich eine einheitliche Ergebnisstruktur zurückliefern. Zusätzlich können mit diesem Verfahren die Dienste (3) unabhängig voneinander und ohne softwaretechnische Abhängigkeit ausgetauscht werden, lediglich die APIs der jeweiligen Cloud-Dienste müssen vorher bekannt sein, damit die Aufruf- und Ergebnisstrukturen zusammenpassen. Je nach Dienst wird eine semantische Verlinkung der erkannten Entitäten oder Konzepte durchgeführt (Stichwort: *Semantic Linking*) (4). Das bedeutet, dass parallel zu den extrahierten Konzepten und Entitäten, diese mit einer im Web verfügbaren Wissensdatenbank wie z.B. DBpedia oder Wikidata verknüpft werden.

NEMEX

NEMEX ist eine Komponente, die von Günter Neumann[19] im Rahmen seiner wissenschaftlichen Arbeiten auf dem Gebiet *Language Technologies* (Sprachtechnologie) und im Bereich *Offene Informationsextraktion* (Open Information Extraction, [EHN08]) am Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI) entwickelt wurde. Diese Komponente bietet über verschiedene verknüpfte Module die Möglichkeit, sehr schnell und effizient Informationsextraktionsaufgaben in Echtzeit und streambasiert zu erledigen. Durch seine verschiedenen Module und dank seiner hohen Konfigurierbarkeit (der Entwickler kann NE-

[19] `http://www.dfki.de/~neumann/`

MEX frei konfigurieren, d.h. die Auswahl des Korpus (Begriffe einer definierten Domäne, die über eine lexikalische Datenbank wie Wordnet[20] vordefiniert worden sind) oder die Ergebnispräzision bestimmen) wird eine automatische Extraktion von Entitäten u.a. Eigennamen wie Personen, Institute, geografische Bezeichner aber auch domänenspezifische Fachbegriffe und ihre relevanten Relationen zueinander aus natürlichsprachlichen Texten und aus unterschiedlichen Quellen realisiert. Im Fall von NEMEX stammen diese Texte von Quellen wie dem EPG oder durch die OCR-Analyse gewonnenen Textstücke (engl. *Chunks*).

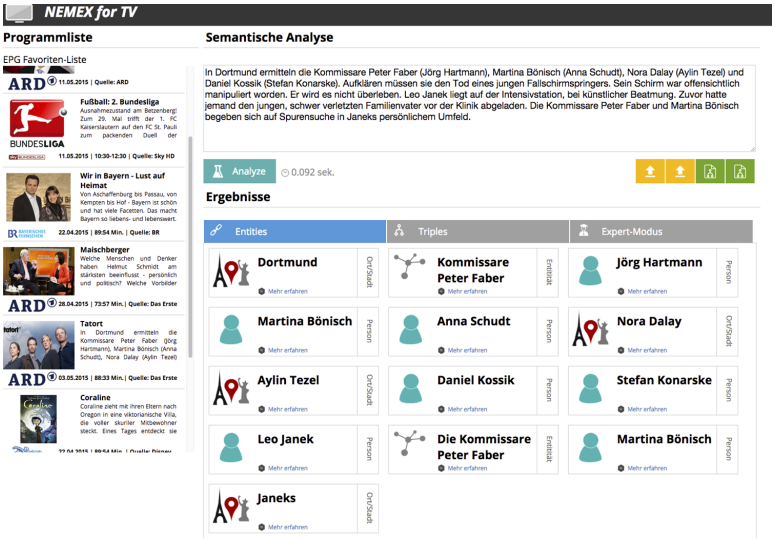


Abbildung 6.8: Grafische Oberfläche von NEMEX for TV

Parallel dazu bietet NEMEX eine Open Relation Extraktions-Funktion in Form von semantischen Tripeln (Subjekt-Prädikat-Objekt), eine Echtzeiterkennung von mehrstelligen Beziehungen und die Identifizierung

[20] https://wordnet.princeton.edu/

von sog. Relationsphrasen. Zusätzlich wurde im Rahmen der Entwicklung von Swoozy innerhalb von NEMEX eine Toolbox namens „NEMEX for TV" entwickelt, die eine dynamische semantische Verlinkung, falls Metadaten und semantische Beschreibung der Medien vorhanden sind, und eine Emotions- und Stimmungsextraktionskomponente (*Opinion Mining* und *Sentiment Analysis* zur Verbesserung der Positiv/Negativ-Bewertung) spezifisch für die TV-Domäne zur Verfügung stellt. Diese wird über eine leicht zu bedienende Oberfläche dargestellt und bietet eine Möglichkeit, die Echtzeitanalyse von NEMEX zu konfigurieren.

Durch seine hohe Konfigurierbarkeit wurde NEMEX so optimiert, dass unstrukturierte und nicht wohlgeformte Texte wie z.B. die aus dem EPG in Echtzeit besser erkannt werden. Zusätzlich wurde in NEMEX eine weitere Komponente integriert, welche die Sätze und Satzteile (engl. *Chunks*) analysiert und nicht valide Texte schon vor der tatsächlichen Analyse eliminiert. Dies bildet einen wesentlichen Vorteil, insbesondere bei der Bereinigung und Analyse von aus der OCR gewonnenen Texten. Durch diese Komponente können qualitativ bessere Ergebnisse geliefert werden.

NEMEX ist als eigenständige Komponente innerhalb von Swoozy integriert. Die Integration wurde so umgesetzt, dass ein dedizierter Server die Schnittstelle zur NEMEX-Komponente bereitstellt. Innerhalb der Textanalysekomponente wird der Dienst mittels der dedizierten REST-Schnittstelle aufgerufen. NEMEX kann während seiner Ausführung konfiguriert werden, d.h. bestimmte Zusatzparameter zur Verfeinerung der Extraktion können während einer laufenden Analyse über die gleiche REST-Schnittstelle angegeben werden. Dadurch kann eine bessere Feinabstimmung der Extraktionsalgorithmen erzielt werden. Eine JSON-Struktur wird als Ergebnis der textuellen Extraktion zurück-

geliefert. Diese Ergebnisstruktur wird später in den Schritten (5) und (6) analysiert, mit der internen Ontologie abgeglichen und im letzten Schritt zusammen mit den Ergebnissen der anderen Cloud-basierten Dienste (z.B. Alchemy, Stanford NER oder AYLIEN) in einem einheitlichen Format zurückgegeben.
Dank seines Cloud-basierten Ansatzes und seiner sehr fein granularen Konfigurierbarkeit der semantischen Informationsextraktion, wird die Benutzung von NEMEX als Textanalysekomponente bevorzugt.

Cloud-basierte Dienste

Parallel zu NEMEX werden andere Dienste für die semantische Extraktion in Swoozy benutzt. Grund für diese Entscheidung war die Notwendigkeit einer zusätzlichen Überprüfung der erkannten Entitäten und Konzepte. Diese Dienste dienen einer gegenseitigen Ergänzung, denn nicht jeder Cloud-basierte Dienst ist in der Lage, für Swoozy qualitativ hochwertige Ergebnisse zurückzuliefern. Unterschiede lassen sich z.B. bei der Anzahl der analysierten Konzepte oder Entitäten erkennen. Die Analyseverfahren, die intern von der Textextraktion benutzt werden wie z.B. die Chunk-Analyse und der dabei eingesetzte Korpus bestimmen die Qualität der Erkennung der semantischen Extraktion. Die Qualität der Ergebnisse der Erkennung hängt davon ab, wie groß der Korpus für die Erkennung ist und welche Analyseverfahren intern benutzt werden. Die gleiche Problematik existiert bei Texten, die in unterschiedlichen Sprachen analysiert werden. Zusätzlich sind die Leistungen der Dienste unterschiedlich. Manche bieten nur eine rudimentäre Verknüpfung zu Wikidata oder DBpedia, andere realisieren nur die Extraktion von Konzepten in festgelegten Klassen (siehe Kapitel 8). Passt während der Analyse ein Konzept nicht in eine der

vordefinierten Klassen, wird dieses als „unbekannt“ bewertet.
Um all diese Defizite zu beheben, wurde bei Swoozy die Entscheidung getroffen, eine kombinierte Dienstabfrage (siehe 3 in Abbildung 6.7) auf verschiedenen Cloud-basierten Textanalysediensten durchzuführen und diese programmtechnisch umzusetzen. Konkret bedeutet es, dass wie bereits im Abschnitt über NEMEX erwähnt, ein generischer Aufbau geschaffen wurde. Somit können diese Cloud-basierten Dienste ein- bzw. ausgeschaltet werden, so dass mehrere mühelos anschließend innerhalb der kombinierten Dienstabfrage hinzugefügt werden können. Innerhalb von Swoozy und parallel zu NEMEX werden drei unterschiedliche Dienste aufgerufen: erstens AYLIEN, dann Alchemy und drittens der Stanford NER (Named Entity Recognizer). Letzterer kann streng genommen nicht als Cloud-basierter Dienst bezeichnet werden, da er keine webbasierte verfügbare REST-Schnittstelle zur Verfügung stellt und eine lokale Java-Installation auf einem Rechner notwendig ist, um die Textanalyse zu starten. Nichtsdestotrotz wurden die Stanford NER-Module innerhalb einer node.js-Serverkomponente in gekapselter Form benutzt (ner-server[21]). Das bedeutet, dass die Kommandos und die zu analysierenden Texte über die node.js basierte Serverkomponente an die Stanford NER Java-basierte Komponente weitergeleitet werden.
Bei dem *Stanford Named Entity Recognizer (NER)* werden die Ergebnisstrukturen als XML und potentielle Tags mit 7-Klassen zurückgeliefert:

- *Location (Ort)*,
- *Time* (Zeitpunkt oder Zeitstempel eines Ereignisses),
- *Person*,

[21] `https://www.npmjs.com/package/ner-server`

- *Organization* (Firmenname, Institutionen),
- *Money* (Finanzbezogene Themen),
- *Percent* (Preisliche und prozentuelle Angaben von Zahlen),
- *Date* (Datum und zeitbezogene Referenzen innerhalb des Textes).

Folgende Ausgabe aus dem *Stanford Named Entity Recognizer (NER)* zeigt das Format der Ergebnisse.

Listing 6.2: Ausgabe einer Analyse aus dem Stanford Named Entity Recognizer

```
In a <ORGANIZATION>SPIEGEL</ORGANIZATION> interview, Egyptian
    President <PERSON>Abdel Fattah el-Sisi</PERSON> discusses his
    country's struggles, the forceful overthrow of his predecessor
     <PERSON>Mohammed Morsi</PERSON> and the threat against <
    LOCATION>Egypt</LOCATION> posed by <ORGANIZATION>Islamic State
    </ORGANIZATION> and the <ORGANIZATION>Muslim Brotherhood</
    ORGANIZATION>. Interview Conducted by <PERSON>Dieter Bednarz</
    PERSON> and <PERSON>Klaus Brinkbaumer</PERSON> more...
```

Dabei ist zu sehen, dass die erkannten Entitäten ohne semantische Referenz zurückgeliefert werden. Bei der API von AYLIEN werden die Ergebnisse mit einer direkten semantischen Referenz (Link Data) auf DBpedia versehen (siehe Abbildung 6.9). Beide Informationen dienen dazu, den semantischen Abgleich zu vereinfachen und die Ergebnisstrukturen mit der internen Ontologie zu vergleichen (siehe Punkt 6 und 7 von Abbildung 6.7).

Die Dienste (AYLIEN, Alchemy, NEMEX und Stanford NER) liefern heterogene Rückgabe- und Ergebnisstrukturen. Diese heterogenen Ergebnisstrukturen, zwar mit unterschiedlichen Inhalten, interner Formatierung und Formaten, werden innerhalb einer dedizierten Komponente

zusammengefügt, kombiniert und in ein einheitliches und bearbeitbares Format umgewandelt. Dieses generische Modul läuft innerhalb des Swoozy-Servers unter der Bezeichnung „Transformer.js-Modul“ (Punkt 5 von Abbildung 6.7). Die transformierten und aggregierten Daten werden in SwoozyML umgewandelt und zur weiteren Verwendung innerhalb des Analyseworkflows per REST als Ausgabe weitergereicht (9)(10).

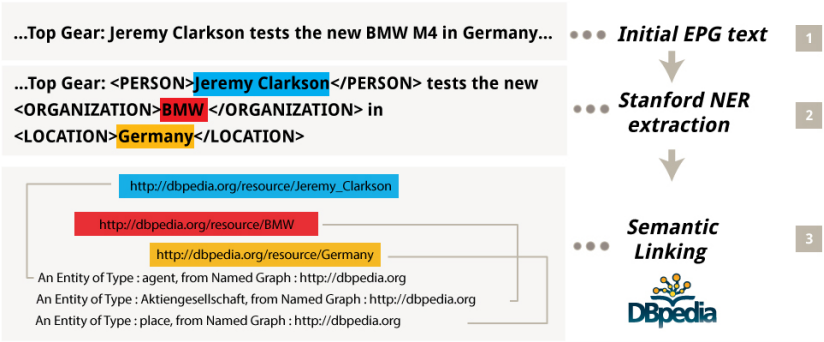


Abbildung 6.9: Prinzip der Stanford Named Entity Extraktionskomponente und des Semantic Linkings nach DBpedia

Video-basierte Wissensextraktion

Technische Architektur

Die OCR-, Objekt-, und Gesichtserkennungen innerhalb von Swoozy spielen eine wichtige Rolle für die Generierung der Grabbables. Die Erkennung wird von Video- und Bildanalysekomponenten verwaltet und angesteuert. Die Aufgaben der Komponenten werden extern und verteilt durchgeführt, d.h. weder auf dem Client, da die aktuelle Rechenleistung dies nicht erlauben würde, noch direkt auf Smart-TV-

Geräten, weil kein direkter Zugriff auf die Videoeinzelbilder per APIs möglich ist, sondern auf einem eigens für die Erkennung dedizierten Server. Dieser speziell für die Videoanalyse ausgelegte Server führt alle Berechnungen und aufwendigen Erkennungen durch. Die Videoanalysekomponente funktioniert dezentral. Die eigentliche Analyse und der Abgleich zu einer Datenbank von Persönlichkeiten (Punkt 8 von Abbildung 6.10) wird ebenso auf einer externen Komponente durchgeführt und erst, wenn diese Analyse durchgeführt wurde, werden die Ergebnisse zum Swoozy-Server zurückgeschickt, damit diese weiterverarbeitet werden (10) und (9).

Die Auswahl dieser Architektur ist dadurch motiviert, dass der Swoozy-Client aus Gründen der Performanz nicht alle Prozesse der Videoanalyse, der DVB-Videosignalanzeige, der Gesteninteraktions- und Interpretationsanalyse und der Anzeige des UI gleichzeitig verwalten kann. Dies bedeutet implizit, dass schon größere vorannotierte Datenmengen lokal beim Client vorliegen müssten, damit der Abgleich schnell und effizient durchgeführt werden kann. Dies kann bei einem fernsehbasierten Echtzeitszenario nie der Fall sein.

Aus diesem Grund wurde die Videoanalyse auf mehrere Videoanalysekomponenten verlagert (Punkt 3 von Abbildung 6.10), die ausschließlich die Analyse der Bilder von nur einem dedizierten Fernsehkanal durchführen. Konkret sind diese Module unabhängige selbstständige Serverkomponenten, die z.B. das Videosignal einer dedizierten DVB-T-Karte abgreifen (4) (eine pro Multiplex) und deren Signal von OpenCV analysieren lassen (5). Ein Abgleich mit einer zentralen Datenbank liefert die Informationen, welche Personen inklusive ihrer grafischen Position im Bild zu sehen sind (6) und zu welcher *Named Entity* diese gehören (7). Als Ergebnis werden diese Informationen zuerst dem Swoozy-Server zur Verfügung gestellt (9). Dieser stellt in einem wei-

teren Schritt die Ergebnisstrukturen dem Swoozy-Client über eine REST-Schnittstelle zur Verfügung (10).

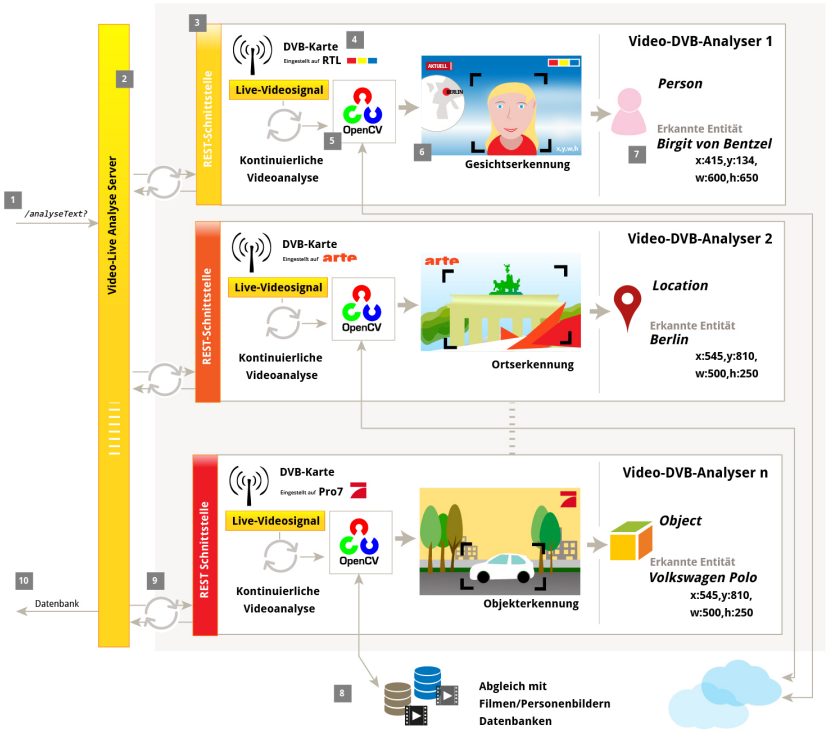


Abbildung 6.10: Architektur der Server-seitigen video-basierten Extraktion in Swoozy

Der Start- und Endzeitpunkt einer Erkennung wird über den Swoozy-Server synchronisiert. In einem weiteren Schritt werden die Ergebnisse der Erkennung vom Swoozy-Server gesammelt und anschließend , u.a. durch die verschiedenen Textanalysekomponenten, mit einer passenden semantischen Beschreibung versehen. Der Swoozy-Server benutzt mehrere Videoanalysekomponenten, die mit DVB-T-Karten

ausgestattet sind, welche die jeweiligen Livestreams mitprotokollieren und auf das Livevideosignal beide im nächsten Abschnitt beschriebenen Verfahren (OCR und Gesichtserkennung) anwenden. Innerhalb der Videoanalysekomponenten werden zwei wesentliche Verfahren für die Erkennung benutzt: einerseits das OCR-Verfahren und andererseits das Gesichtserkennungsverfahren. Die Hauptmotivation bei der gezielten Anwendung dieser beiden Verfahren war:

- Eine zeitkoordinierte Videoeinzelbild- und Annotationsspur zu generieren.
- Über Einblendungen und dank der OCR-Analyse an erste Informationen zum Video (z.B. Name des Gastes/Moderators oder Ort des Geschehenes) zu gelangen. Die intelligente Analyse der Einblendungen kann durch Vorlagen, die speziell für eine jeweilige Sendung angelegt worden sind, unterstützt werden.
- Insbesondere bei Live-Übertragungen bieten die Einblendungen eine schnelle, vom Fernsehsender selbst generierte, erste zwar ungefilterte Beschreibungsquelle für das, was gerade im Livebild zu sehen ist, an. Diese Information kann als Zusatzquelle benutzt werden, falls die Bilderkennung keine Grabbable erzeugen konnte.
- Eine höhere Feingranularität der Information zu gewinnen und die Qualität durch Zusatzinhalte aus den EPG-, Audio- und DVB-Streams zu verbessern.

Die einzelnen Videoanalysekomponenten bzw. Swoozy-Videoanalyse-Server sind mit der Programmiersprache C# realisiert worden und stützen sich auf OpenCV, um die Erkennung durchzuführen. Wenn

eine Analyse durchgeführt wurde, wird als Ergebnis eine SwoozyML-Struktur geliefert. Diese Struktur beinhaltet u.a. die Koordinaten und eine erste semantische Beschreibung, wie z.B. das Konzept und seinen dazugehörigen Namen, der identifizierten Inhalte. Diese Strukturierung der Information wird im Abschnitt über SwoozyML näher beschrieben. Um später eine (z.B. in Falle einer Wiederholung) schnellere Erkennung und Analyse einer Videoszene zu ermöglichen, können die Ergebnisstrukturen persistent in einer Datenbank gespeichert werden. Dieser verteilte Ansatz ist generisch aufgebaut: jeder Swoozy-Videoanalyse-Server ist modular ablauffähig, sodass es technisch möglich ist, die Ergebnisse mehrerer solcher Server gleichzeitig zu analysieren, zu vergleichen und letztendlich über eine SwoozyML-Struktur zu aggregieren. Bei dieser Echtzeitanalyse muss man zwischen zwei „Zuständen" differenzieren.

- Im Fall der Analyse einer Live-Sendung (d.h. mit einer nicht bekannten Sendungsabfolge, z.B. Nachrichtensendung, Sportübertragung, Debatte, usw.) bei der keine Informationen zum Zeitpunkt der Ausstrahlung vorhanden sind, muss die Videoanalyse kontinuierlich durchgeführt werden. Dieser Vorgang läuft automatisch. Die Ergebnisse können anschließend auf Richtigkeit über eine semi-automatische Überprüfungskomponente geprüft werden (Stichwort: Plausibilitätscheck).
- Bei Wiederholungen von Sendungen oder Filmen, deren Ablauf schon bekannt ist, d.h. eine komplette zeitliche Beschreibung vorhanden ist, braucht die Videoanalyse keine neue Analyse durchzuführen. Hier wird auf eine vorhandene Datenbank von Annotationen zurückgegriffen.

OCR und Einblendungserkennung

Prinzip

Die OCR-Komponente (engl. Optical Character Recognition) führt mit einer Taktung von 200 bis 500ms eine automatisierte Texterkennung innerhalb des Videobildes durch, um textuelle Informationen, d.h. Name des Moderators aber auch bestimmte Werbeplakate oder andere Texte, die im Videoeinzelbild zu finden sind, als Zeichenkette auszugeben. Diese Zeichenkette beinhaltet eine textuelle Information, die als Eingabe für die Textanalysekomponente benutzt werden kann, um z.B. Kontextinformationen zu extrahieren.
Die OCR-Erkennung läuft parallel und verteilt ab (ähnlich der Gesichts- und Objekt-Erkennung), wird über den Swoozy-Video-Server verwaltet und im Livemodus betrieben. Die Ergebnisse der Erkennung, die Zeichenkette und die grafische Position dieser im Videoeinzelbild werden für eine spätere Benutzung persistent gespeichert.
Die OCR-Analyse des ausgestrahlten Videobildes kann entweder global, d.h. über das komplette Bild, oder lokal, d.h. über eine bestimmte vordefinierte Zone des Bildes, durchgeführt werden. Bei der globalen Analyse wird das komplette, ungefilterte Videobild (Einzelbild) als Eingabe benutzt. Nachteil dieses Ansatzes ist die mögliche Entstehung von Rauschen und Bildartefakten während der Analyse, die vom Erkenner fälschlicherweise als Zeichenkette erkannt werden kann. Zusätzlich benötigt die OCR-Analyse eines Full-HD Einzelbildes viel Rechenleistung, was die Erkennung verlangsamt (siehe Kapitel 8). Um dies zu verhindern, können zwar Bildanalysefunktionen wie das Benutzen von Filtern (Schwarz-Weiß, Background-Filtering, usw.) abgestellt werden, dadurch wird jedoch die Qualität der Erkennung verschlechtert.

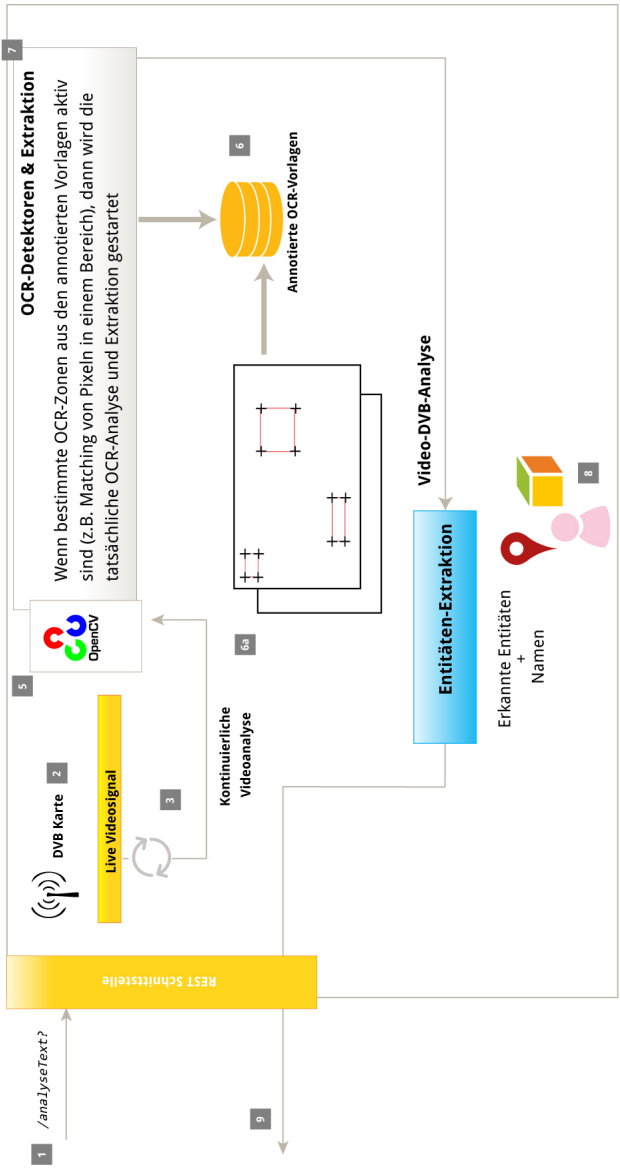


Abbildung 6.11: Prinzip der Video-basierten OCR-Analyse

Aus diesem Grund wurde innerhalb des Swoozy-Videoanalyseservers eine positions- und flächenbegrenzte OCR-Analyse bevorzugt. Das bedeutet, dass nur bestimmte, festdefinierte Bereiche für die OCR-Analyse als Eingabequelle benutzt werden. Auch wenn dies zunächst restriktiv klingt, weist dieser Ansatz folgende Vorteile auf:

- Einzig die vorgegebenen Bildbereiche werden tatsächlich analysiert. Es entsteht während einer Live-Video-Analyse keine unnötige zusätzliche Rechnerlast.
- Die gezielte lokale Analyse kann besser mit dynamischen Einblendungen (z.B. Name des Moderators) arbeiten, da diese während bekannter Sendungen immer an bestimmten festen Positionen innerhalb des Videobildes erscheinen. Das Erscheinen der Einblendung wird als Trigger für die Anzeige der *Grabbables* benutzt.
- Unterschiedlich im Bild positionierte Bereiche können parallel analysiert werden. Dabei können gleichzeitig verschiedene Informationen extrahiert werden (z.B. Ort einer Austragung und Name des Reporters während einer Liveschaltung).
- Die *False-Positives*-Erkennungsrate wird deutlich reduziert, da nur festgelegte Bereiche als potenzielle Kandidaten in Frage kommen. Das System verfügt über das Wissen, an welcher Stelle die Wahrscheinlichkeit höher ist, dass Text eingeblendet wird.

Die lokale OCR-Analyse von Videoeinzelbildern gewährt eine gute Balance zwischen Schnelligkeit, Qualität der Erkennung und Transformation in Zeichenketten. Bei der Sendung *Tagesschau* z.B. können nicht nur der Name des Themengebietes, sondern auch die Namen der Moderatoren gescannt und als Zeichenkettenfolge ausgegeben werden (siehe Abbildung 6.14). Das Hauptproblem bei der OCR-Analyse bleibt

die Effizienz der Erkennung. Da OCR nur auf 2D-Bilderkennung basiert und keine Informationen zur Verfügung stellt, welche Einblendungen sich im Vordergrund befinden (im Gegensatz zum menschlichen Auge) oder welche Textteile innerhalb eines Bildes relevant sind, muss die Erkennungserfolgsquote gesteigert werden.
Diese kann durch die Voraberkennung sogenannter *False-Positives* gesteigert werden. *False-Positives* sind ungewollte „Erkennungen" eines Textes innerhalb des Bildes. Durch diese nicht korrekte oder *„falsche"* Erkennung kann kein Bezug oder Kontext (gehörte der Text zu einem Objekt, einem Ort oder einer Person?) eindeutig bestimmt werden. Dies erzeugt Rauschen, das die Erkennungsrate und die anschließende Umwandlung in Annotationen erschwert.
Während der technischen Realisierung der Live-Video-Analyse konnten folgende Fälle, die Rausch- oder Erkennungsprobleme bei der Analyse verursachen, identifiziert werden:

- Infographie oder Tabellen werden komplett ausgewertet und als Zeichenkette zurückgegeben. Der Zusammenhang ist dem System im Voraus nicht bekannt, d.h. welche Daten zu welchen Begriffen oder Kontexten gehören, ist grafisch nicht ersichtlich.
- Aus einer Wetterkarte werden alle Städtenamen extrahiert. Dies macht für den Zuschauer und für die weitere Analyse wenig Sinn.
- Werbeplakate oder Markennamen werden erkannt und in textueller Form umgewandelt. Die Information, auf welchem gefilmten Gegenstand (z.B. eine Werbung im Hintergrund des Moderators) sich der erkannte Text bezieht, kann nicht eindeutig bestimmt werden.
- Bei transparenten Bannern und Lauftexten bei Nachrichtensen-

dern wie bei CNN oder N24 besteht die Schwierigkeit darin, die verschiedenen Ebenen zu trennen und das OCR-System vorher so einzustellen, dass nur unbewegte Stellen analysiert werden. Unter unbewegten Stellen versteht man Bildausschnitte, bei denen der Hintergrund eines halbtransparenten Banners sich nicht bewegt. Die Bewegungen im Hintergrund eines Banners können die OCR-Algorithmen stören und somit die Erkennung verschlechtern.

OCR-Erkennungsvorlagen

Diese Beispiele zeigen, dass die Verwendung der OCR-Analyse im Rahmen einer Echtzeit- Content-Analyse immer noch viel Rauschen und Nebeneffekte erzeugt. Unter Rauschen versteht man textuelle Informationen (Markennamen auf dem Pullover eines Interviewten, Anzeigetafeln im Hintergrund einer Veranstaltung), die zwar textuell richtig erkannt worden sind, aber in der ausgestrahlten Sendung keinen sinnvollen Kontext bilden und für eine weitere Analyse nicht benutzbar sind. Um diese Nebeneffekte zu reduzieren und um die Erkennungsrate zu erhöhen, wird eine grafisch beschränkte OCR-Analyse durchgeführt. Dabei wird nicht das komplette Videoeinzelbild von der OCR-Komponente analysiert, sondern nur Teilbereiche des Videobildes. Diese werden innerhalb des Swoozy-Frameworks als *OCR-Zonen* bezeichnet. Diese *OCR-Zonen* werden auf Pixelebene definiert (mit den Parametern X-, Y-Position, samt Höhe und Breite), damit die OCR-Komponente nur die aktivierten vordefinierten Zonen analysiert.

Konkret ist die Erkennungsvorlage ein XML-Dokument, welches Vorgaben für die Position von *OCR-Zonen* angibt. Das Dokument beinhaltet

die X-, Y-Koordinaten und die Größe eines Bereiches, in dem eine Entität zu finden ist. Im Fall von Sendungen, die ein festes grafisches Layout für die Einblendungen besitzen, wie z.B. in der folgenden Abbildung bei der *Tagesschau* oder bei einer Sportsendung dargestellt, ist es sogar möglich, die Themen und Namen der Sportler schneller durch dieses Verfahren zu extrahieren, als es mit einem Gesichtserkennungsverfahren möglich wäre. Die Verarbeitungsschritte werden in der Abbildung 6.12 zusammengefasst.

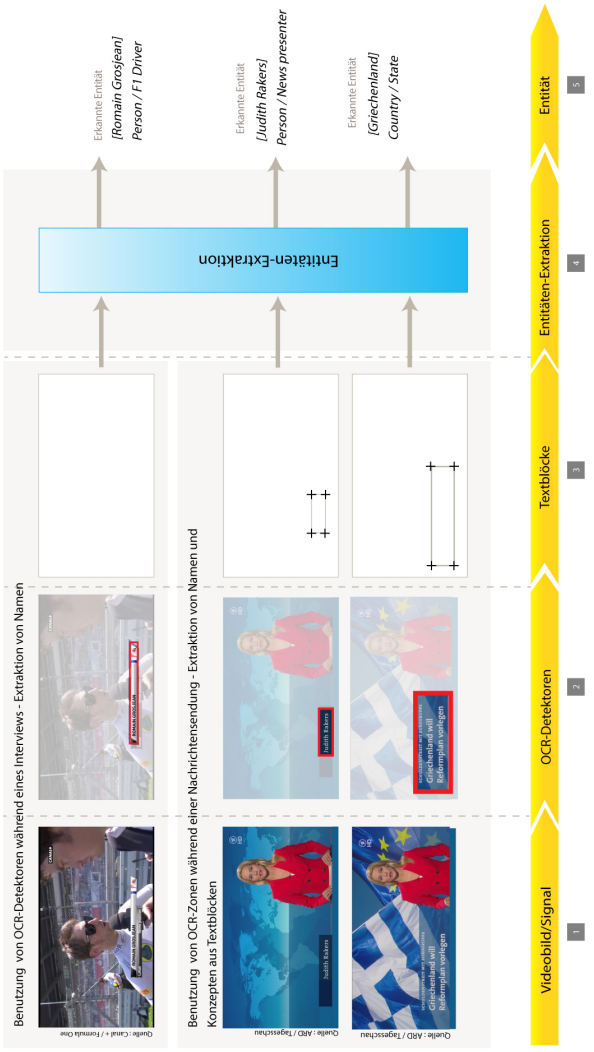


Abbildung 6.12: Prozess der OCR-Erkennung mit Vorlagen

Das Verfahren hat den Vorteil, dass unter der Annahme, dass jede Fernsehsendung fixe grafische Elemente besitzt, wie z.B. Einblendungsblöcke für den Namen des Moderators oder für die Anzeige des Aufnahmeortes des Korrespondenten, der textuelle Kontext (d.h. Ort plus Thema) dank der Position der Zonen hergeleitet werden kann (siehe Abbildung 6.14).
Folgende Abbildungen zeigen für verschiedene Sendungen, wie das Zusammenspiel zwischen *OCR-Zonen* und den Kontextinformationen aussieht.

Abbildung 6.13: Übereinstimmung der OCR-Zonen mit der Ortsangabe bei einer Liveübertragung der Tour de France Quelle: France 2 – Le Tour de France

Abbildung 6.14: OCR-Zone mit dem Namen des Reporters und Angabe des Ausstrahlungsortes Quelle: ARD – Tagesschau

Abbildung 6.15: OCR-Zone mit dem Namen des Interviewten. Quelle: Canal +

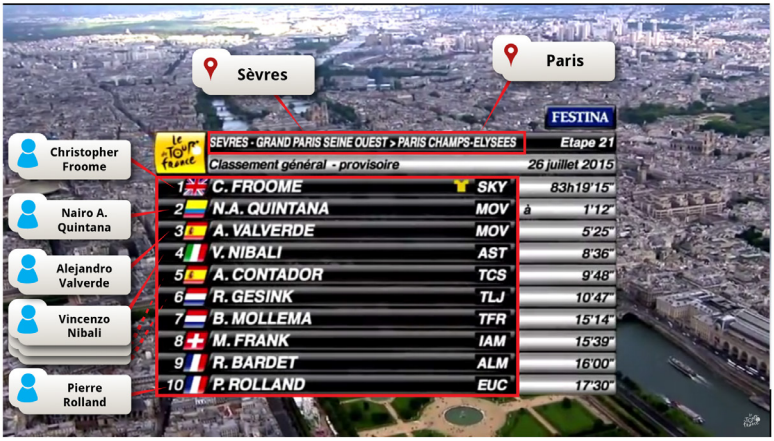


Abbildung 6.16: Übereinstimmung der OCR-Zonen mit einer Tabelle. Hierbei werden Ortsangaben und die Namen der Rennfahrer extrahiert. Quelle: France 2 – Le Tour de France

Die Bestimmung der *OCR-Zonen* wird mittels eines Werkzeugs erreicht, das eine visuelle und softwaretechnische Unterstützung anbietet. Dieses Werkzeug ermöglicht pixelgenau die Platzierung der Zonen in einem Video und die Vorauswahl der Konzepte nach Kategorien (Person, Objekt, Gebäude, usw.) und internen Beschreibungen (z.B. *Börse*, *Drehort* oder Name der Sendung) mittels einer grafischen Editieroption. Diese werden als persistente Vorlagen für eine weitere Verwendung durch die Videoanalysekomponente auf einem Server zwischengespeichert.

Listing 6.3: Struktur einer vordefinierten OCR-Zone mittels einer Vorlage für die Sendung ARD-Tagesschau zur Erkennung des Namens der Moderatorin und des Themas

```
{
  "channel":"ARD",
    "showdescription":"Tagesschau - 19.11.2015 | 09:00 Uhr",
    "ocrZones":[
    {
      "id":"1",
            "description":"Moderator Name",
      "rectangle":{
        "x":"442",
        "y":"912",
                "height":"434",
                "width":"130"
      },
            "semanticInformation": {
                          "type": "Person"
            }
    },
        {
      "id":"2",
            "description":"News Topic",
      "rectangle":{
        "x":"150",
        "y":"630",
                "height":"860",
                "width":"300"
      },
            "semanticInformation": {
            }
    }
  ]
}
```

Die Vorlagen der *OCR-Zonen* werden manuell editiert und können mittels einer C# basierten Applikation überprüft werden (siehe Abbildung 6.17). Zusätzlich können sogenannte „Detektoren“ eingesetzt werden, die mittels einer Pixelanalyse die Farbveränderungen der OCR-Zonen analysieren.

Abbildung 6.17: Bildschirmauszug der C#-Applikation zur Kontrolle der OCR-Zonen

Diese Detektoren sind für den Fall nützlich, dass Einblendungen nicht persistent oder nur für ein paar Sekunden angezeigt werden. Um keine unnötigen OCR-Analysen durchzuführen, werden diese nur dann gestartet, wenn eine unmittelbare, deutlich farblich erkennbare Veränderung der OCR-Zone stattfindet. Ab diesem Zeitpunkt wird die eigentliche OCR-Analyse durchgeführt. Abbildung 6.18 zeigt wie eine Logoeinblendung (hier das Logo der Tour de France) benutzt werden kann, um den Start der OCR-Erkennung zu triggern. Die Einblendung

des Logos deutet darauf hin, dass ein textueller Inhalt z.B. die Namen der Radfahrer oder des Ortes in einer vordefinierten Zone vorhanden sind.

Abbildung 6.18: Beispiele eines Logo-basierten Detektors

Nachdem die Positionen der *OCR-Zonen* und die Detektoren über die dedizierte Software Swoozy-Livana (siehe Kapitel 7) eingestellt worden sind, können diese, wie bereits erwähnt, persistent in Form einer strukturierten XML-Datei gespeichert werden. Das bedeutet für regelmäßig ausgestrahlte Sendungen (z.B. Nachrichten, Magazine oder Berichte), bei denen die Einblendungspositionen gleich bleiben, dass die gleichen OCR-Zonenvorlagen jedes Mal erneut für die Analyse benutzt werden können. Verändert die Sendung ihre Einblendungspositionen (oder ihr komplettes *Corporate Design*), muss die Vorlage entsprechend nachbearbeitet werden.

Technische Realisierung

Die OCR-Erkennungskomponente wurde in der Programmiersprache C# und mittels der OpenCV-Wrapperbibliothek Emgu-CV[22] implementiert (siehe Punkt 2 von Abbildung 6.11). Die Hauptaufgabe der Komponente ist zum einen, im abgespielten Fernsehbild die Dekodierung der OCR-Zonen durchzuführen, und zum anderen, in Echtzeit Textblöcke daraus zu extrahieren. Dabei werden die oben beschriebenen, rechteckigen OCR-Zonen analysiert (3). Mit einer Taktung von 200 bis 500ms wird die OCR-Analyse angestoßen. Die eigentliche Analyse, d.h. die Umwandlung in maschinenlesbare Textblöcke, wird über die Open Source Tesseract OCR Engine[23] Bibliothek, die eine Vielzahl von gängigen Schriftarten erkennen kann, durchgeführt. Der daraus resultierende Text wird weiterverarbeitet (4) (5) und die darin enthaltenen potentiellen Entitäten extrahiert.

Gesichtserkennung

Die Gesichtserkennungskomponente dient als Erweiterung der OCR-Komponente und ermöglicht es, Personen im Livefernsehbild zu erkennen. Damit die Gesichtserkennungskomponente in der Lage ist, diese Personen zu erkennen, müssen im Vorfeld in Form einer Datenbank Bildinformationen der Personen angelegt werden. Diese wird manuell durchgeführt, indem Portraitbilder von Schauspielern samt einer semantischen Annotation gespeichert und mit einer Verknüpfung zu einer Wissensdatenbasis versehen werden. Die Gesichtserkennungskomponente, die innerhalb des Swoozy-Systems benutzt wird, stützt sich auf die OpenCV-Bibliothek.

[22] `http://www.emgu.com`
[23] `https://github.com/tesseract-ocr`

Diese Bibliothek bestimmt in einem ersten Schritt nur die grafische Position eines Gesichts innerhalb eines Bildes, ohne es zu bewerten. Nachdem dies erfolgt ist, wird anhand der markanten Gesichtspunkte versucht, dieses mit denen des vorannotierten Bildmaterials zu vergleichen. Im Fall, dass mehrere Gesichter gleichzeitig erkannt werden, kann nicht immer gewährleistet werden, dass eine hohe Genauigkeit der Erkennung erzielt wurde. Dadurch steigt die Wahrscheinlichkeit des Auftretens der sog. „False-Positives" verbunden mit einem gleichzeitigen Sinken der Genauigkeit der Erkennung.

Aus diesem Grund wird die Gesichtserkennung nur partiell und in bestimmten Situationen wie z.B. einer moderierten Sendung oder bei Talkshows, bei denen die Gäste relativ nah gefilmt und leicht über eine Datenbank abgeglichen werden können, benutzt. Gesichtserkennung funktioniert am besten mit Personen, die in der Öffentlichkeit stehen oder den sog. Stars, da in diesen Fällen eine große Menge suchbares Bildmaterial in online verfügbaren Fotodatenbanken zu finden ist.

Im Fall des Swoozy-Systems wird die Gesichtserkennung als Unterstützung zur visuellen Positionierung der *Grabbables*, die über das Live-Bild platziert und vom Benutzer initiiert werden (siehe Abbildung 6.24), benutzt. Das bedeutet, dass die grafische Position des Gesichtsvierecks (samt Angaben der X-, Y-Positionen und der Höhe bzw. Breite des Bereichs) innerhalb des Videos dazu dient, die *Grabbables* an der richtigen Position über dem Videosignal einzublenden. In den neueren Versionen von Swoozy wird die gezielte Gesichtserkennung über den Onlinedienst Microsoft Cognitive Services durchgeführt.

Objekterkennung

Die Objekterkennung ist im Rahmen des Systems als prototypisches Zusatzmodul integriert worden und wird nur dann benötigt/aktiviert, wenn über vorangefertigte Vorlagen und Objektdatenbanken bestimmte Objekte und Logos aus Fernsehbildern extrahiert werden müssen. Hierbei spielte die Idee, bestimmte kommerzielle Gegenstände hervorzuheben und damit den Kauf aus dem TV-Programm heraus zu unterstützen eine Kernrolle, ähnlich dem WIREWAX[24]- oder dem LookAt[25]-Ansatz (siehe Kapitel 5).
Dabei ist der Gedanke eines videogestützten Produktplacements besonders wichtig. In Filmen werden oft Markenprodukte gezeigt, die vom Zuschauer entweder indirekt, d.h. das Logo des Herstellers wird im Bild positioniert oder direkt, d.h. ein Produkt des Herstellers wird gezeigt, wahrgenommen werden. Durch eine semi-automatische Objekterkennung kann beispielsweise ein eventueller Kaufvorgang vom System assistiert und interaktiver gestaltet werden. Als Ergebnis wird genau wie bei der videogestützten OCR und der Gesichtserkennung eine semantische Struktur (SwoozyML) ausgegeben, welche die grafischen Koordinaten des gefundenen kommerziellen Objektes und seiner textuellen Beschreibung beinhaltet.

Audio- und Audiodeskription-basierte Wissensextraktion

Eine rein bild-basierte Analyse liefert vorwiegend grafische Details zum Geschehen innerhalb einer Sendung, z.B. durch die Erkennung von Protagonisten, Schauspielern oder Objekten. Oft reichen diese

[24] http://www.wirewax.com
[25] http://www.lookat.io

Informationen nicht aus, um eine komplette und vollständige Beschreibung der Livevideos zu erhalten. Es fehlen Kontextinformationen wie z.B. die Thematik der Sendung oder der Geschehnisse.

Aus diesem Grund wurde in das Swoozy-System eine erste prototypische Komponente implementiert, die in der Lage ist, aus der Original-Tonspur und der Audiodeskription-Tonspur Texte zu generieren. Dieses Verfahren wird als *Speech-To-Text* bezeichnet.

Dem Cloud-basierten Ansatz folgend, wurde bei der Echtzeitverarbeitung, anstatt das Spracherkennungsmodul auf den Swoozy-Clients zu installieren, was nur mit dem Set-Top-Box-Szenario technisch realisierbar wäre, auf zwei Online-Dienste zurückgegriffen, die in der Lage sind, ohne Verzögerung gesprochene (Teil)-Sätze in Texte umzuwandeln. Diese kombinierten Dienste übernehmen die Rolle des *Speech-To-Text*-Moduls (siehe Punkt 2 von Abbildung 6.4) innerhalb der Swoozy-Komponentenarchitektur.

Zusätzlich kann bei Filmen, obwohl nicht immer erforderlich, auf sog. SRT-Dateien (Subtitle Rip (kurz SubRip))-Dateien[26] zurückgegriffen werden, die ein vollständiges Transkribieren von Dialogen und Texten samt Zeitstempel beinhalten. Diese SubRip-Dateien enthalten eine abgeschlossene und korrigierte Version der Audiospur. Werden sie verwendet, kann auf die Echtzeit-Audioanalyse verzichtet werden. Bei Filmen oder Reportagen, die schon mehrmals wiederholt wurden, können die so extrahierten Untertitel als Grundlage für eine anschließend durchgeführte Textanalyse dienen.

[26] `http://zuggy.wz.cz`

Listing 6.4: Ausschnitt einer Transkribierung der BBC-Serie Sherlock

```
[...]
91
00:05:59,480 --> 00:06:01,599
Oh, I’m sure something
will turn up, Sherlock.

92
00:06:01,600 --> 00:06:05,519
A nice murder. That’ll cheer you up.

93
00:06:05,520 --> 00:06:07,399
Mmm. Can’t come too soon.

94
00:06:07,400 --> 00:06:10,120
Hey, what have you done
to my bloody wall?!

95
00:06:11,440 --> 00:06:15,080
I’m putting this on your rent,
young man!

96
00:06:18,200 --> 00:06:19,440
EXPLOSION BOOMS

97
00:06:19,441 --> 00:06:21,000
CAR ALARM BEEPS

[...]
```

Audioanalyse

Die Audioanalyse wird benutzt, um aus einem Video die Handlung und die Dialoge zu extrahieren, und somit die Emotionen, die allgemeine Stimmung, aber auch die angesprochenen Themen zu identifizieren. Unter Original-Audioton oder O-Ton versteht man die unveränderte Audiospur, die mit einem Video verbunden ist. Diese enthält meistens Dialoge, Hintergrundmusik und Hintergrundgeräusche. In Swoozy werden prototypisch zwei Cloud-basierte, Speech-To-Text-Komponenten benutzt, die in der Lage sind, online und in Echtzeit eine Analyse des Audiosignals durchzuführen. Die erste Komponente ist die IBM Watson Cloud Speech-To-Text Service API[27], die auf der *(Cloud) Platform as a Service (PaaS)* von IBM Bluemix beruht. Der IBM Watson Cloud Speech-To-Text-Dienst ermöglicht über eine API, ein Mikrofon und über den Webbrowser die direkte Aufnahme von englischsprachigen Audioeingaben. Alternativ können Audiodateien als Eingabedatei benutzt werden. Zur Verwendung innerhalb des Swoozy-Servers wurde die node.js-basierte API von IBM Watson benutzt, welche die Verbindung zu den IBM-Servern und dem Streaming der Spracheingaben (über Websocket) kapselt. Als Ergebnis der Erkennung wird eine JSON-Struktur samt Metadaten geliefert, die jedes Wort inklusive Konfidenzwert und Zeitstempel auflistet. Auf Basis der Metadaten werden alternative Hypothesen angeboten. Diese JSON-Struktur wird mit dem Ergebnis der zweiten Komponente, der Web Speech API[28] kombiniert. Dafür muss der dedizierte Swoozy-Server mit zwei Mikrofonen (oder Mikrofon-In Anschlüssen) ausgestattet sein, damit beide Speech-To-Text-Komponenten, wie in Abbildung 6.19 skizziert, gleichzeitig die

[27] `http://www.ibm.com/smarterplanet/us/en/ibmwatson/developercloud/doc/speech-to-text/`

[28] `https://dvcs.w3.org/hg/speech-api/raw-file/tip/speechapi.html`

Analyse durchführen können.
Im Falle der Web Speech API, bei der die Erkennung auf den Servern von Google durchgeführt wird, wird ebenfalls das Mikrofon des Webbrowsers (in diesem Fall des Google Chrome Browsers) als Quelle benutzt. Die Web Speech API kann nur über eine browserspezifische Javascript-basierte API gestartet und angesteuert werden. Diese API kann zurzeit nicht von einem zwischengeschalteten Server gekapselt werden. Konkret bedeutet dies, dass der Swoozy-Server als solcher nicht die Spracheingaben verwalten kann und diese nicht innerhalb seiner node.js-Instanz aufgelöst werden können.

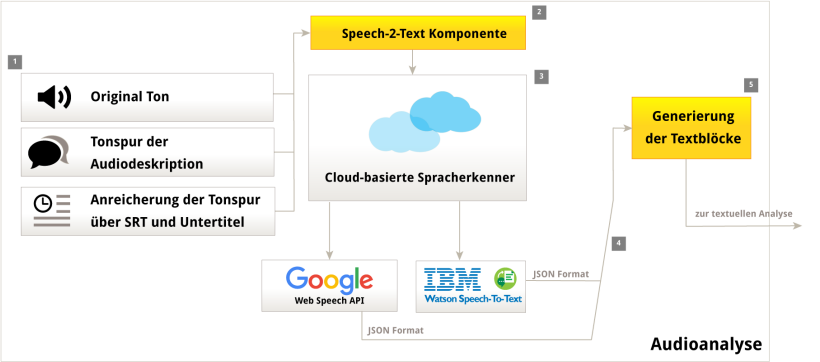


Abbildung 6.19: Übersicht des Extraktions-Worflow und Benutzung der Speech-To-Text-Analyse

Um diese technische Einschränkung zu umgehen, wird eine minimalistische App in Form einer Webbrowserinstanz auf einem dedizierten Rechner gestartet. Diese App extrahiert über Javascript die im Browser aufgezeichneten Texte und übergibt sie dem Swoozy-Server. Das bedeutet, dass die Ergebnisse der Speech-To-Text-Erkennung in textueller Form zum Swoozy-Server weitergereicht werden.
Nachdem die beiden Speech-To-Text-Komponenten ihre Ergebnisse

erzeugt haben, werden diese zusammengefügt und mit einem vereinfachten Verfahren überprüft (siehe Abbildung 6.20). Die gemeinsam erkannten Wörter der beiden Komponenten werden übernommen und zur Weiterverarbeitung durch die Textanalysekomponente weitergereicht.

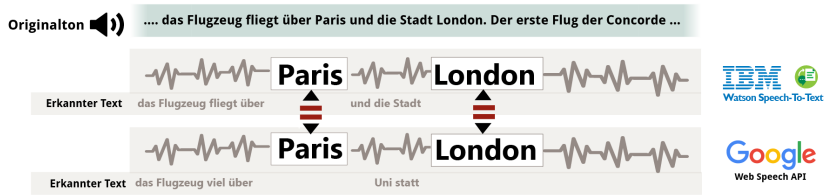


Abbildung 6.20: Überprüfung des Ergebnisses der Audio-Analyse

Audiodeskription

Über die oben genannten Speech-To-Text-Komponenten wird ebenfalls die Audiodeskriptionspur einer Sendung analysiert und in textueller Form für eine semantische Analyse weiterverarbeitet, sodass im Nachhinein eine Szene komplett und semantisch korrekt beschrieben werden kann.

Die Audiodeskription (kurz AD) (auch Audiokommentar genannt) ist ein Verfahren, das von den Fernsehanstalten benutzt wird, um Szenen und Handlungen mit einer akustischen Beschreibung zu versehen. Ziel dabei ist, den Zugang zu Fernsehprogrammen für Blinde und sehbehinderte Menschen zu ermöglichen[29]. Dieses Verfahren wird als barrierefrei bezeichnet. Parallel zur Einblendung der Untertitel wird eine ausführliche audio-basierte Beschreibung des Videoinhalts

[29] `http://www.csa.fr/Television/Le-suivi-des-programmes/L-accessibilite-des-programmes/Pour-les-personnes-sourdes-ou-malentendantes-le-sous-titrage/Principes-techniques`

angeboten. Somit bekommen Personen, die Schwierigkeiten haben die Videos visuell zu erfassen oder die darin enthaltenen Handlungen und Gegenstände zu erkennen, eine akustische Unterstützung.
In Filmen oder Reportagen werden die Szenen mit einer sog. Off-Stimme (Stimme, die nicht im originalen Ton zu finden ist) versehen. Diese Stimme beschreibt so präzise wie möglich das Geschehen innerhalb des Videos (z.B. Orte, Landschaften, Personen, usw.) und die damit verbundenen Aktionen wie z.B. „Er steigt in ein rotes Auto. Das Auto fährt nun die Straße in Berlin mit dem Namen „Kudamm" herunter [...]".
Sehr oft wird dies als Hörfilmfassung bezeichnet. Leider wird die Audiodeskriptionspur nicht bei jeder Sendung mitausgestrahlt, da Aufwand und Kosten für die Vorbereitung (Einlesen, Annotieren und Synchronisieren) noch sehr hoch sind und meistens manuell von einem Redaktionsteam während der Ausstrahlung nachgebessert werden muss. Aus diesem Grund werden nur größere Produktionen wie Spielfilme oder Sportveranstaltungen mit der speziellen Audiodeskriptionspur versehen. Diese Situation wird sich in den kommenden Jahren, dank der Unterstützung durch Vereine und Hilfsgruppen für Blinde und Sehbehinderte, verbessern. Die Bereitstellung einer Audiodeskription durch einen Fernsehsender erfolgt dadurch, dass eine zweite Tonspur parallel zur originalen Audiospur zusätzlich mit dem Video ausgestrahlt wird. Der DVB-Receiver, oder präziser gesagt: der darin integrierte DVB-Demuxer, bietet beide Audiokanäle an. Der Zuschauer kann frei auswählen, welche Audiospur er gerne hören möchte. Die Audiodeskription bietet also in akustischer Form eine Beschreibung der jeweiligen Filmszene. Bei IP-TV oder VOD kann die Audiodeskription bereits als vollständige mit Zeitstempel versehene Textstruktur (in XML/JSON-Format) vom Fernsehsender selbst mitausgestrahlt werden.

Die Audiodeskription kann folgende Zusatzinformationen (nach der Verarbeitung durch eine Entitäts- und Konzeptextraktion) bereitstellen:

- Eine detailliertere Szenen- und Kontextbeschreibung, die mit der videobasierten Erkennung alleine nicht möglich wäre.
- Eine Überprüfungskomponente für die videobasierte Analyse und Erkennung.
- Falls keine der OCR- oder Bilderkennungskomponenten in der Lage war, Elemente des Videos zu erkennen, z.B. aufgrund zu schneller Aktionen in der Szene, komplexen Dialogen, speziellen Kameraeinstellungen wie z.B. Weichzeichnen (engl. Blurring) oder der Verwendung anderer künstlerischer Effekte, wird die Audio-Description-Spur benutzt, um diese Informationslücke zu schließen und ggfs. eine alternative Beschreibung zu gewinnen.
- Eine deutlich feinere Erkennung der einzelnen Aktionen, die mit den klassischen Bilderkennungsalgorithmen semantisch nicht erfassbar sind wie z.B. die körperliche Haltung einer Person (z.B. *Er sitzt gerade in einem Sessel*) oder die Bewegung eines Gegenstandes innerhalb einer Szene (z.B. *Ein Flugzeug landet am Flughafen unter einem blauen Himmel*).
- Die Erkennung der Dramaturgie einer Handlung durch die Analyse (Stichwort: *Sentiment Analysis*) der Satzbildungen und der Gewichtungen bestimmter Begriffe kann helfen, die allgemeine Stimmung einer Szene oder einer Sendung zu erfassen.

All diese Informationen werden zusammengefasst und dienen zur Anreicherung der Bilderkennung und des Programminhaltes.

Die technische Realisierung der Audioanalyse ermöglicht, dass parallel auch textuelle Zusatzinformationen aus dem Videomaterial gewonnen werden können. Obwohl die Cloud-basierten Ansätze zur Audioanalyse sehr vielsprechende Ergebnisse liefern, sollte für eine präzise und rauschfreie Erzeugung der Ergebnisse eine manuelle Nachbearbeitung erfolgen, aus dem Grund, dass nicht alle automatisch extrahierten Entitäten im Kontext einer Szene Sinn machen oder eine Bedeutung für den Zuschauer haben. Diese Nacharbeitung kann durch einen Redakteur über das Swoozy-SKRPTR Tool, das im nächsten Kapitel näher beschrieben wird, erfolgen.

Überprüfung

Die letzte Komponente im Extraktions-Workflow, bevor die Ergebnisse zum Swoozy-Client weitergereicht werden, ist die sog. Überprüfungskomponente. Diese misst mittels eines mehrstufigen Verfahrens die Plausibilität der einzelnen Annotationen innerhalb eines Textes und versucht, eine ausreichende Qualität bei der Erkennung und Anzeige der *Grabbables* zu gewährleisten. Technisch betrachtet kann nicht immer gewährleistet werden, dass alle erkannten Entitäten miteinander korrelieren und zueinander passend sind. Um dieses Problem zu lösen, werden die Ergebnisse der OCR-, der Video- und der Textanalyse miteinander verglichen, die gemeinsam auftretenden Begriffe (Annotationen) aufgelistet und nach Häufigkeit ihres Auftretens angeordnet. Tritt ein Begriff häufiger auf, ist die Wahrscheinlichkeit groß, dass dieser tatsächlich von einer der Analysekomponenten korrekt erkannt wurde. Daraus kann geschlossen werden, dass der Begriff im analysierten Video auftaucht. Zusätzlich kann eine Gewichtung jeder einzelnen Komponente angegeben werden, d.h. welche Komponente

durch die Überprüfungsanalyse als relevant bei der Analyse betrachtet werden muss. Dadurch wird bestimmt, ob während des Erkennungs- und Überprüfungsprozesses die OCR-Erkennung oder die Gesichtsanalyse eine höhere Priorität besitzen soll. Die Ergebnisse der Audiospur- oder Textanalyse können höher gewichtet werden. In Sendungen wie z.B. Talkshows spielt die Audiospur eine deutlich wichtigere Rolle als z.B. bei einer Sportübertragung. Bei letzterer werden die Namen der Spieler über OCR-Verfahren schneller extrahiert als über eine Audioanalyse. Diese Priorisierung und Gewichtung der Komponenten wird über einen Parameter auf Seite des Swoozy-Servers vorgenommen. Nach diesem letzten Schritt werden die *Grabbables* vom Swoozy-Server bereitgestellt (siehe Punkte 13 und 14 von Abbildung 6.4) und dem Swoozy-Client zur Darstellung weitergereicht.

Swoozy-Client

Motivation

In Kapitel 3 "Wissenschaftliche Arbeiten im Kontext intelligentes Fernsehen" wurden die grafischen Benutzerschnittstellen verschiedener Hersteller und Smart-TV-Systeme vorgestellt. Diese sind sehr TV-Appzentrisch und für ein schnelles Suchen und Navigieren im Web nicht geeignet, da jedes Mal ein Interaktionsbruch zwischen dem TV-Programm und der ausgewählten App stattfindet. Ein weiteres Problem bei den aktuellen TV-Benutzerschnittstellen ist die sehr beschränkte Möglichkeit einer Websuche, besonders eine semantische Suche direkt vorm Fernseher aus zu initiieren, ohne dass es zu Interaktionsproblemen kommt. Oft bildet die „klassische" Fernbedienung die einzige Eingabe- und Interaktionsmöglichkeit mit dem Fernseher. Diese Interaktions-

form kann zwar gelegentlich benutzt werden, jedoch hat sie sich nicht als dauerhafte und bequeme Lösung bewährt. Die TV-App-zentrische Bedienung der neuen Smart-TV-Geräte zeigt im Rahmen der Benutzerfreundlichkeit deutliche Grenzen auf. Sehr oft hat der Benutzer Entscheidungsprobleme, welche App er starten soll (die App vom Fernsehsender? Oder doch die Wikipedia-App? Oder vielleicht doch das HbbTV-Angebot?), um die gewünschten Informationen zu erhalten oder zu dem aktuell laufenden Programm zurückzuschalten. Dabei stellen sich die Fragen: Welche App des jeweiligen Fernsehsenders muss gestartet werden? Berücksichtigt diese App auch den aktuellen Kontext? Wie lange muss man innerhalb der ausgewählten App navigieren, damit man zu der gewünschten Information gelangt?
Die Lösung dieses Problems wird bei den heutigen Ansätzen dadurch erschwert, dass Benutzer oft von einem Sender zum anderen „zappen". Die Kernfrage ist, ob es wirklich Sinn macht, dass der Benutzer für jeden ausgewählten Fernsehsender jedes Mal eine neue App starten und seine Suchbegriffe neu eintippen muss.
Die aktuellen Versionen der Smart-TVs bieten zwar die Möglichkeit mehrere Apps direkt auf den Geräten zu installieren, leider ist es aber so, dass die Interaktion, insbesondere das schnelle Wechseln vom Live-TV-Programm zu der jeweiligen App, immer mit einem Interaktionsbruch verbunden ist. Das bedeutet, dass immer eine zeitliche Verzögerung zwischen dem Verlassen des Live-Programms, der Auswahl der zu startenden App und ihrem Start existiert und somit eine flüssige Benutzerinteraktion verhindert wird. Die Interaktion und das Zusammenspiel zwischen TV-Apps und TV-Programmen sind zurzeit nicht optimal umgesetzt. Die Möglichkeiten einer benutzerzentrierten Interaktion sind immer noch stark limitiert.

Ausgehend von diesen Beobachtungen wurde ein neuer Ansatz verfolgt und somit die Art und Weise wie der Zuschauer mit dem Fernsehen interagiert, komplett neu definiert. Hier wurde zu Gunsten einer innovativen und intuitiven benutzerorientierten grafischen Benutzeroberfläche vollständig auf das aktuelle TV-App-Konzept verzichtet.

Prinzip

Wenn der Zuschauer die grafische Oberfläche von Swoozy benutzt, besitzt er die Möglichkeit, ein *Grabbable* per Geste zu selektieren und dieses per Handgeste in eine auf der rechten Seite platzierten sog. *Dropzone* zu platzieren. Diese Interaktion wird vom *Gesteninterpretations-Modul* analysiert und entsprechend in eine interne Aktion (*Grab'n'Drop*) umgewandelt. Diese Interaktionsformen können mittels verschiedenster Eingabegeräte ausgeführt werden wie z.B. der Leap Motion oder der Microsoft Kinect. Mit Hilfe letzteren Moduls wird die technische Verbindung der Hardware mit einem sog. *Device Controller* realisiert. Dieser hat die Aufgabe, sämtliche Hardwarekomponenten zu vereinheitlichen. Somit ist es trotz heterogener Eingabehardware wie z.B. Kinect, Leap Motion oder Gyroskop-basierter Fernbedienung möglich, diese alle mit dem Swoozy-Client zu verbinden und die richtigen Aktionen auf Seiten der grafischen Benutzerschnittstelle auszulösen (siehe Abbildung 6.21).

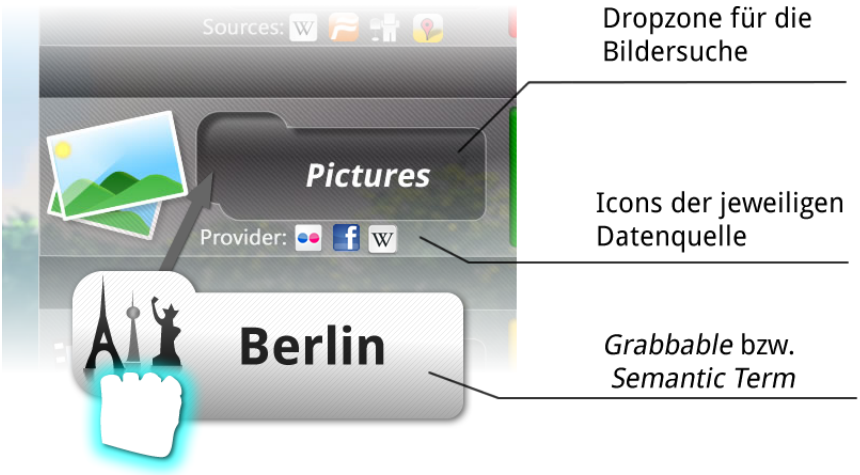


Abbildung 6.21: Detailansicht zu einer Dropzone

Design und Aufbau der Benutzerschnittstelle

Die implementierte grafische Benutzeroberfläche des erstellten Systemprototypen ist absichtlich sehr einfach gehalten und folgt bzgl. ihrer Konzeption dem 10-foot Design [RC06] [Sam13] [Loi11] (siehe Kapitel 3), das einen guten Kompromiss zwischen intuitiver Benutzeroberfläche, Lesbarkeit der Texte und Einfachheit der Interaktion darstellt, damit nicht-computeraffine Benutzer in der Lage sind, das System zu benutzen, ohne komplexere Interaktionen mit der Fernbedienung durchzuführen und unübersichtliche Menüs bedienen zu müssen. Eine weitere Motivation bei der Konzipierung des Designs war, dass verschiedene Designs (Stichwort *Corporate Identity*) für die Benutzerschnittstelle verwendet werden können, ohne die Grundfunktionalität und die Interaktionsform verändern zu müssen. Somit

wurde ein generischer Aufbau der grafischen Benutzerschnittstelle gewährleistet. Das Aussehen der Schnittstelle (Farben der UI, Icons und Logos der Webdienste) kann anschließend von einem Designer verändert werden, z.B. durch Anpassung von CSS-Dateien im Falle der HTML5-basierten Version von Swoozy oder durch die Veränderung der grafischen Flash-basierten Elemente mit einer Software wie z.B. Adobe Illustrator.
Bei der Konzipierung und dem Design der TV-Benutzerschnittstelle von Swoozy wurde besonders viel Wert auf die Einfachheit der Benutzung und möglichst wenig ablenkende grafische Elemente gelegt. Trotz komplexer semantischer Verarbeitungsschritte, die im Hintergrund nahtlos und „versteckt" ausgeführt werden, wirkt die Benutzerschnittstelle nicht disruptiv und behält jederzeit strikte Design- und Interaktionsprinzipien als Grundlage ihrer grafischen Gestaltung bei. Folgende Regeln, die aus wissenschaftlichen Arbeiten abgeleitet wurden, dienten als Grundlage und Inspirationsquelle für das Design der grafischen Swoozy-Benutzeroberfläche:

- Das aktuelle Video sollte jederzeit sichtbar sein und niemals komplett ausgeblendet werden. Somit wird sichergestellt, dass kein Interaktionsbruch und damit keine zeitliche Unterbrechung zwischen dem Wechsel von TV-Bild und dem tatsächlichen Start der App erzeugt wird.
- Jedes Element und jede Funktionalität soll maximal mit nur einer Interaktion aufrufbar sein.
- Um die Interaktion benutzerfreundlicher zu gestalten, wurde explizit auf komplexere Interaktionsformen und Untermenüs verzichtet.

- Alle Elemente sollen entlang des *10-foot Designs* gestaltet werden und eine besonders gute Lesbarkeit auch aus einer größeren Entfernung (3 bis 5 Meter) gewährleisten.
- Der Benutzer soll jederzeit in Erfahrung bringen können, welche Webdienste ihm welche Ergebnisse liefern. Somit wird ihm eine Quellentransparenz während der Suche gewährt.
- Die semantischen Konzepte sollen einfach aber klar visualisiert werden, ohne dabei zu komplex zu wirken. Die Konzepticons sollen gut erkennbar sein und die dahinterliegende Semantik zu Gunsten der Interaktion und der Einfachheit „verstecken".
- Der Benutzer sollte jederzeit ohne Verzögerung in der Lage sein, von einem durch ein Grabbable selektierten Video zum TV-Bild zu wechseln. Dabei sollte die Swoozy-GUI auf Benutzerwunsch visuell „verschwinden", aber trotzdem ohne Verzögerung wieder aufrufbar sein.

Diese Regeln wurden beim Design und Aufbau der Benutzerschnittstelle beachtet. Die Abbildung 6.22 zeigt den genauen grafischen Aufbau der Swoozy-Benutzerschnittstelle.
In der Mitte der Benutzerschnittelle wird das reguläre Fernsehprogramm (z.B. über einen DVB-T-Empfang oder IP-TV) wiedergegeben. Per Geste kann das Swoozy-UI aktiviert oder deaktiviert werden. Das Hauptvideobild (TV- und Videobereich) ist in jeder Interaktionsphase sichtbar: entweder im Großformat oder in einem Kleinformat, wenn sich der Benutzer im Kanalauswahlmenü befindet, oder wenn er sich genauere Informationen über ein bestimmtes Thema anschaut. Sogenannte grafische Annotationen in Form von *Grabbables* können nach einer Gesteninteraktion über dem Hauptvideobild eingeblendet

werden. Die Funktionen werden am Ende des Abschnitts erklärt.
Auf der rechten Seite der grafischen Benutzerschnittstelle wird dem Benutzer eine Seitenleiste mit fünf thematischen *Dropzones* mit den Bezeichnern *Facts* und *News, Pictures, Videos, Shop it!* und *Share It!* angezeigt. Diese werden intern mit spezifischen semantischen Webdiensten verknüpft (engl. Mapping) und dienen als Eingabepunkt und Trigger für die eigentlichen Suchanfragen. Die Metapher der *Dropzones* ist eine grafische und technische Anpassung des *Spotlets*-Mechanismus, einem grafischen, intelligenten, Touchscreen-basierten Suchagenten (siehe Kapitel 5 und [SDB09] [BLS+14] [BDP10]), der grafische Eingaben erlaubt und dementsprechend einen semantischen Bezug zum „abgelegten" Element schafft. Damit der Benutzer erkennen kann, welche Dienste ihm welche Ergebnisse zurückgeliefert haben, werden die Logos der verschiedenen Informationsquellen eingeblendet (siehe Abbildung 6.21). Die *Dropzones* verarbeiten *Grabbables* und führen eine implizite Analyse des semantischen Inhalts durch. Ähnlich sieht es bei den Ergebnissen aus. Sie werden in Form von größeren Vorschauelementen (z.B. *Thumbnails*) grafisch dargestellt. Die Größe wurde so ausgewählt, dass trotz der Entfernung des Benutzers zum Fernsehbildschirm die Vorschauelemente immer noch per Geste gut selektierbar und mittels der virtuell eingeblendeten Hand „greifbar" bleiben. Hier hatte die Interaktionsform eine direkte Auswirkung auf das Design und die Konzipierung der Benutzerschnittstelle. Zusätzlich ist festzuhalten, dass viele für Smart-TV konzipierte Apps nur eine 1:1 Kopie ihres mobilen Pendants sind. Bezüglich der grafischen Darstellung und der Interaktionsmöglichkeiten bedeutet dies, dass die Apps einerseits ihren mobilen Versionen grafisch sehr ähneln und andererseits oft nicht die kompletten Möglichkeiten zur Interaktion mit dem Fernseher bieten, wie dies eine „klassische" Fernbedienung tun würde.

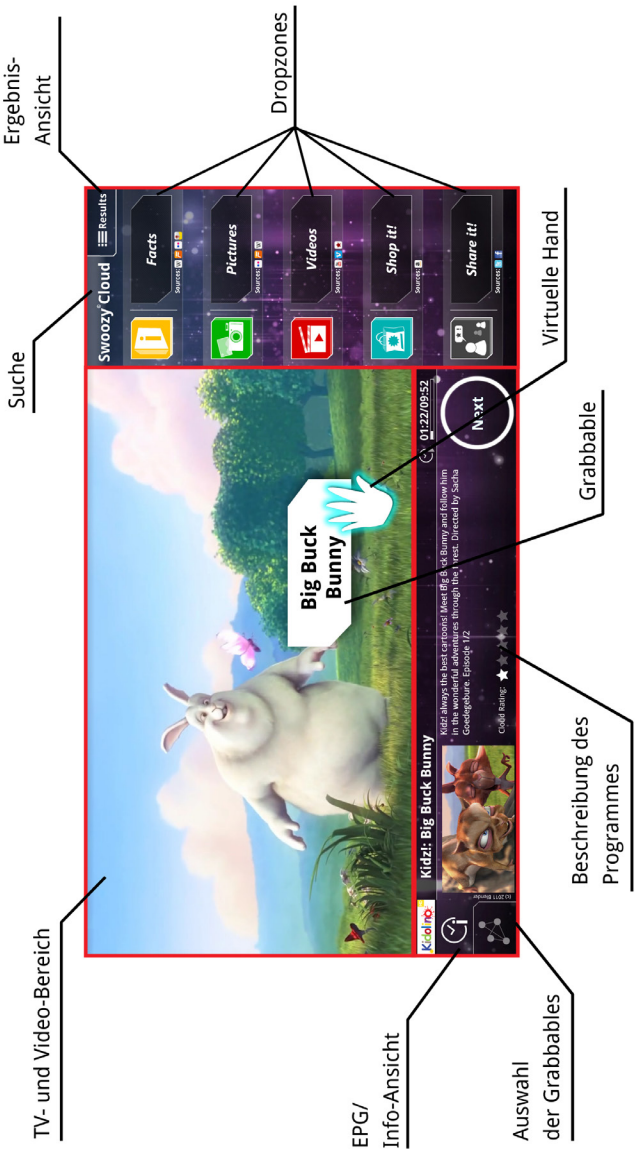


Abbildung 6.22: Benutzerschnittstelle von Swoozy

Aus diesem Grund distanziert sich die Swoozy-Benutzerschnittstelle seitens des Designs sehr stark vom „traditionellen“ grafischen Aufbau einer TV-App.

Die Designfragen wurden bei Swoozy so angegangen, dass die Swoozy-Benutzeroberfläche immer eine einheitliche Strukturierung der Ergebnisse generiert, auch wenn sie von unterschiedlichen und heterogenen Quellen aus dem Web stammen. Die Ergebnisse, die z.B. aus den REST-APIs der Dienste stammen, werden, bevor sie grafisch angezeigt werden, zunächst aggregiert und vereinheitlicht und dann in eine generische Struktur für die Vorschauliste geordnet und kompakt dargestellt. Somit können unterschiedliche Webdienste aufgerufen und deren Ergebnisse einheitlich präsentiert werden.

Abbildung 6.23: Benutzerschnittstelle von Swoozy: Ergebnisansicht

Gesten und Interaktionsformen

Interaktionen

Die Art und Weise, wie der Benutzer mit der Swoozy-Schnittstelle interagieren sollte, wird auch im Rahmen der Designabstimmung festgelegt. Hier wurde explizit auf die Verwendung einer klassischen Fernbedienung verzichtet und stattdessen eine Gesten-basierte Interaktion bevorzugt. Die Gestaltung der Interaktion wurde durch die Ergebnisse der Untersuchung der wissenschaftlichen Arbeiten motiviert. Die Ergebnisse und Beobachtungen wurden in folgende Regeln umgesetzt und innerhalb des Swoozy-Systems als grundlegende Handlungsleitlinien bei der Systemrealisierung benutzt:

- Die Gesten sollen besonders einfach und intuitiv gestaltet werden.
- Jederzeit sollte eine alternative Interaktion angeboten werden. Somit kann gewährleistet werden, dass, wenn der Benutzer nicht immer mit seiner Hand das Swoozy-System kontrollieren möchte, eine Bedienung des Systems trotzdem möglich ist. Hierbei spielen physische und physiologische Fragen wie z.B. Müdigkeit oder physische muskuläre Belastungen eine Rolle.
- Jede Geste sollte mit einem visuellen Feedback seitens der grafischen Benutzerschnittstelle versehen werden, d.h. was der Benutzer selektiert hat, sollte immer durch die Benutzerschnittstelle visualisiert werden (siehe [Mic15]).
- Die Suche oder die Auswahl von Funktionalitäten (wie die Selektion eines Ergebnisses) sollte nur mit einer einzigen Interaktionsphase realisiert werden.

Die Gestensteuerung wurde in folgender Form innerhalb von Swoozy eingebaut. Nachdem der Benutzer seine eigene Hand in Richtung des TV-Bildschirms bewegt hat, erscheint eine virtuelle Hand (siehe Abbildung 6.24). Bewegt der Benutzer seine Hand im Raum, wird diese vom Swoozy-System erfasst. Die Position der Hand kann entweder über eine Tiefenbildkamera, wie die Microsoft Kinect 1, oder für kleinere Wohnräume durch eine Finger-Tracking-Lösung, wie dem Leap Motion Controller-Gerät, verfolgt werden. Die virtuelle Hand spielt gewissermaßen die Rolle eines 3D-Cursors. Die möglichen Interaktionen werden in Swoozy, um u.a. Ermüdungserscheinungen zu unterbinden, auf zwei Varianten beschränkt: *Grab'n'Drop* und *Push*-Geste.

Abbildung 6.24: Darstellung der virtuellen Hand und Erzeugung eines Grabbables

Die Abbildung 6.25 beschreibt den vollständigen Workflow der Benutzerinteraktion innerhalb von Swoozy:

1. Während einer Sportsendung mit Sebastian Vettel erscheint ein *Grabbable* mit dem Begriff *Berlin*.

2. Durch die Leap Motion- oder Kinect-Gestenerkennung wird die Position der virtuellen Hand bestimmt. Bewegt der Benutzer seine (virtuelle) Hand in Richtung des Hauptvideobildes und führt er eine *Push*-Geste (auch Selektionsgeste genannt) aus, erscheint an der ausgewählten Stelle ein *Grabbable* (Punkt 1 von Abbildung 6.25).

3. Der Benutzer greift mit einer *Grab*- Geste das *Grabbable*. In diesem Beispiel werden als interne Repräsentation für das *Grabbable* die GPS-Koordinaten der Stadt Berlin benutzt (2).

4. Das mittels der *Grab*- Geste selektierte *Grabbable* kann genommen und in einer *Dropzone* „losgelassen" werden. Diese Integration triggert die Suche, die im Hintergrund vom Swoozy-Server verwaltet wird. Im abgebildeten Beispiel wird nach Bildern der Stadt Berlin gesucht (3).

5. Nach wenigen Sekunden erscheinen die Ergebnisse der semantischen Suche in Form einer Vorschauliste von typisierten Elementen (Videos, Bilder, Shoppingempfehlungen oder Kurzbeschreibungen) (4).

6. Durch eine *Push-Geste* oder einen *Long-Stay* (längeres Halten der Hand (1.5 Sekunden) über einem grafischen Element) kann ein Element der Vorschauliste ausgewählt werden. Die *Push-Geste* entspricht dabei der aus der Mausverwendung am Computer

benutzten „Klick"-Metapher, die *„Long-Stay"* (dt. *„längeres Draufhalten"*)-Interaktion der Mouse-Over-Metapher (siehe Abbildung 6.23).

7. Nachdem ein Element ausgewählt wurde, wird ein zusätzliches Fenster (Panel) eingeblendet. Dieses zeigt eine vergrößerte Version des selektierten Vorschauelements inklusive Zusatzinformationen (Bildbeschreibung, ähnliche Medien usw.…). Dieses Fenster ist wiederum interaktiv und beinhaltet Schaltelemente, die per *Push-Geste* oder *Long-Stay*-Interaktion ausgewählt werden können. Im Falle des *Shop-It*-Panels dienen die Schaltelemente dazu, einen Warenkorb oder eine Wunschliste aufzustellen, im Falle des *Share It*-Panels eine Nachricht und einen Post-Befehl zu triggern.

8. Möchte der Benutzer zurück zum Hauptvideo gelangen, kann er das reduzierte Livefernsehbild bzw. die Vorschau per Geste selektieren. Ab diesem Zeitpunkt ist das Hauptprogramm wieder im Großformat sichtbar und die Gesteninteraktion kann wiederholt werden.

Abbildung 6.25: Grab'n'Drop Interaktionsworkflow zur Bedienung der Benutzerschnittstelle von Swoozy

Eingabegeräte

Die gestengesteuerte Interaktion von Swoozy beruht auf drei komplett unterschiedlichen Eingabetechnologien, die unabhängig voneinander in der Lage sind, die komplette Kontrolle der Benutzerschnittstelle zu übernehmen.

Microsoft Kinect

Die erste Version der Microsoft Kinect dient als Eingabegerät für die Erfassung und Analyse der Zuschauergeste. Um die Swoozy-Benutzerschnittstelle zu benutzen, wird der komplette Körper des Benutzers mittels einer Tiefenbildkamera erfasst und analysiert. Je nach Position seines Armes und der Hand wird die Position mittels einer virtuellen Hand innerhalb der grafischen Schnittstelle markiert und dargestellt.

Hier werden nur die X- und Y-Positionen im 3D Raum benutzt, da die Tiefe (Z-Achse) bei Bewegungen aufgrund eventueller Okklusionen schlecht bestimmt werden kann, insbesondere, wenn die Hand vor dem Körper bewegt wird.

Abbildung 6.26: Gesteninteraktion mit der Kinect

Aus diesem Grund wurden die Interaktionsmöglichkeiten mit dem Swoozy-System, wie schon im letzten Abschnitt erwähnt, auf nur zwei Gesten beschränkt. Zum einen kann ein Element durch *Long-Stay* oder optional über eine *Push*-Geste ausgewählt werden. Zum anderen können durch die *Grab'n'Drop*-Geste nur *Grabbables* bewegt werden.
Um dies zu realisieren, wird kontinuierlich die Position der Hand des Benutzers bestimmt. Befindet sich diese über einem *Grabbable*, wird es direkt selektiert und „haftet" visuell an der Hand, bis es zu einer

Dropzone geführt wird. Dort löst sich automatisch das *Grabbable* von der virtuellen Hand, so dass der Benutzer wieder seine Hand in Ruheposition bringen kann. Diese Interaktionsform wurde ausgewählt, damit Ermüdungserscheinungen bei mehrfacher Wiederholung der gleichen Geste vermieden werden können. Die Benutzung der Kinect ist immer mit einer Kalibrierung verbunden. Hier muss der Benutzer gewährleisten, dass er jederzeit, insbesondere bei der *Grab'n'Drop*-Geste, im Sichtfeld der Kamera bleibt, damit seine Interaktionen richtig analysiert und in Aktionen umgewandelt werden können. Eine weitere Herausforderung der Gestenerkennung mittels der ersten Version der Kinect ist die genaue Bestimmung des Benutzers, der gerade interagiert. Wenn z.B. mehrere Zuschauer auf einem Sofa sitzen, bestimmt die Komponente anhand der aktivsten Handbewegungen in Richtung des Fernsehers, wer mit dem Swoozy-System interagieren darf.

Leap Motion

Die Leap Motion wird neben der Microsoft Kinect als Fingereingabegerät für die Bedienung der Swoozy-Benutzeroberfläche verwendet. Anstatt der virtuellen Hand wird, wenn die Leap Motion angeschlossen ist, ein virtueller Finger angezeigt. Dieser hat die gleiche Funktionalität wie die virtuelle Hand, jedoch mussten die Interaktionen, bedingt durch das relativ kleine Sichtfeld der Leap Motion, leicht adaptiert werden. Das Selektieren wird nicht mehr über die *Long-Stay*-Interaktion durchgeführt, sondern über eine *„Doppelte Finger"*-Selektion. Dafür muss der Benutzer nur zwei Finger über die Leap Motion bewegen. Auch die *Push*-Geste kann dadurch simuliert werden. Grund für diese Interaktionsauswahl ist, dass eine Zeigefinger-basierte Interaktion in Richtung des Fernsehers im 3D-Sichtfeld, u.a. wegen der Treffsi-

cherheit, schwer zu erfassen ist. Während einer Interaktion mit dem Zeigefinger nach vorne folgt die ermittelte Position des Fingers immer einer etwas gekrümmten Bewegungsbahn, was dazu führt, dass die Treffsicherheit eines grafischen Elementes stark reduziert und die Selektion erschwert wird.

Aus diesem Grund wurde, wie in Abbildung 6.27 dargestellt, die Interaktion auf einen (Bewegung des virtuellen Fingers) und zwei Finger der Hand (zur Selektion eines grafischen Elements der Benutzerschnittstelle) beschränkt. Vordefinierte Leap Motion-Fingergesten, wie *Circle, Swipe, Key Tap* und *ScreenTap* wurden mit Absicht innerhalb des Interaktionskonzepts nicht übernommen, da sie eine Lernphase benötigen und sehr empfindlich für Okklusionen sind, was nachträglich zu fehlerhaften Gesteninterpretationen führen kann.

Ein weiterer Vorteil der Benutzung der Leap Motion in Zusammenhang mit der Swoozy-Benutzerschnittstelle ist, dass der Benutzer nicht über seinen kompletten Körper „getrackt" wird, sondern nur über seine Hand. Somit ist es nicht nötig zu bestimmen, ob sich der Benutzer sitzend oder stehend vor seinem Fernseher aufhält, bevor die Gestenanalyse gestartet wird, daher wird keine neue Kalibrierung des Sensors fällig. Dank ihres kompakten Formfaktors kann die Leap Motion in unmittelbarer Nähe des Benutzers positioniert (z.B. auf der Armlehne eines Sofas) und leichter „transportiert" werden. Das bedeutet, wenn sich z.B. die Position und die Entfernung des Sofas zum Fernseher verändern, muss der Benutzer keine erneute Kalibrierung durchführen.

Abbildung 6.27: Fingerinteraktion über die Leap Motion zur Ansteuerung von Swoozy

Gyroskop-basierte Fernbedienung

Eine weitere Eingabemöglichkeit, die es ermöglicht das Swoozy-System zu bedienen, bieten die sog. Gyroskop-basierten Fernbedienungen. Besonderes Merkmal dieser Geräte ist die Möglichkeit, frei in der Luft Gesten durchführen zu können, ohne den virtuellen Referenzpunkt auf der Benutzerschnittstelle zu verlieren. Diese relativ neue Gerätekategorie ist seit der im Jahr 2006 erschienenen Wii Remote, insbesondere im Home Entertainment-Bereich, immer populärer geworden. Der Trend geht so weit, dass mittlerweile viele Fernsehhersteller ihre eigenen Modelle mit dieser Art von Fernbedienung ausstatten. Bereits 2012 integrierte LG die Gyroskop-basierte InvenSense[30]-Technologie in seine

[30] `http://www.invensense.com/`

Magic Remote. Seit 2014 bietet Samsung ebenfalls eine Gyroskop-basierte Fernbedienung für seine neue Smart-TV-Gerätereihe an.

Abbildung 6.28: Swoozy mit Gyroskop-basierter Fernbedienung

Dieser Trend zeigt, dass durch die Smart- und Connected-TV-Geräte und ihren neuen eingebauten Funktionalitäten, die klassische rechteckige Fernbedienung möglicherweise bald zur Vergangenheit gehören wird. Die Art und Weise, wie Schaltelemente am Bildschirm selektiert werden, tendiert immer mehr in Richtung Gestensteuerung und Benutzung der Maus-Cursor-Metapher. Die Bedienung mit Hilfe der Selektierung über das „Tastenkreuz“, wie es bei traditionellen Fernbedienungen vorgesehen war, gilt als veraltet. In Swoozy werden diese neuen Gyroskop-basierten Eingabegeräte unterstützt. Hierbei werden die gleichen Metaphern wie bei einer Interaktion mittels der Leap Motion benutzt. Die virtuell eingeblendete Hand dient als Cursor und die Auswahl einer Taste der Fernbedienung bedeutet, dass das gerade selektierte Element ausgelöst (*Grab*-Interaktion) wird.

Semantische Verarbeitung

Prinzip und Realisierung

Swoozy und die darin integrierten Module stützen sich auf eine semantische Repräsentation, um eine einheitliche Wissensrepräsentation innerhalb des Systems zu gewährleisten und die Kommunikation mit externen Webdiensten generischer zu gestalten. Diese Verarbeitung wird innerhalb von Swoozy durch mehrere Komponenten in Echtzeit absolviert. Die Komponenten lösen u.a. folgende Aufgaben:

- Die Analyse und Beschreibung von unstrukturierten Texten aus den ausgestrahlten OCR- und EPG-Texten.
- Die Semantifizierung dieser Texte und die Verknüpfung zu Entitäten/Konzepten und Ontologien bzw. Taxonomien.
- Die korrekte semantische Interpretation der Sucheingabe eines Webdienstes samt Konzept und die Rückgabe der Ergebnisse in Form einer semantisch bearbeitbaren Beschreibungsstruktur.

Während der Analyse der unstrukturierten Texte werden die Konzepte, Entitäten oder anderen erkannten textuellen Eigenschaften wie Stimmungen oder Emotionen in verschiedenen Formaten von den Diensten für das System bereitgestellt. Diese Dienste können auf externe Wissensquellen verweisen. Diese Verweise werden zum Zeitpunkt der Erkennung nicht aufgerufen, was dazu führt, dass nur Teilinformationen für die Erkennungskomponente vorhanden sind. Wenn z.B. ein Verweis auf Wikidata von einem Dienst zurückgeliefert wird, bedeutet dies längst nicht, dass die damit in Zusammenhang stehende Wikidata-Seite analysiert wurde. Ähnliches gilt bei Wikidata-Medien (Bilder oder Videos), die erst später aufgerufen werden. Dies hat zur Folge, dass

die semantischen Ergebnisse der verschiedenen Texte-Analysedienste nicht 1:1 miteinander vergleichbar sind, da sie sich bzgl. ihrer Formate von der internen Beschreibung unterscheiden.
Um dieses Problem zu lösen, werden zu dem Zeitpunkt, an dem die Ergebnisse der einzelnen Dienste innerhalb von Swoozy verarbeitet werden, die Strukturen vereinheitlicht und in ein generisches Format umgewandelt. Innerhalb dieses Formats dienen die Verweise als eindeutige Referenzen auf Taxonomien, Wikidata und die leichtgewichtige Swoozy-Ontologie. Dadurch wird eine direkte Referenz auf die Swoozy-internen Konzepte realisiert, sodass ein generischer und einheitlicher Rückverweis innerhalb des Systems jederzeit möglich ist. Konkret wird es so realisiert, indem bestimmte Konzepte von DBpedia oder Wikidata über ein vereinfachtes Mapping-Verfahren auf die Konzepte der internen leichtgewichtigen Swoozy-Ontologie durchgeführt wird. Weiterführende Informationen über die Herausforderung des Ontologie-Mappings und Matchings können in [Ber14b] oder [BTH04] nachgeschlagen werden. Die Abbildung 6.29 zeigt wie die Swoozy-Ontologie aufgebaut ist.

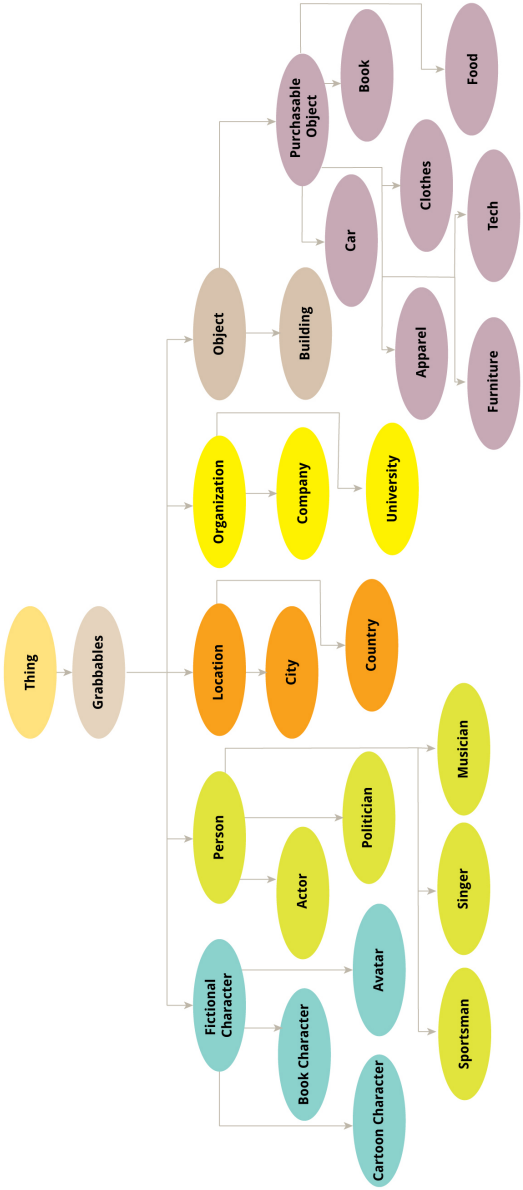


Abbildung 6.29: Auszug aus der Swoozy-Ontologie

Wenn ein „text-identisches“-Konzept gefunden wurde, wird dieses 1:1 gemappt. Falls kein passendes Konzept vorhanden ist, wird das Hauptkonzept benutzt oder das Konzept, das am nächsten in der Konzepthierarchie zu finden ist. In der Abbildung 6.30 wird beispielhaft verdeutlicht, wie zwei Konzepte, die vom Konzept *Person* abgeleitet sind, unterschiedlich gemappt werden.

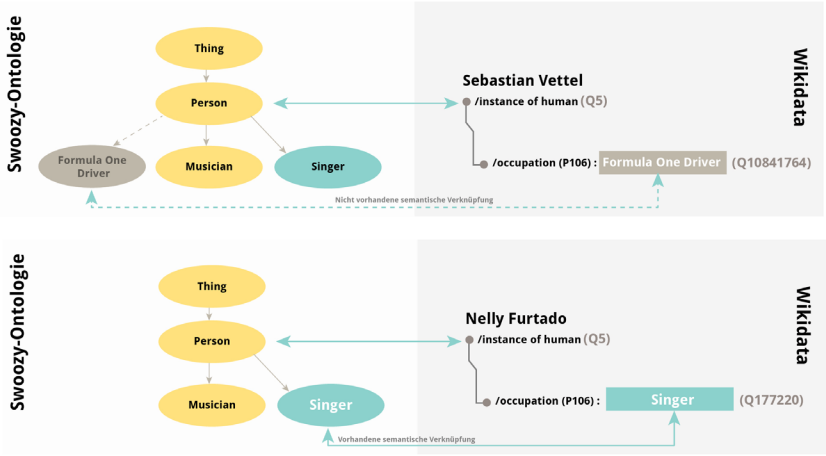


Abbildung 6.30: Semantisches Mapping für die Swoozy-Ontologie

Im Falle des Formel 1- Fahrers Sebastian Vettel wird die *occupation* anhand des Wikidata-Eintrags auf *Formula One Driver* festgelegt. Da kein passendes Konzept (*Formula One Driver*) in der Swoozy-Ontologie für diesen Beruf existiert, wird das nächstbeste oder das Oberkonzept (*Person*) genommen. Umgekehrt sieht es bei der Sängerin *Nelly Furtado* aus, die bei ihrem Wikidata-Eintrag als *occupation* die Eigenschaft *Singer* besitzt. Letzteres Konzept (*Singer*) ist innerhalb der Swoozy-Ontologie vorhanden und kann so 1:1 zugeordnet werden. Dieser vereinfachte Ansatz ermöglicht, dass intern alle Entitäten immer über

eine semantische Beschreibung verfügen und in Folge dessen somit immer ein mehr oder weniger präziser Rückverweis möglich ist.
Die Referenz und Zuordnung der Konzepte werden bei jedem Verarbeitungsschritt innerhalb des Systems mitgeliefert. Konkret wird auf diese Konzepte innerhalb jeder generischen Austauschstruktur verwiesen, so dass theoretisch die zugrundeliegende Ontologie und deren Konzepte ausgetauscht werden können. Dieser Ansatz der generischen semantischen Verlinkung ermöglicht die Bildung sinnvoller Relationen der intern benutzten Konzepte. Somit können optional die semantischen Ergebnisse von Websuchen wieder als Eingabe für eine weitere Such- und Webabfrage benutzt werden. Dieses Prinzip stellt einen der Kernaspekte des semantischen Fernsehens dar.

Dienste und semantische Suche des Web 3.0

Die eigentliche Suchmöglichkeit und die Anbindung zum Semantic Web werden von Swoozy über eine REST-basierte Schnittstelle kombiniert durchgeführt (siehe Abbildung 6.3). Bei diesem Schritt wird zuerst analysiert und dann entschieden, welche Dienste zu den jeweiligen Konzepten und Suchanfragen aufgerufen werden müssen.
Gleiches geschieht, wenn ein *Grabbable* generiert und damit eine Suche initiiert wird. Hier wird jedes Mal eine Referenz auf das Konzept mitgeliefert (z.B. Name + Konzept (PER)), damit die aufzurufenden Dienste, falls diese es unterstützen oder die jeweiligen Parameter anbieten, mit den richtigen Konzeptparametern versehen werden.
Bei Swoozy wird die Suche primär mittels des Dienstes Wikidata durchgeführt, da dieser über eine REST-basierte API anbindbar ist und die Informationsmenge aus Wikipedia strukturiert und mehrsprachig zur Verfügung stellt. Da nicht alle Dienste einheitliche Ergebnisstrukturen

bereitstellen, müssen diese homogenisiert werden. Wenn sich eine API eines dieser Dienste verändert, müssen die Parameter und Ergebnisattribute neu spezifiziert und angepasst werden. Dieser Fall kann bei zusammenhängenden Medien oder Bildern aus unterschiedlichen Diensten eintreten. Nichtdestotrotz, falls bestimmte Medien nicht direkt über Wikidata verfügbar sind, können sie mittels spezieller Aufrufe über Wikimedia gefunden werden. Durch die generische Beschreibung der Dienste innerhalb der Suchkomponente können weitere Webdienste wie z.B. die von Microsoft Bing[31] oder Yahoo Flickr[32] aufgerufen werden, um fehlende Informationen zu ergänzen. Der kombinierte Aufruf dieser Dienste ermöglicht die Generierung einer angereicherten Ergebnisstruktur, die als einheitliches Format zurückgeliefert wird und eine reichhaltigere Basisstruktur für die Visualisierung bietet.
Als Beispiel nehmen wir die Suche zu einer Person. Im ersten Schritt bekommt die REST-Schnittstelle (Punkt 12 von Abbildung 6.3) nur die Information, dass eine Suche mit der Konzeptkombination *Person* + *Name des Fahrers* durchgeführt werden muss. Diese Suche wurde nach einem Drop eines *Grabbables* vom Benutzer initiiert. Im nächsten Schritt werden zuerst faktische Informationen über die Person gesucht. Dies wird über den Aufruf der Dienste DBpedia und Wikidata innerhalb des Swoozy-Servers realisiert, wobei Wikidata als primärer Dienst benutzt wird. Wenn die wichtigsten biografischen Informationen (Geburtsdatum, Geburtsort, Hintergrundinformationen über die Karriere) von Wikidata gefunden und extrahiert worden sind, wird diese Information mit Zusatzmedien ergänzt, die entweder von DBpedia oder Wikidata selbst stammen oder z.B. von Diensten wie Bing, Flickr und YouTube geliefert werden. Zusätzlich können Dienste wie Amazon

[31] `https://www.bing.com`
[32] `https://www.flickr.com`

oder Twitter eingebunden werden, um käufliche Artikel oder Kommentare über die gesuchte Person zu finden und mit der bereitgestellten Biographie zu verknüpfen. Dieser Vorgang der Zusammenstellung einer einheitlichen Ausgabestruktur mittels Quellen von verschiedenen Webdiensten kann als *Mashup* (siehe [Car08] [BDS08] [LHSL07] [KTP09] [NHK10]) bezeichnet werden. Innerhalb des Swoozy-Servers wurden für die folgenden Suchanfragen folgende Dienste über ihre dedizierten REST-APIs verwendet:

- Faktsuche: Wikidata, DBpedia und Wikipedia. Als visuelle Anreicherung der Faktendarstellung für geografische Orte wurde Google Maps benutzt. Für passende Bilder wurde Wikimedia angefragt.
- Bildersuche: Wikidata, Flickr, Bing und Wikimedia. Für die erweiterte Beschreibung eines Bildes wurden Textabschnitte aus Wikipedia benutzt.
- Videosuche: YouTube, Vimeo, teilweise Mediatheken von Fernsehsendern bzw. Kinotrailer-Providern.
- Shopping-Suche: Amazon.com API.
- *Share It!*-Funktion: Anbindung zu Twitter-Feeds über die offizielle Twitter-API.

Als Ergebnis dieser kombinierten Suche wird eine generische Antwortstruktur im SwoozyML-Format erzeugt, die über die REST-Schnittstelle des Swoozy-Servers zum Client weitergereicht und von dort über die Visualisierungskomponente dargestellt wird.

Visualisierung der Ergebnisse und Multimediainhalte

Unter Visualisierung der Ergebnisse verstehen wir die grafische Darstellung der unterschiedlichen Ergebnistypen, die sowohl von heterogenen Diensten stammen als auch von heterogenen Mediatypen (Video, Bilder und Texte). Diese Ergebnisse müssen in einem weiteren Schritt dargestellt werden, ohne dabei das Benutzererlebnis zu beeinträchtigen.

Die Art der Visualisierung und deren Auswahl hängen sehr stark von den vorher genannten Design- und Interaktionsregeln ab. Die Listendarstellung der Ergebnisse erwies sich als guter Kompromiss zwischen einer kompakten visuell orientierten Darstellung (der Benutzer sieht sofort die Ergebnisse dank der Vorschau) und einer Möglichkeit, die Auswahlinteraktion schnell durchzuführen (die Interaktion ist einfach und erfordert keine komplexeren Gesten, wie dies bei einer Liste mit Bildlaufleiste (engl. Scrollable list) der Fall wäre). Möchte der Benutzer weitere Ergebnisse sehen, wird ihm dieses mittels eines 3D-Blättereffekts signalisiert.

Durch eine *Push*-Geste kann er weitere Ergebnisse einblenden lassen, ohne das TV-Bild grafisch zu überfrachten.

Jede Suche, die von einem Benutzer per Geste initiiert wurde, wird zuerst durch den Swoozy-Server weitergereicht. Nach der Analyse des Anfragetyps werden die entsprechenden Dienste, passend zum Konzept und zur Art der Suche, aufgerufen. Um die Verbindung von Ergebnissen mit der Listendarstellung zu ermöglichen, wurde ein spezielles Modul entwickelt, dass die Ergebnisstrukturen des Swoozy-Servers in für die GUI parsbare Formate und die semantischen Strukturen, umwandelt. Diese Struktur wird als SwoozyML-Format ausgegeben und wird später in diesem Kapitel vorgestellt. Da diese Referenz sowohl für

eine weitere Suche benötigt als auch für die Generierung benutzt wird, können jederzeit gemäß des *„No presentation without representation“*-Paradigmas, Rückschlüsse über den Status der Benutzerschnittstelle gezogen werden. Kontextbezogene Begriffe können dadurch automatisch eingeblendet werden. Aus diesem Grund kann die ausgewählte Lösung als „semantische Visualisierung“ der Ergebnisse bezeichnet werden. Die Visualisierung innerhalb von Swoozy wurde zunächst mit der Adobe Flash-Technologie (2013er- und 2014er-Demonstratoren) realisiert. Ab der 2015er-Version wurde für die Programmierung der Interaktions- und Anzeigekomponenten HTML5 verwendet. Diese Entscheidung lässt sich dadurch erklären, dass die Adobe Flash Technologie von den Fernsehherstellern nicht mehr unterstützt wird. Alle Smart-TV-Hersteller (die auch zur Smart-TV Alliance gehören – siehe Kapitel 4), empfehlen die Benutzung von HTML5 für die Implementierung von TV-Apps. Im folgenden Abschnitt werden detailliert die verschiedenen Suchformen und deren Ergebnisvisualisierungen präsentiert.

Faktensuche

Die Faktensuche ist eine spezifische Suche, die mehrere Ausgabemöglichkeiten und Visualisierungsvarianten anbietet. Im Falle einer Suche nach einem Gebäude erwartet der Benutzer andere Eckdaten als bei einer Recherche über einen Schauspieler. Aus diesem Grund werden die Ergebnisse anders dargestellt und dementsprechend sieht der Inhalt des ausgewählten Ergebnispanels (rechts innerhalb der Swoozy-Benutzerschnittstelle) unterschiedlich aus.

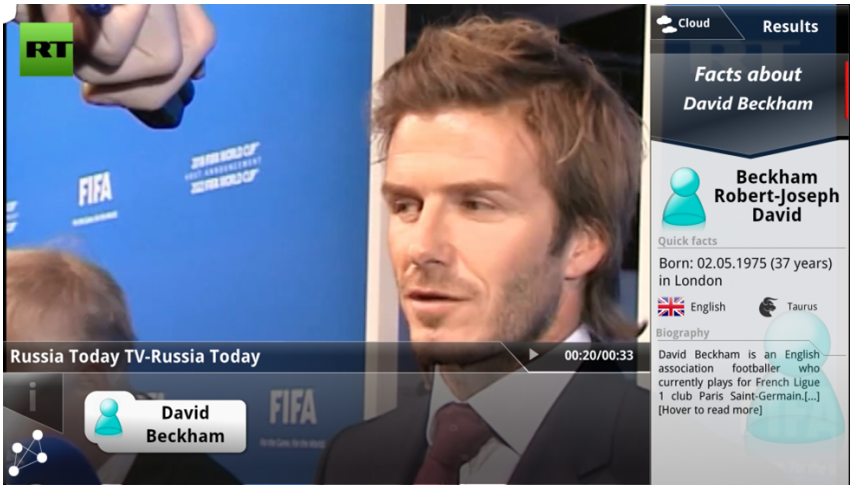


Abbildung 6.31: Darstellung von Fakten über David Beckham

Im Fall einer Suche nach einem Schauspieler oder einer Person werden neben dem Namen und Vornamen Angaben zum Geburtsort und eine zusammengefasste Vita dargestellt (siehe Abbildung 6.31). Zusätzlich werden das Sternzeichen und ein ausgewähltes Portraitbild, inklusive Kurzbeschreibung der Person, eingeblendet. Wird im Namen des Geburtsortes ein auflösbarer Ortsname erkannt, wird dieser als weiteres *Grabbable* angeboten und erscheint in der unteren Programmleiste der Benutzerschnittstelle.

Dieses *Suggested Grabbable* kann vom Benutzer selektiert werden, um mehr über den Geburtsort der gerade ausgewählten Person zu erfahren. Dieses *Suggested Grabbable* wird erst nach der Anzeige der Ergebnisse erzeugt. Abbildung 6.32 zeigt die Generierung des *Suggested Grabbables* „Victoria“, welches parallel zur Anzeige der Biografie der Sängerin Nelly Furtado auch den Namen ihres Geburtsorts (Victoria in Kanada) darstellt. Die *Suggested Grabbables* können in einem weiteren Schritt als Eingabe für eine neue Suche benutzt werden.

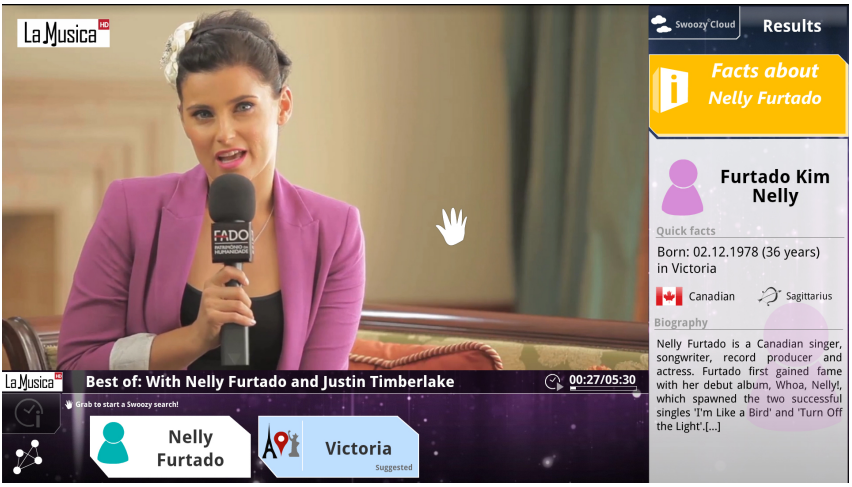


Abbildung 6.32: *Suggested Grabbable* bei der Sängerin Nelly Furtado

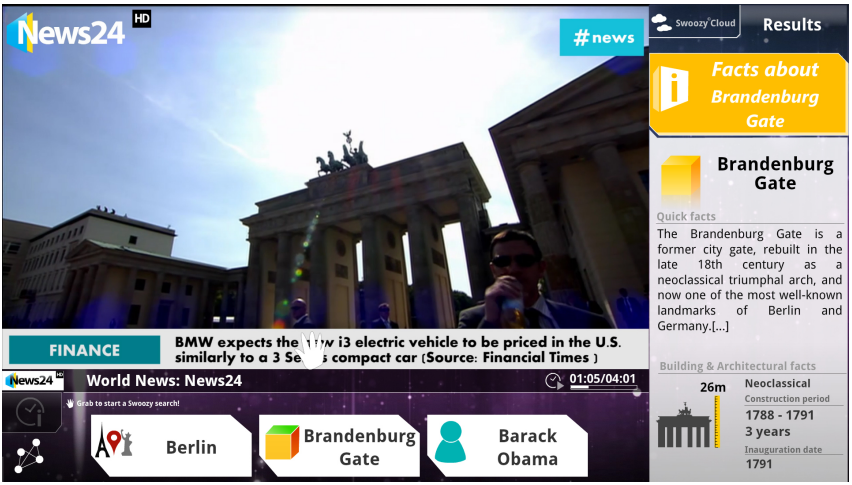


Abbildung 6.33: Beispiel der Visualisierung von Fakten bei einer Suche über das Brandenburger Tor in Berlin

Bei Objekten (Gebäude, Auto oder Kaufgegenstand) erfolgt eine dreischichtige Aufbereitung der Ergebnisse. Als Hauptkonzept wird die

semantische visuelle Repräsentation des Konzepts *Object* benutzt. Konkret werden eine kurze Beschreibung und eine Referenz auf die Webseite der Marke generiert.

Grafisch wird dieses Konzept mittels eines gelben Würfels innerhalb des *Grabbables* dargestellt. Das erzeugte *Object* in Form eines *Grabbables* kann aber auch untertypisiert werden und z.B. als Gebäude markiert werden.

Abbildung 6.33 zeigt die grafische Darstellung dieser Untertypisierung. Hier werden neben dem Namen des Gebäudes auch Daten zu seiner Konstruktion wie die Bauzeit, Höhe und Breite sowie andere technische Daten angezeigt. Ähnlich sieht es für Brücken (z.B. Golden Gate Bridge in San Francisco) oder Denkmäler (z.B. das Capitol in Washington) aus.

Abbildung 6.34: Eckdaten zur Golden Gate Bridge nach einer Faktensuche

Für Ergebnisse des Typs „geografische Orte“ werden die Bevölkerungsdichte und die Fläche mit einem festen Schemata angezeigt. Zusätz-

lich beinhaltet das Ergebnispanel auf der rechten Seite der Swoozy-Benutzeroberfläche eine interaktive Google Maps-Karte mit der exakten geografischen Position des Ortes und dem Wappen der Stadt in Form einer Grafik.

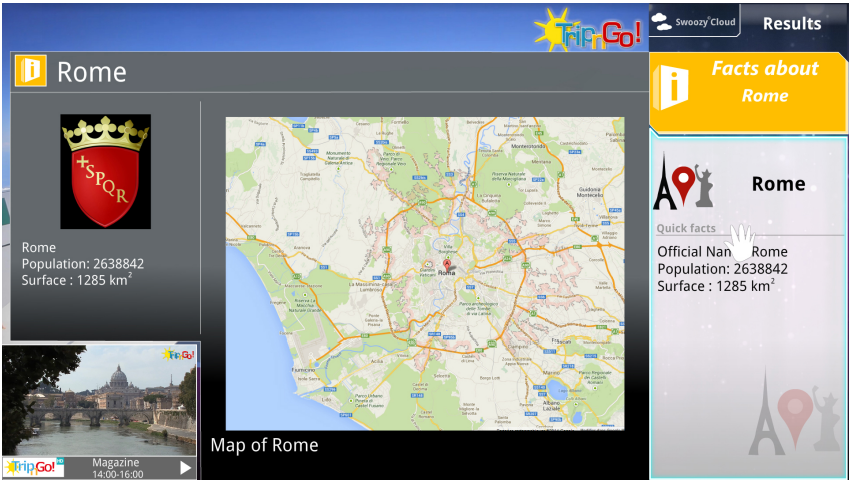


Abbildung 6.35: Visualisierung der Ergebnisse einer Stadtsuche: hier Rom

Bei fiktiven Charakteren (z.B. James Bond oder Big Buck Bunny) werden sog. *Quick Facts* (ähnlich der Wikipedia-Infobox) in Form eines beschreibenden Kurztextes angezeigt.

Die unterschiedlichen Ausprägungen der Ausgabenvisualisierung werden dadurch gewährleistet, dass, je nach Konzept, die Kurzbeschreibung und die dazu passenden Grafiken adaptiv und kontextbezogen, entweder in kombinierter Form innerhalb der *Grabbables* (Konzeptname + Icon) oder in kondensierter Form innerhalb der Ergebnisansicht (siehe Abbildung 6.22), angezeigt werden.

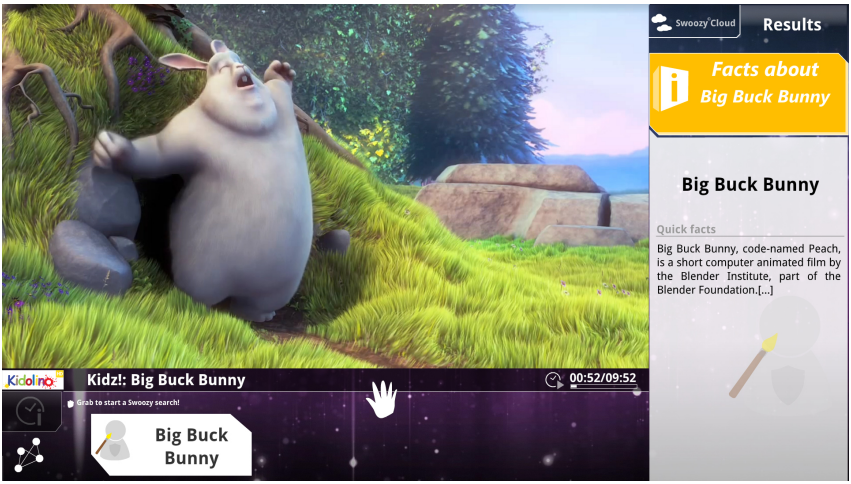


Abbildung 6.36: Eckdaten zum fiktiven Charakter Big Buck Bunny

Bildersuche

Nachdem die Ergebnisse aus den Diensten Wikidata, Wikipedia, Flickr und Bing aggregiert und in einer einheitlichen Struktur zusammengefügt wurden, werden dem Zuschauer zunächst die sieben ersten gefundenen Vorschaubilder (engl. *Thumbnails*) als Ergebnisse (JSON oder SwoozyML-Struktur) zurückgeliefert und von der entsprechenden Anzeigetechnologie (HTML5 oder Adobe Flash) dargestellt. Die Anzahl der angezeigten zurückgelieferten Ergebnisse kann individuell eingestellt werden. Die Größe der dargestellten Ergebnisse wurde so festgelegt, dass die Leitlinien des *10 foot-Designs* eingehalten wurden. Jedes Ergebnis ist mit einer eindeutigen *id* versehen, die eine Referenz auf das gefundene Medium ist. Wie in Abbildung 6.37 zu sehen, nehmen die Ergebnisse der Bildersuche die Form einer auswählbaren Liste innerhalb der Swoozy-Benutzeroberfläche an. Innerhalb des Systems wurden mit Absicht die Ergebnisstrukturen sehr generisch

gehalten, sodass zukünftig weitere semantische Operationen mit den Ergebnissen möglich sein könnten wie z.B. eine Bildähnlichkeitssuche oder eine Personalisierung der zurückgelieferten Medienobjekte.

Abbildung 6.37: Visualisierung der Ergebnisse der Bildersuche

Videosuche

Genau wie bei der Bildersuche liefert die Videosuche eine strukturierte Liste von Ergebnissen. Die Vorschaubilder werden in Form eines kurzen Preview-Videos (Video-Thumbnails) an der rechten Seite der Benutzeroberfläche dargestellt, die parallel zum Livefernsehbild (unten links) abgespielt werden. Wird eines dieser Video-Thumbnails selektiert, erscheint das Originalvideo innerhalb eines Videoplayers samt Schaltelementen (Abspiel-, Pause-, Vorlauf- und Rücklaufknopf). Ab diesem Zeitpunkt wird das komplette Video von den jeweiligen, im Swoozy-Server über APIs integrierten Onlinevideo-Anbietern (YouTube und Vimeo) gestreamt, ohne jedoch das Abspielen der Video-Thumbnails oder des Hauptfernsehbildes zu beinträchtigen.

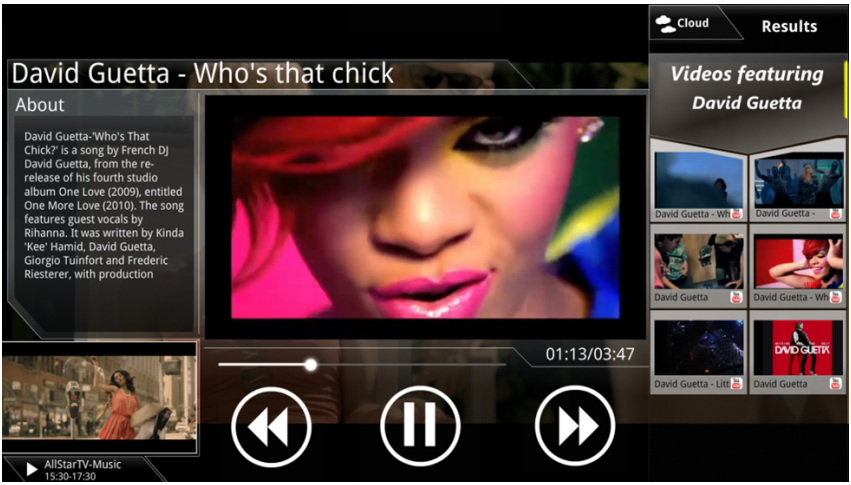


Abbildung 6.38: Visualisierung der Ergebnisse der Videosuche

Shopping-Suche

Bei der Shopping-Suche werden nach dem Drop eines *Grabbables* die im System verknüpften Einkaufsdienste des Online-Anbieters Amazon aufgerufen. Im Unterschied zu den Ergebnissen der Bild- und Videosuche steht nicht die semantische Weiterverarbeitung im Fokus, sondern die Verbindung von einem Produkt (oder *Object*) zum passenden kommerziellen Angebot. Daher wird nicht nur jedem Element eine eindeutige *id* zugeordnet, sondern auch zusätzliche Angaben wie die *ISBN*, die *ASIN* (Amazon Standard Identification Number) oder die UPC-Nummer (Universal Product Code), damit während des Einkaufsvorgangs („*Shop-It!*"-Vorgangs) die Produkte richtig referenziert werden können. Abbildung 6.39 zeigt die Ausprägung einer visuellen Ausgabe für ein Produkt, das bei Amazon.com verfügbar ist. Durch den generischen Aufbau der Ausgabestruktur lassen sich, falls von der Amazon-API Werte zurückgeliefert werden, andere Produkteigen-

schaften wie Verfügbarkeit oder Angaben zum Wiederverkäufer darstellen. Als mögliche Erweiterung könnten spezifische Angaben zum Produkt (Technische Abmessungen, SKU, Reseller und natürlich der Preis) mittels der API der Produktdatenbank Semantics3[33] aufgerufen und aggregiert dargestellt werden.

Abbildung 6.39: Visualisierung der Ergebnisse der Shopping-Suche (Amazon-Ergebnisse)

Im Second Screen

Als Alternative zur Interaktion am TV-Bildschirm wird dem Benutzer die Möglichkeit gegeben, über eine mobile Tablet-Applikation (Swoozy Mobile App für Android und iOS) Anfragen zu stellen und die Swoozy-Funktionalitäten an einem externen Bildschirm zu benutzen. Hierfür musste keine Anpassung seitens der Interaktion durchgeführt werden. Die Swoozy Mobile App ist eine 1:1-Abbildung der originalen

[33] `semantics3.com`

TV-basierten Version. Somit wird gewährleistet, dass das Benutzererlebnis nicht gestört wird und die Wiedererkennbarkeit erhalten bleibt. Durch diese Designentscheidung wird garantiert, dass der Zuschauer nicht zwei komplett getrennte und grafisch unterschiedliche Systeme (wie es bei der gleichzeitigen Verwendung einer App für TV und Second Screen der Fall ist) mit unterschiedlichen Interaktionsmetaphern verwenden muss.

Abbildung 6.40: Visualisierung der Ergebnisse im Second Screen Modus (Bildschirmauszug aus der Android-Version der Swoozy-App)

Da der Second Screen-Ansatz ein distribuierter Ansatz ist, muss, falls mehrere Benutzer verbunden sind, gewährleistet werden, dass die Ergebnisse auf den richtigen Tablets ankommen. Dies wird intern über die REST-Schnittstelle des Swoozy-Servers realisiert (siehe Abbildung 6.42). Jedes Tablet besitzt seinen eigenen dedizierten Zugang zu einer REST-Schnittstelle. Diese kann vollständig unabhängig und losgelöst vom aktuell auf dem TV laufenden Programm aufgerufen werden. Einziger Unterschied zur klassischen Funktionalität der Swoozy TV-

Benutzeroberfläche ist die Möglichkeit, über das Tablet Ergebnisse auf den größeren Swoozy-Client bzw. die Fernseher-basierte Visualisierungsoberfläche zu senden. Dies wird über eine sog. *Sling Geste* realisiert, die eine große Ähnlichkeit zur 3D-Frisbee-Interaktion hat, die in [BLS+14] im Zusammenhang mit einem Kiosksystem implementiert und angewandt wurde. Folgende Abbildung 6.41 zeigt den Interaktionsvorgang des *Sling-To-Beam*-Modus.

Abbildung 6.41: Bildschirmabzug des Second Screens im Beam-Modus und Prinzip der Synchronisation über WLAN

Folgende Abbildung bildet den Aufbau der Architektur und das Zusammenspiel zwischen der TV-Benutzerschnittstelle und den Second Screens ab. Die Kommunikation zwischen den Second Screen Apps und dem TV erfolgt über WLAN. Die Befehle zur Kontrolle der Swoozy TV-Benutzerschnittstelle werden von einer REST-Schnittstelle empfangen und intern als Kommando zur Ansteuerung der verschiedenen Visualisierungen und grafischen Elementen umgesetzt.

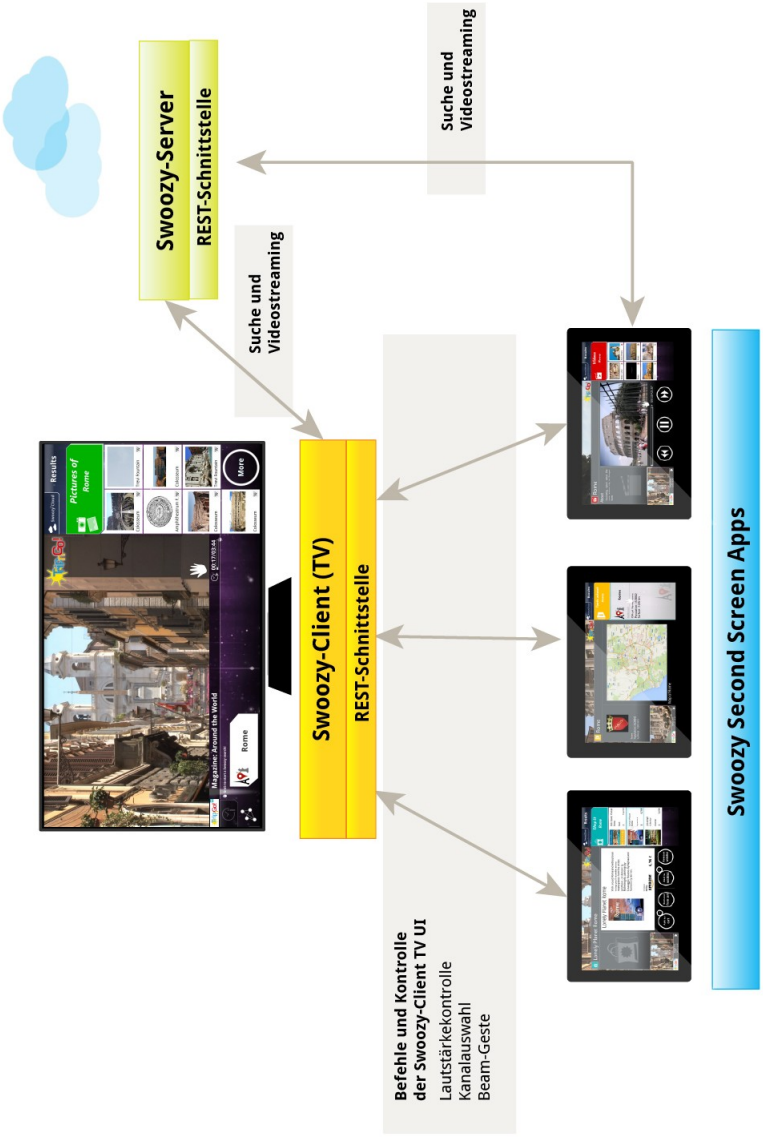


Abbildung 6.42: Aufbau der Second Screen-Architektur und Kommunikationskanälen

SwoozyML als einheitliches Ausgabeformat

Motivation und Prinzip

Die REST-basierte Client-Server-Architektur des Swoozy-Systems benötigt jederzeit einen Austausch von Daten und Zuständen seitens der Benutzerschnittstellen (Clients) und des Swoozy-Servers. Jedes durch die Anzeige der Ergebnisse angestoßene Update wird vom Swoozy-Server initiiert. Genauso sieht es bei Benutzerabfragen und Interaktionen aus, die in Form einer kompakten „Befehls"-Nachricht dem Server mitgeteilt werden. Um den bidirektionalen Nachrichtenaustausch zu vereinheitlichen und synchron ablaufen zu lassen, wurde bei der technischen Umsetzung des Swoozy-Framework ein generisches, sowohl für Maschinen als auch für Menschen lesbares Format mit Namen *SwoozyML* (Swoozy Markup Language) entwickelt.

SwoozyML bildet ein leichtgewichtiges Austauschformat für die Definition und Annotation der Videos samt der semantischen Zusatzinformationen. Zusätzlich dient es zur Ansteuerung der Ergebnisanzeige innerhalb der Swoozy-Clients (auf Fernseh- oder Second Screen-Ebene). Das Format basiert auf XML und wird von verschiedenen Komponenten (inklusive Werkzeuge wie Swoozy-Livana und SKRPTR, siehe Kapitel 7) innerhalb des Frameworks benutzt und verarbeitet.

Optional kann SwoozyML in JSON-Strukturen umgewandelt werden (wie dies der Fall in der 2016er Version von Swoozy ist) und von weiteren UI-Technologien wie z.B. HTML5 verarbeitet werden. Zusätzlich kann SwoozyML, ähnlich AcML (Action Markup Language) [SZE+14] und PreML (Presentation Markup Language) [BLS+14], als Triggerformat für das Auslösen von spezifischen Funktionalitäten seitens der Benutzerschnittstelle (Swoozy-Client) benutzt werden.

Die Austauschformate innerhalb des Swoozy-Systems sind von unter-

schiedlicher Natur und werden in drei Kategorien gruppiert:

- SwoozyML als **Präsentationsnachrichtenformat**

  In diese Kategorie gehören die vom Server initiierten oder zwischengespeicherten Ausgabestrukturen, die später von der Benutzerschnittstelle empfangen und interpretiert werden. Zusätzlich übernimmt die Ausgabestruktur folgende Aufgaben:

- Die Kontrolle der grafischen Benutzerschnittstelle
- Die Anzeige der Ergebnisse nach einer benutzerinitiierten Suche (siehe Beispiel der Videosuche nach Justin Timberlake in Abbildung 6.43 mit den Schritten 1,2 und 3)
- Die zeitgesteuerte Anzeige der *Grabbables* und EPG-Informationen über den Swoozy-Server und seine REST-Schnittstelle
- Die Kontrolle der Echtzeiteinblendung von *Grabbables*, getriggert vom Swoozy-Server

Listing 6.5: Ausschnitt aus einer SwoozyML-basierten Präsentationsnachricht bei der Ergebnisgenerierung einer Videosuche nach der Stadt Rom

```
<swoozyML target="videoSearch">
  <searchResult type="video">
    <videoList amount="6">
      <video id="0">
        <colorSpace><![CDATA[colorfull]]></colorSpace>
        <contentType><![CDATA[Video]]></contentType>
        <content><![CDATA[Rome]]></content>
        <source><![CDATA[YouTube]]></source>
        <abstract><![CDATA[Rome amtlich Roma Capitale, ist die
            Hauptstadt Italiens. Mit etwa 2,8 Millionen Einwohnern
             im Stadtgebiet bzw. rund 3,3 Millionen Einwohnern in
```

```
            der Agglomeration ist sie die groesste Stadt Italiens.
             Rom liegt in der Region Latium an den Ufern des
            Flusses Tiber]]></abstract>
        <description><![CDATA[Video featuring Rome and the
            monuments of the cities]]></description>
        <width><![CDATA[720]]></width><height><![CDATA[1280]]></
            height>
        <title><![CDATA[Rome: a visit]]></title>
        <thumbnail><![CDATA[http://localhost:8085/media/OpenDomain
            /cities/Rome/videos/rome_2.f4v]]></thumbnail>
        <filename><![CDATA[rome_2.f4v]]></filename>
        <infoSource><![CDATA[YouTube]]></infoSource>
        <mediaUrl><![CDATA[http://localhost:8085/media/OpenDomain/
            cities/Rome/videos/rome_2.f4v]]></mediaUrl>
        <copyright><![CDATA[YouTube]]></copyright>
      </video>[...]
    </videoList>
  </searchResult>
</swoozyML>
```

- SwoozyML als **Aktionsnachrichtenformat**

  Der Swoozy-Server erhält „Befehle“ oder Aktionen von der Benutzerschnittstelle (Fernseher oder Second Screen-System) und muss diese interpretieren bzw. zum richtigen Modul weiterleiten. In diesem Falle besitzt SwoozyML folgende Aufgaben:

- Die Kontrolle der TV-basierten Swoozy-Benutzerschnittstelle durch die Second-Screen Apps
- Das Auslösen von nachträglichen Abfragen bereits annotierter Medien
- Das Auslösen von Interaktionsnachrichten nach Benutzerabfragen (semantische Suche) zum Swoozy- Server

- Starten der semantischen Suche

Listing 6.6: Ausschnitt aus einer SwoozyML-basierten Aktionsnachricht für die Kontrolle der Hauptanzeige der Videos am Fernseher

```
<swoozyML>
    <mobile>
        <mediaControl>
            <mainVideo>playing|paused</mainVideo>
            <videoBox>playing|paused</videoBox>
            <beamedVideo>playing|paused|closed</beamedVideo>
        </mediaControl>
    </mobile>
</swoozyML>
```

- SwoozyML als **(Re)Präsentationsformat für Videos und Filme**

  Hierbei handelt es sich um einen hybriden Ansatz bei dem SwoozyML als Containerformat sowohl für die Annotation von Videos und Filmen benutzt werden kann als auch für die finale Ansteuerung der grafischen Präsentation in Form von *Grabbables*. Müssen Videos oder Sendungen vorannotiert werden, werden die temporalen Annotationen in einer Annotationsdatenbank dauerhaft gespeichert. Diese können zu einem späteren Zeitpunkt abgerufen werden und liefern eine komplette Annotation der jeweiligen Sendung in Form von SwoozyML zurück (siehe Abbildung 6.44).

  Hintergrund dieser technischen Entscheidung war, dass in herkömmlichen Systemen, Beschreibungen und Annotationen von Videos über Metadaten gespeichert werden. Diese Formate bedürfen in den meisten Fällen sehr guter IT-Kenntnisse, falls eine redaktionelle Nachbearbeitung notwendig sein sollte (siehe Kapitel 3). Diese Situation führt oft dazu, dass die eigentlichen

Möglichkeiten dieser Formate aufgrund ihrer Komplexität weitgehend nicht benutzt werden.

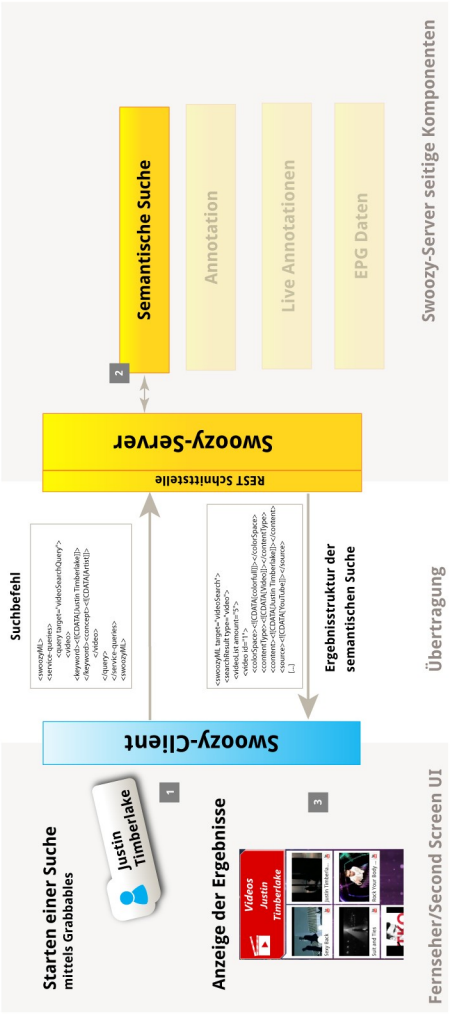


Abbildung 6.43: SwoozyML als Präsentations- und Austauschformat

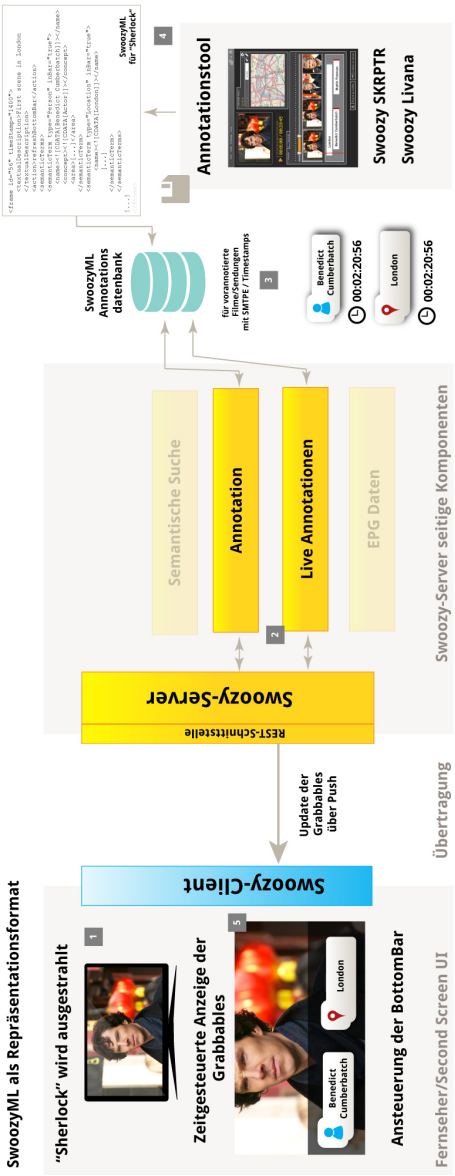


Abbildung 6.44: SwoozyML zur Beschreibung von Videos

Beschreibung der TV- Programme

Die grafische Ansteuerung und Erzeugung von *Grabbables* wird, wie im vorherigen Abschnitt bereits erwähnt, in hybrider Form über das SwoozyML realisiert. Unter hybrid verstehen wir, dass sowohl das Format für die eigentliche Ausgabe zur grafischen Benutzerschnittstelle benutzt werden kann, ähnlich PreML (siehe Kapitel 5), als auch für eine dauerhafte zeitbezogene semantische Beschreibung eines Videos. SwoozyML spielt die Rolle eines Container-Formats, indem eine zeitbezogene Beschreibung samt semantischen Typus jedes im Videoeinzelbild gefundenen Elements mit der jeweiligen grafischen Position fest gespeichert und zu einem späteren Zeitpunkt zur Swoozy-Benutzerschnittstellenkomponente gesendet werden kann. Folgende Struktur (siehe Quellcode 6.7) zeigt die in Swoozy ML enthaltenen Elementdefinitionen zur kompletten semantischen Beschreibung eines Videos. Diese Beschreibung bestimmt den Zeitpunkt der Anzeige der *Grabbables*. Die zu beschreibenden Attribute innerhalb von SwoozyML sind:

Listing 6.7: Beschreibung eines Videos mittels SwoozyML

```
<swoozyML>
<video>
    <name>The Football Mag</name>
  <title>Football mag</title>
    <uuid>9096684f-6321-4c3a-9617-5f9db202a612</uuid>
  <creationDate>26-11-2014</creationDate>
  <duration>01:25:00:00</duration>
  <type>Sports</type>
  <file>/server/archive_sp/videos/wc_2014_210_452.mpeg</file>
  <live>false</live>
  <cloudRating>5</cloudRating>
  <!-- Show in the bottom bar the EPG information -->
```

```
<description></description>
<frame uuid="e3fa45adf2eefe" timeStamp="1" timeCode="00:00:00:01"
    >
      <textualDescription>Start of the retrospective about the
          World Cup 2014. Transition to intro animation.</
          textualDescription>
      <action>refreshBottomBar</action>
</frame>
[...]
<frame uuid="e68b40c1eb9d" timeStamp="30251" timeCode
    ="00:20:10:01">
    <action>refreshBottomBar</action>
    <textualDescription>The football player Miroslav Klose is
        giving an interview after his goal during the football
        match Brasil against Germany in Belo Monte (Brasil)</
        textualDescription>
    <semanticTerms>
        <semanticTerm type="Person" inBar="true" uuid="3584e3ba-8
            f4d-4a9e-8904-00d0879876ed">
            <name>
                <![CDATA[Miroslav Klose]]>
            </name>
            <concept>
                <![CDATA[Football player]]>
            </concept>
            <linkdata type='Wikidata'>
                <reference>Q80471</reference>
                <content>http://www.wikidata.org/entity/Q80471</
                    content>
            </linkdata>
            <area type="square" uuid="71bdfa14-fc8a-4999-bedb-
                dcc270f932da">
            <x>729</x>
            <y>15</y>
            <width>100</width>
            <height>831</height>
```

```
            </area>
        </semanticTerm>
        [...]
    </semanticTerms>
</frame>
</video>
</swoozyML>
```

- *id* steht für eine eindeutige Bildnummer, die nur einmal pro Filmmaterial benutzt werden kann. Diese kann für eine spätere Indizierung benutzt werden.
- *timeStamp* definiert den Zeitpunkt (falls es sich um ein vorannotiertes Video handelt), zu dem das *Grabbable* generiert werden soll. Der *Timestamp* wird in Millisekunden angegeben. Für eine noch präzisere Zeitpunktdefinition kann der Timecode-Wert benutzt werden.
- *timeCode (optional)* beinhaltet den SMPTE (Society of Motion Picture and Television Engineers) EBU-Timecode. Dieses Timecode-Format wird im audiovisuellen Bereich benutzt (Fernsehen, Montage, Filmerstellung), um eine Synchronisation zwischen Video und Audiomaterial während der Produktion und Postproduktion zu gewährleisten. Der Timecode wird im Format HOURS:MINUTES:SECONDS:FRAMES angegeben[34] und dient dazu, ein bestimmtes Einzelbild (Videoframe) innerhalb eines Videos eindeutig zu referenzieren bzw. wiederauffindbar zu machen, z.B. für Videoschnitte.
- *textualDescription* (optional) ermöglicht den Redakteuren eine Beschreibung einer Szene festzulegen. Dieser Textblock kann

[34] `http://filmmakeriq.com/2008/11/understanding-time-code/`

entweder manuell als Notiztext benutzt oder automatisch mit Texten aus den Untertiteln oder der Audiodeskription gefüllt werden. Dieser Block wird dem Zuschauer (Swoozy-Benutzer) nicht angezeigt.

- Der *action*-Block dient dazu, ein Befehl zum Swoozy-Client zu versenden und kann vom Fernsehsender als neuer Synchronisationszeitpunkt für das Video und die parallel angesteuerte Anzeige der *Grabbables* benutzt werden. Dieser Block kann als technischer Ansteuerungsbefehl, wenn der Server einen automatischen Refresh der Oberfläche erzwingen möchte, dienen. Die verfügbaren Werte für die Befehle (*action*) sind:

  - *refreshBottombar*: alle *Grabbables*, die sich in der unteren Leiste (der sog. *BottomBar*) (siehe Abbildung 6.44) befinden, werden gelöscht. Dies kann vorkommen, wenn zu viele *Grabbables* in einer vorherigen Szene/Einzelbild erzeugt wurden oder wenn ein Kanal- oder Sendungswechsel stattfindet.
  - *hideUI*: die grafische Benutzerschnittstelle von Swoozy kann pro-aktiv vom Fernsehsender ausgeblendet werden.

  Per Definition hat der Benutzer keine Möglichkeit, die Auslösung der *action*-Befehle zu unterdrücken. Wenn sich währenddessen eine Benutzerinteraktion ereignet, hat die Interaktion eine höhere Priorität. Dies bedeutet, dass die Interaktion vom Benutzer ungestört bis zum Ende durchgeführt werden kann und erst, wenn er damit fertig ist, wird der Befehl im *action*-Block ausgeführt.

- *semanticTerms* beinhaltet die Liste der *Semantic Terms* bzw. *Grabbables*, welche die eigentlichen Annotationen und die semanti-

sche Beschreibung des jeweiligen Einzelbildes beinhaltet.

- *inBar=true|false* ermöglicht seitens des Fernsehsenders zu entscheiden, ob ein definiertes *Grabbable* eher im unteren Balken (BottomBar) oder nur im sichtbaren Videobereich erscheinen soll. Grund für diese Unterscheidung ist, dem Fernsehsender eine gewisse Flexibilität anzubieten und die visuelle Überlagerung von *Grabbables* innerhalb des unteren Balkens zu vermeiden. Ein Beispiel für diese Anwendung findet man bei Videos, in denen mehrere Personen vorkommen wie z.B. bei Gruppen (z.B. Fußballmannschaft). In diesem Fall wäre es aus Interaktionssicht ungünstig, alle Namen der Spieler als *Grabbables* im unteren Balken erscheinen zu lassen. Trotzdem besitzt der Zuschauer immer noch die Möglichkeit, im Videobereich eine Grab-Interaktion durchzuführen und dadurch ein *Grabbable* zu erzeugen.

- *area* bestimmt die grafische Position und die Fläche des jeweiligen Elementes im Bild, das mit dem angegebenen *Semantic Term* korrespondiert. Die Fläche wird mittels der Typen *circle/square/polygon* und der passenden Koordinatenlisten angegeben, ähnlich der HTML Map/Area Spezifikation[35].

  Falls zusätzliche 3D-Bildinformationen im Videomaterial vorhanden sind, können optional die Z-Positionen angegeben werden. Wobei die Werte 0 für den Vordergrund und 1 für den Hintergrund stehen. Da diese Z-Positionswerte zurzeit mittels herkömmlicher 2D-Bilderkennungsverfahren nicht automatisch extrahiert werden können, müssen diese Werte redaktionell und manuell eingetragen werden.

[35] `https://www.w3.org/wiki/HTML/Elements/map`

- *type*: bestimmt den Haupttyp des semantischen Konzepts. Dieser Parameter wird auch benutzt, um die Icons und das grafische Aussehen der *Grabbables* zu bestimmen.
- *concept*: Gibt die Klassifikation des *Grabbables* mit semantischem Bezug auf den Obertyp an.
- *linkData*: eindeutige Referenz auf das Konzept samt Entität. Hier bestimmt das Attribut *type*, welche semantische Wissensdatenbank für eine weitere Auflösung benutzt werden soll. Dieses Feld kann für interne Zwecke benutzt werden und kann auch, im Falle eines Produkts, die DOI, die ISBN oder eine direkte Referenz auf Wikidata bzw. eine externe Ontologie spezifizieren.

Semantifizierung von Videos

Im Rahmen der technischen Entwicklung von Swoozy spielt die Generierung von Annotationen (*Grabbables*) die zentrale Rolle. In den letzten Abschnitten wurde gezeigt, wie diese effizient und kontextabhängig aus Videos extrahiert und mittels einer benutzerzentrierten Darstellung grafisch visualisiert werden können. Im Falle einer automatischen Extraktion müssen Details beachtet werden, damit diese Annotationen sinnvoll, kontextbezogen und zuschauerfreundlich während Sendungen eingeblendet werden können. Die Herausforderung bei der vollautomatischen Wissensextraktion aus TV- Programmen ist, dass die bei der Extraktion gewonnene Informationsmenge nicht in einer Informationsüberflutung endet, die letzten Endes den Zuschauer überfordern wird.

Herausforderungen bei der automatischen Extraktion

Durch verschiedene Werkzeuge wurde gezeigt, dass eine Szene mittels automatischer oder semi-automatischer Verfahren gut beschrieben werden kann. Durch Gesichtserkennungsalgorithmen oder OCR-Verfahren lassen sich Personen und Gegenstände synchron zum Einzelbild erkennen. Als Ergebnis dieser Erkennung werden die erkannten Entitäten innerhalb der Swoozy-Benutzeroberfläche als *Grabbables* angeboten. Die Herausforderung bei diesem Ansatz ist, im Falle mehrerer gleichzeitig möglichen *Grabbables*, die Priorisierung dieser zu bestimmen, damit diese anschließend korrekt und zum richtigen Zeitpunkt dargestellt werden. Bei der automatischen Erkennung besteht die Gefahr, dass zu viele Elemente innerhalb eines Videoeinzelbildes registriert werden, was dazu führen kann, dass sehr viele *Grabbables* schnell generiert und eingeblendet werden, ohne dass der Benutzer tatsächlich diese Informationen zeitlich wahrnehmen kann. Danach muss die Relevanz der Elemente ausgewertet werden.

Um all diese Probleme zu lösen, wurde innerhalb der Swoozy-Videoanalysekomponente über eine Option in Form einer Konfigurationsdatei gewährleistet, dass die Anzahl der zu erkennenden Personen und Objekte, aber auch die zu analysierenden OCR-Bereiche, begrenzt werden können. Hier taucht das Problem der Präzision und Granularität der Annotationen auf und ob diese Annotationen noch editiert werden können, bevor sie dem Zuschauer über die Swoozy-Benutzerschnittstellen (Swoozy-Client) angeboten werden.

Präzision, Granularität und Anzeigezeitpunkt der Annotation

In manchen Fällen spielt die Granularität einer Annotation eine wichtige Rolle. Unter Granularität verstehen wir den Grad der Beschreibungsgenauigkeit, mit der eine Szene, ein Film oder eine Sendung annotiert werden soll. Diese Entscheidung hat eine direkte Auswirkung auf die Auswahl der Analysekomponente, die für die Analyse des Videos benutzt werden müssen. Die Granularität wird vom Inhalt des Videos bestimmt. Bei einer Naturreportage tauchen weniger Annotationen auf als bei einer Nachrichtensendung, bei der sich die Themen innerhalb eines kurzen Zeitrahmens verändern können.
Ein Redakteur kann, dank der Werkzeuge Swoozy-Livana und SKRPTR, die Granularitätsstufe bestimmen, indem er zusätzliche *Grabbables* einfügt. Einerseits kann er dadurch bestimmen, welche Ergebnisse aus der fernseheigenen Mediathek als Zusatzinformation geliefert werden und andererseits, durch bewusstes Vorenthalten von zu vielen Hintergrundinformationen, den Spannungsbogen aufrechterhalten.
Innerhalb des Swoozy-Annotationsvorgangs sind drei Stufen der Granularität für die Anzeige der *Grabbables* vorgesehen. Die Granularitätsstufen, niedrig, mittel und hoch, werden wie folgt definiert:

- Bei „niedrig“ wird nur ein *Grabbable* angezeigt und zwar vom Hauptdarsteller oder vom ersten erkannten Gegenstand innerhalb des Videos. Markennamen spielen dabei keine Rolle.
- Bei der Stufe „mittel“ werden sekundäre Objekte oder Personen miteinbezogen. Markennamen können als *Grabbable* eingeblendet werden und fungieren als Werbung oder Produktplatzierung.
- Bei einer hohen Granularität werden alle bedeutsamsten und

erkannten Elemente des Videos als *Grabbable* aufgenommen. Diese Stufe ist die höchste und praktisch nur bei Filmen oder aufgezeichneten Reportagen anwendbar. Im Livefernsehen bedürfte diese Stufe einer manuellen und redaktionellen Kontrolle der anzuzeigenden Grabbables, z.B. durch Swoozy-Livana.

Eine weitere Herausforderung stellt das Problem des Einblendezeitpunktes und der Anzeigedauer der *Grabbables* dar, d.h. wann diese innerhalb des Swoozy UI bzw. des Swoozy-Clients angezeigt werden sollen. Obwohl diese Aufgabe auf den ersten Blick trivial erscheinen mag (*„Das Grabbable soll doch erscheinen, wenn die Information vorhanden ist!"*) existieren Handlungsempfehlungen, die beim semantischen Fernsehen beachtet werden sollen.

Ein *Grabbable* kann zeitkodiert sein und zur ausgewählten Zeit in der Taskleiste erscheinen. Es liegt keine Pause zwischen der Erkennung und der Generierung des *Grabbables* vor. Das bedeutet, dass ein *Grabbable* genauso schnell, wie das Videoelement (Person, Objekt, Gebäude), erscheint und verschwindet. Seine Anzeige erfolgt so ohne Verzögerung. Dies mag bei Videos mit sehr langsamer Handlung und Kamerabewegung ganz gut passen, ist für handlungsreiche Szenen wie z.B. Trailer oder Aktionsszenen als Anzeigemodus ungeeignet. Das Überangebot an Informationen wäre für den Zuschauer zeitlich und kognitiv nicht handhabbar. Zusätzlich würde es bedeuten, dass die *Grabbables* zeitgenau vom Benutzer gegriffen („gegrabbt") werden müssten. Dies ist aus Interaktionssicht dem Benutzer nicht zumutbar. Erstens aufgrund der Schnelligkeit der Anzeige und zweitens wegen der kognitiven Belastung während der visuellen, parallelen Erfassung des Fernsehbildes und der *Grabbables*. Die zweite Variante der zeitkodierten Anzeige, die mittels SwoozyML definiert wurde, ermöglicht

eine längere Einblendung der *Grabbables*, damit diese vom Benutzer erfasst werden können.

Beim zeitversetzten Verfahren können zwei Anzeigemethoden angewandt werden. Die erste wird vom Fernsehsender selbst kontrolliert und redaktionell unterstützt, indem er über die dedizierte REST-Schnittstelle des Swoozy-Clients und mittels SwoozyML die Anweisung der Benutzerschnittstelle übermittelt, wann und wie lange ein *Grabbable* tatsächlich eingeblendet werden soll. Die Dauer, wie lange ein *Grabbable* eingeblendet wird, kann entweder redaktionell (z.B. das Hauptthema der Sendung), kommerziell (z.B. eine Automarke soll zu Werbungszwecken länger eingeblendet werden) oder gestalterisch (z.B. bei Serien oder Filmen, bei denen die Handlung spannend bleiben soll) bedingt sein. Bei der zweiten Anzeigevariante findet der Ausblendezeitpunkt später statt. Wenn ein neues *Grabbable* erscheint, wird es zur Liste der aktuellen hinzugefügt, ohne diese Liste komplett zu löschen. Der Benutzer hat so die Möglichkeit, die *Grabbables* bewusst wahrzunehmen und diese für eine Suche per Geste auszuwählen.

Künstlerische Aspekte

Im letzten Abschnitt wurden die unterschiedlichsten Möglichkeiten und Zeitpunkte der Einblendung der *Grabbables* erwähnt. Unter einem kommerziellen und redaktionellen Punkt betrachtet, ist der Zeitpunkt des Erscheinens sehr Einzelbild-nah, d.h. die Annotationen erscheinen zeitgenau zu bestimmten eingeblendeten Gegenständen oder Personen, die innerhalb des Videos angezeigt werden. Bei sehr vielen Sendungstypen wie z.B. Live-Sportsendungen oder Nachrichten macht es wenig Sinn, die Anzeige und Generierung der Annotationen bzw. *Grabbables* zu verhindern, zu unterdrücken oder sie zu einem späte-

ren Zeitpunkt einzublenden. Diese Situation ist bei Filmen nicht so einfach. Hier droht immer der sog. „Spoiler-Effekt", d.h. die frühzeitige Enthüllung eines Handlungselements, die sowohl die Spannung als auch das Interesse seitens des Zuschauers für die Handlung raubt. Folgende Beispiele verdeutlichen das Problem:

- Wenn der Zuschauer sich den ersten Teil des Films *Herr der Ringe* anschauen würde und gleichzeitig eine *Grabbables*-gestützte Suche über den sich mutmaßlich zu Tode stürzenden Magier namens „Gandalf" initiieren würde, käme als ein Ergebnis dieser Suche heraus, dass er im zweiten Teil wieder erscheinen wird. Hat der Zuschauer den zweiten Teil noch nicht gesehen, würde Swoozy ihm schon zu viel verraten und somit einen interessanten Aspekt der Filmhandlung vorab preisgeben (engl. *Spoiler*).
- Während eines Krimis kann die Suche nach dem Namen eines Schauspielers indirekt die Identität des Täters, der gerade ausgestrahlten Folge, verraten (textuelle Swoozy Fakt-Suche Ausgabe: „Schauspieler X, spielte im Krimi Y die Rolle des Mörders. . . ").
- Bei Wiederholungen von bekannten Filmen ist die Sinnhaftigkeit der Einblendung mancher Begriffe fraglich. In James Bond-Filmen können prominente Laptop- oder Handymarken erscheinen. Es stellt sich die Frage, ob es nach der x-fachen Wiederholung des Filmes immer noch kommerziell sinnvoll ist, *Grabbable* vorzuschlagen, da der „Werbeeffekt" längst vorbei ist. Ähnliches gilt bei älteren Filmen oder Reportagen deren Inhalte nur noch einen „historischen" Wert besitzen. Hier stellt sich eher die Frage, in wie weit die Annotationsgranularität und der Einblendezeitpunkt der *Grabbables* vom Kontext der Ausstrahlung abhängig ist.

- Bei sehr emotionalen Sendungen, z.B. über Krankheiten oder Schicksalsschläge während des Krieges, kann die Einblendung von *Grabbables* als visuell störend oder sogar als unangemessen gelten. Daher sollte bei solchen Sendungen nur bedingt und mit Vorsicht die Einblendung von *Grabbables* angestoßen werden.

Live vs. vorgefertigte Produktionen

Die vorangegangenen Beispiele zeigen, wie wichtig der Kontext bei der Einblendung der *Grabbables* während einer Fernsehsendung ist. Eine redaktionelle Überprüfung und Validierung der *Grabbables*, bevor diese beim Swoozy-Client, d.h. dem Zuschauer, eingeblendet werden, ist aus diesem Grund sinnvoll. Während der redaktionellen Überprüfung werden die semi-automatisch generierten *Grabbables* von einer Person auf Richtigkeit und Plausibilität hin überprüft und freigegeben. Erst nach diesem letzten Schritt werden die *Grabbables* tatsächlich angezeigt. Erschwerend kommt hinzu, dass bei der Analyse des Live-Fernsehsignals, z.B. durch die OCR-Verfahren, immer die Gefahr besteht, dass die semi-automatischen Verfahren bestimmte Informationen oder Annotationen im Video falsch interpretieren und ausliefern, z.B. bei einer Live-Übertragung eines Fußballspiels könnte die OCR-Analyse der Einblendung, den Namen eines Fußballspielers nicht korrekt erkennen (statt *Ronaldo*, *Donaldo*). Ließe man also die automatischen Prozesse unkontrolliert laufen, so bekäme der Zuschauer (sicherlich zu seiner großen Verwunderung) zum Bild nicht passende *Grabbables* angezeigt. Um solche unpassenden Anzeigen von Grabbables zu vermeiden, können Redakteure live Korrekturen durchführen und fehlerhafte Erkennungen beheben.
Bei Filmen oder Beiträgen, deren Inhalte bereits bekannt sind, ist der

redaktionelle Aufwand niedriger, da in den meisten Fällen der inhaltliche und zeitliche Ablauf vor der Sendung bekannt ist. Ähnlich einem Storyboard wird bestimmt, welche *Grabbables* zu welchem Zeitpunkt angezeigt werden. Die Zeitpunkte der *Grabbables*-Einblendung können in Form einer SwoozyML-Datei dauerhaft gespeichert werden. Bei Wiederholungen eines Filmes braucht der Redakteur nur diese SwoozyML-Beschreibungsdatei zu laden.
Diese Beispiele zeigen, dass die komplett automatische Extraktion zwar prinzipiell technisch möglich ist, aber aus Qualitätsgründen der Fernsehsender immer noch die Ergebnisse überprüfen sollte. Dies wird durch eine redaktionelle Nachbearbeitung realisiert. Um diese Aufgabe ohne großen Zeitbedarf lösen zu können, kommen die Unterstützungswerkzeuge Swoozy-Livana und SKRPTR , die im nächsten Kapitel detaillierter vorgestellt werden, zum Einsatz.

Zusammenfassung

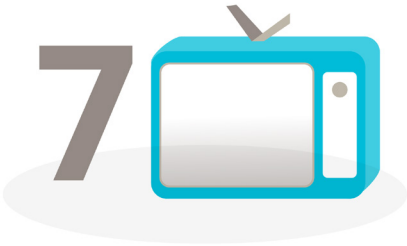
Dieses Kapitel hat gezeigt, dass die Realisierung von Swoozy sowohl architektur- als auch softwaretechnisch realisierbar ist. Die Echtzeitanalyse (von Audio-, Text- und Videosignalen) wird verteilt und von mehreren, generisch zusammengeknüpften Komponenten durchgeführt. Dadurch wird eine Modularität im Systemaufbau gewährleistet, d.h. bestimmte Analysekomponenten und Dienste können nahtlos ersetzt werden, ohne die interne Verarbeitung zu beinträchtigen. Ein weiterer Vorteil des verteilten Ansatzes besteht darin, dass die Verarbeitungsintelligenz (Analyse des Fernsehsignals) unabhängig von der Rechenleistung des Clients (Fernsehhardware) durchgeführt wird. Dies bildet eine Grundlage für das Cloud-basierte Fernsehen. Die Interakti-

on wurde in so weit implementiert, als dass sie den komplexen Prozess der Auswahl des Aufrufs und der Ergebnisanzeige der Webdienste unkompliziert, aber trotzdem interaktiv, dem Benutzer zur Verfügung stellt. Hierfür wurden innerhalb der Swoozy-Benutzeroberfläche die neuen *Grab'n'Drop*- und *Grabbables*-Paradigmen entwickelt und grafisch umgesetzt. Diese Umsetzung von Swoozy erfolgte über die Konzipierung einer intuitiv gestalteten Benutzerschnittstelle mit der permanenten Berücksichtigung der *10-foot Design*-Leitlinien.

Parallel dazu wurde gezeigt, welche neuen Herausforderungen sich bei der automatischen Semantifizierung von Fernsehprogrammen stellen und welche möglicherweise auftretenden Nebeneffekte dabei vermieden werden sollten. Insbesondere bei der automatischen Extraktion von Informationen und der Darstellung von *Grabbables* muss eine kohärente und kontextorientierte Anzeige gewährleistet werden. Zusätzlich, auch wenn es technisch machbar ist, möglichst viele Annotationen oder *Grabbables* aus einem Fernsehprogramm extrahieren zu lassen, sollte die Anzeige der *Grabbables* immer kohärent, passend und sinnvoll erfolgen.

Die Entscheidung, welche *Grabbables* tatsächlich beim Zuschauer eingeblendet werden dürfen, erfordert bei bestimmten Sendungen einen redaktionellen Zwischenschritt zu Gunsten der Qualität und der besseren Interaktion. Werden einem Benutzer innerhalb eines kürzeren Zeitraums zu viele Annotationen angezeigt, ist es möglich, dass er den Überblick und die Möglichkeit, vernünftig mit den *Grabbables* zu interagieren, verliert. Bekommt er aber *Grabbables*, die kohärent sind und zum Sendungskontext passen, angezeigt, auch wenn dies mit einer geringen Verzögerung geschieht, wird das Nutzererlebnis gesteigert. Um diesen Qualitätsanforderungen zu entsprechen, müssen den Redakteuren softwaretechnische Werkzeuge zum Einfügen von An-

notationen bzw. *Grabbables* in Echtzeit, inklusive der Verknüpfung zu den dahinterliegenden semantischen Konzepten, Nacheditieren und Bestimmung des Einblendzeitpunkts zur Verfügung gestellt werden. Diese Betrachtung ist Bestandteil des nächsten Kapitels.



Werkzeuge für Swoozy

Im vorherigen Kapitel wurden die Architektur, die Konzepte und die technische Realisierung des semantischen Fernsehens detailliert vorgestellt. Dabei wurde deutlich, dass die automatischen Verfahren zur Erzeugung und Anzeige der *Grabbables* eine zentrale Rolle bei dem Swoozy-System spielen. Auch wenn die Automatisierung der Extraktion aus einem Live-Fernsehbild mehrere Vorteile in puncto Schnelligkeit bietet, muss in der aktuellen Medienlandschaft berücksichtigt werden, dass Fernsehsender (zumindest öffentlich-rechtliche Fernsehsender) noch andere Faktoren beachten müssen, wie z.B. die redaktionelle Qualität von Sendungen. Bei privaten Fernsehsendern stehen wirtschaftliche Faktoren im Vordergrund, so dass der Fokus mehr auf Werbeeinahmen und gezielterer individualisierter Werbung (Stichwort: Adressable-TV, siehe Kapitel 1) gesetzt ist. Um diese Randbedingungen im aktuell beschriebenen Swoozy-Workflow zu berücksichtigen, muss für die Fernsehsender eine technische Möglichkeit geschaffen und eine redaktionelle Unterstützung zur Live-Generierung semantischer Informationen in Form von *Grabbables* angeboten werden. Zusätzlich sollten hauseigene Medieninhalte wie z.B. aus einer Mediathek oder einem Videostream-Angebot als Teilergebnisse parallel zu den Ergebnissen externer Webdienste über die interaktive Benutzerschnittstelle

von Swoozy zurückgeliefert werden können. Möchte ein Fernsehsender semantische Inhalte entlang der Ausstrahlung seiner Sendungen anbieten, muss eine technische Möglichkeit angeboten werden, die Erzeugung und das Editieren von *Grabbables* auf eine leichte und unkomplizierte Art und Weise zu realisieren. Für die Verbreitung und den Erfolg des semantischen Fernsehens muss sich Swoozy dieser Herausforderung stellen.

Diesbezüglich wurden Diskussionen mit Redakteuren und Fernsehjournalisten mehrerer deutschen Fernsehanstalten (ARD, ZDF, SR, SWR, RTL) durchgeführt. Als Ergebnis dieser Diskussionen kristallisierte sich heraus, dass, wenn eine automatische Extraktion aus deren Sicht vorteilhaft ist, dennoch die Notwendigkeit besteht, die Ergebnisse der Extraktion fallweise redaktionell zu validieren, um in einem weiteren Schritt eigene *Grabbables* erzeugen zu können. Diese notwendigen Schritte erforderten die Realisierung von zwei weiteren Werkzeugen: Swoozy-Livana und SKRPTR. Diese werden im Folgenden näher präsentiert. Ihr Zusammenspiel mit den Komponenten innerhalb der Swoozy-Architektur wird mittels Beispielen veranschaulicht.

Swoozy-Livana: online Annotierung von TV-Sendungen

Livana (das Akronym steht für *Liv*e *An*notation *A*pplication) ist die erste Software, die innerhalb der redaktionellen Verarbeitung von Live-Videoinhalten zum Einsatz kommt.

Editier- und Annotations-Workflow

Abbildung 7.1 zeigt den genauen online-basierten Editier- und Annotations-Workflow bei der Annotation von Live-Programmen mit Swoozy-Livana. Die entwickelte Software dient als direkte Schnittstelle zwischen dem Redakteur bei einem Fernsehsender und der eigentlichen Anzeige von *Grabbables* innerhalb des Swoozy-Clients. Die Echtzeitansteuerung der Anzeige der *Grabbables* wird online durchgeführt, was bedeutet, dass der Swoozy-Client eine dauerhafte Verbindung zum Swoozy-Server besitzt. Im ersten Schritt wird ein Videostream als Eingabe (Punkt 1 in Abbildung 7.1) benutzt. Wie im letzten Kapitel beschrieben, wird nach einer automatischen Extraktion aus Audio, Videos und Texten (Schritt 2) eine Liste möglicher Begriffe und Entitäten erzeugt. Diese dienen als Vorschlagsbasis für den Redakteur, d.h. er besitzt eine vollständige Liste der extrahierten *Grabbables*. Diese Liste kann entweder nacheditiert werden (3) oder, durch die Eingabe und ggfs. die Korrektur der erkannten Begriffe und die Festlegung der damit verbundenen Konzepte, erweitert werden. Im Beispiel der Abbildung werden die Texte „Bastian Schweinsteiger“ und „Allianz Arena“ nachträglich mit den Konzepten „Person“ und „Object“ verbunden. In einem nächsten Schritt, dem sog. redaktionellen Schritt (5), werden dem Redakteur zwei Optionen angeboten.

Die erste Option besteht darin, angegebene und verifizierte Informationen bzw. *Grabbables* für die Ausstrahlung frei zu geben. Bei der zweiten kann der Redakteur, falls er über diese Kompetenz verfügt, eine semantische Verlinkung durchführen, z.B. durch das Hinzufügen einer Wikidata- oder Mediathek-URL oder eines Verweises auf Konzepte einer vordefinierten verwendeten Ontologie entlang der Festlegung der *Grabbables*. Nachdem diese Schritte erfolgreich durch-

geführt worden sind, wird die Liste der *Grabbables* zum Swoozy-Server gesendet (6). Parallel dazu erfolgt eine Speicherung bzw. Archivierung der *Grabbables* in einer Annotationsdatenbank mittels des SwoozyML-Formats. Diese gespeicherten *Grabbables* können zu einem späteren Zeitpunkt (z.B. während einer Wiederholung) neu abgerufen werden. Über einen Push-Befehl signalisiert der Swoozy-Server den verbundenen Swoozy-Clients, dass neue *Grabbables* generiert worden sind. Die grafische Benutzeroberfläche wird erneuert und die neu erzeugten *Grabbables* werden eingeblendet (7). Ab diesem Zeitpunkt kann der Zuschauer diese neuen *Grabbables* für eine Suche benutzen. Um den Workflow erfolgreich durchführen zu können, kann sich der Redakteur auf Swoozy-Livana stützen. Die Bedienoberfläche von Swoozy-Livana ist mit Absicht einfach gestaltet worden, damit die Bearbeitung und Erzeugung der *Grabbables* schnell und intuitiv verläuft. Diese wurde mit der Webtechnologie HTML5 implementiert und kommuniziert über eine dedizierte REST-Schnittstelle direkt mit dem Swoozy-Server (siehe Abbildung 7.2).

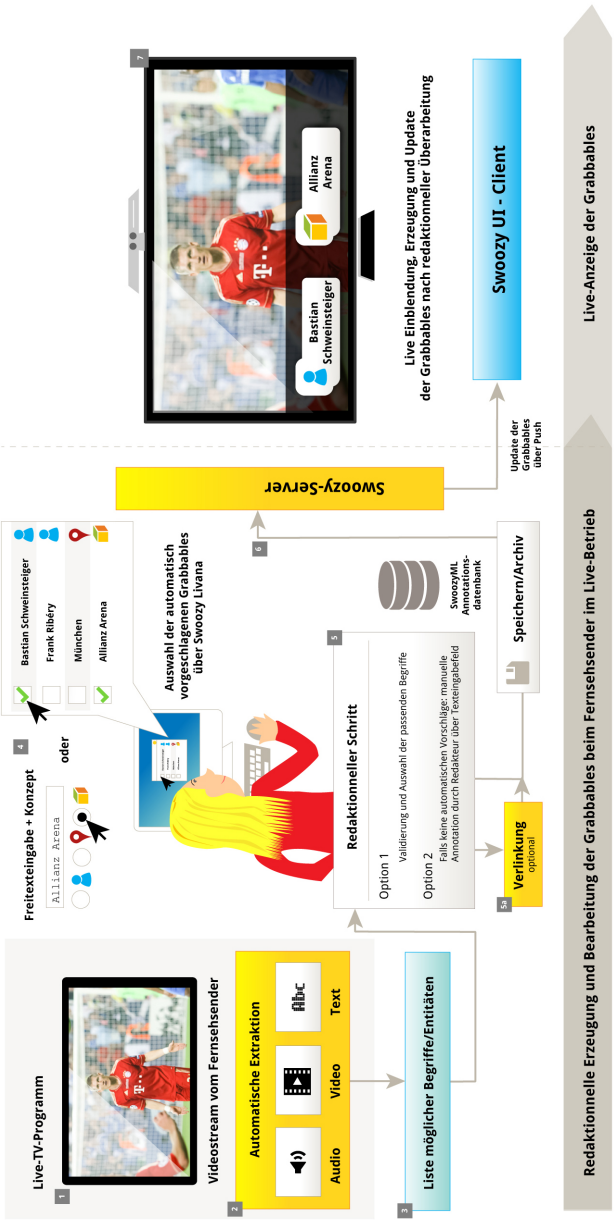


Abbildung 7.1: Swoozy-Annotationsvorgang im Live-Kontext

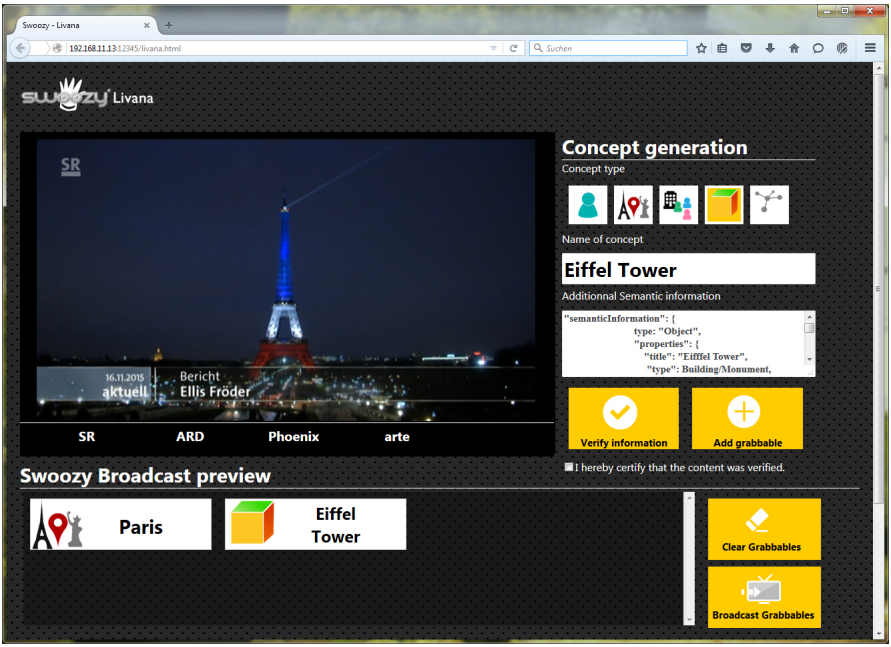


Abbildung 7.2: Web-basierte Bedienoberfläche von Swoozy- Livana

Im zentralen Abschnitt der Benutzerschnittstelle können die verschiedenen Fernsehkanäle ausgewählt werden und der Live-Videostream wird entsprechend eingeblendet (Schritt 1 von Abbildung 7.1). Auf der rechten Seite befinden sich die Optionen für die Konzept- und Entitäten-Generierung durch den Redakteur. Zuerst muss eines von fünf globalen Konzepten ausgewählt werden (*Person*, *Location*, *Organization*, *Object* und *Other/Misc.*). Die Auswahl bestimmt im Hintergrund, welche Ergebnisarten angezeigt werden können (z.B. Biografie für eine Person oder eine bebilderte Karte zu einer Stadt). Nachdem das Konzept ausgewählt wurde, wird der Redakteur den Namen des Konzepts eintragen. Der Name wird von einer Plausibilitätskomponente (um vorab mögliche Tippfehler zu beseitigen) und einer Autovervollständi-

gungsfunktion überprüft, die bereits im System vorhandene und eingetragene Konzepte beim Eintippen vorschlägt. Optional können die automatisch extrahierten Entitäten aus OCR-, Video- und Audioanalyse vorgeschlagen bzw. als *Grabbable* automatisch zum Vorschaubereich (in der Abbildung 7.2 als *Swoozy Broadcast Preview* bezeichnet) hinzugefügt werden (Schritt 2-3). Die vorgeschlagenen Konzepte können beliebig erweitert und von der Redaktion mit zusätzlichen semantischen Informationen versehen werden. Dazu steht ein Freitextfeld zur Verfügung, bei dem Eigenschaften wie z.B. GPS-Koordinaten, Einwohnerzahl oder Geburtsdaten hinzugefügt werden können. Diese Eigenschaften wurden anhand einer vordefinierten Beschreibung (bzw. Taxonomie) festgelegt und dynamisch vom Swoozy-Server geladen (Schritt 3).

Nachdem diese Informationen eingetragen worden sind, kann der Redakteur sie mittels des *Verify Information* Buttons überprüfen lassen. Diese Unterstützungsmöglichkeit besteht darin, dass eine Webseite oder ein Dienst über den Webbrowser gestartet wird und die eingetragenen Informationen manuell und redaktionell überprüft werden (Schritt 5). Optional stehen zwei Möglichkeiten für den redaktionellen Schritt zur Verfügung.

Entweder kann die Validierung und Auswahl der automatisch erkannten Konzepte „per Klick" durchgeführt werden oder, falls kein gültiger Vorschlag aus der Liste der möglichen Begriffe/Entitäten extrahiert werden konnte, kann der Redakteur diesen manuell hinzufügen (Schritt 3). Alternativ können direkte Verlinkungen zu Mediatheken (Schritt 5a) über die Eigenschaft *linkDatareference* des Blocks *semanticInformation* hinzugefügt werden. Diese Funktion gibt besonders Redakteuren die Möglichkeit, direkt das Konzept mit einem Beitrag einer hauseigenen Mediathek zu verknüpfen und somit an den Swoozy-

Client das Signal zu übermitteln. Möchte der Redakteur eine geografische Lokalisation, z.B. als *Location-Grabbable* angeben, kann er dies über eine interaktive Karte tun. Die GPS-Informationen des Ortes werden im Hintergrund gespeichert.

Im Falle einer Person werden die Daten von Wikidata angezeigt. Dadurch kann gewährleistet werden, dass die vom Fernsehsender stammenden *Grabbables* vom Redakteur im Vorfeld kontrolliert wurden. Mit dem Ankreuzen einer Checkbox wird die Entscheidung, die erstellten *Grabbables* zu generieren, nochmals bestätigt. Mit Hilfe des Buttons *Add Grabbable* kann das generierte Grabbable im Vorschaubereich (*Swoozy Broadcast Preview*) gesichtet werden. Mehrere *Grabbables* können zu diesem Balken hinzugefügt werden, aber jedes Hinzufügen muss proaktiv vom Redakteur gesichtet und validiert werden. Nachdem die *Grabbables* zum Vorschaubereich hinzugefügt wurden, kann der Redakteur sie zum Swoozy-Server senden (Schritt 6). Zu Archivierungszwecken und nachträglicher Wiederverwendung der *Grabbables* (z.B. bei einer Wiederholung der gleichen Sendung zu einem späteren Zeitpunkt) können sie mit Zeitstempel in der SwoozyML-basierten Annotationsdatenbank gespeichert werden.
Im letzten Schritt werden die *Grabbables* direkt zum Swoozy-Client über eine RESTSchnittstelle versendet und auf Seite des Clients entsprechend dargestellt (Schritt 7).

Swoozy-SKRPTR: offline Annotierung von TV-Sendungen

Swoozy-SKRPTR (Aussprache: *Skripter*) ist eine dedizierte Software, die speziell für sog. *Content Producer* und Fernsehsender erstellt wurde, damit diese schnell und effizient Videos mit semantischen Annotationen für eine Benutzung durch das Swoozy-System versehen können. Dieses Werkzeug kommt zum Einsatz, wenn automatische Verfahren nicht anwendbar sind oder wenn eine besonders hohe Granularität bei den Einblende-Zeitpunkten für die Anzeige der *Grabbables* erreicht werden soll. Die Software wird ausschließlich offline betrieben, da diese speziell nur für das Hinzufügen der *Grabbables* in eine Annotationsdatenbank, z.B. durch eine Redaktion, vorgesehen ist.
Dies kann bei handlungs- oder informationsreichen Szenen der Fall sein. Swoozy-SKRPTR eignet sich besonders für vorgefertigte Produktionen wie Filme, Berichte oder Reportagen. Dank der mit einer Zeitleiste versehenen Schnittstelle können die *Grabbables* zeitlich exakt positioniert werden.

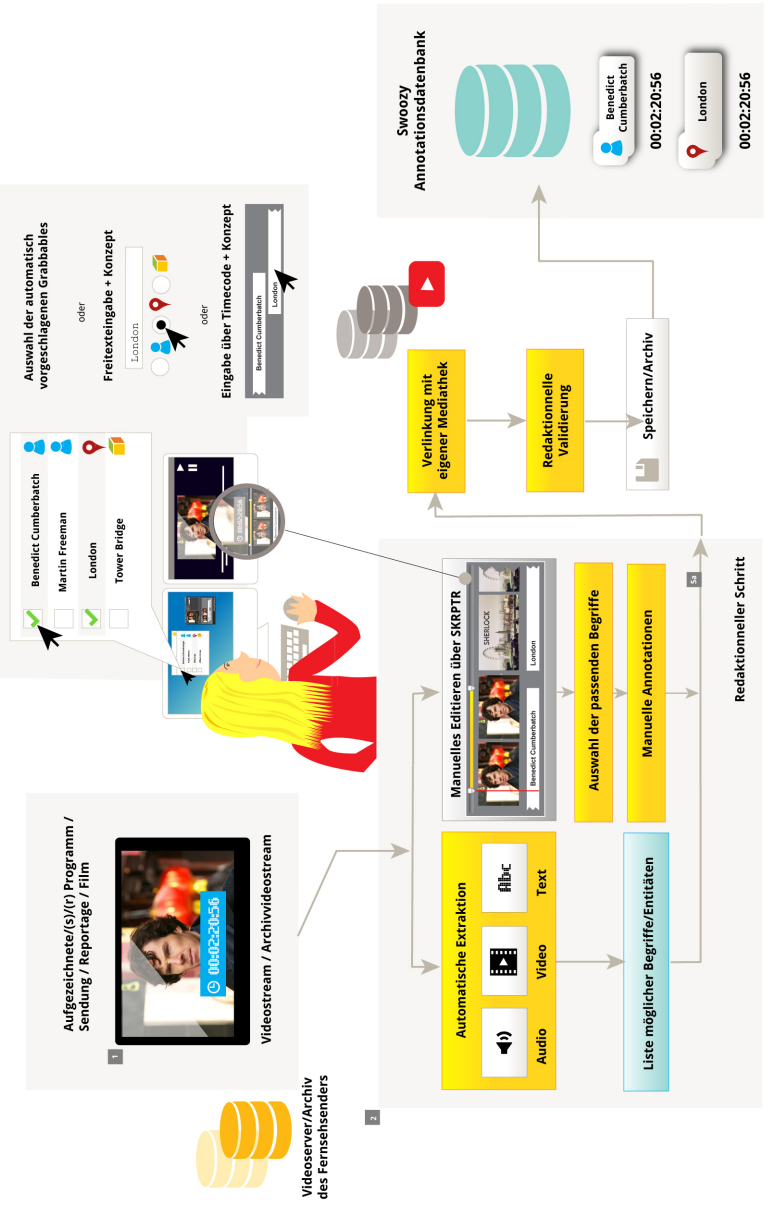


Abbildung 7.3: Swoozy Annotationsvorgang mit SKRPTR

Benutzerschnittstelle, Editier- und Annotations-Workflow

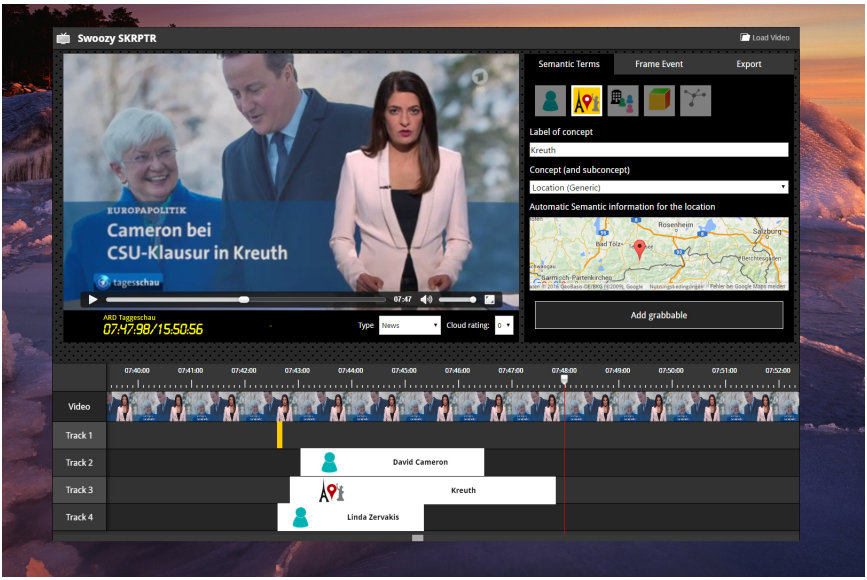


Abbildung 7.4: Bildschirmabzug von Swoozy-SKRPTR

Ähnlich wie bei Swoozy-Livana ist die Benutzerschnittstelle von Swoozy-SKRPTR (siehe Abbildung 7.4) sehr einfach und kompakt gestaltet worden. Angelehnt an das Prinzip der Editierbarkeit des Videomaterials unter Verwendung einer Zeitleiste, die von den gängigsten Videoschnittprogrammen bekannt ist, ermöglicht SKRPTR eine genaue zeitliche Positionierung der Begriffe (*Grabbables*) auf einer Zeitachse parallel zur Videowiedergabe.

Die grafische Zeitleiste bildet die zentrale Bedienkomponente der Applikation. Durch einen beweglichen Slider kann der Redakteur sehr schnell zu einer Vorschau jedes einzelnen Videobilds gelangen. Über die Abspieltasten kann er von einem Zeitstempel zum anderen navigieren und mit dem Abspielkopf die *Grabbables* genau auf das ge-

wünschte Videoeinzelbild (engl. Frame) positionieren.
In der aktuellen Version von SKRPTR ist die Zeitleiste in fünf sog. semantische Spuren unterteilt, die fünf globale Konzepte repräsentieren (*Person*, *Location*, *Organization*, *Object* und *Other/Misc*) und als Grundlage für das zeitlich angesteuerte Anlegen und Editieren der *Grabbables* dienen. Die Tatsache, dass diese fünf Spuren gleichzeitig angezeigt werden, ermöglicht eine bessere Kontrolle darüber, welche Informationen innerhalb einer Zeitspanne angezeigt werden sollen. Wird ein *Grabbable* per Drag'n'Drop hinzugefügt, kann es innerhalb der Zeitleiste bewegt und versetzt werden. Die Dauer, der Erscheinungszeitpunkt, die Anzeige und die zeitliche Gültigkeit des jeweiligen *Grabbables* wird ebenso über die Zeitleiste angesteuert. Um die Dauer der Anzeige eines gerade hinzugefügten *Grabbable* festzulegen, wird dieser grafisch per Mausklick über die Zeitleiste gestreckt (siehe Abbildung 7.4).
SKRPTR bietet eine Unterstützung der Annotation. Will z.B. ein Redakteur eine bestimmte Szene mit dem Konzept *Location* versehen, kann er dies entweder manuell, z.B. mittels der freien Texteingabe, oder über eine interaktive Karte, realisieren. Somit lassen sich dank GPS-Koordinaten ganz präzise die Drehorte einer Filmszene oder eines Events (siehe Abbildung 7.5 mit der geografischen Referenz des Orts „Nürburgring") nachträglich festlegen.
Die *Grabbables* oder Begriffe, die auf die Zeitleiste platziert werden können, stammen entweder direkt vom Swoozy-Videoserver oder aus der Swoozy-Archivdatenbank (für den Fall, dass ein Redakteur diese schon bei einer Live-Annotation über Swoozy-Livana angelegt hat). In einem weiteren Schritt können entweder die passenden Begriffe samt manueller Annotationen angelegt oder über die automatische Extraktion, z.B. über OCR oder eine Untertitel-Extraktion automatisch erzeugt und mit einem Zeitstempel versehen werden. Die Liste der

extrahierten und möglichen Begriffe wird dem Redakteur zur Auswahl gestellt oder kann über eine Freitexteingabe samt Konzepten manuell spezifiziert werden.

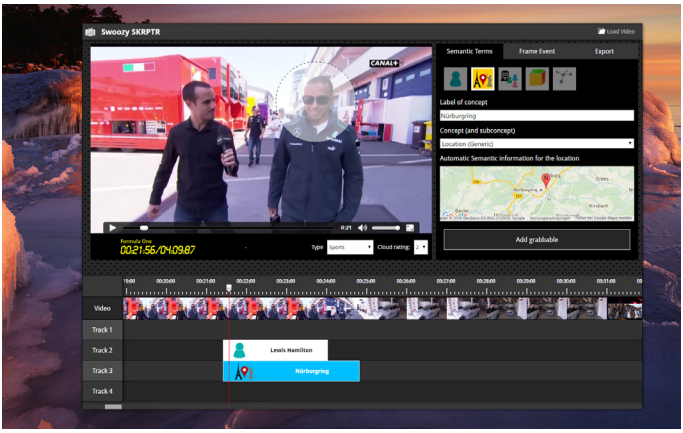


Abbildung 7.5: Bildschirmabzug von Swoozy-SKRPTR: Annotation einer Reportage über die Formel 1

Dieser Schritt wird als redaktioneller Schritt bezeichnet (Schritt 5a von Abbildung 7.3). Hierbei kann über das Hinzufügen von Eigenschaften gewährleistet werden, dass Zusatzinformationen samt Konzept mitgespeichert und mit einer Verlinkung zur eigenen Mediathek versehen werden können. Nachdem die Begriffe von der Redaktion überprüft und validiert worden sind, werden sie in einer Annotationsdatenbank dauerhaft gespeichert.

Diese Datenbank enthält pro Sendung, Beitrag oder Video eine präzise zeitbasierte Definition der Anzeigezeitpunkte der *Grabbables*, gespeichert im SwoozyML-Format. Dieses ermöglicht unter anderem eine bessere Vorbereitung des Semantifizierung-Schritts eines Beitrages durch den Redakteur und im Falle von Wiederholungen einen schnelleren und gezielteren Zugriff auf die kompletten Annotationen

(und *Grabbables*). Somit können bekannte Begriffe schnell wieder aufgerufen und an Swoozy-Clients zur Anzeige gesendet werden. Der komplette Workflow des Annotationsvorgangs mittels SKRPTR wird in Abbildung 7.3 detailliert vorgestellt.

Merkmale

Beide Werkzeuge, Swoozy-Livana und SKRPTR, ermöglichen es Fernsehsendern, im LiveBetrieb genauere Annotationen zu einer Sendung hinzuzufügen und entsprechend zu publizieren, sodass die Swoozy-Clients diese Information in Form von *Grabbables* anzeigen können. Bei der Konzipierung der Werkzeuge wurde besonders auf die einfache Handhabung des Annotationsvorganges geachtet. Diese Faktoren sind bei heutigen Produktionsprozessen von enormer Bedeutung.
Ein weiteres Merkmal der Swoozy-Werkzeuge bildet die Einfachheit der dahinter liegenden Sprachen, der Formulierung der semantischen Konzepte mittels SwoozyML und der vereinfachten JSON-basierten Repräsentation der Konzepte. Aktuell kommerziell verfügbare Videoschnittprogramme wie Adobe Premiere oder Apple Final Cut Pro[1] bieten zwar eine Unterstützung für die Annotation und Exportmöglichkeiten an, jedoch sind diese innerhalb eines Produktions-Workflows aufgrund ihrer teils komplexen proprietären Ausgabenformate für Laien und Nicht-Programmierer nur schwer les- und verwertbar, sodass sie nicht prozessübergreifend eingesetzt werden können. Andere Formate wie BMF (Broadcast Metadata Exchange Format)[2], BIFS (Binary Format for Scene) oder MPEG-7, die theoretisch von der Film- und Produkti-

[1] https://developer.apple.com/library/mac/documentation/AppleApplications/Reference/FinalCutPro_XML/Documents/Documents.html

[2] http://bmf.irt.de/document_downloads/BMF-XML-Schema-einfuehrung.pdf

onsindustrie eingesetzt werden könnten, leiden unter dem Problem der Komplexität der Formate und der fehlenden Unterstützung auf Softwareebene. D.h., es gibt keine für das breite Publikum und für Redakteure spezifischen benutzerfreundlichen Werkzeuge, die diese Formate innerhalb eines Videoeditiervorgangs einsetzen und eine erste Unterstützung auf dem Weg zum semantischen Fernsehen bieten. Zusätzlich ist die Komplexität der teilweise semantisch repräsentierten Beschreibungen (wie z.B. XMP), die von den kommerziellen Videoapplikationen eingesetzt werden, ein Hindernis für Nicht-Programmierer oder Nicht-Ontologen. Daher ist aus redaktioneller Sicht die Verwendung dieser Software als Grundlage für ein semantisches Fernsehen ungeeignet. Aus diesem Grund wurde bei den Werkzeugen SKRPTR und Livana besonders viel Wert auf die Einfachheit der Bedienung und die Wiederverwendbarkeit der daraus resultierenden Strukturen und Datenformate innerhalb des Annotationsworkflows gelegt.

Architektur

Swoozy-Livana und -SKRPTR integrieren sich in die im letzten Kapitel vorgestellte Cloud-basierte Architektur. Einerseits besteht eine kontinuierliche Kommunikation zwischen den beiden Werkzeugen und dem Swoozy-Server, damit eine Benutzung auch im Live-Betrieb möglich ist, und andererseits spielt der Swoozy-Server nach wie vor die Rolle als zentrale Komponente, welche die erzeugten *Grabbables* verwaltet und zu den Swoozy UI-Clients weitereicht. Die drei Komponenten Swoozy-Server, Swoozy UI-Client und Annotier-Werkzeuge sind verzahnt. Dabei unterscheidet man zwei Fälle:

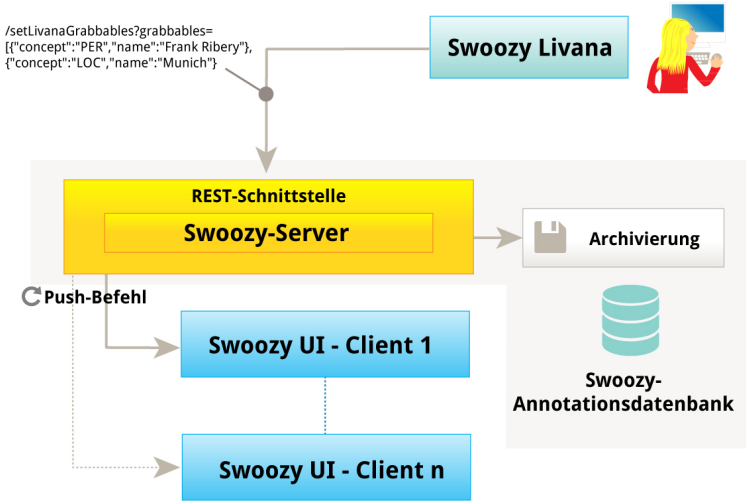


Abbildung 7.6: Workflow einer Live-Annotation mittels Swoozy-Livana

Der erste, wie in der Abbildung 7.6 zu sehen, betrifft den Live-Modus. Nachdem ein Redakteur bestimmte *Grabbables* über Livana editiert hat, werden diese über einen REST-Befehl (hier */setLivanaGrabbables*) zum Swoozy-Server weitergereicht.

Der Befehl wird über eine dedizierte REST-basierte Schnittstelle vom Swoozy-Server empfangen. Optional können die live editierten *Grabbables* dauerhaft archiviert werden. Dies erfolgt durch die Speicherung der *Grabbables* in einer Annotationsdatenbank, die persistent SwoozyML-basierte Beschreibungen von Sendungen beinhaltet. Die Befehle *Grabbables* ein- bzw. auszublenden werden über Push-Befehle realisiert.

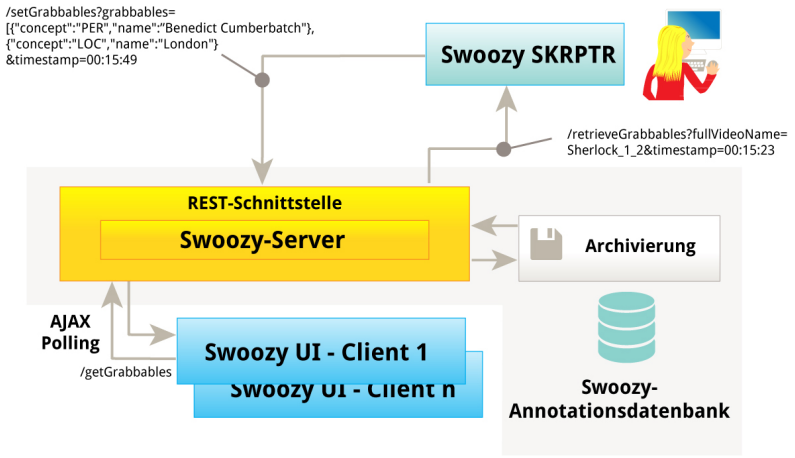


Abbildung 7.7: Workflow einer Annotation mittels Swoozy-SKRPTR

Dadurch sendet der Swoozy-Server allen an ihn gebundene Swoozy-Clients den Befehl, die Anzeige der *Grabbables* zu erneuern und die neu generierten *Grabbables* an die korrekten Stellen innerhalb der Leiste der Benutzerschnittstelle (dem sog. *Bottom Bar* des Swoozy-Clients) zu platzieren.

Im zweiten Fall, z.B. wenn eine Sendung von einem Redakteur vorannotiert wurde (bei Video On Demand), müssen die Swoozy-Clients kontinuierlich per REST-Aufruf den Server nach den korrespondierenden *Grabbables* abfragen (siehe Abbildung 7.7). Dies wird über einen AJAX-Polling-Mechanismus durchgeführt, d.h. der Client fragt kontinuierlich den Swoozy-Server nach neuesten *Grabbables* ab. Bei der Verwendung von Swoozy-SKRPTR kann der Redakteur auf die Annotationsdatenbank zurückgreifen.

Diese Operation wird über den Aufruf */retrieveGrabbables* kombiniert

mit einem Zeitstempelparameter durchgeführt. Während des Annotationsvorgangs wird die Veränderung seitens der Swoozy-SKPRTR-Software ebenfalls über eine dedizierte REST-Schnittstelle parallel zum Swoozy-Server gesendet. Der Befehl */setGrabbables*, samt *timestamp*-Parameter, führt dazu, dass die gerade editierten *Grabbables* dauerhaft gespeichert und für eine weitere Verwendung innerhalb der SKRPTR-Software zur Verfügung gestellt werden.

Fazit

In diesem Kapitel wurden zwei Werkzeuge vorgestellt, die mit einer webbasierten und intuitiven Benutzeroberfläche das Ziel verfolgen, Redakteure bei der semantischen Annotation der Videoinhalte zu unterstützen. Die Werkzeuge SKRPTR und Livana funktionieren nahtlos mit der Swoozy-Architektur zusammen und können verteilt benutzt werden, da ihnen der Swoozy-Server über dedizierte REST-Schnittstellen die Möglichkeit gibt, *Grabbables* und Zusatzinformationen über einen Webbrowser anzugeben. Mechanismen wie die Anzeige, das Editieren und das spätere Suchen der *Grabbables* werden ebenfalls über die REST-basierten Schnittstellen des Swoozy-Servers durchgeführt. Da Annotationen auch bei Reportagen oder Filmen eingesetzt werden können, können diese als SwoozyML-Dateien persistent in einer Annotationsdatenbank gespeichert und jederzeit über das Werkzeug SKRPTR aufgerufen werden. Somit erhält der Redakteur beim Editieren eine Unterstützung in Form von Vorschlägen.
Diese Werkzeuge verfolgen das primäre Ziel, über die Benutzung von einfachlesbaren SwoozyML-basierten semantischen Beschreibungen den Fernsehredakteuren, die mit semantischen Technologien nicht ver-

traut sind, dennoch eine Unterstützung beim Annotations-Workflow anzubieten. Mithilfe dieser Werkzeuge können Fernsehsender, ohne größeren Aufwand und Veränderung des redaktionellen Produktionsworkflows, Live-Videos für das semantische Fernsehen bzw. Swoozy vorbereiten und den Zuschauern über die Swoozy-Clients in Echtzeit anbieten.

Das nächste Kapitel befasst sich mit den Anwendungen und Technologiedemonstratoren, die auf der Swoozy-Architektur aufbauen.

Demonstratoren

Das Swoozy-System und die einzelnen Komponenten wurden als Gesamtsystem in mehreren gleichnamigen Demonstratoren vorgestellt. Diese Demonstratoren zeigen mit ihren verschiedenen Versionen die anwendungsnahe Umsetzung des Systems. Sie dienen als Beweis, dass die vorgestellten Konzepte softwaretechnisch erfolgreich umgesetzt werden konnten. Zusätzlich wurde gezeigt, dass die ausgewählte und implementierte Architektur zum semantischen Fernsehen mittels der skizzierten Verfahren erstens komplett funktionsfähig ist und sich zweitens nahtlos in die aktuelle *„Connected TV"*-Landschaft integrieren lässt.

Das Swoozy-System ist über mehrere Iterationen entstanden, wurde kontinuierlich erweitert und mit neuen innovativen Funktionalitäten versehen. Die verschiedenen Versionen und deren technologische Merkmale sowie Entwicklungsstadien werden auf folgender Zeitachse graphisch präsentiert.

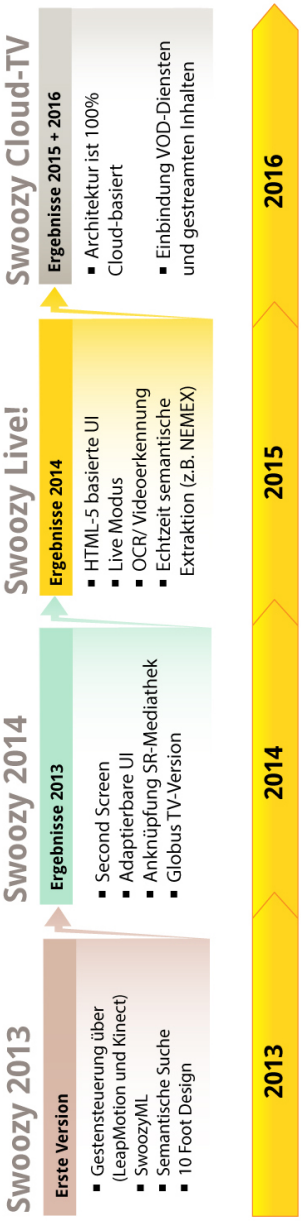


Abbildung 8.1: Zeitachse der Swoozy-Versionen

Insgesamt wurden fünf Versionen bzw. Varianten von Swoozy im Zeitraum März 2013 - November 2016 implementiert (siehe Abbildungen 8.1 und 8.15). Bei jeder Iteration wurden die rechtlichen und design-technischen Aspekte der TV-Branche (z.B. bzgl. der Overlay-Frage oder der parallelen Anzeige von kommerziellen Angeboten bei einem öffentlich-rechtlichen TV-Programm) einbezogen und softwaretechnisch gelöst. Die fünf Versionen von Swoozy wurden nacheinander dem breiten Publikum und Experten der TV-Branche, inklusive der Möglichkeit die Systeme selbst auszuprobieren, vorgestellt, was der Robustheit der Systeme zu Gute kam. Dies geschah u.a. während der CeBIT-Messen 2013 und 2014 und seit 2015 während Veranstaltungen bei Institutionen wie der Landesmedienanstalt Saarland[1], der Handwerkskammer des Saarlandes[2], des Fernsehsenders ARD[3], des Saarländischen Rundfunks[4], dem französischen Fernsehsender Mirabelle TV[5] und dem Conseil Départemental de la Moselle[6]. Das Kapitel wird mit der Darstellung der fünf Versionen bzw. Demonstratoren fortgeführt. Eine detaillierte Tabelle der Versionen rundet das Kapitel ab.

Swoozy 2013 - CeBIT 2013-Demonstrator

Die erste Version von Swoozy erhielt einen CeBIT Innovation Award. Sie wurde erstmalig im Rahmen der CeBIT 2013 vorgestellt und ist mehrmals während medienwirksamer Veranstaltungen dem breiten

1 https://www.lmsaar.de/2014/09/internet-der-dinge
2 http://www.hwk-saarland.de
3 http://www.ard.de
4 http://www.sr-online.de
5 http://www.mirabelle.tv
6 http://www.cg57.fr/Pages/default.aspx

Publikum präsentiert worden. Hauptfokus der 2013er Version von Swoozy war die intelligente Bedienung eines Fernsehsystems über Gestensteuerung. Zu diesem Zweck wurde nach einer genauen Analyse der schon existierenden Interaktionsmetaphern die Entscheidung getroffen, eine stark benutzerzentrierte grafische Bedienoberfläche zu entwerfen und zu implementieren. Die erste Version der Swoozy-Benutzeroberfläche wurde mittels der Adobe AIR-Technologie implementiert. Dies hatte den Vorteil, dass die Medien und grafischen Varianten bei unterschiedlichen Bildschirmauflösungen perfekt dargestellt und die Multimedia-Ergebnisse (von unterschiedlichen Medientypenarten (JPEG, PNG oder Video-Material)) aus verschiedenen Webdiensten unterstützt und mühelos und dynamisch in das Design eingebunden werden konnten. Die Anbindung der interaktiven Benutzeroberfläche zur Gestensteuerung wurde mittels der ersten Version der Microsoft Kinect realisiert und über eine AIR-Native Extension Komponente[7] mit der grafischen Ausgabekomponente verknüpft. Somit war es möglich, die Verarbeitung der Skelettpunkte direkt im Swoozy-Client zu realisieren und dementsprechend die interaktiven grafischen Elemente besser mit Gesten, wie der *Grab'n'Drop*-Geste, zu verbinden. Zusatzgesten wurden innerhalb dieser ersten Swoozy-Version eingebunden, z.B., wenn der Benutzer die Hände nach unten bewegt, wechselte Swoozy in den Vollbild-Modus.

Die 2013er-Version von Swoozy beinhaltete eine erste rudimentäre Unterstützung der Leap Motion, die als fester Bestandteil in die 2014er-Version integriert wurde. Ein weiterer Schwerpunkt des Systems lag in der Anbindung an Dienste des Semantic Web. Dafür wurde, angelehnt an das Spotlet-Prinzip, das Konzept der *Grabbables* entwickelt, das die Lücke zwischen Suche im Semantic Web, benutzerzentrierter Anzeige

[7] http://www.adobe.com/devnet/air/native-extensions-for-air.html

der Ergebnisse und Gestensteuerung zu schließen half. Zur Annotation der abgespielten Videos wurde das Format SwoozyML definiert (siehe Kapitel 6).

Swoozy 2014 – CeBIT 2014-Demonstrator

Die 2014er-Version von Swoozy ist eine Weiterentwicklung der ersten Swoozy-Version. Hierbei wurde der Fokus auf die Verwendung eines Second Screen gesetzt, die schon die von der TV-basierten Bedienoberfläche bekannte Interaktion des *Grab'n'Drop*-Paradigmas auf ein Tablet portiert und in Form einer dedizierten mobilen Swoozy-App implementiert (siehe Abbildung 8.2).

Abbildung 8.2: Second Screen-Version von Swoozy

Parallel dazu wurden die Funktionalitäten der Cross-Devices-Interaktion (d.h. eine Interaktion vom Fernseher zum Tablet und umgekehrt) technisch umgesetzt. In der 2014er-Version wurden neue Bedienfunk-

tionalitäten in die TV- und Tablet-Versionen von Swoozy integriert. Darunter die *Sling-to-Beam* Funktion, die es ermöglicht auf dem Tablet gefundene Medien direkt auf den Fernseher über eine Wischgeste „herüberzuschleudern“ und größer am Fernsehbildschirm anzuzeigen (siehe Kapitel 6).

Diese Funktionalität wird über einen WLAN-Hotspot realisiert. Wird ein Video in Richtung des Fernsehers per Touch-Geste geschleudert, „verwandelt“ sich das Tablet in eine intelligente Fernbedienung. Die mobile Swoozy-App übernimmt die Interaktionsparadigmen der TV-basierten Version. Somit können jederzeit ein oder mehrere Zuschauer, abgekoppelt von der primären TV-basierten Benutzerschnittstelle von Swoozy, eine Suche nach Multimediainhalten über das Tablet durchführen. Dem Benutzer werden die Ergebnisse auf dem Tablet angezeigt und er kann analog die *Grabbables* per Touch-Interaktion als Eingabe für eine Suche benutzen. Der umgekehrte Weg wird mittels der *Auto-Sync*-Funktion realisiert. Diese ermöglicht, dass die komplette Anzeige des TV-Bildschirms samt Ergebnissen 1:1 dupliziert und auf allen an das Swoozy-System per WLAN angeschlossenen Tablets angezeigt wird. Wird eine Suche mit der TV-Benutzeroberfläche von Swoozy durchgeführt, werden die Ergebnisse nicht nur primär auf dem TV-Bildschirm angezeigt, sondern sie erscheinen gleichzeitig innerhalb der mobilen Swoozy App. Zusätzlich wurde die Bedienoberfläche grafisch überarbeitet, so dass eine klare visuelle Trennung zwischen der Ergebnisanzeige und dem Hauptbild geschaffen wurde. Somit wurde die Overlay-Problematik umgangen.

Die Bedienungsmöglichkeiten von Swoozy mittels Gestensteuerung wurden mit Hilfe der softwaretechnischen Integration der Leap Motion erweitert. Diese ermöglicht die Ansteuerung der Swoozy-Benutzeroberfläche mit einem Finger der Hand, parallel zur Steuerung mittels der

Microsoft Kinect. Aus diesem Grund wurde die Bedienoberfläche leicht angepasst und die Gestenmetaphern vereinfacht. Bei der Benutzung von Swoozy mit der Leap Motion wird die Position des Fingers mittels eines virtuellen Cursors direkt in die Swoozy-Benutzerschnittstelle eingeblendet. Die Selektion eines Schaltelements wird mit zwei Fingern ausgeführt.

SR Mediathek - Integration in Swoozy

Als Erweiterung der CeBIT 2014-Version von Swoozy wurde erstmals testweise gezeigt, dass es mühelos möglich ist, schon existierende Multimediainhalte eines Fernsehsenders in die existierende Swoozy-Architektur einzubinden. Hierfür wurde, mit freundlicher Genehmigung des Saarländischen Rundfunks (SR)[8], die Anbindung von Swoozy an Medien der Nachrichtensendung *Aktueller Bericht* softwaretechnisch umgesetzt.
Ziel dieser Anbindung war es zu demonstrieren, inwiefern die Ansätze von Swoozy und die Verknüpfung von Multimediadaten innerhalb aktueller Infrastrukturen realisierbar waren.
Während dieser Integration wurden die ursprünglichen Datenquellen von Swoozy (z.B. Flickr, Wikipedia) mit ausgewählten Medienobjekten der SR-Mediathek und Inhalte der *Aktueller Bericht*-Facebook-Seite ersetzt. Dadurch wurde erstmalig der generische Ansatz von Swoozy mit realen, vom Fernsehsender stammenden Multimediainhalten getestet.
Diese Version wurde im Rahmen der CeBIT 2014 erstmalig exponiert.

[8] `http://www.sr-online.de`

Abbildung 8.3: Swoozy-Version mit Inhalten vom Saarländischen Rundfunk

Swoozy Globus TV

Eine spezielle Version von Swoozy wurde prototypisch für die SB-Warenhäuser des Handelskonzerns Globus[9] entwickelt und ist im Innovative Retail Laboratory (IRL)[10], einem Forschungslabor des Deutschen Forschungszentrums für Künstliche Intelligenz (DFKI), in der Zentrale der Globus SB-Warenhaus Holding in St. Wendel installiert (siehe Abbildung 8.4). Bei dieser Version wurde der Fokus auf das Szenario einer Online-Bestellung von Globus eigenen Waren direkt am Fernsehbildschirm gesetzt. Hierfür wurden zuerst gestreamte Videoin-

[9] http://www.globus.de
[10] http://www.innovative-retail.de

halte von Globus (intern als Globus TV bezeichnet) wie z.B. Werbefilme oder Reportagen über einzelne Abteilungen des Marktes (z.B. der Meisterbäckerei) innerhalb von Swoozy abgespielt.

Abbildung 8.4: Swoozy mit Globus-Inhalten. Quelle: www.dfki.de/web/living-labs-de/irl

Der Zuschauer kann per Gesteninteraktion die einzelnen Begriffe selektieren und erhält Zusatzinformationen zur Herstellung seiner Globus Lieblingsprodukte oder Hintergrundinformationen zu Gütesiegel und Qualitätsüberprüfungsmethoden innerhalb des Globus-Konzerns. Eine weitere innovative Funktionalität ist die direkte Verschmelzung von Swoozy in einer cyberphysischen Umgebung. Führt der Zuschauer eine Interaktion mittels der *Shop It!*-Funktion aus, werden nicht nur die vom Zuschauer per *Grab'n'Drop* selektierten Produkte bestellt, sondern die Bestellung wird mit einem sog. intelligenten Kühlschrank synchronisiert. Dieser SmartFridge [Hau13] ist ein Kühlschrank, der

über verschiedenste Sensoren in der Lage ist, alle Lebensmittel, die sich im Inneren des Kühlschranks befinden grafisch auf einem in die Kühlschranktür integrierten Bildschirm anzuzeigen.
Darüber hinaus überwacht der SmartFridge mittels seiner eingebauten RFID-Antennen, welche Produkte aus dem Kühlschrank entnommen werden und es kann bei Bedarf eine interaktive Einkaufsliste mit den fehlenden Artikeln für den Benutzer erstellt werden. Diese kann über verschiedene Endgeräte (Smartphones oder Tablets) bearbeitet werden. Diese Funktionalität des SmartFridge wird bei Swoozy benutzt, indem über eine ereignisgesteuerte Schnittstelle (*Event Broadcasting Service*) [Kah14] die einzelnen vom Benutzer vorher am Swoozy-TV-Bildschirm ausgewählten Waren direkt mit dem Kühlschranksystem synchronisiert werden. Validiert der Benutzer seine Einkaufsliste am Bildschirm des Kühlschranks, wird eine automatische Online-Bestellung initiiert.
Zusätzlich liefert dieses Szenario einen interessanten Einblick in crossmediale Interaktionen zwischen Fernsehen und dem Internet der Dinge, indem sich Gesteninteraktionen in Zusammenhang mit einem Video konkretisieren und wahlweise entweder zu weiteren Geräten propagiert werden oder in eine tatsächliche Warenbestellung münden.

Swoozy Live

Vorstellung

Die Implementierung der fünften Version von Swoozy startete im Sommer 2015 und integrierte das neue Paradigma des Cloud-TV-Ansatzes. Die technische Grundlage für diese Version wurde ständig erweitert und findet sich ebenso in der 2016er Version von Swoozy wieder (siehe

Abbildung 8.15). Die Herausforderung bei dieser letzten Version war es, verteilte und heterogene Dienste zu implementieren und zusammenzufügen, die trotzdem in der Lage sind, in Echtzeit eine Analyse des Fernsehsignals durchzuführen.

Abbildung 8.5: Swoozy mit Live-Grabbables

Diese Analyse wurde über dedizierte Komponenten realisiert (hierfür wurde eigens eine C#-basierte Komponente entwickelt), die eine OCR-Analyse des Bildes ermöglichen. Die Auswahl der Programmiersprache C# wurde dadurch motiviert, dass die erste Implementierung von Swoozy-Live, im Sommer 2015, das Live-TV Bild mit dem Framework Microsoft DirectShow[11] und einer DVB-T-Karte aufgezeichnet hat und eine OCR-Komponente mittels der .NET-basierten OpenCV-Wrapperbibliothek Emgu-CV[12] (siehe Kapitel 6) realisiert wurde, ohne dabei die Laufzeiten zu beeinträchtigen.

Bei Swoozy Live wird parallel zur Wiedergabe von vorannotierten Vi-

[11] `http://directshownet.sourceforge.net/about.html`
[12] `http://www.emgu.com`

deos und VOD das DVB-Fernsehsignal angezeigt. Hier muss jedoch differenziert werden, dass in der 2015er Version das Fernsehsignal von einer DVB-T Karte (siehe Kapitel 6) empfangen und als Stream zur grafischen Oberfläche weitergeleitet wurde.
In der 2016er Version dagegen wurde dieser Stream durch ein IP-TV und einen Sat-To-IP-basierten Videostream ersetzt. Somit können deutlich mehr Fernsehsender empfangen und das Szenario erweitert werden. Zusätzlich ermöglicht diese Variante eine technische Abkopplung zwischen dem Empfang bzw. der Wiedergabe des Videostreams und dem Rest der grafischen Benutzerschnittstellen. Diese klare Trennung war u.a. dadurch motiviert, dass die Benutzerschnittstelle von Swoozy auf ein Smart-TV-Gerät portiert werden sollte.

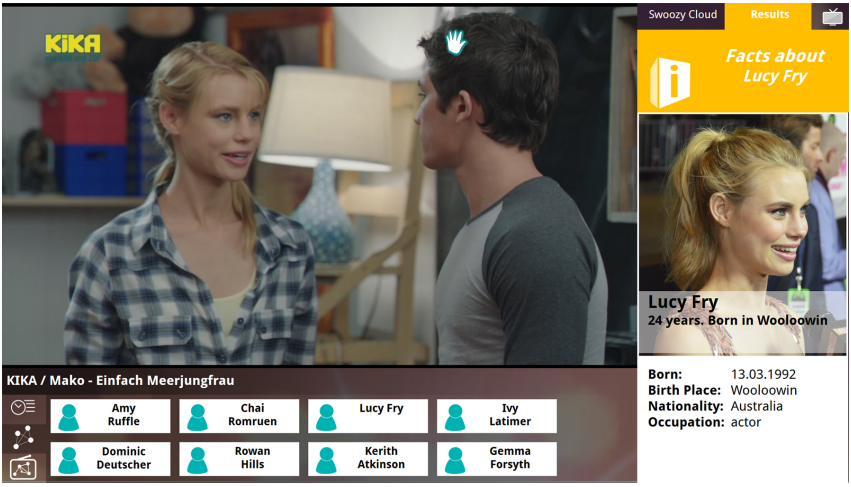


Abbildung 8.6: Beispiel einer Faktsuche nach der Schauspielerin Lucy Fry

Darauf aufbauend wird eine Echtzeitanalyse der aus dem DVB-Stream stammenden Daten durchgeführt. Hierfür dienen z.B. EPG-Texte als Eingabe für die Cloud-basierte textuelle semantische Analyse z.B. mit-

tels AYLIEN. In Zusammenarbeit mit Günter Neumann wurde eine neue, konfigurierbare Textanalysekomponente namens NEMEX als Dienst eingebunden und für die semantische Auswertung der EPG- und OCR-Texte in Echtzeit eingesetzt (siehe Kapitel 6).

Der generische Ansatz der semantischen Suche wurde bei der 2016er Version weiterentwickelt. Wikidata wird als primärer Informationskanal für die Suche benutzt und von zusätzlichen Diensten wie Bing oder Flickr erweitert. Unter Berücksichtigung der Aspekte aus Kapitel 7 wurden zwei Arten von *Grabbables* eingeführt: die *Live Grabbables*, die zeitgleich zu einem Geschehen im Bild direkt nach einer automatischen Erkennung auftauchen und die *Grabbables*, die während einer Sendung zeitlich länger eingeblendet bleiben. Abbildung 8.5 zeigt die grafische Ausprägung von *Live Grabbables* während einer Live-Moderation der Tour de France. Hier werden u.a. die Namen des Moderators (Michael Antwerpes) und des Sendungsproduzenten (Florian Naß) extrahiert und parallel dazu auch der Name der Stadt (Le Havre), von der die Etappe startet. Ein weiteres Beispiel wird in Abbildung 8.6 mit Eckdaten zu der Schauspielerin Lucy Fry dargestellt.

Parallel zu diesen Entwicklungen wurden die Software Swoozy-Livana und Swoozy-SKRPTR entwickelt, die eine redaktionelle Unterstützung für die Annotation und semantische Beschreibung der (Live-) Videoinhalte gewährleisten. Diese sind sogenannte hybride HTML5-Desktopapplikationen und können verteilt von einem oder mehreren Redakteuren betrieben werden. Die dadurch erzeugten Annotationen können persistent innerhalb einer Datenbank verteilt und persistent gespeichert werden (siehe Kapitel 7).

Technische Hintergrundinformationen zur Implementierung von Swoozy

In folgenden Abschnitten werden Hintergrunddetails über die Implementierung und die Tests gegeben, die durchgeführt wurden, um das Swoozy-System intern zu validieren und somit robuster zu gestalten. Dieser Abschnitt sollte auch begeisterten Lesern Anregungen und Idee für die Erstellung ihres eigenen semantischen Systems geben.
Der HTML-5 basierte Swoozy-Client besteht aus ca. 63.000 Zeilen Code, davon etwa 42.000 Zeilen Javascript-Quellcode, die unter 110 Dateien verteilt sind, 18 CSS-Dateien und 9 HTML-Dateien, die als sog. Templates benutzt werden. Zum Vergleich beträgt in der Swoozy 2014-Flash-Version die Anzahl der Actionscript-Dateien etwa 114 (insgesamt 14871 Zeilen Actionscript-Code). Diese sind für die Ansteuerung und Anzeige der grafischen Elemente zuständig. Hierbei darf jedoch nicht vergessen werden, dass Animationen unter Flash nicht nur per Quellcode instanziiert, sondern auch über eine .fla-Datei, die u.a. die Effekte wie Schlagschatten oder Farbverläufe über die dedizierte Entwicklungsumgebung (Adobe Flash bzw. Adobe Edge Animate[13]) definiert. Ähnliche Effekte könnten bei der 2016er Version von Swoozy mit CSS3 und externen Bibliotheken realisiert werden, jedoch mit einer Beeinträchtigung des Benutzererlebnisses.

Implementierung der grafischen Benutzeroberfläche

Die Abbildungen 8.7 und 8.8 stellen die aktuelle HTML5-basierte Benutzeroberfläche der 2016er Version des Swoozy-Systems dar. Eine besondere Herausforderung bei der Implementierung dieser Versi-

[13] `https://www.adobe.com/de/products/edge-animate.html`

on bestand in der Tatsache, dass die grafische Oberfläche von der Flash-Technologie auf die neuere HTML5 migrierte und somit neu implementiert wurde, vor allem wegen des Fokus auf eine höhere Interoperabilität und auf die zukünftige Portierung von Swoozy als eigenständige App für aktuelle Smart-TV Geräte.

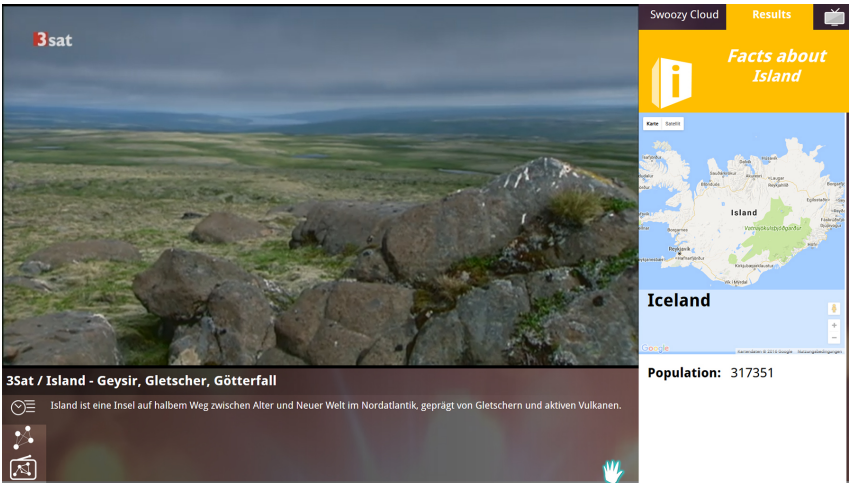


Abbildung 8.7: Swoozy Live mit Ergebnissen einer Faktsuche über Island

Die Neuimplementierung der grafischen Oberfläche weist jedoch auf die aktuell existierenden Unterschiede zwischen der Adobe Flash- und den HTML5-Technologien in punkto Entwicklung und Design des Benutzererlebnisses hin.

Bestimmte grafische Effekte, die Teil der Flash-basierten Version von Swoozy waren, mussten aus der HTML5-basierten Version aus Performanzgründen herausgenommen werden. Im Gegensatz zu Flash wird der Swoozy-Client nicht als eigenständige Applikation gestartet, sondern vom Webbrowser dargestellt. Dies führt dazu, dass je nach Browser, visuelle Effekte wie z.B. das Bouncing (dt. „Hüpfeffekt“, wenn

ein Element selektiert wurde) nicht flüssig genug dargestellt werden konnten und somit aus der HTML5-Version herausgenommen werden mussten. Die Flash-basierte Version benutzte sehr viele visuelle Effekte, um u.a. den Benutzern Feedback zu geben oder eine Aktivität des Systems (während einer Suche oder wenn ein *Grabbable* selektiert wurde) zu signalisieren. Auch wenn diese Effekte mittels CSS3 (mit animation und @keyframes)[14] oder Javascript-Bibliotheken wie Greensock[15] oder Transit[16] realisiert werden könnten, ist die Ausführung dieser nicht optimal und bietet kein zufriedenstellendes Benutzererlebnis an. Aus diesen Gründen mussten bei der 2016er Version von Swoozy sehr viele Abstriche in punkto Animation, grafischer Effekte und Hintergrundgrafik gemacht werden.

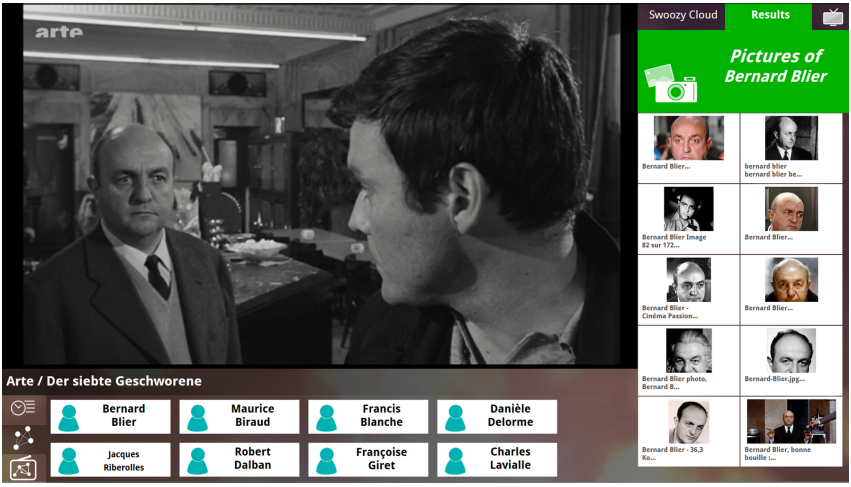


Abbildung 8.8: Beispiel der Einblendung der Ergebnisse der Bildersuche zum Schauspieler Bernard Blier

[14] http://www.w3schools.com/css/css3_animations.asp
[15] https://greensock.com
[16] http://ricostacruz.com/jquery.transit

Die erste 2015er HTML5-Version von Swoozy benutzte den VLC Plugin (application/x-vlc-plugin) als Videokomponente, um das gestreamte DVB-T-Signal wiederzugeben. Hier diente der Mozilla Firefox Webbrowser als Swoozy-Client. Es musste jedoch festgestellt werden, dass wenn ein Grabbable über den Videobereich gezogen wurde, bestimmte grafische Effekte wie die Transparenz oder der Schlagschatten (diese Effekte werden über CSS gesetzt) dessen verschwanden bzw. fehlerhaft dargestellt wurden. Diese Situation war auf die fehlerhafte Unterstützung des VLC-Plugins und seiner Überlappung durch CSS-gestaltete grafische Elemente seitens des Browsers zurückzuführen. In der 2016er Version wurde diese VLC-Komponente zugunsten der HTML5-Videokomponente[17] ersetzt. Diese ermöglicht eine Überlappung ihres Wiedergabebereichs durch CSS-gestaltete Elemente. Die HTML5-Videokomponente musste jedoch erweitert werden, damit sie sowohl MP4-Dateien als auch IPTV-basierte Streams abspielen kann (Stichwort: HLS-HTTP Live Streaming[18]). Bei der Flash-Technologie wären noch ein Zwischenserver bzw. Enkoder (in unserem Fall der Adobe Media Server[19] und ein Encoder[20]), der das Video über das RTMP-Protokoll[21] transportiert und automatisch die Videos für den Flash-Client (hier den Swoozy-Client) in das richtige Format (meistens F4V oder FLV) konvertiert, nötig gewesen. Während der Implementierung der 2015er und 2016er Version von Swoozy musste festgestellt werden, dass durch die gleichzeitige und parallele Benutzung mehrerer HTML5-Videokomponenten die Performanz vom Webbrowser sank

[17] `https://www.w3.org/wiki/HTML/Elements/video`
[18] `https://developer.apple.com/streaming`
[19] `http://www.adobe.com/fr/products/adobe-media-server-professional.html`
[20] `http://www.adobe.com/fr/products/media-encoder.html`
[21] `https://helpx.adobe.com/adobe-media-server/dev/stream-on-demand-media-rtmp.html`

und die Reaktivität des Systems nicht mit der Flash-Version vergleichbar war, insbesondere bei der Rückgabe der Ergebnisse der Videosuche. Um dieses Problem zu beheben, mussten z.B. die verkleinerten Videoergebnisse als GIF-Dateien vom Swoozy-Server konvertiert und zurückgegeben werden oder bestimmte Animationen vereinfacht werden.

Semantische Extraktion

Die Erzeugung der *Grabbables* steht innerhalb der vorgestellten Versionen von Swoozy im Mittelpunkt der semantischen Verarbeitung. Ihre Generierung durch die Analyse des Videobildes, z.B. durch OCR oder eine cloud-basierte textuelle Analyse, hängt stark von der Qualität der Erkennung ab. Entlang der Implementierung des Swoozy-Systems und insbesondere bei der Swoozy Live-Version dienten die Kriterien wie Schnelligkeit der Erkennung, Synchronisation der *Grabbables*-Anzeige und Korrektheit der erkannten Grabbables als Messwerte für die Auswahl bestimmter Dienste und Extraktionskomponenten. Zusätzlich, um bestimmte während der Implementierung gesammelte Erkenntnisse zu verifizieren (z.B. dass OCR-Zonen deutlich bessere und schnellere Ergebnisse als eine Analyse des kompletten Bildes liefern), wurden Messungen durchgeführt, die diese Einschätzung belegen (siehe Test Nr. 1).

Zusätzlich wird im folgenden Abschnitt die textuelle Extraktionskomponente von Swoozy näher betrachtet (siehe Test 2). Ziel dieses Tests ist es zu zeigen, welche der einzelnen Dienste eine höhere Anzahl von *Grabbables* zurückgeben und somit zu belegen, dass die aktuell aus dem EPG extrahierten Informationen eine akkurate Erzeugung von Grabbables gewährleisten können. Alle hier durchgeführten Tests wur-

den auf einem PC mit folgender Konfiguration durchgeführt: Prozessor Intel Core i7 5820K CPU 3.30Ghz mit 32 GB RAM und einer NVIDIA GeForce GTX 960 Grafikkarte. Um den Test 1 durchzuführen, wurde eine C# Komponente implementiert, die parallel zur Videowiedergabe die Rechenzeiten jedes einzelnen Tesseract OCR-Aufruf mitprotokolliert.

Test 1: OCR-Analyse mit/ohne OCR-Zone zur Erzeugung der Grabbables

Bei diesem ersten Test werden die Präzision und die Zeit gemessen, die von der Swoozy OCR-Komponente benötigt werden, um einen Text innerhalb eines Einzelbilds zu erkennen. Unter Text verstehen wir eine Reihenfolge von Buchstaben oder Zeichen, die innerhalb eines Wörterbuches oder Korpus zu finden ist und somit als Eingabe für eine weitere Komponente dient. Eine Reihenfolge von willkürlichen Symbolen oder Zeichen wird nicht als Text bewertet und wird auch nicht als Ergebnis zurückgegeben. Das Ergebnis der Swoozy OCR-Komponente bildet eine Zeichenkette, die als Eingabe für eine Textanalysekomponente (siehe Kapitel 6) benutzt werden kann, um daraus automatisch ein *Grabbable* zu generieren.

Vorgehensweise

Die in diesem Abschnitt vorgestellten Zeitmessungen werden mit folgender Methodik durchgeführt:

- Ein Video mit einer Auflösung von (1440 x 810 Pixel) wird parallel zu der OCR-Erkennung gestartet.
- Eine Liste der potentiell auffindbaren *Grabbables*, d.h. *Grabbables*, die im Videoabschnitt gefunden werden sollten, wird im Vorfeld

angelegt.

- Die OCR-Komponente wird mit einem Intervall von 200ms, 500ms und 1s die OCR-Erkennung durchführen. Bei jedem Analyseintervall werden folgende Parameter überprüft und mitprotokoliert:
  - Zeit, die für die OCR-Analyse mit bzw. ohne Zone benötigt wurde. Bei der OCR-Zonenbasierten Analyse werden pro Intervall und für jede Zone parallele Threads gestartet, d.h. wenn zwei Zonen während eines Intervalls analysiert werden müssen, werden zwei parallele Threads zeitgleich gestartet.
  - Gesamtanzahl der erkannten Grabbables innerhalb des Videoabschnitts. Hierbei wird die im Vorfeld definierte Grabbable-Liste mit der durch die OCR-Analyse gewonnenen Liste verglichen.

Konfiguration und Videomaterial

Zur Ausführung der Vermessung der OCR-Komponente wurde die Standardversion von Tesseract genommen, die innerhalb der Bibliothek EmguCV (siehe Kapitel 3) Version 3.1 eingebunden ist. Es wurden bei der Benutzung des Tesseract-Moduls keine zusätzlichen Trainingsdaten oder vortrainierten Schriftarten benutzt, sodass erkannte Zeichenketten ungefiltert von der OCR-Komponente zurückgegeben werden. Als Eingabe-Set wurden drei unterschiedliche Ausschnitte aus Fernsehsendungen benutzt. Diese Sendungen wurde so ausgewählt, dass in einem ersten Schritt die Schnelligkeit der OCR-Komponente und in einem zweiten Schritt die Korrektheit der zurückgelieferten erkannten Texte berechnet werden können. Zusätzlich ist die Wahl

der Sendung so erfolgt, dass diese Sendungen teils sehr unterschiedliche Einblendungstypen wie z.B. kurze Infographie, Einblendungsgrafik (engl. Overlay) mit Moderator-, Sportler- oder Ortsnamen, Themeneinblendungen mit größerer Schriftart besitzen. In Kapitel 6 wurde die Entscheidung getroffen, OCR-Zonen zu benutzen, um die Schnelligkeit der OCR-Erkennung zu erhöhen. Um dies besser bewerten zu können, werden bei den jeweiligen Videos zwei OCR-Analysenverfahren benutzt:

- **Mit vordefinierten OCR-Zonen**. Die zu analysierenden Bereiche werden durch festgelegte OCR-Zonen an der genauen Position einer möglichen Einblendung innerhalb des Videos bestimmt.
- **Ohne OCR-Zonen bzw. globale Analyse**. In diesem Fall wird die Analyse über den kompletten Videobildbereich (1440 x 810 Pixel) durchgeführt, ohne jegliche OCR-Zonen zu berücksichtigen. Dadurch kann möglicherweise Rauschen entstehen.
  Die hochskalierte Videoauflösung von 1440 x 810 Pixel entspricht den Dimensionen des Live-Bildbereichs innerhalb der Swoozy-Benutzerschnittstelle. Die native Auflösung der dargestellten und wiedergegebenen Testvideos beträgt 1280 x 720 Pixel.

Im folgenden Abschnitt werden drei Videos und die Ergebnisse der OCR-Analysezeiten tabellarisch dargestellt. Die Trefferquote gibt die Anzahl der *String-identischen* Treffer ohne Nachverarbeitung wieder. Das bedeutet: ein Treffer gilt nur, wenn die OCR-Komponente die gesuchte Zeichenkette 1:1 wiederfindet. Da das Tesseract-Modul nach der Bildanalyse Rauschen produziert, d.h. Zeichenketten oder Pixelbereiche, die eigentlich keinen Text enthalten, sondern nur willkürliche ASCII-Zeichen, werden die letzteren gefiltert und nicht berücksichtigt. Eine zusätzliche Konsolidierungskomponente mit einer Zeichenketten-

basierten Überprüfung der Distanz zwischen dem erwarteten und dem erkannten Text, z.B. mittels einer Berechnung der Levenshtein-Distanz[22], kann dieses Problem lösen und somit die Trefferquote erhöhen.

Video 1: Olympia / Sportschau: 200m

Das erste Video ist ein kurzer Ausschnitt aus der Sendung ARD-Sportschau des 100m Sprintfinales der Olympia 2016 in Rio (Quelle: Sportschau YouTube[23], Dauer des analysierten Ausschnitts: 60 Sekunden). Hier werden verschiedene Läufer in der Startaufstellung vorgestellt. Diese Vorstellung wird während der Ausstrahlung mit einer grafischen Einblendung des Namens des Läufers und seines Landeskürzels unterstützt.
Bei diesem Video wurden 13 Grabbables (d.h. Textblöcke, die in einem weiteren Schritt von der semantischen Suche bearbeitet werden können) vordefiniert:

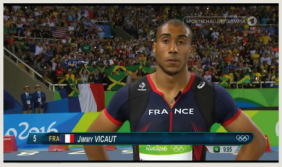
- Einerseits die Namen der Läufer: *Akani Simine*, *Andre de Grasse*, *Usain Bolt*, *Jimmy Vicaut*, *Justin Gatlin*, *Ben Youssef Meite*, *Yohan Blake* und andererseits die Länderkürzel: *JAM*, *FRA*, *CIV*, *CAN*, *USA* und *RSA*.
- Und zwei Zonen von jeweils 340x90 Pixeln (Zone 1) und 100x90 Pixeln (Zone 2) (siehe Abbildung 8.9) wurden festgelegt.

Folgende Tabelle (siehe Abbildung 8.9) fasst die unterschiedlichen Zeiten zwischen einer OCR-Analyse mit bzw. ohne OCR-Zonen zusammen. Die Abbildung 8.9 zeigt anhand einer Zeitachse, wie die

[22] http://www.levenshtein.de
[23] https://youtu.be/avTlP5hnxx4

OCR-Analysezeiten verteilt sind. Hierbei lässt sich sehr gut darstellen, dass sich die OCR-Analysezeiten deutlich erhöhen, wenn textuelle Einblendungen innerhalb des Bildes erkannt werden müssen.

Trotz präziser definierten OCR-Zonen zeigen die Ergebnisse, dass bestimmte Namen nicht korrekt erkannt werden. Dies ist z.B. der Fall bei den Athletennamen *Justin Gatlin* oder *Andre de Grasse*, die nur mit Rauschen von der OCR-Komponente erkannt werden (hier werden die Namen *Jus Tin Gatlin* und *Andre Grasse* von der Komponente zurückgegeben). Eine Anpassung der Namen, z.B. durch eine im Vorfeld festgelegte Liste der Läufernamen, würde diese Ungenauigkeit aufheben.

	OCR-Analyse Intervalle		
	1000 ms	**500 ms**	**200 ms**
 **Zone 1** Sportlername	**45,03 ms** Akani Simine, Justin Gatlin Usain Bolt, Ben Youssef Meite, Yohan Blake	**37,44 ms** Akani Simine, Jimmy Vicaut, Usain Bolt, Ben Youssef Meite, Yohan Blake,	**44,80 ms** Akani Simine, Justin Gatlin, Jimmy Vicaut, Usain Bolt, Ben Youssef Meite, Yohan Blake,
 **Zone 2** Landname als Kürzel	**7,24 ms** RSA, USA, FRA, JAM, CIV, CAN	**3,95 ms** RSA, USA, FRA, JAM, CAN, CIV	**4,32 ms** RSA, USA, FRA, JAM, CAN, CIV
Trefferquote mit OCR-Zonen	**11/13**	**11/13**	**12/13**
	**447,33 ms** Usain Bolt, FRA, USA	**542 ms*** Jimmy Vicaut, Usain Bolt, Ben Youssef Meite, FRA, USA, JAM	**438 ms*** Jimmy Vicaut, Usain Bolt, Ben Youssef Meite, FRA, USA, CAN, CIV JAM
Trefferquote ohne OCR-Zonen	**3/13**	**6/13**	**8/13**

*punktuelle Verlangsamung der OCR-Analyse - siehe Seite 337

Abbildung 8.9: Ergebnisse der Messungen der Zeit für die OCR-Analyse des Videos 1

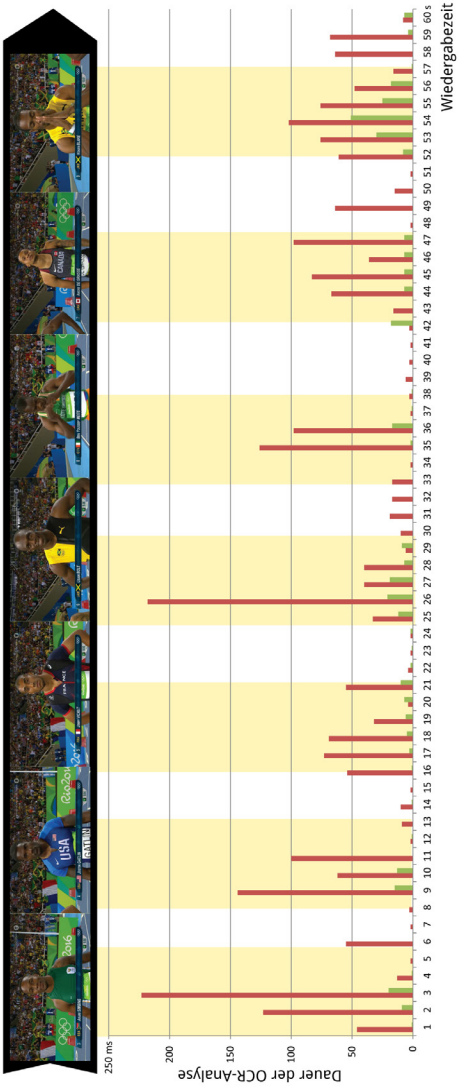


Abbildung 8.10: Korrelation zwischen der Dauer der OCR-Analyse und dem Zeitpunkt der Einblendung der Namen

Bei den durchgeführten Tests ist aufgefallen, dass ein Analyseintervall von 200ms dazu führt, insbesondere wegen der hohen Analyserate und des dadurch entstandenen mangelnden Arbeitsspeichers, dass die Videowiedergabe stark beeinträchtigt wird. Konkret bedeutet dies, dass das Video nicht immer mit der gleichen Bildrate wiedergegeben werden kann und sog. „Ruckler" oder „Hänger" entstehen, welche die Synchronisation mit der OCR-Analysekomponente erschweren. Das bedeutet eine Ungenauigkeit und eine punktuelle Verlangsamung der OCR-Analyse. Dies führt dazu, dass die Messung der Zeit ungenau ist. Aus diesem Grund wurden in Abbildung 8.9 die Zeitwerte mit einem Stern gekennzeichnet.
Die Ergebnisse der Messung zeigen jedoch eindeutig, dass bei diesem Video eine OCR-Zonen-basierte Analyse schneller durchgeführt werden kann als die Analyse des Gesamtbildbereichs. Eine weitere Erkenntnis dieser Analyse ist, dass die Zeiten für die OCR-Erkennung unter 45 ms für die Erkennung des Sportlernamens und unter 7,5 ms für die Erkennung der Länderabkürzungen liegen.
Die Einblendezeiten der Grafik durch die Fernsehregie beträgt im Durchschnitt 2 Sekunden (mit Fade-In/Out Effekt). Wäre das OCR-Analyse-Intervall höher als 1000ms, bestünde die Gefahr, dass die OCR-Komponente die Einblendungen „verpasst" und somit nicht alle dargestellten Texte analysiert werden.

Video 2: ARD Tagesschau vom 25.08.2016

Die ARD Tagesschau-Sendung[24] (Dauer des Videoausschnittes für die Analyse: 2:30 Minuten) wurde ausgewählt, weil Einblendungen in Nachrichtensendungen Kontextinformationen über das gerade angespro-

[24] https://www.tagesschau.de/archiv/sendungsarchiv100.html

chene Thema geben. Zusätzlich bilden die unterschiedlichen Größen der beiden Zonen und der darin enthaltenen Texte eine für die Messungen interessante Variante, insbesondere um festzustellen, ob sich die Analysezeiten je nach Textgröße und Länge verändern werden. Die ARD benutzt halbtransparente grafische Einblendungen innerhalb dieser Sendung, um die Themen und die Moderatoren bzw. Namen der dargestellten Personen hervorzuheben.
Bei diesem Video wurden 7 Grabbables vordefiniert:

- *Thorsten Schröder* (Name des Moderators), *Italien*, *Lorenzo Botti* (Interviewter 1), *Antonio Ragonesi* (Interviewter 2), *Richard Schneider*(Reporter), *Ellen Trapp* (Name der Reporterin), *Amatrice* (Stadt in Italien).
- Zwei OCR-Zonen von jeweils 704x214 Pixeln (Zone 1) und 430x90 Pixeln (Zone 2) (siehe Abbildung 8.11) wurden vordefiniert.

Die Trefferquote mit und ohne OCR-Zonen liefert ähnliche Werte. Einen leichten Vorteil weist die OCR-Zonen basierte Erkennung auf, die im Durschnitt 50ms schneller ist. Die Tatsache, dass nur 6 von 7 Grabbables bei der OCR-Zonenanalyse gefunden wurden, lässt sich dadurch erklären, dass der Stadtname *Amatrice* vom Tesseract-Modul als *Amatricb* zurückgegeben und somit als nicht 1:1 passend bewertet wurde. Ähnlich dem Video 1 erzeugt hier ein hohes OCR-Analyseintervall von 200ms zwar eine gute Trefferquote, jedoch wird die Videowiedergabe beeinträchtigt. Aus diesem Grund sind die Werte in Klammern gesetzt worden.
Die Messungen zeigen, abgesehen von der Fehlerkennung des Namens *Amatrice*, dass ein höheres Analyseintervall keine Auswirkung auf die Trefferquote hat.

Dies lässt sich dadurch erklären, dass die Einblendungen innerhalb der Tagesschau-Sendung im Durschnitt 6 Sekunden dauern und somit die Wahrscheinlichkeit, dass Texte von der OCR-Analyse nicht erfasst werden, nicht auftreten kann.

	OCR-Analyse Intervalle		
	1000 ms	**500 ms**	**200 ms**
 **Zone 1** Thematik des Beitrags	**397,13 ms** Italien	**414,83 ms** Italien	**(472 ms)** Italien
 **Zone 2** Personenname	**394,05 ms** Thorsten Schröder, Lorenzo Botti, Richard Schneider Antonio Ragonesi Ellen Trapp	**427,48 ms** Thorsten Schröder, Lorenzo Botti, Richard Schneider Antonio Ragonesi, Ellen Trapp	**(445 ms)** Thorsten Schröder, Lorenzo Botti, Richard Schneider Antonio Ragonesi, Ellen Trapp Amatrice
Trefferquote mit OCR-Zonen	**6/7**	**6/7**	**7/7**
	**485 ms** Thorsten Schröder, Lorenzo Botti, Richard Schneider Antonio Ragonesi Italien Amatrice Ellen Trapp	**493,37 ms** Thorsten Schröder, Lorenzo Botti, Richard Schneider Antonio Ragonesi Italien Amatrice Ellen Trapp	**(516,81 ms)** Thorsten Schröder, Italien Richard Schneider Lorenzo Botti Antonio Ragonesi Ellen Trapp Amatrice
Trefferquote ohne OCR-Zonen	**7/7**	**7/7**	**7/7**

Abbildung 8.11: Ergebnisse der Messungen der Zeit für die OCR-Analyse des Videos 2

Die Tabelle zeigt zusätzlich im Falle der Tagesschau-Sendung, dass

ein OCR-Analyseintervall von 1000ms ein guter Kompromiss zwischen Schnelligkeit und Präzision der Erkennung darstellt.

Video 3: Tagesschau in 100 Sekunden

Die Tagesschau in 100 Sekunden wurde aus dem Grund ausgewählt, da die Themen mit einer sehr großen Schriftart über 2/3 des Bildbereiches eingeblendet werden. Zudem überlagern die Texte teils Standbilder, teils Videobeiträge, welche die angesprochene Thematik visuell unterstützen. Hier steht die Frage, ob die OCR-Analyse des gesamten Bildbereichs genauso gute Ergebnisse liefert wie die OCR-Zonenbasierte Erkennung im Mittelpunkt.
Bei diesem Video wurden 11 Grabbables vordefiniert, wobei das Grabbable *Kristen Gerhard* nur von der globalen OCR-Analyse erkannt werden kann.

- *Kristen Gerhard* (Moderatorin), *Grüne* (als Organisation), *ARD*, *Merkel* (Person), *Kolumbien* (Land), *Frankreich* (Land), *Cazeneuve* (Person), *Burkini* (Objekt), *NASA* (als Organisation), *Mars* (Ort) und *Hawaii* (Land).
- Eine einzige OCR-Zone von 1200x160 Pixeln (siehe Abbildung 8.12) wurde vordefiniert.

Die Ergebnisse zeigen, dass die Trefferquote mit OCR-Zonen, unabhängig vom OCR-Analyseintervall, perfekt ist. Alle Grabbables wurden unter einer Analysezeit von 170ms gefunden. Bei der Analyse ohne OCR-Zonen wurden jedoch die Grabbables *Cazeneuve, Burkini* und *Frankreich* nicht erkannt.
Dies lässt sich auf den kontrastierten Hintergrund des Gesamtbildbereichs zurückführen. Die Einblendungen der Sendung *Tagesschau*

in 100 Sekunden dauern im Durchschnitt 20 Sekunden. Hier würde gar noch ein niedrigeres OCR-Analyseintervall (z.B. alle 5 Sekunden) gute Ergebnisse liefern.

	OCR-Analyse Intervalle		
	1000 ms	**500 ms**	**200 ms**
 **Zone 1** Thematik	**167,37 ms** Grüne, ARD, Merkel, Kolumbien, Frankreich, Cazeneuve, Burkini, Mars, Hawaii, NASA	**152,60 ms** Grüne, ARD, Merkel, Kolumbien, Frankreich, Cazeneuve, Burkini, Mars, Hawaii, NASA	**160,98 ms** Grüne, ARD, Merkel, Kolumbien, Frankreich, Cazeneuve, Burkini, Mars, Hawaii, NASA
Trefferquote mit OCR-Zonen	**10/10**	**10/10**	**10/10**
	**478,46 ms** Kirsten Gerhard, ARD,Grüne, Merkel, Kolumbien, Mars, Hawaii, NASA	**509,23ms** Kirsten Gerhard, ARD,Grüne, Merkel, Kolumbien, Mars, Hawaii, NASA	**469,91 ms** Kirsten Gerhard, ARD,Grüne, Merkel, Kolumbien, Mars, Hawaii, NASA
Trefferquote ohne OCR-Zonen	**8/11**	**8/11**	**8/11**

Abbildung 8.12: Ergebnisse der Messungen der Zeit für die OCR-Analyse des Videos 3

Fazit

Die Zeitmessungen zeigen, dass insgesamt das Vorhaben einer OCR-Erkennung mit festgelegten Zonen schneller durchgeführt werden kann als eine Analyse des kompletten Bildes. Im Schnitt dauert die OCR-Analyse unter 1 Sekunde. Die Auswahl eines zu hohen Intervalls, z.B. unter 200ms, führt auf dem Zielrechner wegen des Speichers

zu einer Desynchronisation zwischen der Videowiedergabe und der OCR-Analyse. Da jedoch die grafischen Einblendungen der Fernsehsender deutlich länger als 1 Sekunde angezeigt werden (im Video 1: 2 Sekunden, bei Video 2: 6 Sekunden und im Video 3: 20 Sekunden pro Einblendung), wäre ein zu hohes Intervall für die OCR-Erkennung nicht sinnvoll. Ein niedrigeres Intervall (z.B. 2000ms) wäre zwar akzeptabel, jedoch besteht die Gefahr, dass bei schnellen textuellen Einblendungen (z.B. bei Laufbändern von Nachrichtensendern wie bei dem Fernsehsender N24) bestimmte Einzelbilder von der OCR-Analyse verpasst und somit nur partielle passende Grabbables generiert werden. Eine weitere Erkenntnis ist, dass je grösser eine definierte OCR-Zone ist, desto mehr Zeit braucht die OCR-Komponente, um eine Analyse durchzuführen. Dieser Fakt muss bei der Vordefinition der Zonen berücksichtigt werden. Zusätzlich beweisen die Messungen, dass der in dieser Arbeit vorgestellte Ansatz der OCR-Zonenbasierten Analyse realistisch und für eine echtzeitbasierte semantische textuelle Extraktion von Informationen aus dem Fernsehbild anwendbar und präzise genug ist.

Test 2: Semantische textuelle Analyse der EPG-Texte

Vorgehensweise

Der zweite Test befasst sich mit dem Erkennungserfolg der textbasierten semantischen Analyse. Hierbei werden beispielhaft 8 EPG-Texte aus unterschiedlichen deutschsprachigen Fernsehsendungen von den Cloud-basierten Entitäten- bzw. Konzeptextraktionskomponenten AYLIEN, Microsoft Cognitive Service, Alchemy, Stanford NER und NEMEX (siehe Kapitel 3 und 6) kombiniert analysiert. Der Stanford Named Entity Tagger (siehe Kapitel 3 und 6) wurde mit dem Klassifizierer

german.hgc_175m_600.crf benutzt. Der Microsoft Cognitive Service benutzt zurzeit keinen deutschen Korpus, aber bildet die Ergebnisse auf Wikipedia-Einträgen ab. Die aus dem EPG stammenden Texte sind teils sehr ausführlich mit einer kompletten Beschreibung des Filmes samt Schauspieler, teils sehr kompakt und geben sehr wenig Information über den Inhalt und die Einzelheiten der ausgestrahlten Sendung. Aus Entwicklersicht ist es wichtig zu erfahren, welche der fünf Komponenten, die innerhalb von Swoozy kombiniert integriert und benutzt werden, die meisten Entitäten bzw. *Grabbables* aus den EPG-Texten zurückgibt.

In Tabelle (8.14) werden die Ergebnisse zusammengefasst, d.h. die gefundenen Entitäten der jeweiligen Dienste werden aufgelistet, bewertet und überprüft, um festzustellen, ob die erkannten Konzepte korrekt und passend sind.

Die erkannten Konzepte oder Typen werden durch Klammern signalisiert und folgen folgender Nomenklatur: *LOC*: Ort oder Stadt, *LOCderiv*: Derivat von Ort oder Stadt, *PER*: Person, *ORG*: Organisation, Verein oder Unternehmen, *MISC*: Konzept als Erweiterung einer existierenden Entität (z.B. „hessischen" stammt aus der Entität „Hessen"). Folgende EPG-Texte aus unterschiedlichen Sendungstypen und Fernsehsendern wurden für diese Analyse verwendet:

- **Text 1: ZDF – Soko Wien**
  Die SOKO steht vor einem Rätsel. Am Donauufer wurde eine Frauenleiche gefunden. Zwei Tage später wird dort erneut eine Frau entdeckt, die zwar lebt, aber ihr Gedächtnis verloren hat. Weder bei der Toten noch bei der zweiten Frau wurden Verletzungen oder Hinweise gefunden, die auf die Todesursache und auf den Grund für die Amnesie schließen lassen. Dann meldet

der Gynäkologe Dr. Wagner seine Frau als vermisst. Ist es die Frau ohne Gedächtnis? Carl Ribarski deutet den Zustand der Frau ohne Gedächtnis als posttraumatisch. Sein Kollege Helmuth Nowak hingegen glaubt, dass die Frau den Ermittlern etwas vorspielt. Gleichzeitig wurde Wagners neuer Sportwagen gestohlen. Im Zuge der Ermittlungen stößt das Team der SOKO Wien im Internetportal auf die Partnervermittlung „Zweiter Frühling". In diesem Zusammenhang taucht immer wieder der Name „Mr. M" auf. Gleichzeitig findet Penny Lanz heraus, dass das Auto von Dr. Wagner von dem Kleinganoven Loschek gestohlen wurde. Als Carl Ribarski und Helmuth Nowak Loschek verhören, machen sie eine interessante Entdeckung.

- **Text 2: RTL - sternTV** Streit um Religionsfreiheit: Braucht Deutschland ein Burka-Verbot? Eltern der getöteten Anneli äußern sich zum Urteil: "Gemessen an der Tat ist das zu wenig"; Olympiasieger Fabian Hambüchen besucht seine Fans: Wie gut turnen die Deutschen? (RTL – stern TV)

- **Text 3: ZDF Neo - Columbo**
  Columbo. Des Teufels Corporal. Colonel Rumford ist entsetzt, als sein Vorgesetzter, William Haynes, die Militärschule zu einem gemischtgeschlechtlichen College umfunktionieren will. Ein „Unfall" soll Haynes beseitigen. Die polizeilichen Untersuchungen ergeben, dass es sich um einen Unfall handeln muss, der durch ein Tuch im Kanonenrohr verursacht worden ist. Aber Inspektor Columbo vermutet vorsätzlichen Mord. Der teuflische Colonel Rumford wird von Patrick McGoohan gespielt, der für seine Rolle als Bösewicht mit dem Emmy ausgezeichnet wurde.

- **Text 4: ARTE - In der Hängematte auf dem Amazonas**
  Amazonien, das ist undurchdringlicher tropischer Regenwald, durchzogen vom längsten Fluss der Welt. Städte gibt es nur wenige, Straßen kaum. Wer hier reisen muss, kann das nur per Schiff. „Recreios" heißen die Passagierboote, die gleichzeitig auch Frachtkähne sind. „Recreio" bedeutet „Pause", „Auszeit". Die Passagiere bringen ihre eigene Hängematte mit und begeben sich auf eine Reise, die in der Großstadt Manaus beginnt und in den Weiten der Amazonaslandschaft endet.

- **Text 5: Eurosport 1 - Motorrad: MotoGP 2016 – Großer Preis von San Marino in Misano(ITA)**
  Ausblick auf die 13. von 18 Saisonstationen. Auch die 68. Saison der MotoGP bringt die besten motorisierten Zweiradfahrer der Welt wieder rund um den Globus. Sie umfasst wie gewohnt die drei Klassen MotoGP, Moto2 und Moto3. Der Rennkalender beinhaltet wieder 18 bekannte Circuits und schickt die Fahrer nach Asien, Europa, Australien, sowie Süd- und Nordamerika. Ein Blick auf die abgelaufene Saison verspricht Spannung, denn der WM-Kampf 2015 hatte es in sich: Die Saison hielt einige aufregende Rennen bereit. Der Kampf um den WM-Titel zwischen Valentino Rossi und Jorge Lorenzo blieb bis zum letzten Duell in Valencia offen.

- **Text 6: ARD - ARD-Buffet**
  Heimatküche: Forelle Müllerin mit Salzkartoffeln und Kopfsalat, zubereitet von Sören Anders; Zuschauerfragen zum Thema: Badepralinen für trockene Haut nach dem Sommer; Gute Idee: Getränkehalter zum Wandern; Mein neues Leben: Gabriele Braun: Von der Topmanagerin zur Schuhmacherin; Seifen für Männer;

Fisch aus Saalfeld; Bienenparadies Waldökologie; Wandern in der hessischen Rhön; Quiz. Moderation: Holger Wienpahl.

- **Text 7: SAT.1-Frühstücksfernsehen**
  Gäste: Vanessa Blumhagen, Peer Kusmagk. Moderation: Matthias Killing, Karen Heinrichs.
- **Text 8: Sport 1 – Handball Live**
  Die DKB Handball-Bundesliga Handball Live.

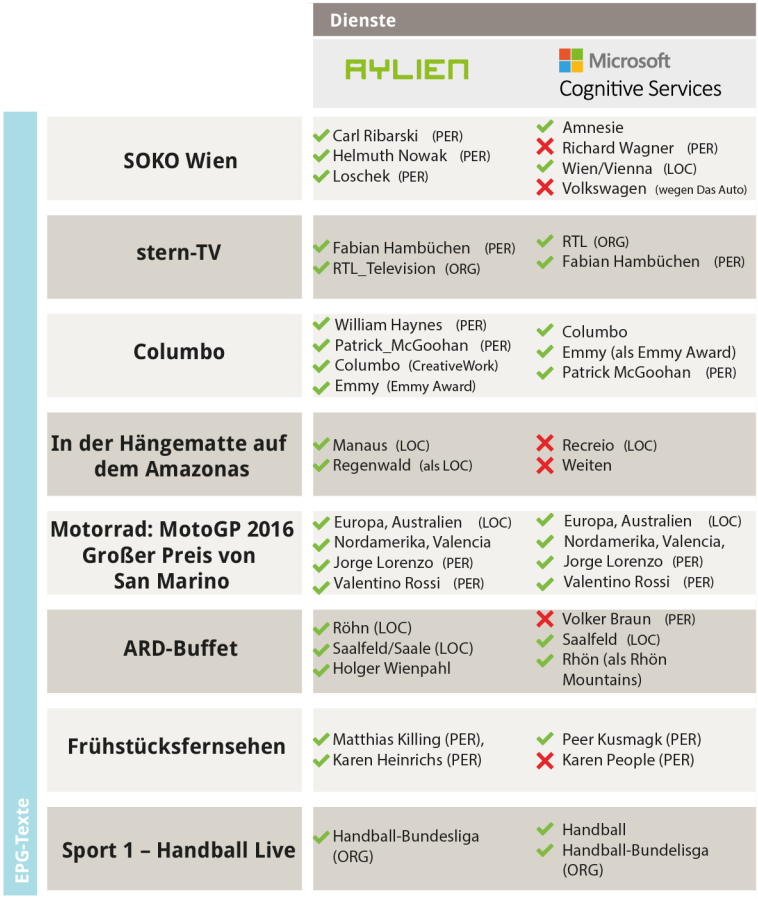


Abbildung 8.13: Übersicht der Ergebnisse der Extraktion der EPG-Texte (1/2)

Abbildung 8.14: Übersicht der Ergebnisse der Extraktion der EPG-Texte (2/2)

Fazit

Die Cloud- und offline-basierten Dienste für die Analyse der Texte zeigen, dass die extrahierbare Informationsmenge sehr unterschiedlich sein kann. Bei dem Dienst Microsoft Cognitive Service werden Fehlinterpretationen der Entitäten ausgelöst wie z.B. bei dem Namen *Wagner*, der auf den Musiker *Richard Wagner* auf Wikipedia referenziert. Der Text „*das Auto*" wird zu der Marke Volkswagen verbunden, womöglich weil dies der Slogan der Automarke ist. Die Erklärung für solche Fehlinterpretationen seitens des Microsoft Cognitive Service Dienstes liegt darin, dass die Entitätenextraktion zur Zeit nur für die englische Sprache verfügbar ist und sich auf englische Wikipedia-Einträge bezieht. Beim Text Nr. 2 (stern-TV) und Text Nr. 6 liefert die NEMEX-Komponente die meisten korrekten Ergebnisse. Die Alchemy API fügt zusätzliche andere Konzepte wie *HealthCondition* ein und erkennt als einzige den Begriff „Amnesie" als Gesundheitskonzept, wobei Microsoft Cognitive Services nur auf den Wikipedia-Eintrag verweist. Der Text Nr. 7 (SAT1. - Frühstücksfernsehen) ist sehr kompakt und erzeugt ein Problem bei der automatischen Erkennung der Sprache durch den Alchemy-Dienst. Hier kann, vermutlich wegen eines zu sperrigen textuellen Inhalts, die Sprache „Deutsch" nicht ermittelt werden und somit scheitert die semantische Analyse.

Eine weitere Feststellung ist, dass der Name *Vanessa Blumenhagen* entweder nicht erkannt wird oder fälschlicherweise als Organisation (wie z.B. beim Stanford NER-Dienst) oder Ort klassifiziert wird, obwohl eine Wikipedia-Seite über die Moderatorin existiert. Hier könnte die Benutzung eines angepassten und aktualisierten Korpus von bekannten deutschen TV-Moderatoren die Ungenauigkeit beheben. Zusammenfassend kann behauptet werden, dass eine kombinierte semantische

Extraktion von Texten, wie sie in Swoozy integriert und implementiert wurde, einen sinnvollen Ansatz zur Echtzeitgenerierung der Grabbables bildet.

Swoozy-Versionen

Die Tabelle in Abbildung 8.15 fasst die einzelnen Eigenschaften der jeweiligen Swoozy-Versionen zusammen und stellt die Besonderheiten sowie die benutzten Werkzeuge bzw. Technologien dar. Das nächste Kapitel fasst die Beiträge dieser Arbeit zusammen und bietet einen Überblick über wissenschaftliche Erkenntnisse, die mit der technischen Realisierung des Swoozy-Systems gewonnen wurden. Anschließend wird ein Ausblick auf mögliche zukünftige Erweiterungen des Systems gegeben.

Swoozy Versionen	Programmierung		Semantische Bearbeitung/Erkennung			
	Swoozy Client	Swoozy Server	Live-Erkennung	Cloud-basierte Extraktions-komponenten	Mediathek/ Video Anbindung	Semantische Dienste/ Web 3.0
Swoozy 2013	Adobe AIR	Java über JSE [1]	✗ nicht zutreffend	nur lokal über Annotationen	✓ lokale Videos	✓
Swoozy 2014	Adobe AIR	Java über JSE [1]	✗ nicht zutreffend	nur lokal über Annotationen	✓ lokale Videos	✓
Swoozy Globus TV	Adobe AIR	Node.js	✗ nicht zutreffend	nur lokal über Annotationen	✓ lokale Videos	✓ Verbindung mit Smart Fridge
Swoozy Live 2015-2016	HTML5	Node.js C# Raspberry PI [2]	✓ OCR/Emgu CV	✓ NEMEX	✓ DVB-T Signal	✓ Wikidata/DBPedia
Swoozy Live 2016	HTML5	Node.js C#	✓ OCR/Emgu CV	✓ NEMEX	✓ DVB-S over IP	✓ Wikidata/DBPedia

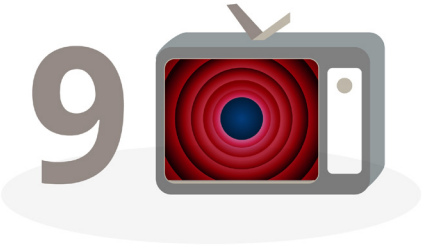
1 - Siehe "Foundations of Semantic TV" Bergweiler & Deru und [Ber14b] [Ber14a]

2 - Als DVB-T Streaming Komponente

Abbildung 8.15: Vergleichstabelle: Gesamtübersicht der Swoozy-Versionen von 2013 bis 2016 (1/2)

Interaktionen und Funktionen			Werkzeuge/Formate		Besonderheiten
Gesten-steuerung	**Second Screen**	**Live TV**	**SwoozyML Swoozy JSON**	**Livana/ SKRPTR**	
LeapMotion Kinect Fernbedienung	✗	✗ nur VOD	✓ SwoozyML	✗ nur partiell	Erste Version vom semantischen TV Komplett Gestengesteuerte UX
LeapMotion Kinect Fernbedienung	✓	✗ nur VOD	✓ SwoozyML	✗ nur partiell	Full HD Überarbeitetes Overlay-Prinzip Second Screen Unterstützung
LeapMotion Kinect Fernbedienung	✓	✗	✓ SwoozyML	✗ nur partiell	Verbindung mit Smart Fridge vom IRL Anbindung mit spezifischen Medien
LeapMotion Kinect Fernbedienung	✓	✓	✓ JSON	✓	Live-Analyse vom DVB-T-Signal NEMEX Cloud-basiertes Fernsehen
LeapMotion Kinect Fernbedienung	✓	✓	✓ JSON	✓	Live-Videoanalyse (Signalquelle: IP-TV) NEMEX Cloud-basiertes Fernsehen

Abbildung 8.16: Vergleichstabelle: Gesamtübersicht der Swoozy-Versionen von 2013 bis 2016 (2/2)



Fazit

In diesem Buch wurde das neue Konzept des semantischen Fernsehens präsentiert, indem zuerst die Grundlagen des Semantic Web und seiner Benutzung näher definiert wurden. Des Weiteren wurde bewiesen, dass durch eine geschickte und verteilte Analyse heterogener, im Fernsehsignal eingebetteter Datenquellen (Videosignal über OCR, Extraktion von EPG-Daten) eine semantische Extraktion in Echtzeit möglich ist. Auf diesem Wege können Annotationen generiert werden, die passend zum abgespielten Video über die grafische Benutzerschnittstelle eingeblendet werden. Diese Einblendung der Annotationen über sog. *Grabbables* stellt auf der Interaktionsebene einen echten Mehrwert für den Zuschauer dar. Der zuvor bei den klassischen App-orientierten Smart-TV auftretende Interaktionsbruch wurde mit dem vorgestellten Swoozy-Ansatz vermieden. Die dadurch realisierte Vereinfachung der Interaktion spiegelt sich in der benutzerzentrierten Wiedergabe von multimedialen Ergebnissen aus dem Semantic Web innerhalb der grafischen Schnittstelle wider.

Die technische Realisierung von Swoozy hat auch gezeigt, dass eine semantische Verknüpfung von Fernsehbildern kombiniert mit einfachen Interaktionen und Aufrufen von Webdiensten direkt vor dem Fernseher einen echten Mehrwert für den Zuschauer darstellt. Da-

mit geht der Swoozy-Ansatz weit über die Möglichkeiten aktueller kommerzieller Smart-TV-Systeme hinaus. Während der technischen Entwicklung von Swoozy wurde festgestellt, dass sich die Medien- und Fernsehindustrie in einer Umwandlungsphase befindet. Immer mehr konkurrierende Angebote wie z.B. IP-TV, Video on Demand und Internetvideoportale können vom Zuschauer über die neuen TV-Geräte aufgerufen werden. Dies wirft die Frage auf, wie „klassische" Fernsehsender, trotz dieser Konkurrenz, immer noch ihren Zuschauern wirtschaftlich hochqualitative Inhalte anbieten können. Die Lösung wird höchstwahrscheinlich im Bereich der neuen Interaktionsmöglichkeiten mit zuschauerzentrierten Inhalten (Stichwort: Personalisierung von TV-Inhalten und Adressable-TV - siehe Kapitel 1) und der nahtlosen Verbindung mit dem Web liegen. Für diese Herausforderung bietet Swoozy schon erste Ansätze an, indem es gezeigt hat, dass das semantische Fernsehen technisch mit der aktuellen Web- und Fernsehinfrastruktur realisierbar ist.
Anschließend liste ich hier ein paar inspierende Fragen, die mir während Vorführungen von Swoozy und Konferenz gestellt wurden. Diese fassen sehr gut das System zusammen und öffnen meiner Meinung nach neue Wege für die Zukunft des interaktiven Fernsehens.

- **Inwieweit lässt sich das Konzept großflächig einsetzen und kann es überhaupt in die bestehende Fernsehsenderinfrastruktur integriert werden?**
  Die technische Implementierung des Swoozy Systems (siehe Kapitel 6) hat bewiesen, dass das Konzept dank seines verteilten Ansatzes schon heute in die aktuelle Fernsehsenderinfrastruktur integriert und eingesetzt werden kann. Die angewandten Verfahren setzen auf die bereits verfügbaren Standards und Cloud-

basierten Dienste, die allmählich ihren Weg in die Smart-TVs finden. Darüber hinaus können Fernsehsender Kontextinformationen samt Videosignal zu den ausgestrahlten Datenpaketen hinzufügen, ohne dabei die DVB-Spezifikationen zu „verletzen" oder erweitern zu müssen. Spezielle Datenplätze oder Platzhalter (z.B. in der EIT-Tabelle DVB-SI-Spezifikation) werden zurzeit noch nicht verbreitet genutzt und könnten für den Transport von Zusatzinformationen dienen. Ähnliches gilt für den Transport und die Ausstrahlung von Metadaten oder zeitgesteuerten Informationen (ähnlich der *Content Purchasing API*- Spezifikation bei DVB-MHP [ETS08]). Diese könnten benutzt werden, um erweiterte semantische Informationen anzuzeigen. Zusätzlich zeigen Initiativen der EBU (wie „Semantic Web@EBU") konkrete Ziele, semantische Technologien in Form von Ontologien und Vokabularen in der Fernsehproduktion zu etablieren. Mit der stetig steigenden Anzahl von mit dem Internet verbundenen Fernsehern und der Verbreitung von HbbTV stellen Live-Zugriffe auf Server und Mediatheken, parallel zu einer laufenden Sendung, keine wesentliche technische Hürde mehr dar. In den meisten Fällen bieten Fernsehsender einen Rückkanal (z.B. über Second Screen-Apps) an, der über eine dedizierte Client-Server-Infrastruktur Zugang zu vorbereitenden Medien anbietet. Neben Fernsehgeräten könnten somit auch Second Screens (Tablets oder mobile Endgeräte) über den Swoozy-Server Medien aufrufen, herunterladen und abspielen.

Die größte Herausforderung, der sich Fernsehsender bei der Einführung von Semantic TV stellen werden müssen, liegt darin, dass sie sich intern auf eine neue Art der Produktion und redaktionellen Bearbeitung von Videos einstellen müssen. Hierzu

muss der Produktions-Workflow adaptiert werden, damit eine *„semantische redaktionelle"* Verarbeitung bzw. Annotation erfolgen kann. Das bedeutet, dass Videos vor der Ausstrahlung semiautomatisch annotiert werden müssen und dass diese Informationen im kompletten Workflow bis zur Archivierung des Videomaterials gespeichert werden müssen, damit bei einer eventuellen Wiederholung die Informationen und Zeitpunktedefinitionen schon im Archiv vorhanden und direkt aufrufbar sind. Zusätzlich kann durch die Annotationen bzw. *Grabbables* die kommerzielle Verwertbarkeit eines Films oder einer Reportage erhöht werden.

Dabei stellt sich natürlich die Frage der Aktualität der Annotationen und der damit verbundenen Abfrage der Dienste. Es wäre in Zukunft denkbar, dass basierend auf z.B. dem Google Knowledge Graph oder Wikidata die Annotationen zu eingepflegten Medienobjekten sich automatisch aktualisieren und somit immer ihren neuesten Stand behalten. Neue onlinebasierte Extraktionsdienste wie z.B. Diffbot[1] könnten aus existierenden Webseiten helfen, diese neuen Medienobjekte mittels einer einheitlichen API aufzufinden. Dies hätte den Vorteil der Wiederverwendbarkeit der bestehenden Annotationen und würde dem Redakteur eine mühsame und manuelle Überprüfung jeder Annotation und jedem Link (z.B. zu Medienobjekten der Mediathek) ersparen. Es wäre hierzu die Definition eines interoperablen Industriestandards zur aktiven Unterstützung und Einbindung semantischer Information im Broadcast-Bereich wünschenswert. Ein leichtgewichtiges Format wie SwoozyML könnte eingesetzt werden.

[1] `https://www.diffbot.com`

- **Von welchen Verfahren der künstlichen Intelligenz könnte das semantische Fernsehen profitieren?**
  In den Kapiteln 2 und 3 wurde deutlich gezeigt, dass immer mehr Dienste KI-Verfahren benutzen, um eine Cloud-basierte semantische Analyse durchzuführen. Diese neue Tendenz ist besonders bei den cloud-basierten Diensten zur Text und Bild- bzw. Videoanalyse sichtbar. Streambasierte Analysen, welche bereits in NEMEX integriert sind, würden eine kontinuierliche Inhaltsanalyse des TV-Programms durchführen und somit in Echtzeit das ausgestrahlte und wiedergegebene Video mit Informationen anreichern. Deep Learning (siehe Kapitel 3) wird dem semantischen Fernsehen eine ganz neue Bedeutung geben, u.a. dank einer schnelleren und präziseren Erkennung von Gesichtern, Filmsituationen und Kontextinformationen aus Videomaterial. In naher Zukunft wird es möglich sein, eine komplette Semantifizierung der Medienobjekte bzw. des Fernsehprogramms zu erreichen (siehe [JKFF16], [KTS+14] und [VXD+14b]): alles was der Zuschauer sieht, kann gesucht werden. Des Weiteren könnte eine Deep Learning-gestützte granulare Emotionserkennungskomponente die Möglichkeit geben, eine personalisierte Programmauswahl für den Benutzer anzubieten. Bei der automatischen Extraktion von Wissen (siehe Kapitel 6 und 7) wurde deutlich, dass die Identifikation der Domäne (z.B. Fußball, geografische Reportage über ein bestimmtes Land, politische Debatte) bei der Erkennung des Kontexts und des Inhalts einer Sendung dazu beitragen kann, die richtigen *Grabbables* zu generieren.

  Hierbei könnte das Swoozy-System von weiteren KI-gestützten Ansätzen profitieren, welche in [KGS14] präsentiert werden, um

u.a. diese automatische Erkennung der Domäne zu realisieren. Eine weitere Herausforderung bei der Analyse von internationalen und mehrsprachigen Fernsehprogrammen ist die automatische Erkennung der Sprache anhand der verschiedenen extrahierten Texte (Stichwort: Automatic Language Identification (LID) [LMGDP14] [LLB14] [PSPY12]). Die Benutzung dieser automatischen Erkennung würde dazu führen, dass sich die Eigennamen- und Konzepterkennung dynamisch zu der jeweiligen Sprache anpassen würde. Zusätzlich könnten die Wissensquellen und die Dienste, die für die Suche benutzt werden, je nach erkannter Sprache dynamisch ausgewählt werden. Hierfür könnten KI-Verfahren zur automatischen Komposition von Wissensquellen eingesetzt werden (siehe [Ber14a] [KLKR08] [SPAS03]).

- **Bilden nun nach den Smart-TVs das semantische oder sogar das Cloud-TV eine neue Chance für das interaktive Fernsehen?**
  Die aktuellen Smart-TVs stoßen sehr oft an technische Grenzen (siehe Kapitel 5) wie z.B. die herstellerbedingten Begrenzungen der Zugriffsmöglichkeiten der APIs auf Hardwarekomponenten, das gleichzeitige Abspielen verschiedener Videoströme oder die Anzeige von Zusatzinformationen mittels *Overlay*, welches die Entwicklung von innovativen Konzepten verlangsamt. Dies wirft die Frage auf, ob ein Paradigmenwechsel von Smart-TV zu Cloud-TV eine Chance für die Verbreitung des semantischen Fernsehens sein könnte. Immer mehr Online-Provider bieten Fernsehkanäle an, die nicht nur über die traditionellen DVB-T/S-/C-Kanäle ausgestrahlt werden, sondern auch als IP-TV Streams verfügbar sind. Denkt man dieses Konzept zu Ende, entwickelt sich der

Fernseher immer mehr zu einem rein grafischen Ausgabegerät für Videoströme. Grafiken und Videos werden serverseitig generiert, über einen IP-basierten Kanal übermittelt und vom „Cloud-TV"-Gerät dargestellt. Lediglich die Benutzerinteraktionen müssen über einen dafür vorgesehenen Rückkanal versendet werden. Ähnliche, bereits existierende Ansätze wie DaaS[2] (Display as a Service) zeigen, dass das Konzept des Cloud-basierten Bildschirms schon technisch realisierbar ist. Die eigentliche Verarbeitungsintelligenz ist dabei nicht mehr im Fernsehgerät „verbaut", sondern komplett in der Cloud verteilt und wird je nach Bedarf mit dem Videosignal synchronisiert. In diesem technischen Kontext würden das Einblenden grafischer Zusatzinformationen (Bilder, Videos oder *Grabbables*) und die Verknüpfung zu den Inhalten von Wissensdatenbanken (dank dauerhafter Internetverbindung) nur einen geringen Zusatzaufwand bedeuten. Der Cloud-TV-Ansatz setzt voraus, dass eine IP-basierte Kommunikation dauerhaft besteht und eine Rückkopplung zu einem (Rendering)-Server existiert. Beide Bedingungen können als Grundlage für das semantische Fernsehen dienen.

- **Welche Vorteile kann der Einsatz von Semantic TV im Geschäftsmodell eines Fernsehsenders mit sich bringen?**
  Semantic TV kann als neue interaktive Plattform für eine geschickte Kombination aus Wissen und Werbung dienen. In der aktuellen Version dient die Anzeige der *Grabbables* und *Suggested Grabbables* zunächst nur als interaktives Wissenselement bzw. als Hinweis, der zeitlich eingeblendet wird. Verfolgt man diese Idee weiter, lässt sich damit ein neuer hybrider Ansatz definie-

[2] http://www.daas.tv

ren. Fernsehsender könnten ihre eigenen Inhalte einblenden und über die zeitlich definierten *Grabbables* durch die Verlinkung zu internen Shops oder über kostengünstige Zugänge zu exklusiven Archivinhalten und animierten Infographiken vermarkten. Kommerzielle, TV-bezogene Dienste und Content-Provider könnten sich bestimmte Zeiträume in Sendungen mieten und zur Sendung passende *Grabbables* einfügen, um damit gezielt gesponserte Inhalte anbieten zu können. Bei Formel1- Übertragungen könnten Eckdaten der verschiedenen verwendeten Reifen eines Herstellers zur Verfügung gestellt werden oder während eines Fernsehberichts zur Elektromobilität könnte der Autohersteller mehr Informationen über sein Fahrzeug in Form von längeren Werbespots (engl. Informercials) anbieten. Durch die nicht disruptive Anzeige der *sponsored Grabbables* (*Grabbables*, die für kontextbezogene Werbezwecke von Unternehmen erworben werden können) würde diese indirekte Werbungsform als nicht intrusiv empfunden werden, da die eigentliche Anzeige und das Abspielen dieser Infomercials proaktiv vom Benutzer getriggert werden müssen.

- **Semantic TV als Vorstufe zu einem hybriden neuen MyTV-Konzept?**
  Die aktuelle Neuausrichtung der Fernsehsender, insbesondere durch die explosionsartige Verbreitung von Second Screen-Angeboten zeigt, dass die Aktivität *„Fernsehen"* sich kontinuierlich verändert und deutet stark darauf hin, dass die Konvergenz mit dem Internet und dessen Angeboten unaufhaltbar ist. Das wird heute sichtbar beim sehr weit verbreiteten parallelen Zusammenspiel von TV-Signal und Zusatzinformation aus dem Internet. Die-

ser Trend wird in den kommenden Jahren stark zunehmen und dem Zuschauer ein komplett neues interaktives Fernseherlebnis ermöglichen. Zusätzlich mehren sich die Video On Demand- und OTT-Angebote (wie z.B. Hulu oder Netflix) und spielen, neben den traditionellen Fernsehsendern, eine immer bedeutendere Rolle.

Es kann sogar davon ausgegangen werden, dass es in ein paar Jahren keine zeitlich fest vordefinierten TV-Programmabläufe mehr geben wird, sondern nur noch personalisierte Programmzusammenstellungen. Jeder Zuschauer kann möglicherweise individuell und je nach persönlicher Interessenslage, inhaltlich und thematisch selbst seine eigene Sendung zusammenstellen inklusive aus dem Internet heruntergeladener Beiträge oder alternativer Videoinhalte von VOD-Anbietern und aus Mediatheken.

Dieser individuelle und persönlich gestaltete Mix aus linearem und nicht-linearem Videoangebot könnte von den Ansätzen des semantischen Fernsehens profitieren und somit die Vision des Fernsehers als interaktives *„Fenster zur Wissenswelt“* weiter voranbringen.

Abbildungsverzeichnis

Index

Literaturverzeichnis

[AAB06] Paul Akkermans, Lora Aroyo, and Pieter Bellekens. iFanzy: Personalised filtering using semantically enriched TV-Anytime content. In *Demo at the Third European Semantic Web Conference*, 2006.

[ABB+07] Lora Aroyo, Pieter Bellekens, Martin Björkman, Geert-Jan Houben, Paul Akkermans, and Annelies Kaptein. SenSee Framework for Personalized Access to TV Content. In Pablo César, Konstantinos Chorianopoulos, and Jens F. Jensen, editors, *EuroITV*, Seiten 156–165. Springer, 2007.

[Abe13] Peter Abeles. *Speeding Up SURF*, Seiten 454–464. Springer Berlin / Heidelberg, 2013.

[ABP06] Jan Alexandersson, Tilman Becker, and Norbert Pfleger. Overlay: The basic operation for discourse processing. In *SmartKom: Foundations of Multimodal Dialogue Systems*, Seiten 255–267. Springer, 2006.

[Abr87] Albert Abramson. *The history of television, 1880-1941*. McFarland & Company, 1987.

[ACC+13] H. Agrawal, N. Chavali, Mathialagan C., A. Alfadda, P. Banik, and D. Batra. CloudCV: Large-Scale Distributed Computer Vision as a Cloud Service, 2013.

[ACD+09] Lora Aroyo, Alex Conconi, Stefan Dietze, Annelies Kaptein, Lyndon Nixon, Christoph Nufer, Davide Palmisano, Luca Vignaroli, and Milena Yankova. NoTube–making TV a medium for personalized interaction. 2009.

[Ado04] Adobe. 7 steps to Understanding XMP Metadata, 2004.

[ADWMH+12] Roland Aigner, Hrvoje Benko Daniel Wigdor, David Lindbauer Michael Haller, Alexandra Ion, and Jeffrey Tzu Kwan Valino Koh Shengdong Zhao. Understanding Mid-Air Hand Gestures: A Study of Human Preferences in Usage of Gesture Types for HCI. Bericht, November 2012.

[AH15] Rakesh Achanta and Trevor Hastie. Telugu OCR Framework using Deep Learning. *Journal - arXiv:1509.05962*, 2015.

[Aky03] Suat Akyol. *Nicht-intrusive Erkennung isolierter Gesten und Gebärden*. Dissertation, Bibliothek der RWTH Aachen, 2003.

[AMY10] Freedman Barak Shpunt Alexander, Machline Meir, and Arieli Yoel. Depth mapping using projected patterns, May 2010. Patent 20100118123.

[AND09] Lora Aroyo, Lyndon Nixon, and Stefan Dietze. Television and the future internet: the NoTube project. 2009.

[ANM11] Lora Aroyo, Lyndon Nixon, and Libby Miller. NoTube: the television experience enhanced by online social and semantic data. In *2011 IEEE International Conference on*

Consumer Electronics-Berlin (ICCE-Berlin), Seiten 269–273. IEEE, 2011.

[AP05] Olivier Aubert and Yannick Prié. Advene: active reading through hypervideo. In *Proceedings of the sixteenth ACM conference on Hypertext and hypermedia*, Seiten 235–244. ACM, 2005.

[BAH+07] Pieter Bellekens, Lora Aroyo, Geert-Jan Houben, Annelies Kaptein, and Kees Van Der Sluijs. *Semantics-based framework for personalized access to TV content: the iFanzy use case*. Springer, 2007.

[Bas74] J.V. Basmajian. *Muscles alive: their functions revealed by electromyography*. Williams & Wilkins, 1974.

[BBC02] BBC. An introduction to MHP 1.0 and MHP 1.1. Bericht, BBC, 2002.

[BBE+15] Julia Bernd, Damian Borth, Benjamin Elizalde, Gerald Friedland, Heather Gallagher, Luke R. Gottlieb, Adam Janin, Sara Karabashlieva, Jocelyn Takahashi, and Jennifer Won. The YLI-MED Corpus: Characteristics, Procedures, and Plans. *Computing Research Repository*, 2015.

[BBEF05] Tamara L. Berg, Alexander C. Berg, Jaety Edwards, and David A. Forsyth. Who's in the picture. *Advances in neural information processing systems*, 2005.

[BDG+00] P. Bernzott, J. Dilworth, D. George, B. Higgins, and J. Knight. Optical character recognition method and apparatus, March 14 2000. US Patent 6,038,342.

[BDNS08] Paul Buitelaar, Thierry Declerck, Jan Nemrava, and David Sadlier. Cross-media semantic indexing in the soccer domain. In *2008 International Workshop on Content-Based Multimedia Indexing*, Seiten 296–301. IEEE, 2008.

[BDP10] Simon Bergweiler, Matthieu Deru, and Daniel Porta. Integrating a Multitouch Kiosk System with Mobile Devices and Multimodal Interaction. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces. ACM International Conference on Interactive Tabletops and Surfaces (ITS-2010), November 7-10, Saarbrücken, Germany*, 1515 Broadway New York, New York 10036, 2010. Association for Computing Machinery, The Association for Computing Machinery.

[BDPM+09] Roberto Borgotallo, Roberto Del Pero, Alberto Messina, Fulvio Negro, Luca Vignaroli, Lora Aroyo, Chris van Aart, and Alex Conconi. Personalized Semantic News: Combining Semantics and Television. In *User Centric Media*, Seiten 137–140. Springer, 2009.

[BDS08] Djamal Benslimane, Schahram Dustdar, and Amit Sheth. Services mashups: The new generation of web applications. *IEEE Internet Computing*, 2008.

[BEJ+14] Andy Brown, Michael Evans, Caroline Jay, Maxine Glancy, Rhianne Jones, and Simon Harper. HCI over multiple screens. In *CHI'14 Extended abstracts on human factors in computing systems*, Seiten 665–674. ACM, 2014.

[Bel14] Matheus Giovanni Soares Beleboni. A brief overview of Microsoft Kinect and its applications. Bericht, 2014.

[Ben09] Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2009.

[BEP+08] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, Seiten 1247–1250. ACM, 2008.

[BEPS08] Rajesh Balchandran, Mark E. Epstein, Gerasimos Potamianos, and Ladislav Seredi. A multi-modal spoken dialog system for interactive TV. In *Proceedings of the 10th international conference on Multimodal interfaces*, Seiten 191–192. ACM, 2008.

[Ber14a] Simon Bergweiler. A Flexible Framework for Adaptive Knowledge Retrieval and Fusion for Kiosk Systems and Mobile Clients. In *UBICOMM 2014, The Eighth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, Seiten 164–171, 2014.

[Ber14b] Simon Bergweiler. Interactive Service Composition and Query. In Wolfgang Wahlster, Hans-Joachim Grallert, Stefan Wess, Hermann Friedrich, and Thomas Widenka, editors, *Towards the Internet of Services: The Theseus Program*, Cognitive Technologies. Springer Berlin / Heidelberg, 2014.

[BI98] M. Baca and Getty Information Institute. *Introduction to metadata: pathways to digital information*. Introduction To Series. Getty Information Institute, 1998.

[Bir06] Klaus Birkenbihl. *Semantic Web: Wege zur vernetzten Wissensgesellschaft*, chapter Standards für das Semantic Web, Seiten 73–88. Springer Berlin / Heidelberg, 2006.

[BK14] Andreas Butz and Antonio Krüger. *Mensch-Maschine-Interaktion*. Walter de Gruyter GmbH & Co KG, 2014.

[BKM+15] Evgenia Belyaeva, Aljaž Košmerlj, Andrej Muhič, Jan Rupnik, and Flavio Fuart. Using Semantic Data to Improve Cross-lingual Linking of Article Clusters. *Web Semant.*, December 2015.

[BKU11] Damian Borth, Christian Kofler, and Adrian Ulges. Smart Video Buddy - Content-based Live Recommendation. In *Proceedings of the International Conference on Multimedia and Expo, IEEE.*, 2011.

[BKVH03] Jeen Broekstra, Arjohn Kampman, and Frank Van Harmelen. Sesame: An architecture for storing and querying RDF data and schema information. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*, 2003.

[BL+07] Yoshua Bengio, Yann LeCun, et al. Scaling learning algorithms towards AI. *Large-scale kernel machines*, 2007.

[Bla13] Christian Blank. Generierung von Tiefenbildern mittels Stereoskopie, 2013. Bachelorarbeit.

[BLHL+01] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic Web. 2001.

[BLS+14] Tilman Becker, Markus Löckelt, Christian H. Schulz, Simon Bergweiler, Matthieu Deru, and Norbert Reithinger. A Unified Approach for Semantic-based Multimodal Interaction. In Wolfgang Wahlster, Hans-Joachim Grallert, Stefan Wess, Hermann Friedrich, and Thomas Widenka, editors, *Towards the Internet of Services: The Theseus Program*, Cognitive Technologies. Springer Berlin / Heidelberg, 2014.

[Bol80] Richard A. Bolt. Put-that-there: Voice and Gesture at the Graphics Interface. *SIGGRAPH Comput. Graph.*, July 1980.

[Bor14] Damian Borth. Visual Learning of Semantic Concepts in Social Multimedia. *KI-Künstliche Intelligenz*, 2014.

[Bou11] Roland Bouman. MQL to SQL - A JSON based RDBMS Query Language, 2011.

[BP08] Andreas Blumauer and Tassilo Pellegrini. *Social Semantic Web: Web 2.0 - Was nun?* Springer, Berlin, November 2008.

[BPG+05] Yolanda Blanco, José J. Pazos, Alberto Gil, Manuel Ramos, Ana Fernández, Rebeca P. Díaz, Martín López, and Belén Barragáns. AVATAR: an approach based on semantic reasoning to recommend personalized TV programs. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, Seiten 1078–1079. ACM, 2005.

[Bra11] Falk Brauer. *Extraktion und Identifikation von Entitäten in Textdaten im Umfeld der Enterprise Search*. Dissertation, Universität Potsdam, 2011.

[BSP07] Johannes Bauer, Niko Sunderhauf, and Peter Protzel. Comparing several implementations of two recently published feature detectors. In *Proc. of the International Conference on Intelligent and Autonomous Systems*, 2007.

[BSUB08] Damian Borth, Christian Schulze, Adrian Ulges, and Thomas M. Breuel. Navidgator-similarity based browsing for image and video databases. In *Annual Conference on Artificial Intelligence*, Seiten 22–29. Springer, 2008.

[BTH04] Liviu Badea, Doina Tilivea, and Anca Hotaran. Semantic Web Reasoning for Ontology-Based Integration of Resources. In Hans-Jürgen Ohlbach and Sebastian Schaffert, editors, *Principles and Practice of Semantic Web Reasoning*, Seiten 61–75. Springer Berlin / Heidelberg, 2004.

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, Seiten 404–417. Springer, 2006.

[Buc97] Michael K. Buckland. What Is a "Document"? *Journal of the American Society of Information Science*, 1997.

[BVDSAH08] Pieter Bellekens, Kees Van Der Sluijs, Lora Aroyo, and Geert-Jan Houben. Convergence of Web and TV broadcast data for adaptive content access and navigation. In *Adaptive Hypermedia and Adaptive Web-Based Systems*, Seiten 361–365. Springer, 2008.

[BVKK09] Pieter Bellekens, Geert Van Kerckhove, and Annalies Kaptein. iFanzy–A Ubiquitous Approach Towards a Personalized EPG. *Online paper*, 2009.

[BVLG11] Gilles Bailly, Dong-Bach Vo, Eric Lecolinet, and Yves Guiard. Gesture-aware Remote Controls: Guidelines and Interaction Technique. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, Seiten 263–270, New York, NY, USA, 2011. ACM.

[Car05] Bob Carpenter. *The logic of typed feature structures: with applications to unification grammars, logic programs and constraint resolution*. Cambridge University Press, 2005.

[Car08] D. Carl. *Mashups programmieren*. O'Reilly, 2008.

[CBDC14] Tao Chen, Damian Borth, Trevor Darrell, and Shih-Fu Chang. Deepsentibank: Visual sentiment concept classification with deep convolutional neural networks. *Journal - arXiv:1410.8586*, 2014.

[CBZL13] Jingyuan Cheng, Mathias Sundholm Bo Zhou, and Paul Lukowicz. Smart Chair: What can simple pressure sensors under the chairs legs tell us about user activity? IARIA XPS Press, 2013.

[CCC+11] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J. Wu, and Andrew Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *2011 International Conference on Document Analysis and Recognition*, Seiten 440–445. IEEE, 2011.

[CDF+04] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, Seiten 1–2, 2004.

[Cha13] Tom Chatfield. OCR. In *50 Schlüsselideen Digitale Kultur*, Seiten 132–135. Spektrum Akademischer Verlag, 2013.

[Chn13] Amina Chniti. Gestion des dépendances et des interactions entre Ontologies et Règles Métier, 2013.

[CKL+96] Gwo-Ching Chang, Wen-Juh Kang, Jer-Junn Luh, Cheng-Kung Cheng, Jin-Shin Lai, Jia-Jin J Chen, and Te-Son Kuo. Real-time implementation of electromyogram pattern recognition as a control command of man-machine interface. *Medical engineering & physics*, 1996.

[CKLS07] Mohamed Cheriet, Nawwaf Kharma, Cheng-Lin Liu, and Ching Suen. *Character recognition systems: a guide for students and practitioners*. John Wiley & Sons, 2007.

[CKN12] Adam Coates, Andrej Karpathy, and Andrew Y. Ng. Emergence of Object-Selective Features in Unsupervised Feature Learning. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, Seiten 2681–2689. Curran Associates, Inc., 2012.

[CLST15] Henry Chen, Austin S. Lee, Mark Swift, and John C. Tang. 3D Collaboration Method over HoloLens™ and Skype™ End Points. In *Proceedings of the 3rd International Workshop on Immersive Media Experiences*, Seiten 27–30. ACM, 2015.

[CM08] Steven J. Castellucci and I. Scott MacKenzie. Graffiti vs. unistrokes: an empirical comparison. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Seiten 305–308. ACM, 2008.

[Com11] Z_punkt GmbH The Foresight Company. TV-2020 - Die Zukunft des Fernsehens, 2011.

[CQ69] Allan M. Collins and M. Ross Quillian. Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior*, 1969.

[CSS08] Jordi Conesa, Veda C. Storey, and Vijayan Sugumaran. Improving web-query processing through semantic knowledge. *Data & Knowledge Engineering*, 2008. Including Special Section: Natural Language Processing and Information Systems (NLDB 2006) – Four selected and extended papers.

[Dag07] Gregg Daggett. Deliverable 2.1 DICIT Architecture Tools, Standards, Hardware and Software for the First Prototypes. 2007.

[DASLP10] Stefano De Amici, Andrea Sanna, Fabrizio Lamberti, and Barbara Pralio. A Wii remote-based infrared-optical tracking system. *Entertainment Computing*, 2010.

[Daw10] Alexander Dawson. Ultimate Guide To MicroFormats. *http://sixrevisions.com/web-development/ultimate-guide-to-microformats-reference-and-examples/*, 2010.

[DBBF14] Svilen Dimitrov, Jochen Britz, Boris Brandherm, and Jochen Frey. Analyzing Sounds of Home Environment

for Device Recognition. In *Proceedings of the European Conference on Ambient Intelligence 2014. European Conference on Ambient Intelligence (AmI-14), November 11-13, Eindhoven, Netherlands*, Seite 16. Eindhoven University of Technology, AmI, 2014.

[DC07] Thierry Declerck and Andreas Cobet. *Towards a Cross-Media Analysis of Spatially Co-located Image and Text Regions in TV-News*, Seiten 288–291. Springer Berlin / Heidelberg, 2007.

[DDSP08] KC Dohse, Thomas Dohse, Jeremiah D. Still, and Derrick J. Parkhurst. Enhancing multi-user interaction with multi-touch tabletop displays using hand tracking. In *Advances in Computer-Human Interaction, 2008 First International Conference on*, Seiten 297–302. IEEE, 2008.

[Den08] K. Denecke. Using SentiWordNet for multilingual sentiment analysis. In *IEEE 24th International Conference on Data Engineering Workshop (ICDEW) 2008,*, Seiten 507–512, April 2008.

[DGH+14] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge Vault: A Web-scale Approach to Probabilistic Knowledge Fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, Seiten 601–610, New York, NY, USA, 2014. ACM.

[DGL+11] Stamatia Dasiopoulou, Eirini Giannakidou, Georgios Litos, Polyxeni Malasioti, and Yiannis Kompatsiaris. A

Survey of Semantic Image and Video Annotation Tools. In Georgios Paliouras, ConstantineD. Spyropoulos, and George Tsatsaronis, editors, *Knowledge-Driven Multimedia Information Extraction and Ontology Evolution*, Seiten 196–239. Springer Berlin / Heidelberg, 2011.

[DGM06] Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, Seiten 177–190. Springer, 2006.

[DHSW02] Erik Duval, Wayne Hodgins, Stuart Sutton, and Stuart L. Weibel. Metadata principles and practicalities. *D-lib Magazine*, 2002.

[DJM94] Bertrand Dominique and Gatto Jean-Marie. Système de traitement de l'information utilisant la télévision dans un réseau de type numérique ou analogique, 1994.

[DKP+04] Witold Drozdzynski, Hans-Ulrich Krieger, Jakub Piskorski, Ulrich Schäfer, and Feiyu Xu. Shallow Processing with Unification and Typed Feature Structures Foundations and Applications. *Künstliche Intelligenz*, 2004.

[DL97] Carlo J De Luca. The use of surface electromyography in biomechanics. *Journal of applied biomechanics*, 1997.

[DM11] Julien Plu Didier Mouronval, Thibaut Cuvelier. Schema.org - une initiative pour que les moteurs de recherche comprennent les sites Web, 2011.

[DMKA15] John Dowell, Sylvain Malacria, Hana Kim, and Edward Anstead. Companion apps for information-rich television programmes: representation and interaction. *Personal and Ubiquitous Computing*, 2015.

[DN15] Matthieu Deru and Robert Neßelrath. autoUI-ML: A Design Language for the Flexible Creation of Automotive GUIs Based on Semantically Represented Data. In *Proceedings of the 4th International Symposium on Pervasive Displays. International Symposium on Pervasive Displays (PerDis-15), 4th, June 10-12, Saarbrücken, Germany*, Seiten 235–236. ACM, 2015.

[Don06] Kevin Donnelly. SNOMED-CT: The advanced terminology and coding system for eHealth. *Studies in health technology and informatics*, 2006.

[DRS02] Stefan Kokkelink DCMI and Dublin Core Roland Schwänzl. Expressing Qualified Dublin Core in RDF / XML, 2002.

[DSW06] John Davies, Rudi Studer, and Paul Warren. *Semantic Web technologies: trends and research in ontology-based systems*. Wiley, 2006.

[Duc07] Andrew Duchowski. *Eye tracking methodology: Theory and practice*. Springer, 2007.

[DWC01] Thierry Declerck, Peter Wittenburg, and Hamish Cunningham. The automatic generation of formal annotations in a multimedia indexing and searching environment. In *Proceedings of the workshop on Human Language Technology and Knowledge Management-Volume*

2001, Seite 17. Association for Computational Linguistics, 2001.

[DZ14] Guanglong Du and Ping Zhang. Markerless human–robot interface for dual robot manipulators using Kinect sensor. *Robotics and Computer-Integrated Manufacturing*, 2014.

[Ede12] J.S. Eder. Knowledge graph based search system, June 21 2012. US Patent App. 13/404,109.

[EFC+11] Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. Open Information Extraction: The Second Generation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, Seiten 3–10, 2011.

[EHN08] Kathrin Eichler, Holmer Hemsen, and Günter Neumann. Unsupervised Relation Extraction From Web Documents. In *Language Resources and Evaluation Conference (LREC)*, Seiten 1674–1679, 2008.

[Eik93] Line Eikvil. Optical Character Recognition. *Norsk Regnesentral - Report No. 876*, 1993.

[EP06] Ralf Engel and Norbert Pfleger. Modality fusion. In *SmartKom: Foundations of Multimodal Dialogue Systems*, Seiten 223–235. Springer, 2006.

[ES06] Andrea Esuli and Fabrizio Sebastiani. SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining. In *5th Language Resources and Evaluation Conference (LREC 2006)*, Seiten 417–422, 2006.

[ETS08] ETSI. ETSI TS 102 757 Content Purchasing API. Bericht, 2008.

[ETS13] ETSI. ETSI TS 102 809 Signalling and carriage of interactive applications and services in Hybrid broadcast-broadband environments. Bericht, 2013.

[ETS14a] ETSI. ETSI EN 300 468 Specification for Service Information (SI) in DVB systems. Bericht, 2014.

[ETS14b] ETSI. ETSI EN 300 743 Digital Video Broadcasting (DVB) Subtitling systems. Bericht, 2014.

[ETS14c] ETSI. ETSI EN 301192 DVB Implementation Guidelines for Data Broadcasting. Bericht, 2014.

[Eva09] Jean-Pierre Evain. Semantic TV — is the semantic web a part of broadcasting's future? Bericht, 2009.

[FEM+16] Michael Färber, Basil Ell, Carsten Menne, Achim Rettinger, and Frederic Bartscherer. Linked Data Quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. 2016.

[FEMR15] Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. A Comparative Survey of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. 2015.

[FGM05] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, Seiten 363–370,

Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[Fie00] Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. Dissertation, University of California, Irvine, 2000.

[For13] Clinton Forry. Like-able Content: Spread Your Message with Third-Party Metadata. *http://alistapart.com/article/like-able-content-spread-your-message-with-third-party-metadata*, 2013.

[FPAN+06] Y.B. Fernandez, J.J. Pazos Arias, M.L. Nores, A.G. Solla, and M.R. Cabrer. AVATAR: an improved solution for personalized TV based on semantic inference. *IEEE Transactions on Consumer Electronics*, 2006.

[Fre14] Cecille Freeman. *Feature selection and hierarchical classifier design with applications to human motion recognition*. Dissertation, University of Waterloo, 2014.

[Fre15] Jochen Frey. *ASaP - Integrationsplattform für Smart Services in Intelligenten Umgebungen*. Dissertation, Universität des Saarlandes, Postfach 151141, 66041 Saarbrücken, 2015.

[FW95] William T. Freeman and Craig Weissman. Television control by hand gestures. In *Proc. of Intl. Workshop on Automatic Face and Gesture Recognition*, Seiten 179–183, 1995.

[FWL02] Dieter Fensel, Wolfgang Wahlster, and Henry Lieberman. *Spinning the Semantic Web: Bringing the World Wide*

Web to Its Full Potential. MIT Press, Cambridge, MA, USA, 2002.

[GAS09] Pragati Garg, Naveen Aggarwal, and Sanjeev Sofat. Vision based hand gesture recognition. *World Academy of Science, Engineering and Technology*, 2009.

[GDPM08] Luigi Gallo, Giuseppe De Pietro, and Ivana Marra. 3D interaction with volumetric medical data: experiencing the Wiimote. In *Proceedings of the 1st international conference on Ambient media and systems*, Seite 14. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[GHGM14] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik. Using k-poselets for detecting people and localizing their keypoints. In *CVPR*, 2014.

[GHM+14] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: an OWL 2 reasoner. *Journal of Automated Reasoning*, 2014.

[GJP+14] Jože Guna, Grega Jakus, Matevž Pogačnik, Sašo Tomažič, and Jaka Sodnik. An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking. *Sensors*, 2014.

[GLDG+14] David Geerts, Rinze Leenheer, Dirk De Grooff, Joost Negenman, and Susanne Heijstraten. In Front of and Behind the Second Screen: Viewer and Producer Perspectives on a Companion App. In *Proceedings of the*

2014 ACM International Conference on Interactive Experiences for TV and Online Video, TVX '14, Seiten 95–102, New York, NY, USA, 2014. ACM.

[Got15] ML Gottmer. Merging Reality and Virtuality with Microsoft HoloLens. 2015.

[GVC+08] Eric Galmar, Olivier Villon, Milos Cernak, Mohamed Benzeghiba, Christian J Wellekens, Mohamed Faouzi BenZeghiba, Remy Etheve, Daniel Riccio, Jean-Luc Dugelay, Claudio Bertotti, et al. Representation and Analysis of Video Content for Automatic Object Extraction. *Eurecom.fr*, 2008.

[Hö11] Benjamin Höferlin. Einführung OpenCV. Bericht, 2011.

[Hah10] Daniel Hahn. Eletromyographe. Bericht, TU München, 2010.

[Hau13] Jens Haupert. *DOMeMan : Repräsentation, Verwaltung und Nutzung von digitalen Objektgedächtnissen*. Phdthesis, Saarländische Universitäts- und Landesbibliothek, Postfach 151141, 66041 Saarbrücken, 1 2013.

[Hbb16] HbbTV.org. HbbTV Specification 2.0.1. Bericht, 2016.

[HBPH14] Jens Haupert, Simon Bergweiler, Peter Poller, and Christian Hauck. IRAR: Smart Intention Recognition and Action Recommendation for Cyber-Physical Industry Environments. In Juan Carlos Augusto, Vassilis Bourdakis, Daniela Braga, Simon Egerton, Kaori Fujinami, Gordon Hunter, Fahim Kawsar, Ahmad Lotfi, Davy Preuveneers, Abdul Wahab bin Abdul Rahman, and Victor

Zamudio, editors, *10th International Conference on Intelligent Environments (IE) 2014*, Seiten 124–131. IEEE, 7 2014.

[Hep08] Martin Hepp. Goodrelations: An ontology for describing products and services offers on the web. In *Knowledge Engineering: Practice and Patterns*, Seiten 329–346. Springer, 2008.

[Hes10] Rob Hess. An open-source SIFTLibrary. In *Proceedings of the international conference on Multimedia*, Seiten 1493–1496. ACM, 2010.

[HF09] W.D. Hillis and B. Ferren. Knowledge web, March 10 2009. US Patent 7,502,770.

[HHMW12] Volker Haarslev, Kay Hidde, Ralf Möller, and Michael Wessel. The RacerPro knowledge representation and reasoning system. *Semantic Web*, 2012.

[Hin07] Geoffrey E. Hinton. Learning multiple layers of representation. *Trends in cognitive sciences*, 2007.

[HJC12] Michael E. Holmes, Sheree Josephson, and Ryan E. Carney. Visual attention to television programs with a second-screen application. In *Proceedings of the symposium on eye tracking research and applications*, Seiten 397–400. ACM, 2012.

[HK02a] Thomas Heider and Thomas Kirste. Architecture considerations for interoperable multi-modal assistant systems. In *International Workshop on Design, Specification,*

and Verification of Interactive Systems, Seiten 253–267. Springer, 2002.

[HK02b] Thomas Heider and Thomas Kirste. Supporting goal-based interaction with dynamic intelligent environments. In *European Conference on Artificial Intelligence (ECAI 2002)*, Seite 1436, 2002.

[HKR09] Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC, 2009.

[HKS01] Thorsten Herfet, Thomas Kirste, and Michael Schnaider. EMBASSI: multimodal assistance for infotainment and service infrastructures. *Computers & Graphics*, 2001. Intelligent Interactive Assistance and Mobile Multimedia Computing.

[HMW12] Ian Horrocks, Boris Motik, and Zhe Wang. The HermiT OWL Reasoner. In *OWL Reasoner Evaluation Workshop (ORE)*, 2012.

[HNSS07] Christian Hentschel, Andreas Nürnberger, Ingo Schmitt, and Sebastian Stober. SAFIRE: Towards Standardized Semantic Rich Image Annotation. In Stéphane Marchand-Maillet, Eric Bruno, Andreas Nürnberger, and Marcin Detyniecki, editors, *Adaptive Multimedia Retrieval: User, Context, and Feedback*, Seiten 12–27. Springer Berlin / Heidelberg, 2007.

[Hoi11] Derek Hoiem. How the Kinect works. Bericht, University of Illinois, 2011.

[Hor05] Ian Horrocks. *Logic Programming: 21st International Conference, ICLP 2005, Sitges, Spain, October 2-5, 2005. Proceedings*, chapter OWL: A Description Logic Based Ontology Language, Seiten 1–4. Springer Berlin / Heidelberg, 2005.

[HS85] Robert M. Haralick and Linda G. Shapiro. Image segmentation techniques. *Computer vision, graphics, and image processing*, 1985.

[HS00] Axel Hildebrand and Vítor Sá. EMBASSI: electronic multimedia and service assistance. In *Proceedings IMC 2000: Intelligent Interactive Assistence & Mobile Multimedia Computing*, Seiten 50–59, 2000.

[HSB07] Martin Hepp, Katharina Siorpaes, and Daniel Bachlechner. Harvesting wiki consensus: Using wikipedia entries as vocabulary for knowledge management. *Internet Computing, IEEE*, 2007.

[HSBW13] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 2013.

[HSS+09] Andreas Holzinger, Selver Softic, Christian Stickel, Martin Ebner, and Matjaz Debevc. Intuitive E-Teaching by Using Combined HCI Devices: Experiences with Wiimote Applications. In Constantine Stephanidis, editor, *Universal Access in Human-Computer Interaction. Applications and Services*, Seiten 44–52. Springer Berlin / Heidelberg, 2009.

[HSXS13] Jungong Han, Ling Shao, Dong Xu, and Jamie Shotton. Enhanced computer vision with Microsoft Kinect sensor: A review. 2013.

[IACM15] Anastasia Ioannidou, Evlampios Apostolidis, Chrysa Collyda, and Vasileios Mezaris. A web-based tool for fast instance-level labeling of videos and the creation of spatiotemporal media fragments. *Multimedia Tools and Applications*, 2015.

[IDF+05] G. Iyengar, P. Duygulu, S. Feng, P. Ircing, S. P. Khudanpur, D. Klakow, M. R. Krause, R. Manmatha, H. J. Nock, D. Petkova, B. Pytlik, and P. Virga. Joint Visual-text Modeling for Automatic Retrieval of Multimedia Documents. In *Proceedings of the 13th Annual ACM International Conference on Multimedia*, MULTIMEDIA '05, Seiten 21–30, New York, NY, USA, 2005. ACM.

[IDS12] IDS Imaging Development Systems GmbH. Obtaining Depth Information from Stereo Images. Bericht, 2012.

[IPT05] IPTC International Press Telecommunications Council. IPTC Core Schema for XMP, 2005.

[IPT10] IPTC International Press Telecommunications Council. IPTC Standards - Photo Metadata, 2010.

[JGAK07] Sergi Jordà, Günter Geiger, Marcos Alonso, and Martin Kaltenbrunner. The reacTable: exploring the synergy between live music performance and tabletop tangible interfaces. In *Proceedings of the 1st international conference on Tangible and embedded interaction*, Seiten 139–146. ACM, 2007.

[JKFF16] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. DenseCap: Fully Convolutional Localization Networks for Dense Captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[JLK+14] Hojun Jaygarl, Cheng Luo, YoonSoo Kim, Eunyoung Choi, Kevin Bradwick, et al. *Professional Tizen Application Development*. John Wiley & Sons, 2014.

[JSB01] Mehrdad Jalali-Sohi and Feza Baskaya. A Multimodal Shopping Assistant for Home E-Commerce. In *FLAIRS Conference*, Seiten 2–6, 2001.

[JSD+14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, Seiten 675–678. ACM, 2014.

[JT05] Frederic Jurie and Bill Triggs. Creating efficient codebooks for visual recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, Seiten 604–610. IEEE, 2005.

[Kah14] Gerrit Matthias Kahl. *Dual reality framework : Basistechnologien zum Monitioring und Steuern von Cyber-Physischen Umgebungen*. Dissertation, 2014.

[KAR+86] Alfred Kobsa, Jürgen Allgayer, Carola Reddig, Norbert Reithinger, Dagmar Schmauks, Karin Harbusch, and Wolfgang Wahlster. Combining deictic gestures and natural language for referent identification. In *Proceedings*

of the 11th coference on Computational linguistics, Seiten 356–361. Association for Computational Linguistics, 1986.

[KE12] Kourosh Khoshelham and Sander Oude Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 2012.

[KGS14] Pawan Kumar, Raj Kumar Goel, and Prem Sagar Sharma. Domain Specific Named Entity Recognition (DSNER) from Web Documents. *International Journal of Computer Applications*, 2014.

[Kip07] Michael Kipp. Anvil: The video annotation research tool. *http://www.anvil-software.org*, 2007.

[KLKR08] Ulrich Küster, Holger Lausen, and Birgitta König-Ries. Evaluation of semantic service discovery—a survey and directions for future research. In *Emerging Web Services Technology, Volume II*, Seiten 41–58. Springer, 2008.

[Kon05] Peter Konrad. EMG-Fibel: Eine praxisorientierte Einführung in die kinesiologische Elektromyographie. *Noraxon INC. USA*, 2005.

[KS05] Maria Karam and M. C. Schraefel. A Taxonomy of Gestures in Human Computer Interactions. 2005.

[KSA13] Pooja Kamavisdar, Sonam Saluja, and Sonu Agrawal. A Survey on Image Classification Approaches and Techniques. *International Journal of Advanced Research in Computer and Communication Engineering*, 2013.

[KSDB15] Sebastian Kalkowski, Christian Schulze, Andreas Dengel, and Damian Borth. Real-time analysis and visualization of the YFCC100M dataset. In *Proceedings of the 2015 Workshop on Community-Organized Multimodal Mining: Opportunities for Novel Solutions*, Seiten 25–30. ACM, 2015.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, Seiten 1097–1105, 2012.

[KT13] Elsa Andrea Kirchner and Marc Tabie. Closing the gap: combined EEG and EMG analysis for early movement prediction in exoskeleton based rehabilitation. In *Proceedings of the 4th European Conference on Technically Assisted Rehabilitation-TAR 2013*, 2013.

[KTP09] Agnes Koschmider, Victoria Torres, and Vicente Pelechano. Elucidating the mashup hype: Definition, challenges, methodical guide and tools for mashups. In *Proceedings of the 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web at WWW*, Seiten 1–9, 2009.

[KTS+14] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale Video Classification with Convolutional Neural Networks. In *CVPR*, 2014.

[Lal13] Rajesh Lal. *Digital Design Essentials: 100 ways to design better desktop, web, and mobile interfaces*. Rockport Pub, 2013.

[Lan14] Markus Lanthaler. *REST: Advanced Research Topics and Practical Applications*, chapter Leveraging Linked Data to Build Hypermedia-Driven Web APIs, Seiten 107–123. Springer New York, 2014.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 2015.

[LC08] Mathias Lux and Savvas A. Chatzichristofis. Lire: lucene image retrieval: an extensible Java CBIR Library. In *Proceedings of the 16th ACM international conference on Multimedia*, Seiten 1085–1088. ACM, 2008.

[LE+12] Thomas Lin, Oren Etzioni, et al. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, Seiten 84–88. Association for Computational Linguistics, 2012.

[Le13] Quoc V. Le. Building high-level features using large scale unsupervised learning. In *2013 IEEE international conference on acoustics, speech and signal processing*, Seiten 8595–8598. IEEE, 2013.

[Leb10] Marie Lebert. Le Projet Gutenberg (1971-2009), 2010.

[Lee08] Johnny Chung Lee. Hacking the Nintendo Wii Remote. *Pervasive Computing, IEEE*, 2008.

[Len95] Douglas B. Lenat. CYC: A Large-scale Investment in Knowledge Infrastructure. *Communications of the ACM*, November 1995.

[Lev14] Michal Levin. *Designing Multi-device Experiences: An Ecosystem Approach to User Experiences Across Devices*. O'Reilly Media, Inc., 2014.

[LG12] Markus Lanthaler and Christian Gütl. On using JSON-LD to create evolvable RESTful services. In *Proceedings of the Third International Workshop on RESTful Design*, Seiten 25–32. ACM, 2012.

[LHSL07] Xuanzhe Liu, Yi Hui, Wei Sun, and Haiqi Liang. Towards service composition based on mashup. In *Services, 2007 IEEE Congress on*, Seiten 332–339. IEEE, 2007.

[LIJ+14] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, et al. DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 2014.

[LJB+95] Yann LeCun, LD. Jackel, Léon Bottou, Corinna Cortes, John S. Denker, Harris Drucker, Isabelle Guyon, UA. Muller, E. Sackinger, Patrice Simard, et al. Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural networks: the statistical mechanics perspective*, 1995.

[LKG04] Mathias Lux, Werner Klieber, and Michael Granitzer. Caliph & Emir: semantics in multimedia retrieval and annotation. In *Proceedings of the 19th International CODATA Conference*, Seiten 64–75. Citeseer, 2004.

[LKP03] Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection. In Bernd Michaelis and Gerald Krell, editors, *Pattern Recognition*, Seiten 297–304. Springer Berlin / Heidelberg, 2003.

[LLB14] Marco Lui, Jey Han Lau, and Timothy Baldwin. Automatic Detection and Language Identification of Multilingual Documents. *Transactions of the Association for Computational Linguistics*, 2014.

[LMGDP14] Ignacio Lopez-Moreno, Javier Gonzalez-Dominguez, and Oldrich Plchot. Automatic Language Identification Using Deep Neural Networks. In *Proceeding International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

[Loi11] Daria Loi. Changing the TV Industry through User Experience Design. In Aaron Marcus, editor, *Design, User Experience, and Usability. Theory, Methods, Tools and Practice*, Seiten 465–474. Springer Berlin / Heidelberg, 2011.

[Lov07] Rachel Lovinger. RDF & OWL. *http://www.slideshare.net/rlovinger/rdf-and-owl*, 2007.

[Low99] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The pro-*

ceedings of the seventh IEEE international conference on, Seiten 1150–1157. IEEE, 1999.

[LPLT15] Ziwei Liu, Xiaogang Wang Ping Luo, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[LTS03] Ching-Yung Lin, Belle L. Tseng, and John R. Smith. VideoAnnEx: IBM MPEG-7 annotation tool for multimedia indexing and concept learning. In *IEEE International Conference on Multimedia and Expo*, Seiten 1–2, 2003.

[Lux09] Mathias Lux. Caliph & Emir: MPEG-7 photo annotation and retrieval. In *Proceedings of the 17th ACM international conference on Multimedia*, MM '09, Seiten 925–926, New York, NY, USA, 2009. ACM.

[LVJ05] Rémi Landais, Laurent Vinet, and Jean-Michel Jolion. Evaluation of Commercial OCR: A New Goal Directed Methodology for Video Documents. In Sameer Singh, Maneesha Singh, Chid Apte, and Petra Perner, editors, *Pattern Recognition and Data Mining*, Seiten 674–683. Springer Berlin / Heidelberg, 2005.

[LXU11] Hong Li, Feiyu Xu, and Hans Uszkoreit. TechWatchTool: Innovation and Trend Monitoring. In *Recent Advances in Natural Language Processing (RANLP)*, Seiten 660–665, 2011.

[LZYN11] Quoc V. Le, Will Y. Zou, Serena Y. Yeung, and Andrew Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace

analysis. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, Seiten 3361–3368. IEEE, 2011.

[M+67] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Seiten 281–297. California, USA, 1967.

[MABE08] Roberto Manione, Fiorenza Arisio, Rajesh Balchandran, and Mark E. Epstein. Deliverable 5.1 Language Modelling, Dialogue and User Interface the First Set-top-box Related DICIT Prototype. *Language*, 2008.

[Mac92] I. Scott MacKenzie. Fitts' Law As a Research and Design Tool in Human-computer Interaction. *Hum.-Comput. Interact.*, March 1992.

[Mac11] John MacCormick. How does the Kinect work? *http://users.dickinson.edu/ jmac/selected-talks/kinect.pdf*, 2011.

[Mac12] Valentina Maccatrozzo. Burst the filter bubble: using Semantic Web to enable serendipity. In *The Semantic Web–ISWC 2012*, Seiten 391–398. Springer, 2012.

[May06] Wolfgang May. *Semantic Web: Wege zur vernetzten Wissensgesellschaft*, chapter Reasoning im und für das Semantic Web, Seiten 485–503. Springer Berlin / Heidelberg, 2006.

[MBS14] Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. Yago3: A knowledge base from multilingual wikipedias. In *7th Biennial Conference on Innovative Data Systems Research*. CIDR Conference, 2014.

[MC07] Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, Seiten 233–242. ACM, 2007.

[McN92] David McNeill. *Hand and mind: What gestures reveal about thought*. University of Chicago press, 1992.

[MCWD06] Cynthia Matuszek, John Cabral, Michael J. Witbrock, and John DeOliveira. An Introduction to the Syntax and Content of Cyc. In *AAAI Spring Symposium: Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, Seiten 44–49, 2006.

[MDD06] Bassem Makni, Stefan Dietze, and John Domingue. Towards semantic TV services a hybrid Semantic Web Services approach, 2006.

[Mer11] Klaus Merkel. HbbTV - Status und Ausblick. Bericht, Institut für Rundfunktechnik, 2011.

[MG10] Nicolai Marquardt and Saul Greenberg. Applying Proxemics to Mediate People's Interaction with Devices in Ubiquitous Computing Ecologies. In *ACM International Conference on Interactive Tabletops and Surfaces*, Seiten 1–1, 2010.

[MGK+10] Vasileios Mezaris, Spyros Gidaros, Walter Kasper, Jörg Steffen, Roeland Ordelman, Marijn Huijbregts, Franciska de Jong, Ioannis Kompatsiaris, and Michael G. Strintzis. A System for the Semantic Multimodal Analysis of News Audio-visual Content. *Journal on Advances in Signal Processing (EURASIP)*, February 2010.

[MGS08] Boris Motik, Bernardo Cuenca Grau, and Ulrike Sattler. Structured Objects in OWL: Representation and Reasoning. In Jinpeng Huai, Robin Chen, Hsiao-Wuen Hon, Yunhao Liu, Wei-Ying Ma, Andrew Tomkins, and Xiaodong Zhang, editors, *Proc. of the 17th Int. World Wide Web Conference (WWW 2008)*, Seiten 555–564, Beijing, China, April 21–25 2008. ACM Press.

[MHMSW05] Lena Maier-Hein, Florian Metze, Tanja Schultz, and Alex Waibel. Session independent non-audible speech recognition using surface electromyography. In *Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on*, Seiten 331–336. IEEE, 2005.

[Mic15] Microsoft. Kinect Human Interface Guidelines v1.8.0, 2015.

[Mie39] Erwin Miehlnickel. „Electrisches Fernsehen" vor fünfzig Jahren. *Kolloid-Zeitschrift*, 1939.

[MMGK14] Britta Meixner, Katarzyna Matusik, Christoph Grill, and Harald Kosch. Towards an easy to use authoring tool for interactive non-linear video. *Multimedia Tools and Applications*, 2014.

[MOM+08] Marco Matassoni, Maurizio Omologo, Roberto Maniione, Timo Sowa, Rajesh Balchandran, Mark E. Epstein, and Ladislav Seredi. The DICIT project: an example of distant-talking based spoken dialogue interactive system. *Intelligent Information Systems (IIS)*, 2008.

[MRF16] Mateusz Malinowski, Marcus Rohrbach, and Mario Fritz. Ask Your Neurons: A Deep Learning Approach to Visual Question Answering. *Journal - arXiv:1605.02697*, 2016.

[MSC05] Steven Morris and Anthony Smith-Chaigneau. *Interactive TV standards*. Taylor & Francis, 2005.

[MSM+09] L. Marquardt, P. Svaizer, E. Mabande, A. Brutti, C. Zieger, M. Omologo, and W. Kellermann. A natural acoustic front-end for Interactive TV in the EU-Project DICIT. In *Proc. IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing (PacRim)*, 2009.

[MSS02] Bangalore S Manjunath, Philippe Salembier, and Thomas Sikora. *Introduction to MPEG-7: multimedia content description interface*. John Wiley & Sons, 2002.

[MSS+03] Alexander Maedche, Steffen Staab, Nenad Stojanovic, Rudi Studer, and York Sure. Semantic portal-the seal approach. *Spinning the Semantic Web*, 2003.

[MTUK95] Paul Milgram, Haruo Takemura, Akira Utsumi, and Fumio Kishino. Augmented reality: A class of displays on the reality-virtuality continuum. In *Photonics for industrial applications*, Seiten 282–292. International Society for Optics and Photonics, 1995.

[Mul11] Tony Mullen. Introduction to Sentiment Analysis. *Online course: https://lct-master.org/files/MullenSentimentCourseSlides.pdf*, 2011.

[MW98] Mark T. Maybury and Wolfgang Wahlster. *Readings in intelligent user interfaces*. Morgan Kaufmann, 1998.

[MWK+05] Cynthia Matuszek, Michael Witbrock, Robert C. Kahlert, John Cabral, David Schneider, Purvesh Shah, and Doug Lenat. Searching for common sense: populating Cyc™ from the web. In *AAAI*, Seiten 1430–1435, 2005.

[MZD+14] Bernardo Magnini, Roberto Zanoli, Ido Dagan, Kathrin Eichler, Günter Neumann, Tae-Gil Noh, Sebastian Pado, Asher Stern, and Omer Levy. The Excitement Open Platform (EOP) for textual inferences. *Conference of the Association for Computational Linguistics - ACL 2014*, 2014.

[NA09] Robert Nesselrath and Jan Alexandersson. A 3d gesture recognition system for multimodal dialog systems. *6th International Joint Conference on Artificial Intelligence Worksh. on Knowledge and Reasoning in Practical Dialogue Systems*, 2009.

[NBB+97] Günter Neumann, Rolf Backofen, Judith Baur, Markus Becker, and Christian Braun. An information extraction core system for real world german text processing. In *Proceedings of the fifth conference on Applied natural language processing*, Seiten 209–216. Association for Computational Linguistics, 1997.

[NBRH14] Lyndon Nixon, Lotte Belice Baltussen, Lilia Perez Romero, and Lynda Hardman. *A Companion Screen Application for TV Broadcasts Annotated with Linked Open Data*, Seiten 241–244. Springer International Publishing, 2014.

[NBSD08] Jan Nemrava, Paul Buitelaar, Vojtěch Svátek, and Thierry Declerck. Text mining support for semantic indexing and analysis of a/v streams. *Workshop Programme*, 2008.

[NEJ16] Timothy Neate, Michael Evans, and Matt Jones. Designing Visual Complexity for Dual-screen Media. *CHI 2016*, 2016.

[Nes08] Robert Nesselrath. TaKG - Ein Toolkit zur automatischen Klassifikation von Gesten. Mastersthesis, DFKI, 2008.

[Neß16] Robert Neßelrath. SiAM-dp: an open development platform for massively multimodal dialogue systems in cyber-physical environments. 2016.

[NF14] Robert Neßelrath and Michael Feld. SiAM-dp: A Platform for the Model-Based Development of Context-Aware Multimodal Dialogue Applications. In *Proceedings of the 10th International Conference on Intelligent Environments*. IEEE, 7 2014.

[NHK10] Manfred Nowak, Lambert Heller, and Sascha Korzen. *Mashups und Bibliotheken*. 2010.

[Nix13] Lyndon Nixon. *Web and TV Seamlessly Interlinked: LinkedTV*, Seiten 32–42. Springer International Publishing, 2013.

[NJT06] Eric Nowak, Frédéric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In *Computer Vision–ECCV 2006*, Seiten 490–503. Springer, 2006.

[NLS+11] Robert Nesselrath, Chensheng Lu, Christian H. Schulz, Jochen Frey, and Jan Alexandersson. A Gesture Based System for Context-Sensitive Interaction with Smart Homes. In Reiner Wichert and Birgid Eberhardt, editors, *Ambient Assisted Living, 4. AAL-Kongress 2011*, Advanced Technologies and Societal Change, Seiten 209–219, Berlin, Germany, 2011. VDE, Springer.

[NPB+15] Karl Ni, Roger A. Pearce, Kofi Boakye, Brian Van Essen, Damian Borth, Barry Chen, and Eric Wang. Large-Scale Deep Learning on the YFCC100M Dataset. *Computing Research Repository*, 2015.

[NPvdA14] Günter Neumann, Gerhard Paaß, and David van den Akker. Linguistics to structure unstructured information. In *Towards the Internet of Services: The THESEUS Research Program*, Seiten 383–392. Springer, 2014.

[NW+96] Yanghee Nam, K Wohn, et al. Recognition of space-time hand-gestures using hidden Markov model. In *ACM symposium on Virtual reality software and technology*, Seiten 51–58, 1996.

[ODC+08] Eyal Oren, Renaud Delbru, Michele Catasta, Richard Cyganiak, Holger Stenzhorn, and Giovanni Tummarello. Sindice.com: a document-oriented lookup index

for open linked data. *International Journal of Metadata, Semantics and Ontologies*, 2008.

[PB06] Alex Poole and Linden J. Ball. Eye tracking in HCI and usability research. *Encyclopedia of human computer interaction*, 2006.

[PD15] Bernhard Preim and Raimund Dachselt. *Grundlegende 3D-Interaktionen*, Seiten 339–397. Springer Berlin / Heidelberg, 2015.

[PDB+14] Daniel Porta, Matthieu Deru, Simon Bergweiler, Gerd Herzog, and Peter Poller. Building Multimodal Dialogue User Interfaces in the Context of the Internet of Services. In Wolfgang Wahlster, Hans-Joachim Grallert, Stefan Wess, Hermann Friedrich, and Thomas Widenka, editors, *Towards the Internet of Services: The Theseus Program*, Cognitive Technologies. Springer Berlin / Heidelberg, 2014.

[PDP14] Christian Seemann Prof. Dr.Sven Pagel, Tobias Simon. Usability-Analyse von HbbTV. Bericht, Wirtschaft Hochschule Mainz, 2014.

[Pfl07] Norbert Pfleger. *Context-based multimodal interpretation : an integrated approach to multimodal fusion and discourse processing*. Dissertation, Universität des Saarlandes, Postfach 151141, 66041 Saarbrücken, 2007.

[PGE+03] Thomas Portele, Silke Goronzy, Martin C. Emele, Andreas Kellner, Sunna Torge, and Jürgen te Vrugt. SmartKom-Home: An advanced multi-modal interface to home entertainment. In *INTERSPEECH*. Citeseer, 2003.

[PGE+06] Thomas Portele, Silke Goronzy, Martin Emele, Andreas Kellner, Sunna Torge, and Jürgen te Vrugt. SmartKom-Home: The interface to home entertainment. In *SmartKom: Foundations of multimodal dialogue systems*, Seiten 493–503. Springer, 2006.

[PH05] JeffZ. Pan and Ian Horrocks. OWL-Eu: Adding Customised Datatypes into OWL. In *The Semantic Web: Research and Applications*, Seiten 153–166. Springer Berlin / Heidelberg, 2005.

[Pow03] Andy Powell. Expressing Dublin Core in HTML/XHTML meta and link elements. *http://dublincore.org/documents/dcq-html*, 2003.

[Pow09] Shelley Powers. *Practical RDF*. O'Reilly, 2009.

[Pri07] Prime Sense Ltd. Method and System for object reconstruction, 2007.

[Pro07] Robert Prot. *Précis d'histoire de la radio et de la télévision*. Editions L'Harmattan, 2007.

[PSH97] Vladimir I. Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 1997.

[PSN09] Daniel Porta, Daniel Sonntag, and Robert Neßelrath. A Multimodal Mobile B2B Dialogue Interface on the iPhone. In *Proceedings of the 4th Workshop on Speech in Mobile and Pervasive Environments (SiMPE 2009) in Conjunction with MobileHCI*, 2009.

[PSPY12] Thierry Poibeau, Horacio Saggion, Jakub Piskorski, and Roman Yangarber. *Multi-source, multilingual information extraction and summarization*. Springer Science & Business Media, 2012.

[PT06] Peter Poller and Valentin Tschernomas. Multimodal fission and media design. In *SmartKom: Foundations of Multimodal Dialogue Systems*, Seiten 379–400. Springer, 2006.

[QMB+02] Francis Quek, David McNeill, Robert Bryll, Susan Duncan, Xin-Feng Ma, Cemil Kirbas, Karl E. McCullough, and Rashid Ansari. Multimodal Human Discourse: Gesture and Speech. *ACM Trans. Comput.-Hum. Interact.*, September 2002.

[QOP12] M. Atif Qureshi, Colm O'Riordan, and Gabriella Pasi. Short-text Domain Specific Key Terms/Phrases Extraction Using an N-gram Model with Wikipedia. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, Seiten 2515–2518, New York, NY, USA, 2012. ACM.

[Qua90] David L. Quam. Gesture recognition with a dataglove. In *Aerospace and Electronics Conference, 1990. NAECON 1990., Proceedings of the IEEE 1990 National*, Seiten 755–760. IEEE, 1990.

[Que94] Francis K. H. Quek. Toward a Vision-based Hand Gesture Interface. In *Proceedings of the Conference on Virtual Reality Software and Technology*, VRST '94, Seiten 17–31,

River Edge, NJ, USA, 1994. World Scientific Publishing Co., Inc.

[Que95] Francis K. H. Quek. Eyes in the interface. *Image and vision computing*, 1995.

[RAAS12] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, Seiten 1194–1201. IEEE, 2012.

[RASL16] Scott E. Reed, Zeynep Akata, Bernt Schiele, and Honglak Lee. Learning Deep Representations of Fine-grained Visual Descriptions. *Computing Research Repository*, 2016.

[RAT11] Carlos Rodrigues, José Afonso, and Paulo Tomé. *ENTERprise Information Systems: International Conference, CENTERIS 2011, Vilamoura, Portugal, October 5-7, 2011, Proceedings, Part II*, chapter Mobile Application Webservice Performance Analysis: Restful Services with JSON and XML, Seiten 162–169. Springer Berlin / Heidelberg, 2011.

[RBE+05] Norbert Reithinger, Simon Bergweiler, Ralf Engel, Gerd Herzog, Norbert Pfleger, Massimo Romanelli, and Daniel Sonntag. A look under the hood: design and development of the first SmartWeb system demonstrator. In *Proceedings of the 7th international conference on Multimodal interfaces*, Seiten 159–166. ACM, 2005.

[RBN+07] Herwig Rehatschek, Werner Bailer, Helmut Neuschmied, Sandra Ober, and Horst Bischof. *A tool supporting annotation and analysis of videos*. na, 2007.

[RC06] Brian Kralyevich Richard Cardran, Kate Wojogbe. The Digital Home: Designing for the Ten-Foot User Interface. Video, Microsoft, 2006.

[RFZ05] Deva Ramanan, David A. Forsyth, and Andrew Zisserman. Strike a pose: Tracking people by finding stylized poses. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, Seiten 271–278. IEEE, 2005.

[RGHRT14] José Luis Redondo-García, Michiel Hildebrand, Lilia Perez Romero, and Raphaël Troncy. Augmenting TV newscasts via entity expansion. In *European Semantic Web Conference*, Seiten 472–476. Springer, 2014.

[RK06] V. Rodehorst and A. Koschan. Comparison and evaluation of feature point detectors. In *Proc. 5th International Symposium Turkish-German Joint Geodetic Days"Geodesy and Geoinformation in the Service of our Daily Life", Berlin, Germany*, 2006.

[RKD+03] Dennis Reidsma, Jan Kuper, Thierry Declerck, Horacio Saggion, and Hamish Cunningham. Cross document annotation for multimedia retrieval. In *EACL Workshop on Language Technology and the Semantic Web (NLPXML), Budapest, Hungary*, 2003.

[RLR06] Jens Racky, Michael Lützeler, and Hans Röttger. The Sense of Vision: Gestures and Real Objects. In *Smart-*

Kom: Foundations of Multimodal Dialogue Systems, Seiten 153–165. Springer, 2006.

[Rod08] Alex Rodriguez. Restful web services: The basics. *IBM developerWorks*, 2008.

[Roh14] Marcus Rohrbach. *Combining visual recognition and computational linguistics : linguistic knowledge for visual recognition and natural language descriptions of visual content*. Dissertation, Universität des Saarlandes, Postfach 151141, 66041 Saarbrücken, 2014.

[Ros04] Ron Roszkiewicz. Metadata in Context. Bericht, The Seybold Report, 2004.

[RRH+15] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of Textual Phrases in Images by Reconstruction. *Computing Research Repository*, 2015.

[RTR+16] Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Christopher J. Pal, Hugo Larochelle, Aaron C. Courville, and Bernt Schiele. Movie Description. *Computing Research Repository*, 2016.

[RV08a] Mohammadali Rahimi and Matthias Vogt. Gestenbasierte Computerinteraktion auf Basis von Multitouch-Technologie. *Hochschule für Angewandte Wissenschaften Hamburg, Bachelorarbeit*, 2008.

[RV08b] B. Reuse and R. Vollmar. *Informatikforschung in Deutschland*. Springer Berlin / Heidelberg, 2008.

[SAB+06] Rui Ping Shi, Johann Adelhardt, Anton Batliner, Carmen Frank, Elmar Nöth, Viktor Zeißler, and Heinrich Niemann. The Gesture Interpretation Module. In *SmartKom: Foundations of Multimodal Dialogue Systems*, Seiten 209–219. Springer, 2006.

[SAC10] Timo Sowa, Fiorenza Arisio, and Luca Cristoforetti. DICIT: Evaluation of a Distant-talking Speech Interface for Television. In *Language Resources and Evaluation Conference (LREC)*, 2010.

[SAFAKRM11] Jamie Shotton, Andrew Blake Andrew Fitzgibbon, Mark Finocchio Alex Kipman, and Toby Sharp Richard Moore. Real-Time Human Pose Recognition in Parts from a Single Depth Image. IEEE, June 2011.

[Sam13] Samsung. Design Principles for Creating Samsung Apps Content. *Design Principles for Creating Samsung Apps Content*, 2013.

[SB13] Alberto Gil Solla and Rafael G Sotelo Bovino. *TV-Anytime*. Springer, 2013.

[SBB+10] Balthasar Schopman, Dan Brickley, Vicky Buser, Libby Miller, and van Aart. NoTube: making the Web part of personalised TV. 2010.

[SBD+08] Johannes Schöning, Peter Brandl, Florian Daiber, Florian Echtler, Otmar Hilliges, Jonathan Hook, Markus Löchtefeld, Nima Motamedi, Laurence Muller, Patrick Olivier, et al. Multi-touch surfaces: A technical guide. *IEEE Tabletops and Interactive Surfaces*, 2008.

[SBM+06] Padmanabhan Soundararajan, Matthew Boonstra, Vasant Manohar, Valentina Korzhova, Dmitry Goldgof, Rangachar Kasturi, Shubha Prasad, Harish Raju, Rachel Bowers, and John Garofolo. Evaluation Framework for Video OCR. In PremK. Kalra and Shmuel Peleg, editors, *Computer Vision, Graphics and Image Processing*, Seiten 829–836. Springer Berlin / Heidelberg, 2006.

[Sch87] Dagmar Schmauks. Natural and Simulated Pointing. In *Proceedings of the Third Conference on European Chapter of the Association for Computational Linguistics*, EACL '87, Seiten 179–185, Stroudsburg, PA, USA, 1987. Association for Computational Linguistics.

[Sch91] Dagmar Schmauks. *Deixis in der Mensch-Maschine-Interaktion: multimediale Referentenidentifikation durch natürliche und simulierte Zeigegesten*. Niemeyer, 1991.

[Sch03a] Cordelia Schmid. Bag of Words or Bag of Features. Bericht, LEAR Inria Grenoble, 2003.

[Sch03b] Thomas Schmidt. A short introduction to the EXMARaLDA Partitur Editor. 2003.

[Sch04] Thomas Schmidt. Transcribing and annotating spoken language with EXMARaLDA. In *Proceedings of the Language Resources and Evaluation Conference (LREC)- Workshop on XML based richly annotated corpora, Lisbon 2004*, 2004.

[SDB09] Daniel Sonntag, Matthieu Deru, and Simon Bergweiler. Design and implementation of combined mobile and

touchscreen-based multimodal web 3.0 interfaces. In *Proceedings of the International Conference on Artificial Intelligence (ICAI). International Conference on Artificial Intelligence (ICAI-09), July 13-16, Las Vegas,Nevada, USA*, Seiten 974–979, 2009.

[SEA+15] Harpreet Singh Sawhney, Jayakrishan Eledath, Saad Ali, Bogdan C. Matei, Steven S. Weiner, and Xutao Lv. Computer vision as a service, October 6 2015. US Patent 9,152,870.

[Seb02] Fabrizio Sebastiani. Machine Learning in Automated Text Categorization. *ACM Computing Surveys (CSUR)*, 2002.

[SEH+07] Daniel Sonntag, Ralf Engel, Gerd Herzog, Alexander Pfalzgraf, Norbert Pfleger, Massimo Romanelli, and Norbert Reithinger. SmartWeb handheld—multimodal interaction with ontological knowledge bases and semantic web services. In Thomas S. Huang, Anton Nijholt, Maja Pantic, and Alex Pentland, editors, *Artifical Intelligence for Human Computing*, Seiten 272–295. Springer Berlin / Heidelberg, 2007.

[SET09] Toby Segaran, Colin Evans, and Jamie Taylor. *Programming the semantic web*. O'Reilly Media, 2009.

[Sev15] SevenOneMedia. Media Activity Guide 2015. Bericht, SevenOneMedia, 2015.

[SGKM04] Nick Siegel, Keith Goolsbey, Robert Kahlert, and Gavin Matthews. The Cyc System: Notes on Architecture. *Cycorp, Inc*, 2004.

[SH82] Christopher Schmandt and Eric A. Hulteen. The intelligent voice-interactive interface. In *Proceedings of the 1982 conference on Human factors in computing systems*, Seiten 363–366. ACM, 1982.

[Shp10] Alexander Shpunt. Depth mapping using multi-beam illumination, January 2010. Patent 20100020078.

[SKB12] Thomas Scharrenbach, Yamuna Krishnamurthy, and Christian Bockermann. ViSTA-TV: Video Stream Analytics for Viewers in the TV Industry. 2012.

[SKW07] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, Seiten 697–706, New York, NY, USA, 2007. ACM.

[SLJ+14] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. *Computing Research Repository*, 2014.

[SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[Smi07] Ray Smith. An Overview of the Tesseract OCR Engine. In *International Conference on Document Analysis and Recognition (ICDAR)*, Seiten 629–633, 2007.

[SP12] Christian Schulze and Sebastian Palacio. Retrieving Objects, People and Places from a Video Collection: TRECVID'12 Instance Search Task. In *Proc. TRECVID*, 2012.

[SPAS03] Katia Sycara, Massimo Paolucci, Anupriya Ankolekar, and Naveen Srinivasan. Automated discovery, interaction and composition of semantic web services. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2003.

[SPG+07] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 2007.

[SPPS08] Jan Schehl, Alexander Pfalzgraf, Norbert Pfleger, and Jochen Steigner. The babbleTunes system: talk to your iPod! In *Proceedings of the 10th international conference on Multimodal interfaces*, Seiten 77–80. ACM, 2008.

[Ste05] Achim Stein. *Semantische Repräsentation italienischer Verben: automatische Disambiguierung mit Konzepthierarchien*. Walter de Gruyter, 2005.

[STM+09] T. Scott Saponas, Desney S. Tan, Dan Morris, Ravin Balakrishnan, Jim Turner, and James A Landay. Enabling always-available input with muscle-computer interfaces. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, Seiten 167–176. ACM, 2009.

[STMB08] T. Scott Saponas, Desney S. Tan, Dan Morris, and Ravin Balakrishnan. Demonstrating the feasibility of using

forearm electromyography for muscle-computer interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Seiten 515–524. ACM, 2008.

[SVR12] Mark Sammons, V.G. Vinod Vydiswaran, and Dan Roth. Recognizing Textual Entailment. In Daniel M. Bikel and Imed Zitouni, editors, *Multilingual Natural Language Applications: From Theory to Practice*, chapter 6, Seiten 209–258. IBM Press, Pearson, May 2012.

[SVT+12] Thomas Steiner, Ruben Verborgh, Raphaël Troncy, Joaquim Gabarro, and Rik Van de Walle. Adding Realtime Coverage to the Google Knowledge Graph. In *11th International Semantic Web Conference (ISWC 2012)*. Citeseer, 2012.

[SvWMB09] Michael Schreiber, Margeritta von Wilamowitz-Moellendorff, and Ralph Bruder. New Interaction Concepts by Using the Wii Remote. In Julie A. Jacko, editor, *Human-Computer Interaction. Novel Interaction Methods and Techniques*, Seiten 261–270. Springer Berlin / Heidelberg, 2009.

[SW91] Dagmar Schmauks and Michael Wille. Integration of communicative hand movements into human-computer-interaction. *Computers and the Humanities*, 1991.

[SW09] Thomas Schmidt and Kai Wörner. EXMARaLDA: Creating, analyzing and sharing spoken language corpora

for pragmatics research. *Pragmatics-Quarterly Publication of the International Pragmatics Association*, 2009.

[SZE+14] Daniel Sonntag, Sonja Zillner, Patrick Ernst, Christian Schulz, Michael Sintek, and Peter Dankerl. Mobile radiology interaction and decision support systems of the future. In *Towards the Internet of Services: The THESEUS Research Program*, Seiten 371–382. Springer, 2014.

[SZK+11] Christian H. Schulz, Ingo Zinnikus, Patrick Kapahnke, Jochen Frey, Robert Nesselrath, and Jan Alexandersson. Universally Accessible Interactive Services on TV: A Case Study on the Provisioning of Internet Services to Elderly People. *Ambient Assisted Living-AAL*, 2011.

[SZZ+15] Wei Song, Shiqi Zhao, Chao Zhang, Hua Wu, Haifeng Wang, Lizhen Liu, and Hanshi Wang. Exploiting Collective Hidden Structures in Webpage Titles for Open Domain Entity Extraction. In *Proceedings of the 24th International Conference on World Wide Web*, Seiten 1014–1024. International World Wide Web Conferences Steering Committee, 2015.

[TBF+15] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, Seiten 4489–4497. IEEE, 2015.

[TCV00] Louis G. Tassinary, John T. Cacioppo, and Eric J. Vanman. The skeletomotor system: Surface electromyography. *Handbook of psychophysiology*, 2000.

[TGLZ02] Xiaoou Tang, Xinbo Gao, Jianzhuang Liu, and Hongjiang Zhang. A spatial-temporal approach for video caption detection and recognition. *IEEE Transactions on Neural Networks*, 2002.

[TH06] Dmitry Tsarkov and Ian Horrocks. *Automated Reasoning: Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006. Proceedings*, chapter FaCT++ Description Logic Reasoner: System Description, Seiten 292–297. Springer Berlin / Heidelberg, 2006.

[Tha14] ThalmicLabs. Myo Developer User Experience Guidelines. 2014.

[THY15] Ahmad Pahlavan Tafti, Hamid Hassannia, and Zeyun Yu. Siftservice.com - Turning a Computer Vision algorithm into a World Wide Web Service. *Journal - arXiv:1504.02840*, 2015.

[TK13] Marc Tabie and Elsa Andrea Kirchner. EMG Onset Detection - Comparison of different methods for a movement prediction task based on EMG. In *In Proceedings of the 6th International Conference on Bio-inspired Systems and Signal Processing. International Conference on Bio-inspired Systems and Signal Processing (Biosignalis-2013), February 11-14, Barcelona, Spain*, 2013.

[TLP97] Hong Z. Tan, Ifung Lu, and Alex Pentland. The chair as a novel haptic user interface. In *Proc. of the Workshop on Perceptual User Interfaces*, Seiten 19–21, 1997.

[TM14] Dorothea Tsatsou and Vasileios Mezaris. *LUMO: The LinkedTV User Model Ontology*, Seiten 268–272. Springer International Publishing, 2014.

[Tob14] Tobii. 4 Consideration designing UIs eye interaction. *http://developer.tobii.com/4-considerations-designing-uis-eye-interaction/*, 2014.

[TP89] David S. Touretzky and Dean A. Pomerleau. What's hidden in the hidden layers. *Byte, August*, 1989.

[TSF+16] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. YFCC100M: The new data in multimedia research. *Communications of the ACM*, 2016.

[TTR03] Tien Tran-Thuong and Cécile Roisin. Multimedia modeling using MPEG-7 for authoring multimedia integration. In *Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval*, Seiten 171–178. ACM, 2003.

[TVS+16] Thomas Pellissier Tanon, Denny Vrandecic, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. From Freebase to Wikidata: The Great Migration. In *World Wide Web Conference*, 2016.

[TYRW14] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Seiten 1701–1708, 2014.

[VAAR+09] Chris Van Aart, Lora Aroyo, Y. Raimond, D. Brickley, G. Schreiber, Michele Minno, L. Miller, Davide Palmisano, M. Mostarda, R. Siebes, et al. The NoTube Beancounter: aggregating user data for television programme recommendation. *Social Data on the Web (SDoW2009)*, 2009.

[VBB+14] Jakob Voß, Susanna Bausch, Jasmin Bogner, Julian Schmitt, Viktoria Berkelmann, Franziska Ludemann, Oliver Löffel, Janna Kitroschat, Maiia Bartoshevska, and Katharina Seljuzki. *Normdaten in Wikidata*. Lulu.com, 2014.

[vdSGS10] K. van de Sande, T. Gevers, and C. Snoek. Evaluating Color Descriptors for Object and Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Sept 2010.

[VG11] Jan C. Van Gemert. Exploiting photographic style for category-level image classification by generalizing the spatial pyramid. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, Seite 14. ACM, 2011.

[VGJ13] Biljana Veselinovska, Marjan Gusev, and Toni Janevski. Overview of current trends in IPTV related FP7 projects. In *Information, Communication and Energy Systems and Technologies (ICEST) 2013 48th International Scientific Conference on*, Seiten 111–114, 2013.

[VJ01] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Com-*

puter Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, Seiten I–511. IEEE, 2001.

[VK14] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 2014.

[VM14] Radu-Daniel Vatavu and Matei Mancas. Visual attention measures for multi-screen TV. In *Proceedings of the 2014 ACM international conference on Interactive experiences for TV and online video*, Seiten 111–118. ACM, 2014.

[VMD09] Rekha B. Venkatapur, VD Mytri, and A. Damodaram. Effective Algorithm Using Similarity Based Technique for Image and Video Databases. *Data Mining and Knowledge Engineering*, 2009.

[VXD+14a] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *Journal - arXiv:1412.4729*, 2014.

[VXD+14b] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond J. Mooney, and Kate Saenko. Translating Videos to Natural Language Using Deep Recurrent Neural Networks. *CoRR*, 2014.

[W3C09] W3C. OWL 2 Web Ontology Language Profiles - W3C Recommendation 27 October 2009, 2009.

[W3C12a] W3C. OWL 2 Web Ontology Language, 2012.

[W3C12b] W3C. OWL 2 Web Ontology Language RDF-Based Semantics (Second Edition) W3C Recommendation 11 December 2012, 2012.

[W3C12c] W3C. RDFa Lite 1.1 - W3C Recommendation 07 June 2012, 2012.

[W3C13a] W3C. HTML Microdata W3C Working Group Note 29 October 2013, 2013.

[W3C13b] W3C SPARQL Working Group. SPARQL 1.1 Overview. Bericht, W3C, 2013.

[W3C16a] W3C. SPARQL Implementations, 2016.

[W3C16b] W3C Schools. W3C Schools - RDF Dublin Core Metadata Initiative, 2016.

[Wah98] Wolfgang Wahlster. *User and discourse models for multimodal communication*. Morgan Kaufmann, San Francisco, 1998.

[Wah02] Wolfgang Wahlster. SmartKom: Fusion and fission of speech, gestures, and facial expressions. In *Proceedings of the 1st International Workshop on Man-Machine Symbiotic Systems*, Seiten 213–225, 2002.

[Wah04] Wolfgang Wahlster. SmartWeb: Mobile applications of the semantic web. In *KI 2004: Advances in Artificial Intelligence*, Seiten 50–51. Springer, 2004.

[Wah06a] Wolfgang Wahlster. Dialogue systems go multimodal: the SmartKom experience. In *SmartKom: foundations*

of multimodal dialogue systems, Seiten 3–27. Springer, 2006.

[Wah06b] Wolfgang Wahlster. *SmartKom: foundations of multimodal dialogue systems*. Springer, 2006.

[WBRF13] Frank Weichert, Daniel Bachmann, Bartholomäus Rudak, and Denis Fisseler. Analysis of the accuracy and robustness of the Leap Motion controller. *Sensors*, 2013.

[WCZ+14] Bing Wang, Jingyuan Cheng, Bo Zhou, Orkhan Amiraslanov, Paul Lukowicz, and Mengfan Zhang. Smartchairs: Ubiquitous Presentation Evaluation Based on Audience's Activity Recognition. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, MOBIQUITOUS '14, Seiten 350–351, ICST, Brussels, Belgium, Belgium, 2014. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[WDT+06] Wolfgang Wahlster, Andreas Dengel, Deutsche Telekom, Contributions Dietmar Dengler, Dominik Heckmann, Malte Kiesel, Alexander Pfalzgraf, Thomas Rothberghofer, Leo Sauermann, et al. Web 3.0: Convergence of Web 2.0 and the semantic web. In *In Technology Radar, Feature Paper, 2nd ed.; Deutsche Telekom Laboratories*, 2006.

[Web10] Answers Semantic Web. What is the difference between OWL Full, OWL DL and OWL Lite, 2010.

[WGW+14] Wolfgang Wahlster, Hans-Joachim Grallert, Stefan Wess, Hermann Friedrich, and Thomas Widenka. *Towards*

the internet of services: The THESEUS research program. Springer, 2014.

[WHW09] Daniel S. Weld, Raphael Hoffmann, and Fei Wu. Using Wikipedia to bootstrap open information extraction. *ACM SIGMOD Record*, 2009.

[WN07] Rui Wang and Günter Neumann. Recognizing Textual Entailment Using Sentence Similarity Based on Dependency Tree Skeletons. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, RTE '07, Seiten 36–41, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[WW10] Fei Wu and Daniel S. Weld. Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Seiten 118–127. Association for Computational Linguistics, 2010.

[XT08] Lei Xie and Xi Tan. A Heuristic Approach to Caption Enhancement for Effective Video OCR. In De-Shuang Huang, II Wunsch, DonaldC., DanielS. Levine, and Kang-Hyun Jo, editors, *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*, Seiten 347–355. Springer Berlin / Heidelberg, 2008.

[ZCT14] Qi-Jie Zhao, Peng Cao, and Da-Wei Tu. Toward intelligent manufacturing: label characters marking and recognition method for steel products with machine vision. *Advances in Manufacturing*, 2014.

Swoozy

Intelligente Interaktionen für das semantische Fernsehen

Fernsehen ist die beliebteste Aktivität der Deutschen, immer noch vor der Nutzung des Radios und Internets. Doch die Art und Weise fernzusehen, hat sich in den letzten Jahren stark verändert. Dieses Buch befasst sich mit der spannenden Thematik der parallelen Benutzung der Medien Fernsehen und Internet bzw. des semantischen Web, mit dem Fokus auf einer Suche und benutzerzentrierten Interaktionen, die intuitiv vorm Fernsehgerät durchgeführt werden können.

In dieser überarbeiten Version seiner Dissertation führt der Autor zuerst eine detaillierte Analyse der heutigen technischen Ansätze im Bereich Smart-TV durch. Dabei werden ihre Möglichkeiten in puncto Interaktion, Design und funktionaler Erweiterbarkeiten präsentiert. Zusätzlich wird das System namens Swoozy, welches die Grundlagen des neuen interaktiven semantischen Fernsehens (Semantic TV) setzt, detailliert vorgestellt.

Durch die technische Realisierung von Swoozy wurde eine nahtlose Verbindung zwischen (nicht- bzw. linearen) Videos, den Cloud-basierten Diensten und der semantischen Verarbeitung geschaffen. Somit erhalten Zuschauer durch das semantische Fernsehen eine neue Möglichkeit mit Videoinhalten zu interagieren und effizient nach Hintergrundwissen zu suchen.

Dr. Matthieu Deru ist Senior Software Ingenieur und UX-Designer für interaktive Systeme im Forschungsbereich "Intelligente Benutzerschnittstellen" am Deutschen Forschungszentrum für Künstliche Intelligenz GmbH (DFKI) in Saarbrücken. Für die Realisierung von "Swoozy" erhielt er 2013 einen CeBIT Innovation Award.