Konzeption
Web-UIS 3.0

C. Düpmeier (Gesamtredaktion);
T. Schlachter; R. Weidemann
Karlsruher Institut für Technologie
Institut für Angewandte Informatik
Hermann-von-Helmholtz-Platz 1
76344 Eggenstein-Leopoldshafen

F. Chaves; U. Bügel; J. Moßgraber; B. Schnebel; T. Usländer
Fraunhofer Institut für Optronik, Systemtechnik und Bildauswertung
Fraunhoferstr. 1
76131 Karlsruhe

W. Schillinger; R. Ebel; M. Tauber, B. Nonnenmann; A. Koch; D. Bollinger;
E. Schöpflin-Reichmann
LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg
Griesbachstr. 1
76185 Karlsruhe

K. Zetzmann; R. Rossi
Ministerium für Umwelt, Klima und Energiewirtschaft Baden-Württemberg
als Träger des F & E Vorhabens MAF-UIS
Kernerplatz 9
70182 Stuttgart

Version	***Datum***
1.0	*21.05.2013*

INHALTSVERZEICHNIS:

1 Einführung

Das Ministerium für Umwelt, Klima und Energiewirtschaft Baden-Württemberg (UM) und die Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg (LUBW) betreiben für das Umweltinformationssystem Baden-Württemberg (UIS BW) seit 1996 Webangebote mit dem Ziel, Umweltinformationen auf wirtschaftliche Weise für die Öffentlichkeit verfügbar zu machen. Im Intranet der beiden Häuser und im Landesverwaltungsnetz bestehen ebenfalls Webangebote für den verwaltungsinternen Gebrauch bei den Umweltdienststellen von Land und Kommunen (UIS-Landesintranet).

Als Content-Management-System (CMS) [1] wird seit 2004 im Geschäftsbereich die Entwicklungsplattform WebGenesis [9] vom Fraunhofer Institut für Optronik, Systemtechnik und Bildauswertung (IOSB) als technische Grundlage der Webpräsentationen der LUBW [19], des UM [20] und vieler weiterer Unterangebote eingesetzt. Damit werden vor allem Texte, Bilder und Dokumente im Landesdesign präsentiert. WebGenesis ist jedoch vom IOSB nicht als reines CMS, sondern eher als Entwicklungsplattform konzipiert. Es wird im Geschäftsbereich des UM als Entwicklungsplattform für diverse UIS-Komponenten, wie das Umweltportal Baden-Württemberg [21], das Fachdokumenten-Managementsystem FADO [22] oder den Themenpark Umwelt [23], eingesetzt.

Seit 2008 ist für die Unterstützung einer effizienten Suche nach Umweltinformationen eine Google Search Appliance (GSA) [37] im Einsatz, für die ebenfalls Erweiterungen von WebGenesis entwickelt wurden. Damit wird im Unterschied zu der Standardsuchfunktion eines CMS, die immer nur beschränkt auf die eigene Website suchen kann, eine übergreifende Suche in allen Domänen und Websites des Geschäftsbereichs und darüber hinaus ermöglicht.

Sowohl WebGenesis als auch die GSA werden von unterschiedlichen Partnern (andere Bundesländer, Forschungseinrichtungen, Landesministerien) im Rahmen von Kooperationen mit genutzt. Alle diese WebGenesis-Angebote werden von der LUBW gehostet.

WebGenesis [9] wird seit 2004 zusammen mit dem Fraunhofer IOSB und dem Karlsruher Institut für Technologie (KIT) in bedarfs- und finanzorientierten Schritten den zunehmenden Anforderungen des UIS angepasst und weiterentwickelt. Es ist jedoch festzustellen, dass moderne CMS inzwischen über Funktionen verfügen, die WebGenesis nicht oder nur unvollständig bietet, da sich die funktionale Erweiterung von WebGenesis nur in geringem Maße auf CMS-Funktionalitäten konzentriert. Als Beispiele für fehlende Funktionalitäten seien genannt:

- Komfort bei der redaktionellen Arbeit (Bildbearbeitung etc.)
- Einbindung von Social-Media-Plattformen
- flexible Anpassung von Inhalten an unterschiedliche Medien bzw. Hardware (mobile Geräte)

Bei der Präsentation von Umweltdaten in Webangeboten werden dynamische Techniken benötigt, um möglichst zeitnah auch aktuelle Daten präsentieren zu können. Auch dafür wird teilweise WebGenesis im Sinne einer Entwicklungsplattform eingesetzt, z.B. im Portal zur

Kernreaktorfernüberwachung (KFÜ) des UM oder dem Angebot Fließgewässer der LUBW. Vorteilhaft ist hierbei, dass WebGenesis in Java programmiert ist, was auch als Standard für Entwicklungen von Fachsystemen im UIS BW festgelegt wurde (etwa in der aktuell gültigen UIS-Rahmenkonzeption) [38]. Daher lassen sich fachanwendungsnahe Funktionalitäten wirtschaftlich mit WebGenesis umsetzen. Eine Ablösung von Webanwendungen bzw. die Portierung solcher Funktionalitäten in eine neue Programmierumgebung ist deutlich komplexer und aufwändiger als eine reine Portierung von Inhalten („Content“) in Form von Texten und Bildern.

Mit der Einführung des CMS wurde eine weitgehende Vereinheitlichung für die Publikation der bis dahin statischen HTML-Seiten erreicht. Neben WebGenesis sind aber noch einige weitere Web-basierte Fachanwendungen, z.B. „Umwelt-Datenbanken und -Karten Online“ [39] auf Basis von Cadenza Web [40] im UIS im Einsatz, die nicht mit WebGenesis oder einem anderen CMS realisiert sind. Solche Systeme werden über Verlinkung und eine übergreifende Suchfunktion mit den Portalen und Homepage-Anwendungen vernetzt, um dem Nutzer den Eindruck eines homogenen Angebots zu vermitteln.

1.1 Veränderte Rahmenbedingungen im Land

Das Staatsministerium (StM) hat über eine europaweite Ausschreibung die Entwicklung einer neuen Systemplattform für die Internetangebote zumindest des Landesportals und der Ministerien auf Basis des Open-Source-CMS TYPO3 [11] vergeben. Das Landesportal ist seit 1. Februar 2013 auf dieser neuen Plattform in Betrieb. Wesentliche Neuerungen sind das neue Landesdesign mit dem Schwerpunkt auf Multimedia-Inhalten, der Einbindung von Social Media und die automatisierte Anpassung der Darstellung für mobile Geräte. Die neue Systemplattform soll den Ministerien ab März 2013 für eigene Auftritte zur Verfügung stehen. Es soll jedoch auch möglich sein, die technischen Elemente für das einheitliche Landesdesign in eigene Systemumgebungen zu übernehmen.

Die bisher bekannten Restriktionen der Landesplattform scheinen eine Vernetzung über eine übergreifende Suche nicht mehr zu erlauben. Die Landesplattform erscheint lediglich für eine Publikation von Bildern, Videos, Texten und Dokumenten geeignet. Die dynamische Einbindung von Inhalten und Daten auf programmtechnischer Ebene über Programmierschnittstellen (API) ist nicht vorgesehen. Damit ist die Landesplattform als Entwicklungsplattform für Webanwendungen im Umweltbereich nicht geeignet.

Das Kultusministerium hat sich bereits entschieden, seine bisherige Plattform pirobase [41] beizubehalten und lediglich das Design anzupassen.

Das Innenministerium lässt durch T-Systems bereits jetzt eine Plattform auf Basis von MS Sharepoint [42] betreiben, die prinzipiell ebenfalls von anderen Ministerien genutzt werden könnte. Nutzungsbedingungen sind dafür jedoch nicht bekannt. Die Sharepoint-Plattform eignet sich dabei besonders gut zur Integration Microsoft-basierter Anwendungen, wie die Bürokommunikationsdienste von Microsoft, in Webanwendungen.

Im März 2013 wurde ein neues Landeslayout (Styleguide) vorgestellt, an dem sich die Webangebote des Landes zu orientieren haben.

Die bestehende Vielfalt unterschiedlicher Plattformen im Land wird vom Rechnungshof immer wieder kritisiert. Im Sinne von Synergieeffekten wird zunehmend auf eine Vereinheitlichung gedrängt. Die besonderen Anforderungen des Umweltressorts für komplexe Webanwendungen zur Publikation von Umweltdaten werden vom Rechnungshof jedoch nur unzureichend berücksichtigt.

1.2 Ziel der Konzeption

Im UM ist das Thema „Präsenz im Internet“ zur Chefsache erklärt worden. Struktur und Inhalte der UM-Homepage werden derzeit auf der bestehenden Plattform an die neuen Rahmenbedingungen der (zu vermutenden) neuen Layoutvorgaben angepasst, um sie dann zeitnah unverändert auf eine neue Plattform übernehmen zu können. Auf konzeptioneller und technischer Ebene wurde erkannt, dass geklärt werden muss, ob auch künftig WebGenesis die Plattform für möglichst viele Webangebote bleiben kann. Dafür ist zu untersuchen, ob und mit welchem Aufwand WebGenesis für die neu entstandenen Anforderungen ertüchtigt werden kann.

Sollte das wirtschaftlich und innerhalb eines verträglichen Zeitrahmens nicht oder nur unzureichend möglich sein, ist zu klären, ob künftig ggf. für bestimmte Funktionen und Angebote eine andere Plattform genutzt werden muss. Das UM erwägt für den „politischen“ Teil des UM-Angebots, mit den beschriebenen Restriktionen, die neue Plattform des Landes zu nutzen. Die Anforderungen an Entwicklungsplattformen für Webanwendungen des UIS erlauben diesen Plattformwechsel jedoch nicht. Damit wäre die Infrastruktur für die Webangebote und -anwendungen im UIS noch vielfältiger als bisher.

Deshalb ist eine vergleichende Betrachtung von alternativen Systemen geboten, die den Anforderungen des UIS im Sinne der UIS-Rahmenkonzeption gerecht werden:

- Investitionssicherheit für die bestehenden UIS-Webanwendungen
- Zukunftssicherheit der UIS-Entwicklungsstrategie „Wirtschaftlicher Einsatz verfügbarer personeller und finanzieller Ressourcen“
- Berücksichtigung von künftig sinkenden personellen und finanziellen Ressourcen

Berücksichtigt werden sollte auch der politische Wille zur weitgehenden Herstellerunabhängigkeit (Open Source).

Die vorliegende, insbesondere von KIT und Fraunhofer IOSB ausgearbeitete Konzeption Web-UIS 3.0 soll die oben genannten Fragen näher beleuchten und Lösungsvorschläge machen. Besonders im Blick soll dabei die Entwicklung des UIS BW hin zu einem modernen, service-orientierten Internet-basierten Dienstleistungsangebot sein, das Umweltinformationen nicht nur für einen rein Browser-basierten Zugriff über Desktop-Computer bereitstellt, sondern auch mobilen und anderen Anwendungen offene Schnittstellen zum Zugriff auf die Informationen bietet.

Dabei sollen nicht nur rein Öffentlichkeitsarbeits-nahe Informationen, wie Pressemitteilungen, Social Media Beiträge oder Blogposts, über das Web zugänglich gemacht werden, sondern

gerade auch im Sinne des Umweltinformationsgesetzes Umwelt-Fachinformationen für die Öffentlichkeit in sinnvoller Weise bereitgestellt werden. Hierfür müssen Informationen aus verschiedenen Fachinformationssystemen auf intelligente Weise über Services miteinander verknüpft und integriert angeboten werden („Linked Data“ [43] oder „Linked Open Data“ [44]), was zugleich eine wesentliche Grundlage des Web 3.0 darstellt.

Außer der bisher verwendeten Web-Entwicklungsplattform WebGenesis [9] und dem CMS TYPO3 [11], das für den Landesserver eingesetzt wird, wurde für eine Einschätzung, was heute bereits mit moderner Portal-Software [6] [7] realisierbar ist, die Open-Source-Software Liferay [45] [30] als alternative Java-basierte Lösung zum Aufbau von Enterprise Portalen in den Vergleich mit einbezogen. Liferay gehört laut einer Studie von Gartner Inc. von 2012 [15] [16] als einzige Open Source Portal-Software zu den führenden Web-Portallösungen und wird in der „Open Government Data Deutschland“-Studie des Bundesministerium des Innern (BMI) [46] als Open Source Lösung für Behördenangebote empfohlen. Liferay wird weiter in einigen Portalen der öffentlichen Verwaltung, z.B. leo-bw.de und govdata.de, bereits eingesetzt.

Weiter soll das Enterprise Content-Management-System pirobase, das ebenfalls im Land Baden-Württemberg bei Behörden im Einsatz ist, als mögliche Alternative mit in den Vergleich (siehe Kapitel 5) einbezogen werden.

1.3 Gliederung der Studie

Im folgenden Kapitel 2 wird zunächst eine Bestandsaufnahme der momentan vorhanden Web-basierten Informationssysteme im UIS durchgeführt. Die Systeme werden dabei grob in 3 Klassen eingeteilt:

- reine **CMS-Anwendungen** (Systeme zum Management öffentlichkeitsarbeitsnaher Webinhalte)
- **Web-Systeme**, die Web-basierte Fachanwendungen oder komplexe interaktive Fachinformationsinhalte enthalten (fachanwendungsnahe Systeme)
- **Portale** (Plattformen zur Integration verschiedener Inhalte aus Fremdsystemen)

Dabei wird auch in Systeme für verschiedene Nutzergruppen unterschieden:

- **allgemeine Öffentlichkeit** (Internet mit Zielgruppe allgemeine Öffentlichkeit)
- **Fach-Öffentlichkeit** (Internet-basiertes System mit Zielgruppe Fachanwender)
- **Fachanwender Intranet** (Intranet-Webanwendung für interne Fachanwender)

Zu jedem System wird auch die technische Plattform gelistet, mit der das System implementiert ist.

Kapitel 3 gibt einen Überblick über den augenblicklichen „State oft the Art“ des Web und konzentriert sich dabei auf die Beschreibung der aktuellen und zukünftigen Webtechnolo-

gien, die für das Verständnis der zukünftigen Anforderungen und des anschließenden Konzeptes für das Web-UIS 3.0, das in Kapitel 7 beschrieben wird, notwendig sind.

Kapitel 4 definiert den Katalog von Anforderungen, der aus Sicht von UM und LUBW bei der Konzeption des Web-UIS 3.0 zu berücksichtigen ist.

In Kapitel 5 werden die auf Basis der Beschreibung des aktuellen Technologiestandes in Kapitel 3 sowie der Anforderungen in Kapitel 4 ermittelten Ergebnisse einer Analyse der bestehenden Systeme aufgelistet, Defizite innerhalb der aktuellen Web-UIS Systemlandschaft identifiziert und notwendiger Modernisierungsbedarf formuliert.

Da die Analyse von WebGenesis in Kapitel 5 auf die für das Web-UIS eingesetzten Versionen von WebGenesis basiert, die nicht mehr den aktuellen Entwicklungsstand von WebGenesis widerspiegeln, gibt Kapitel 6 einen kurzen Überblick über den aktuellen Entwicklungsstand sowie die momentan geplanten Entwicklungsarbeiten von WebGenesis.

Kapitel 7 enthält schließlich einen detaillierten Vorschlag eines Web-UIS 3.0 für die Neuausrichtung der Webangebote des UIS und Kapitel 8 fasst die wesentlichen Ergebnisse nochmals kurz zusammen und gibt einen Ausblick auf ein mögliches weiteres Vorgehen.

2 Bestandsaufnahme

Die Nutzung des World Wide Web für die Bereitstellung von Informationen aus dem UIS BW hat bereits 1996 begonnen. Hinzu kamen die Selbstdarstellung der einzelnen Dienststellen und Präsentation ihrer Arbeitsergebnisse im Web, später dann der Aufbau von Portalen, in denen die vielfältigen Webangebote zusammengefasst werden. Bereits Ende der neunziger Jahre wurde erkannt, dass ein Content-Management-System für die vielfältigen Webdarstellungen benötigt wird.

Üblicherweise wird bei neuen Webangeboten die Nutzung des seit 2004 eingesetzten Systems WebGenesis vorgeschrieben und nur in begründeten Ausnahmefällen werden andere Techniken, wie PHP-Programme, Active-Server-Pages, etc. verwendet. Diese werden dann per IFrame in die WebGenesis-Seiten eingebunden, verlinkt oder durch Suchportale vernetzt.

Einige ältere Webangebote konnten nicht nach WebGenesis umgestellt werden. Außerdem werden die Webseiten der Hochwasservorhersagezentrale Baden-Württemberg und der Messnetzzentrale Luft nicht mit WebGenesis realisiert, sondern regelmäßig in der Messnetzzentrale als HTML-Seiten generiert und dann auf dem Webserver www2.lubw.baden-wuerttemberg.de und zusätzlich bei einem externen Provider bereitgestellt.

Für Geoanwendungen und das UIS-Berichtssystem stehen eigene Server, z.T. auch im Internet, zur Verfügung. Die Inhalte werden in die WebGenesis-Seiten eingebunden oder verlinkt.

Aufgrund von Umstrukturierungen der Ministerien kamen in den letzten Jahren auch Webangebote hinzu, die von anderen Organisationseinheiten in Auftrag gegeben wurden und bei externen Providern bereitgestellt werden. Diese Webangebote stellen „Insellösungen“ dar, wenn es sich um UIS-relevante Themen handelt, sollten sie langfristig in die UIS-Infrastruktur überführt werden.

2.1 WebGenesis-Webangebote

2.1.1 WebGenesis Standard + Corporate-Design-Baukasten

Diese Webangebote beinhalten den größten Teil der Web-Selbstdarstellung der Dienststellen im Ressort des Ministeriums für Umwelt, Klima und Energiewirtschaft. Die Autoren können mit einer einfachen Oberfläche Webseiten erstellen, die automatisch dem Corporate Design des Landes entsprechen. Die LUBW hat derzeit beispielsweise ca. 200 Autoren, die jeweils ihr Internet-Angebot bearbeiten. Es gibt einige Formulare für Spezialanwendungen, z.B. für den LUBW-Bestellshop, für Pressemitteilungen, etc., die spezielle Autoren nutzen können.

Im Intranet werden dieselben Techniken wie im Internet eingesetzt.

Webangebote der LUBW

Intranet

- LUBW Intranet http://cms.lubw.bwl.de/
- UIS Landesintranet http://www.lubw.bwl.de/
  - Grundwasser Dokumente WIBAS http://gwdok.lubw.bwl.de/

Internet

- LUBW Internet http://www.lubw.baden-wuerttemberg.de/
  - REACH http://www.reach.baden-wuerttemberg.de/
  - PLENUM http://www.plenum-bw.de/

Webangebote des UM

- Intranet http://www.um.bwl.de/
  - Informationsangebot für Land und Kommunen http://portal.um.bwl.de/
- Internet Hauptseite http://www.um.baden-wuerttemberg.de/
  - Weitere WebGenesis-Angebote http://www2.um.baden-wuerttemberg.de/, mit verschiedenen Quereinstiegen, d.h. eigene Adressen für Unterangebote

Webangebote der Gewerbeaufsicht (GWA)

- Intranet http://www.gaa.bwl.de/
- Internet http://www.gaa.baden-wuerttemberg.de/

2.1.2 WebGenesis KIT/IAI

Diese Webangebote beinhalten zusätzliche Funktionen, die mit der WebGenesis-Entwicklungsplattform programmiert wurden. Der Autorenkreis ist beschränkt auf Experten, die Autoren- bzw. Erfassungs-Oberfläche komplexer oder erfordert spezielles Informationstechnisches Know-how.

- Fachdokumente Online (FADO) http://www.fachdokumente.lubw.baden-wuerttemberg.de/
- Themenpark Umwelt http://www.themenpark-umwelt.baden-wuerttemberg.de/
- Umweltportale
  - Portal Umwelt BW http://www.umwelt.baden-wuerttemberg.de/
  - Energieportal BW http://www.energie.baden-wuerttemberg.de/

2.1.3 WebGenesis - Sonstige Anwendungen

Diese Spezialanwendungen wurden im Auftrag von UM und Fachabteilungen der LUBW vom Fraunhofer IOSB entwickelt und können derzeit auch nur vom IOSB gepflegt werden. Ziel ist es, die Anwendungen in die UIS-Standard-Infrastruktur zu integrieren.

- Jahresdatenkatalog Fließgewässer http://jdkfg.lubw.baden-wuerttemberg.de/
- Jahresdatenkatalog Grundwasser http://193.197.158.205/

Folgende Anwendungen werden vom IOSB in Zusammenarbeit mit T-Systems im Auftrag des UM und des IM entwickelt und betrieben. Eine Umstellung auf ein anderes System ist bisher nicht vorgesehen:

- Elektronische Lagedarstellung für den radiologischen Notfallschutz (nicht öffentlich)
- Portal der Kernreaktor-Fernüberwachung Baden-Württemberg (nicht öffentlich)
- Elektronische Lagedarstellung für den Bevölkerungsschutz im Auftrag des IM (nicht öffentlich)

Folgende Anwendungen wurden im Auftrag des UM und der Gewerbeaufsicht 2012 entwickelt und werden 2013 in Betrieb gehen:

- Datenbank für Vorschriften und Erlasse (DAVE) integriert im Gewerbeaufsicht-Intranet (geht demnächst in Betrieb)
- Visualisierung aktueller Messdaten zur Fließgewässerbeschaffenheit (FISGeQua) http://www.lubw.baden-württemberg.de/servlet/is/78491/

2.2 Andere Webangebote

Es handelt sich überwiegend um Webangebote, die nicht mit einem Content-Management-System erstellt werden können, z.B. dynamisch generierte HTML-Seiten aus den Messnetzen oder Kartenanwendungen.

Intranet der LUBW:

- BRS-Web Berichtssystem (Cadenza-WebVersion inklusive GISterm) http://uisprod.lubw.bwl.de/brs-web/index.html
- DVZ Diensteverzeichnis des BRS http://dvz.lubw.bwl.de/wiki/index.php/Hauptseite
- TA Luft WIKI http://taluftwiki-leitfaden.lubw.bwl.de/
- DRS Document Retrieval System des Fachdienstes Wasser http://drs.lubw.bwl.de/
- HVZ Hochwasservorhersagezentrale http://hochwasser.lubw.bwl.de/

Internet der LUBW

- UDO Umwelt-Datenbanken und -karten Online (Cadenza-WebVersion) http://brsweb.lubw.baden-wuerttemberg.de/brs-web/index.xhtml
- LINUX-Webserver der LUBW für Webangebote ohne WebGenesis (z.B. alte Webangebote, HTML-Seiten oder PHP-Anwendungen, die nicht umgestellt werden konnten), die überwiegend per IFrame in das Hauptwebangebot der LUBW eingebunden werden http://www2.lubw.baden-wuerttemberg.de
- RIPS (Metadatenauskunft) http://rips-uis.lubw.baden-wuerttemberg.de/rips/ripsmdk/
- Kartendienste http://rips-dienste.lubw.baden-wuerttemberg.de/ …
- DRS (Document Retrieval System) des Fachdienstes Wasser http://www.drs.baden-wuerttemberg.de/

- HVZ Hochwasservorhersagezentale http://www.hvz.lubw.baden-wuerttemberg.de/
- MNZ Messnetzzentrale Luft http://mnz.lubw.baden-wuerttemberg.de/
- Bodensee Online http://www.bodenseeonline.de/
- TA Luft WIKI im Internet noch nicht freigeschaltet

Internet des UM

- Betrieblicher Umweltschutz http://www.umweltschutz-bw.de/
- Versorgerportal Baden-Württemberg http://www.versorger-bw.de/
- Weitere Angebote bei Providern, die für einzelne Veranstaltungen, Projekte und Aktionen entwickelt wurden, z.T. bei Umstrukturierungen der Ministerien hinzukamen

3. State of the Art

Die Entwicklung der Webtechnologien verlief über die letzten 20 Jahre rasant. Während in den Anfängen des Webs Webserver nur HTML-Dateien als statische Dokumente an Webbrowser auslieferten, kam schnell der Wunsch auf, Webseiten auch dynamisch zum Zeitpunkt des Abrufs vom Server über Programme erzeugen zu können.

Hierzu wurden für Webserver Schnittstellen, zunächst das Common Gateway Interface (CGI) [47] entwickelt, mit denen sie externe Programme, die diese Schnittstellen bedienen, zur Erzeugung von Webseiten dynamisch aufrufen können (siehe Abb. 3.1). Auf diesen Webserver-Schnittstellen aufbauend entstanden für die unterschiedlichsten Programmiersprachen schnell spezifische Programmierbibliotheken, mit denen man CGI-Programme in der jeweiligen Programmiersprache erstellen kann (z.B. für die Sprachen Perl, C, C++). Alternativ hierzu wurden aber auch vollständig neue Programmiersprachen wie PHP [13] entwickelt, die auf das Schreiben von CGI-Programmen für Webanwendungen spezialisiert waren und sich mit bestehenden Webservern bestens integrieren.

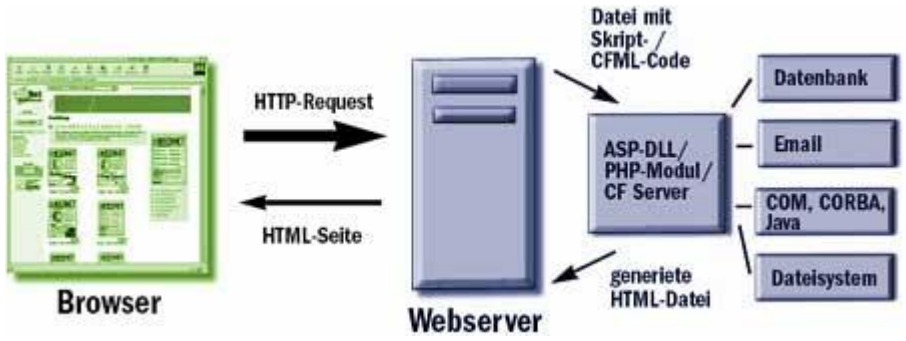


Abb. 3.1: Webserver mit CGI-basierten Erweiterungen (ASP-DLL; PHP-Modul, …)

In der Java- oder .NET-Welt wiederum wurden nicht neue Sprachen, sondern auf dynamische Webanwendungen spezialisierte Webserver entwickelt (Servlet-Engines [48] in Java, IIS-Server in .NET [49]), die Programmierern von vorneherein ein stark vereinfachtes, komponentenorientiertes Programmiermodell zum Schreiben von dynamischen Webanwendungen zur Verfügung stellen. Abb. 3.2 zeigt, wie eine Webanfrage von einem Client innerhalb eines Java-Webservers an eine dedizierte Softwarekomponente (Servlet [48]) zur Bearbeitung weitergeleitet wird, die den Web-Request dann beantwortet.

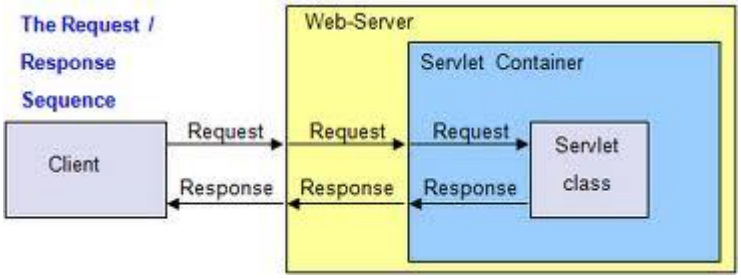


Abb. 3.2: Java Servlet-Engine als Webserver: Servlets sind hier Komponenten, die von der Servlet-Engine gehostet werden und Anfragen zu bestimmten URLs bearbeiten und beantworten

Aufbauend auf solchen Grundtechnologien zur dynamischen Erzeugung von Webseiten entstanden dann oberhalb der Grundfunktionalität angesiedelte, sprachspezifische Softwarebaukästen (Web-Frameworks), die das Schreiben von dynamischen Webanwendungen auf der Serverseite weiter vereinfachen.

Parallel zu den Web-Frameworks zur Programmierung von Webanwendungen entstanden eine Vielzahl von konkreten Webanwendungen für bestimmte Anwendungsbereiche: Spezialisierte Systeme zur Verwaltung von Webseiten und zugehörigen Medieninhalten (Content-Management-Systeme, CMS [1] [3] [4]), auf die Verwaltung von vielen Dokumenten oder binären Inhalten spezialisierte Systeme (webbasierte Dokumentenmanagementsysteme [51]), Online-Foren, Systeme zur Verbreitung aktueller Nachrichten (Blogs und Newssysteme), Bildverwaltungs- und Präsentationsdienste sowie spezialisierte Dienstleistungsangebote, wie Suchmaschinen, Hosting-Dienste für Bilder, Kalenderdaten, Dokumente, Bookmarking-Dienste, Soziale Netzwerke etc. Dabei wurden die Anforderungen dieser Anwendungen an die darunterliegenden Web-Entwicklungsframeworks und Webstandards immer größer.

Bei der Entwicklung solcher komplexer Webanwendungen stellte sich heraus, dass sich mit den zu dieser Zeit verfügbaren Hilfsmitteln HTML-Seiten zwar sehr gut zur Anzeige beliebiger Informationen in einem Webbrowser dynamisch erzeugen ließen, ergonomischere Oberflächen mit einem hohen Grad an Benutzerinteraktivität (wie sie z.B. für Autorenumgebungen und Datenerfassungsszenarien notwendig sind) aber nur unzureichend implementierbar waren. Für solche Anwendungen müssen innerhalb des Webbrowsers komplexere Abläufe als Code auf Clientseite ablaufen können, die abhängig von der Interaktion des Benutzers die Webseite auf der Clientseite im Browser dynamisch in Teilbereichen modifizieren und dabei im Hintergrund bei Bedarf Daten in einem maschinenlesbaren Format vom Server nachladen, ohne dass der Benutzer hier explizit eingreifen oder die Webseite komplett neu laden muss.

Dies führte zunächst zur Implementierung einer neuen Programmiersprache JavaScript [33], mit der lauffähiger Code in HTML-Seiten eingebettet werden kann, so dass dieser im Browser auf der Clientseite ausgeführt wird. Zum Durchgriff auf die Struktur der Webseite von JavaScript aus wurde eine Programmierschnittstelle für JavaScript entworfen, die das gesamte HTML-Dokument einer Webseite als Baum von JavaScript-Objekten (Document Object Model, kurz DOM [51]) in JavaScript zur Verfügung stellt, um die Webseite dynamisch modifizieren zu können. Hierzu musste auch das zugehörige HMTL-Modell, das die Basis für den DOM darstellt, entsprechend aktualisiert werden (dies führte zu den Standards HTML 4 [52] und JavaScript 1.5 [53]).

JavaScript-Funktionen lassen sich an Interaktionen der Benutzer (Anklicken von Elementen der Webseite, Mausbewegungen, etc.) mit der Webseite binden (Registrierung von Ereignis-Behandlungsfunktionen) und können so dynamisch im Browser mit dem Nutzer interagieren. In einem nächsten Evolutionsschritt wurden die JavaScript-Funktionalitäten in den Webbrowsern so erweitert, dass JavaScript-Code asynchron zu anderen Aktivitäten des Browsers (z.B. bei Interaktion mit dem Nutzer) mit Internet-Servern kommunizieren kann, um maschinenlesbare Daten (z.B. im XML- oder JSON-Format) für Nutzerinteraktionen im Hintergrund von Servern nachzuladen, ohne dass hierdurch die Interaktion mit dem Nutzer im Vordergrund unterbrochen und die gesamte Webseite neu geladen werden muss.

Diese Kommunikationsmechanismen bezeichnet man wegen ihrer prinzipiellen Arbeitsweise daher auch als Asynchronous JavaScript and XML (AJAX) [54], wobei die Daten durchaus nicht nur in XML-Datenformat, sondern auch anderen Formaten wie dem JSON-Format vom Server auf den Client übertragen werden können. Mit JavaScript- und AJAX-Unterstützung lassen sich nun viel komplexere, interaktivere und ergonomischere Webanwendungen entwickeln, bei denen die Interaktion des Nutzers mit der Webanwendung im Vordergrund steht, und die Nutzer aktiv Inhalte für das Internet erstellen oder hochladen und anderen Internetnutzern zur Verfügung stellen. Sowohl die hierfür notwendige Web-Technologie (JavaScript, AJAX-Unterstützung) als auch die hierdurch entstandene Art von hochgradig interaktiven und Nutzer-Inhaltsgetriebenen Webanwendungen bezeichnet man heute mit dem Schlagwort „Web 2.0“.

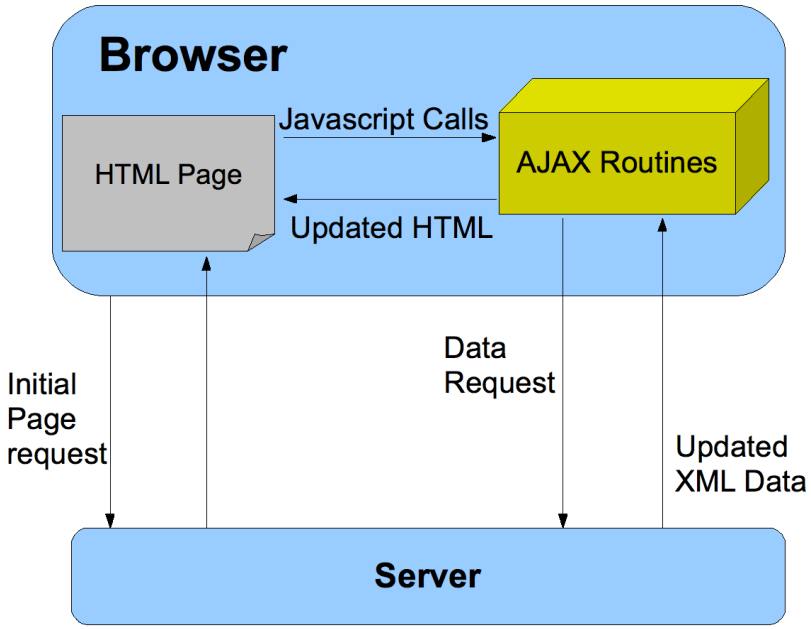


Abb. 3.3: Asynchrones Nachladen von Informationen im Browser über Asynchronous JavaScript and XML (AJAX)

Damit asynchron ausgeführter JavaScript-Code im Browser Daten von Servern nachladen kann, sind serverseitige Kommunikationsdienste notwendig, die vom JavaScript-Code angesprochen werden können und auf Anfrage die benötigten Daten in maschinenlesbaren Formaten liefern (vgl. auch Abb. 3.3). Mit dem Aufkommen der „Web 2.0“-Anwendungen wurde daher die Entwicklung und Standardisierung von solchen serverseitigen Kommunikationsdiensten (sogenannten Web-Services [56]) nicht nur für die AJAX-Kommunikation innerhalb einer Webanwendung, sondern auch zum Austausch von Daten zwischen Webanwendungen, z.B. für Business-to-Business-Anwendungen (B2B), immer wichtiger.

Im Rahmen der internationalen Standardisierung von Web-Services wurden zunächst die SOAP-basierten Web-Services [55] [56] entwickelt. Diese folgen den Prinzipien von klassischen Verteilten Kommunikationsschnittstellen und kapseln den Kommunikationsverkehr zwischen den Kommunikationspartnern vollständig in ein eigenes XML-basiertes Kommunikationsprotokoll, das SOAP (Simple Object Access Protocol) genannt wird (daher der Name SOAP-basierte Web-Services oder klassische Web-Services). Die Nutzung eines eigenen Kommunikationsprotokolls und die Abbildung klassischer verteilter Kommunikationsfunktio-

nalitäten oberhalb des eigentlichen Webkommunikationsprotokolls HTTP (Hypertext Transfer Protokoll) stellt aber durchaus einen beträchtlichen zusätzlichen Aufwand da, da insbesondere das HTTP-Protokoll ganz andere Eigenschaften wie die Kommunikationsprotokolle, die für klassische Objektkommunikationsmechanismen entwickelt wurden, aufweist. Aus diesem Grunde entstand parallel zu den SOAP-basierten Web-Services eine weitere Art von Web-Service-Standard (REST-basierte Web-Services) [57], der die Grundfunktionalitäten des Webs weitaus optimaler nutzt als die SOAP-basierten Web-Services. REST steht dabei für Representational State Transfer.

REST-basierte Web-Services benötigen im Gegensatz zu SOAP-basierten Diensten kein zusätzliches Verpackungsformat und sind daher wesentlich einfacher zu implementieren. Darüber hinaus nutzen sie in natürlicher Weise vorhandene Webinfrastrukturen wie Cache-, Proxyserver und Lastverteilungsdienste und sind somit auch performanter und skalierbarer als SOAP-basierte Web-Services. Daher werden für viele größere Internetdienste bevorzugt REST- und nicht SOAP-basierte Web-Services eingesetzt. REST-basierte Services sind auch die idealen serverseitigen Dienstschnittstellen zur Unterstützung von AJAX-Funktionalitäten des JavaScript-GUI-Codes im Browser.

Für die komfortable Programmierung von Webanwendungen mit Desktop-ähnlichem Komfort reichten die durch den HTML-4-Standard gegebenen Grundfunktionalitäten und die ersten Webframeworks auf Basis dieser Standardgeneration bei Weitem nicht aus. Außerdem erforderte das Aufkommen neuer Mobiltechnologien ebenfalls eine Verbesserung und Überarbeitung der bestehenden Webstandards. Daher wurden die bestehenden Standards im Laufe der Zeit erweitert und ergänzt.

Die Arbeiten an neuen Standards für Webdesign und Web-Anwendungsaspekte werden beim W3C unter dem Stichwort „Web Design und Applications“ [58] zusammengefasst. Sie betreffen die Arbeiten an neuen HTML-Standards (HTML5 und HTML5.1) und zugehörigen Standards, wie CSS, JavaScript, den JavaScript-DOM-Schnittstellen und JavaScript-Web APIs, aber auch weiteren Standards, z.B. für behindertengerechtes Design von Webseiten und Anwendungen, Einbindung von Vektorgrafik, Medien oder auch mathematischen Beschreibungen in Webseiten oder besseren Zugang für mobile Geräte u.a.

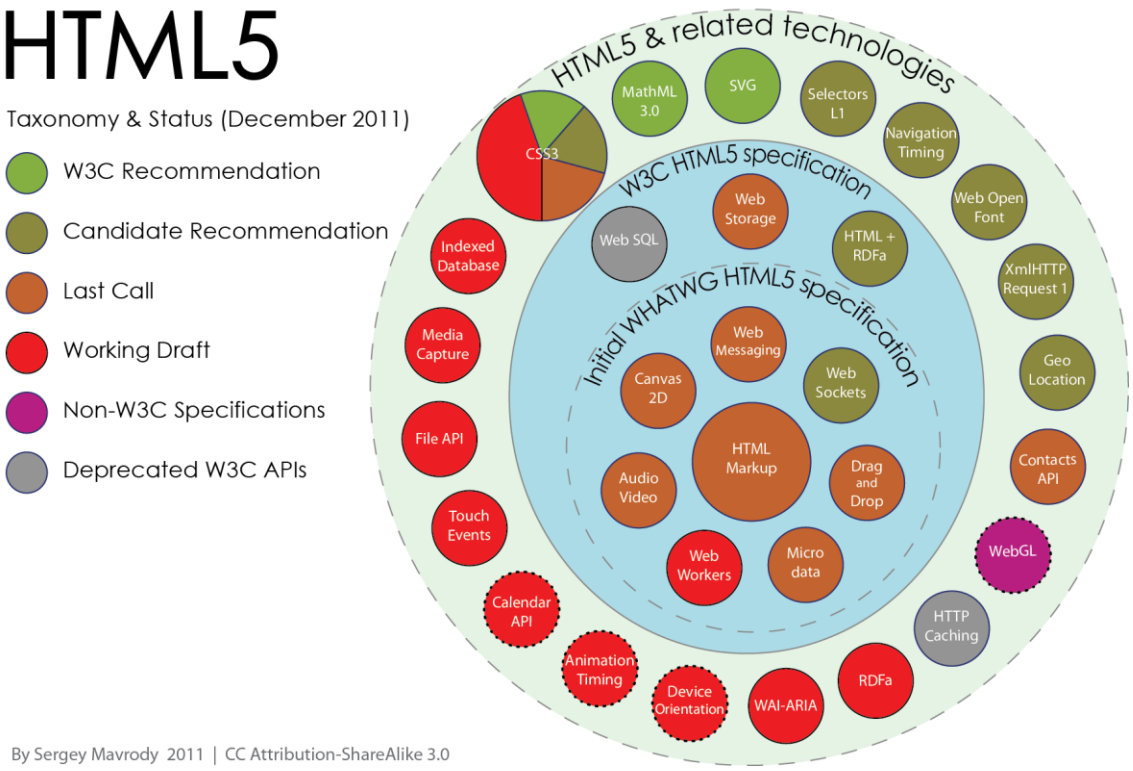


Abb. 3.4: W3C-Standards im Umfeld der HTML5-Standardisierung

Wichtige Elemente neuer Standards wie HTML5, die in neueren Browsern auch weitgehend implementiert sind, betreffen die Unterstützung neuer Sprachelemente für fluide Designs, die sich automatisch an verschiedene Browsergrößen anpassen, die Implementierung eines HTML-Elements als grafische Zeichenfläche, das die Darstellung hochperformanter 2D- und 3D-Grafik unter Nutzung von nativen Grafikbeschleunigern der jeweiligen Hardwareplattform erlaubt oder z.B. auch spezielle Elemente für das standardkonforme Abspielen von Medien, wie Audio oder Video. Die Standardisierung dieser Elemente geht einher mit der Erweiterung der JavaScript-APIs zum programmatorischen Zugriff auf diese Funktionen. Weitere wichtige Funktionalitäten betreffen die verbesserte Unterstützung von Formularelementen zum Aufbau ergonomischer interaktiver GUI-Elemente sowie für mobile Anwendungen die Unterstützung von Touchscreen-Ereignissen, um die Interaktion von Nutzern mit Touchscreens ebenfalls in Browseranwendungen unterstützen zu können. Auch die Unterstützung besser optimierter Kommunikationsschnittstellen zwischen Browser und Server, wie die WebSocket-API, und die Integration asynchroner Mechanismen in das HTTP-Protokoll sind wichtige Ergänzungen für verbesserte Webanwendungen.

Unter dem W3C-Konsortium sind mittlerweile unzählige Arbeitsgruppen zusammengefasst, die an einer Vielzahl miteinander in Bezug stehender Standards arbeiten, so dass ein Überblick über sämtliche Standards und ihre Bezüge zueinander fast unmöglich ist. Da die Standardisierung des W3C Browserherstellern wie Apple, Mozilla Corporation oder Opera nicht schnell genug voranschritt und theoretisch überfrachtet erschien, wurde von diesen die „Web Hypertext Application Technology Working Group“ (WHATWG) [59] gegründet, die mittlerweile eigene pragmatischere Standardisierungen der Kernstandards für Browser wie HTML vornimmt, die ihnen als Grundlage von neuen Funktionalitäten in ihren Webbrowsern dienen. Der HTML Standard der WHATWG-Gruppe nennt sich „Living HTML“ [60], verzichtet mittler-

weile also auf Versionsnummern und ist ein „Work in Progress“. Er deckt sich zum großen Teil mit wichtigen Elementen aus der HTML5-Standardisierung, weicht aber in einzelnen Punkten durchaus vom HTML5-Standard ab. In Bezug auf die Implementierung neuer Funktionalitäten in Browsern bietet der Living-HTML-Standard heutzutage eine gute Grundlage.

Die Versionslosigkeit des Living-HTML-Standard dokumentiert sich auch gut in der einleitenden DOCTYPE-Deklaration einer HTML5-Webseite: <DOCTYPE! html>. Diese enthält ebenfalls keinen Versionsbezug mehr.

Entscheidend bei der praktischen Arbeit mit Webstandards ist daher heutzutage nur noch bedingt was durch das W3C standardisiert ist, sondern was von den Browserherstellern und Web-Framework-Herstellern aktuell auch auf breiter Front implementiert ist. Für Entwickler von Webanwendungen ist es dabei von grundlegender Bedeutung, eigene Entwicklungen nicht mehr auf den Basisfunktionalitäten aufzusetzen, sondern moderne Web-Frameworks als Hilfsmittel zu verwenden. Diese bieten nicht nur einen größeren Programmierkomfort, sondern schützen den Programmierer auch vor Implementierungsdifferenzen in den einzelnen Browsern.

Nach dieser an der Entwicklung der Webtechnologien orientierten Übersicht sollen im Folgenden einzelne Aspekte dieser Technologien, die für die die Konzeption der Web-UIS 3.0-Infrastruktur von besonderer Bedeutung sind, detaillierter vorgestellt werden. Dabei werden zunächst Grundtechnologien, dann darauf aufsetzende Konzepte für moderne Applikationsframeworks zum Schreiben von Webanwendungen und schließlich Komplettpakete für gewisse Anwendungsgebiete, wie Content-Management-Systeme und am Schluss Serviceumgebungen, die für einzelne Anwendungsgebiete genutzt werden können, betrachtet.

3.1 HTML5 und assoziierte Standards

Die Beschreibungssprache HTML (Hypertext Markup Language) für Inhalt und Struktur von Webseiten wurde im Rahmen der HTML5-Standardisierung um einige wesentliche Funktionalitäten erweitert. Weitere neue Funktionalitäten sind durch mit HTML5 assoziierte Standards beschrieben (siehe auch die Abb. 3.4 in der Einleitung dieses Kapitels), die man zum engeren Umfeld des HTML5 Standards zählt. So sind z.B. die neuen Möglichkeiten der visuellen Stilisierung von HTML5 Elementen im CSS3 Standard festgehalten und auch die Einbettung von mathematischen Formeln (MathML 3.0) oder Vektorgrafiken (SVG) ist durch eigene Standards beschrieben. Viele neue HTML5-Funktionalitäten benötigen auf der JavaScript-Seite ebenfalls neue Schnittstellen für den programmatorischen Zugang, so dass mit dem HTML5-Standard eine Reihe von Ergänzungen der JavaScript-APIs einhergehen. Eine detaillierte Beschreibung von HTML5 findet sich in [31].

Im Folgenden sollen die Neuerungen aus dem HTML5-Umfeld vorgestellt werden, die für das Web-UIS 3.0-Konzept wichtig sind.

3.1.1 Seitenstrukturierung und semantische Auszeichnung von HTML-Elementen

Der HTML5-Standard definiert neue HTML-Tags zur Beschreibung funktionaler Teile einer Seite, wie dem Kopf- (header) und Fußbereich (footer), Menübereichen (nav) mit Navigati-

onselementen, Inhaltsbereiche (section), einzelne Artikel (article) oder Seitenboxen (aside). Mit diesen Elementen lassen sich Webseiten organisatorisch in funktionale Bereiche unterstrukturieren.

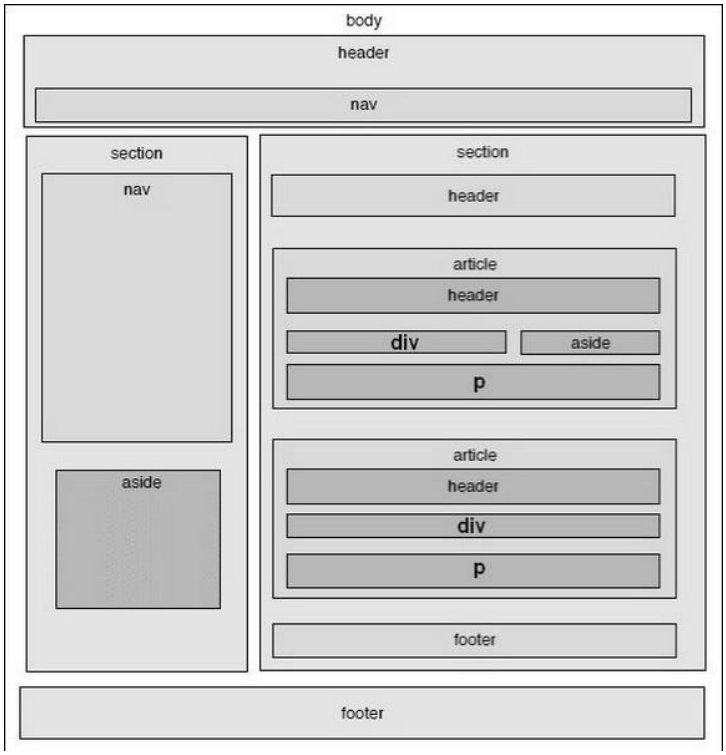


Abb. 3.5: Unterstrukturierung von Webseiten mit HTML5

Abbildung 3.5 zeigt, wie der Inhalt einer Webseite in einen Kopf- (header) und Fußbereich (footer) aufgeteilt ist, zwischen dem zwei verschiedene Inhaltsbereiche (section) platziert sind. Der Inhaltsbereich auf der linken Seite ist als eine Spalte der Webseite gedacht, die Navigationselemente (z.B. ein Menü mit Navigationslinks (nav)) und Teaserboxen oder ähnliches aufnimmt (aside). Der rechte Inhaltsbereich (section-Element rechts) nimmt den eigentlichen Inhalt der Seite auf und hat eine Blog-ähnliche Struktur (bei Blogseiten werden mehr als ein Artikel zu einer bestimmten Themenrubrik untereinander auf einer Seite gelistet). Nach einem Kopfbereich (header) folgen ein oder mehrere inhaltlich eigenständige Artikel (article, hier in der Beispielgrafik sind zwei Artikel angedeutet) zu einer Themenrubrik. Jeder Artikel hat wiederum einen Kopfbereich (header), dem andere HTML-Elemente (wie div- oder p-Elemente) folgen. Man beachte, dass jeder Artikel wiederum die Möglichkeit vorsieht, dass zum eigentlichen Inhalt (div) eine Seitenbox (aside) rechts vom Artikelinhalt stehen kann, die zusätzliche Information zum Artikel aufnehmen kann (z.B. Liste von Links auf ähnliche Inhalte, etc.). Die rechte Inhaltsspalte wird wiederum von einem eigenen Fußbereich abgeschlossen. Header- und Fußbereiche nehmen in der Regel Inhaltselemente auf, die als Metadaten eine Seite oder einen Seitenbereich näher beschreiben. Header können dabei Metadaten, wie Titel des Artikels, Informationen über den Autor, Datum der letzten Änderung oder ähnliche Dinge enthalten. Fußbereiche nehmen häufig Metadaten, wie z.B. Copyrightangaben, Link auf weiterführenden Artikel oder einen Link auf zu dem Artikel gehörige Kommentare auf.

Der folgende Codeabschnitt zeigt einen Ausschnitt des HTML5-Codes (ein konkretes Artikelelement) einer zugehörigen Webseite.

```
<article itemscope itemtype="http://schema.org/BlogPosting">
 <header>
 <h1 itemprop="headline">The Very First Rule of Life</h1>
 <p><time itemprop="datePublished" datetime="2013-02-20">3 days
ago</time></p>
 <link itemprop="url" href="?comments=0">
 </header>

 <p>If there's a microphone anywhere near you, assume it's hot and
 sending whatever you're saying to the world. Seriously.</p>
 <p>...</p>

 <footer>
 <a itemprop="discussionUrl" href="?comments=1">Show comments...</a>
 </footer>
</article>
```

Interessant an dem HTML-Ausschnitt ist, dass im HTML-Code nicht nur eine Strukturierung des Artikels in Kopfbereich, eigentlichen Inhalt und Fußbereich mit den entsprechenden Strukturierungstags vorgenommen wurde, sondern einzelne HTML-Elemente über eine Microdata-Beschreibung (die Attribute „itemscope“, „itemtype“ und „itemprop“ und weitere zugehörige Attribute des Microdata-Formates für Blogs, wie „datetime“) semantisch ausgezeichnet sind.

Über die Attributangabe itemtype="http://schema.org/BlogPosting" wird der Inhalt des gesamten Artikelelementes als „Blogeintrag“ semantisch annotiert. Das erste <h1>-Tag innerhalb des header-Bereiches des Artikels wird dabei als „headline“-Element des Blogeintrags gekennzeichnet (itemprop="headline"). Die Zeitangabe eine Zeile darunter wird entsprechend als Publikationsdatum des Blogeintrages (itemprop="dataPublished"), das Linkelement als „url“ und der Hypertextlink <a> im Fußbereich als „discussionUrl“ des Blogeintrags semantisch beschrieben.

HTML5 eignet sich also nicht nur für eine grobe strukturelle Auszeichnung von Inhaltselementen sondern erlaubt eine detaillierte semantische Beschreibung von Inhalten entweder über Microdata- (hier im Beispiel) oder RDFa-Formate (Resource Description Framework-in-attributes). Die Anzahl der semantischen Objekte, die sich dabei über solche Formate beschreiben lassen, wächst dabei ständig (siehe z.B. http://schema.org).

In Zukunft kann davon ausgegangen werden, dass immer mehr Inhalte von Webseiten auf diese Weise semantisch ausgezeichnet werden. Dies ermöglicht Suchmaschinen und anderer auf Crawlern basierender Software, gezielter Information im Internet zu erfassen und damit z.B. bessere Suchmöglichkeiten zu bieten.

Damit die Seiten einer Webanwendung und deren Inhalte entsprechend strukturiert und ausgezeichnet sind, bedarf es eines HTML5-basierten Rahmendesigns, das die passenden Strukturierungselemente verwendet. Softwareanwendungen wie CMS müssen die entsprechenden semantischen Formate dazu nutzen, die HTML-Elemente mit den entsprechenden Mikroformat- oder RDFa-Attributen zu kennzeichnen. Hierfür müssen im CMS von den Autoren entweder bereits semantisch vorstrukturierte Formulare oder aber WYSIWYG-

Webeditoren verwendet werden, welche die semantische Annotation von Inhaltselementen unter Vorgabe von semantischen Formaten unterstützen. Für solche Editoren gibt es bereits erste Versionen in der Forschung oder entsprechende Plugins für Webanwendungen, wie Wordpress, Drupal oder Joomla.

3.1.2 Verbesserte Formulare und interaktive Formulareingabeelemente

Der HTML5-Standard enthält weiterhin eine Reihe neuer Eingabetypen für Formularfelder, welche die Eingabe spezieller Datenelemente wie Datums- und Zeitangaben, Telefonnummern, Email-Adressen oder mathematischer Zahlenräume erlauben (color, date, datetime, datetime-local, email, month, number, range, seach, tel, time, url, week).

Andere neue Formulartags erlauben eine verbesserte Darstellung von vorgegebenen Aufzählungswerten für ein Eingabefeld, z.B. „Montag“, „Dienstag“, „Mittwoch“, „Donnerstag“, „Freitag“, „Samstag“ und „Sonntag“ (datalist). Beim Eintragen von Werten werden dem Benutzer automatisch die erlaubten Werte angeboten. Weiter gibt es Elemente für die automatische Generierung von sicheren Schlüsseln (keygen) und die Ausgabe mathematischer Kalkulationen (output-Tag).

Der begleitende CSS3-Standard ermöglicht eine deutlich verbesserte und individuellere Darstellung von Formularen und Formularelementen, da jetzt alle wesentlichen visuellen Eigenschaften von Formularfeldern dem Design der Webseite angepasst werden können (siehe Abb. 3.6).

Zugehörige JavaScript-Erweiterungen ermöglichen eine verbesserte programmatorische Unterstützung des Vorabfüllens und der Überprüfung der Inhalte von Formularfeldern vor dem Abschicken des Formulars, so dass in der Summe Web-Frameworks auf Basis von HTML5 dem Programmierer hochwertige Interaktionskomponenten für Formulare bereitstellen können, die Benutzern einer Webanwendung eine Desktopanwendungs-ähnliche Ergonomie ermöglichen.

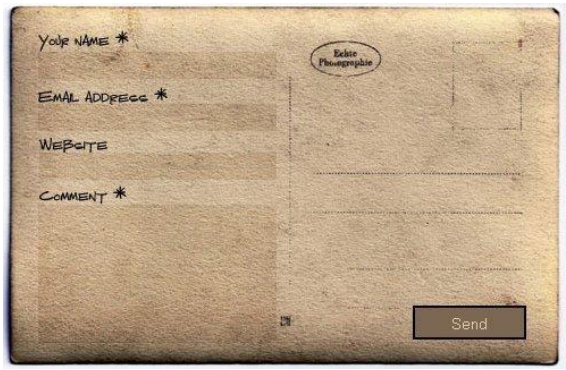


Abb. 3.6: Individuelles Design von Formularen und deren Eingabeelementen

3.1.3 Vektorgrafik und Zeichenflächen (Canvas-Element)

Der HTML5-Standard erlaubt die durchgängige Einbettung von SVG-basierter Vektorgrafik in HMTL-Seiten. SVG (Scalable Vector Graphic) ist ein XML-basierter W3C-Standard zur Be-

schreibung von Vektorgrafiken, der von den meisten Webbrowsern unterstützt wird [61]. SVG-basierte Vektorgrafiken können direkt in HTML-Seiten eingebunden werden. Da SVG die Manipulation über JavaScript und auch die Animation von Vektorgrafiken erlaubt, können auch animierte und interaktive Darstellungen erzeugt werden. SVG wird auch dazu verwendet, interaktive Grafiken oder Karten innerhalb einer Webseite zu implementieren.

Das <canvas>-Element von HTML5 erlaubt die Einbindung von Zeichenflächen in eine HTML-Seite, die dynamisch über eine zugehörige JavaScript-API beschrieben werden können. Auf diese Weise lassen sich ebenfalls komplexe 2D- und 3D-Grafiken dynamisch erzeugen und in Realzeit modifizieren.

Aufbauend auf dem Canvas-Element können daher interaktive grafische Spiele oder animierte interaktive Grafiken erzeugt werden, wie sie z.B. Google verstärkt für seine interaktiven Doodles nutzt. Eine weitere Nutzung besteht in der grafischen Darstellung von Daten in interaktiven Diagrammen, die sich vom Benutzer dynamisch modifizieren lassen (vgl. auch Abb. 3.7).

Abb. 3.7: Datendarstellung über Canvas-Element

Canvas-Elemente lassen sich aber auch mit transparentem Hintergrund über andere HTML-Elemente, wie z.B. eine Google Maps-Karte, legen (s. Beispiel in Abb. 3.8). Sie werden daher in modernen kartenbasierten Webanwendungen häufig dazu genutzt, dynamische Layer über eine Hintergrundkarte zu legen, in denen sich Objekte in Realzeit bewegen müssen, zum Beispiel bei Verkehrsanimationen wie der Visualisierungen des Flugverkehrs, der Bewegung von Sternen oder der Ausbreitung von Gegenständen innerhalb einer Strömung. Für die Google Maps-API [62] gibt es entsprechende Zusätze, mit denen man auf einfache Art und Weise einen Canvas-basierten Layer einer Google Maps-Grundkarte hinzufügen kann.

Abb. 3.8: Google Maps Karte mit Canvas-erzeugtem Layer im Vordergrund (Die Karte zeigt in Realzeit Netzwerkverbindungen, wie sie dynamisch zwischen Rechnern (den gelben Knoten) und zentralen Verteilknoten (rote Knoten) entstehen

3.1.4 Standardisierte Multimedia-Funktionalitäten

Ziel der neuen Multimedia-Funktionalitäten im Umfeld der HTML5-Standardisierung ist es, das Abspielen von Audio- und Videodateien und Live-Sendungen sowie das Schreiben von Browser-basierter Abspielsoftware oder anderer webbasierter Audio- oder Videosoftware rein auf Basis des HTML5-Standards zu erlauben, ohne dass – wie bisher üblich – hierzu proprietäre Browserplugins benötigt werden.

Neueste Statistiken zeigen, dass mittlerweile mehr Browser und Geräte den HTML5-Videostandard als das Abspielen von Flash-basierten Videos unterstützen. Vor allem auf Mobilgeräten ist die Flashtechnologie mittlerweile nicht mehr sehr verbreitet. Trotzdem stellt das Abspielen von HTML5-Videos auf allen Geräten eine kleine Herausforderung dar, da die unterschiedlichen Browser hierfür mit Anteil 50/50 zwei verschiedene Videoformate (entweder WebM oder MP4) nutzen. Damit Videos daher auf allen Browsern abspielbar sind, sollte man sie gleichzeitig in WebM- und MP4-Format bereitstellen.

Die Implementierungsdetails (d.h. die Vollständigkeit aller Attribute und der zugehörigen JavaScript-API) differieren von Browser zu Browser. Um eine bestmögliche Kompatibilität zwischen den verschiedenen Geräten zu bewahren, ist es daher für die Implementierung von Videofunktionalitäten sinnvoll, auf JavaScript-basierte Frameworks (wie JWPLayer) zurückzugreifen, die die verschiedenen Implementierungszustände der verschiedenen Browser berücksichtigen und den Programmierer damit von dieser Arbeit befreien.

Ein Problem ist nach wie vor das Abspielen von Live-Videostreams mit adaptiven Streamingmethoden, die sich automatisch der verfügbaren Übertragungsbandbreite anpassen. Hier gibt es bislang nur auf Apples HTTP-Live-Streaming-Protocol (HLS) basierende Lösungen, die daher nur für iOS-basierte Geräte implementiert sind. Ein vorübergehender Support von HLS wurde mittlerweile wieder aus dem Android-Betriebssystem herausgenommen. Google und Microsoft arbeiten zurzeit an eigenen Lösungen basierend auf dem DASH-Standard (Dynamic Adaptive Streaming over HTTP) des MPEG Konsortiums. Dieser hat zum Ziel, eine standardkonforme Streaminglösung direkt als JavaScript-API in alle Brow-

ser zu integrieren. Der DASH-Standard könnte damit auf Dauer eine universelle Lösung für Live-Streaming in Browsern bieten.

3.1.5 Weitere Verbesserungen und Erweiterungen für das Schreiben von Web-Anwendungen

Im Rahmen der HTML5-Standardisierung sind weitere neue Funktionalitäten und Programmierschnittstellen hinzugekommen, die vor allem JavaScript-Programmierern die Arbeit erleichtern sollen. Dies sind insbesondere die GeoLocation- und Web-Storage-APIs sowie die API für Web-Workers.

Die GeoLocation-API ermöglicht es JavaScript-Programmierern, die Position eines Gerätes, auf dem ein Browser läuft, zu ermitteln. Grundvoraussetzung hierfür ist allerdings, dass der Nutzer des Browsers bzw. des Geräts die Weitergabe der Positionsdaten aktiviert hat. Unter Nutzung der Geolocation-API können Webanwendungen geschrieben werden, die bei bestimmten Funktionen wie einer Suche automatisch die Position des Nutzers mit einbeziehen (Ortsbezogene Anwendungen).

Mit der Web-Storage-API kann JavaScript-Code in einer Webanwendung Daten lokal auf dem Rechner des Nutzers in einem geschützten Web-Storage-Bereich schreiben und wieder auslesen. Dies ermöglicht es Nutzern von Browsern, Webanwendungen bis zu einem gewissen Grad auch dann zu nutzen, wenn aktuell keine Internetverbindung besteht. Die Webanwendung arbeitet dann temporär mit Daten aus dem Web-Storage auf der lokalen Maschine des Nutzers. Steht die Verbindung zum Internet wieder zur Verfügung, können diese lokalen Daten dann wieder mit serverseitigen Diensten synchronisiert werden.

Die Web-Workers-API ermöglicht es, Funktionen in Webanwendungen zu realisieren, die im Hintergrund (ohne Interaktion mit dem Nutzer) als eigenständige, unabhängige Bearbeitungsvorgänge ablaufen. Solche Abläufe können z.B. eine im Hintergrund laufende Synchronisierung zwischen lokalen und serverseitigen Daten vornehmen, ohne dass der Nutzer durch diese im Vordergrund gestört wird.

Vervollständigt man diese Funktionalitäten noch mit der Anwendungscache-Funktionalität von HTML5, so kann man Webanwendungen schreiben, deren vollständiger Code und Inhalte auf der Clientseite gecacht werden und auch offline mit lokalen Daten arbeiten können. Solche Anwendungen sind dann vollständig offline nutzbar, z.B. Anwendungen, wie Google Drive.

3.1.6 Bessere Unterstützung für mobile Clients und Touchscreen-Geräte

Mit dem Aufkommen von iPhone und iPad hat Apple für den mobilen Safari-Browser Funktionalitäten (neue Ereignisse wie touchstart, touchmove oder touchend) zur Unterstützung von Multitouch-Eingaben (Mehrfinger-Eingabegesten) entwickelt. Diese zunächst proprietären Erweiterungen wurden dann für andere mobile Browser (z.B. auf Androidgeräten) übernommen und implementiert.

Mit dem verstärkten Aufkommen von Touch-sensitiven Notebooks und anderen Desktop-ähnlichen Computergeräten stieg auch der Bedarf an Implementierungen der Touchfunktionalitäten für Browser auf Desktopsystemen, so dass mittlerweile auch eine W3C-Arbeitsgruppe an der Standardisierung von Touch-Ereignissen für Webanwendungen arbeitet und es erste experimentelle Unterstützung solcher Schnittstellen in aktuellen Browserversionen von Firefox, Chrome und IE gibt.

In Zukunft kann man also erwarten, dass sich standardisierte Webanwendungen schreiben lassen, die sowohl auf Desktop-Computern als auch auf mobilen Geräten Unterstützung für Mehrfinger-Eingabegesten bieten.

Browserimplementierungen auf mobilen Geräten weisen heute eine gute Unterstützung des HTML5-Standards auf, ohne dabei alle seine vom W3C standardisierten Facetten zu implementieren. Die Browserunterstützung von HTML5 folgt, wie bereits geschildert, eher dem HTML-Living-Standard der WHATWG-Gruppe und implementiert die für die mobilen Geräte sinnvollen HTML5-Funktionalitäten, ohne zu sehr auf Rückwärtskompatibilität mit allen HTML 4-Funktionalitäten zu achten.

Für mobile Webanwendungen ist daher die Implementierung direkt in HTML5 oftmals die bessere Wahl, und HTML5-Anwendungen finden zurzeit auf mobilen Geräten ihre größte Verbreitung.

3.2 Neue Funktionalitäten auf der Serverseite

Die Änderungen an den Webstandards im Umfeld der HTML5-Standardisierung betreffen aber nicht nur die Client-, sondern auch die Serverseite. Hier bieten moderne Webstandards neue Formen der Kommunikation zwischen Client und Server.

In früheren Webanwendungen musste man für das aktive Ereignis-orientierte Übertragen von neuen Nachrichten vom Server auf einen Client die Webseite im Client so programmieren, dass sie regelmäßig beim Server anfragt, ob neue Informationen verfügbar sind (Clientseitiges periodische Ziehen am Server, Pull-Kommunikation). Über die Implementierung von Server Send Events (SSE) in Webservern und Browsern kann ein Server jetzt neue Nachrichten Ereignis-orientiert vom Server an den Browser senden (Push-Kommunikation) [63]. Eine in der Clientwebseite registrierte Ereignisbehandlungsfunktion für solche SSE-Nachrichten bekommt eine Nachricht vom Browser zugeleitet und kann diese asynchron innerhalb der Browserseite darstellen. Typische Anwendungsbeispiele dafür sind die Facebook-Nachrichtenkommunikation, die automatische Aktualisierung von Börsenkursen, das automatische Aktualisieren von Neuigkeiten-Feeds oder das Anzeigen von Sportresultaten.

Während sich Server Send Events gut für das immer wiederkehrende kontinuierliche Senden von Nachrichten vom Server an den Clientwebbrowser eignen, erlaubt die Verwendung von asynchronen Kommunikationsmechanismen über HTTP die Entkoppelung der Rückmeldung über die Beendigung einer langandauernden Serveraktion, die z.B. mehrere Minuten auf dem Server laufen muss, bevor sie beendet ist, vom Start einer Aktion und das sichere Ausführen solcher langandauernder Aktionen. Asynchrone Kommunikation über HTTP eignet sich daher sehr gut für das kontinuierliche Übertragen großer Datenströme zwischen Server und Client, bei denen der Client bzw. Server nicht ständig auf die Beendigung der langan-

dauernden Aktion des jeweiligen Partners warten soll. Zur Implementierung solcher asynchroner Kommunikationsmechanismen über HTTP benötigen die Programmierer Hilfsmechanismen zur Implementierung auf einem Server (z.B gemäß der Comet-Architektur [64]) sowie zugehörige Clientbibliotheken, die z.B. für Browser als JavaScript-Erweiterungen erhältlich sind.

Das neue WebSocket-Protokoll [65] ist schließlich ein zum HTTP-Protokoll alternatives Kommunikationsprotokoll, das eine binäre Datenübertragung analog zur Socketschnittstelle eines Betriebssystems über eine stehende Verbindung zwischen Server und Client erlaubt. Eine Verbindung zu einem Server unter Nutzung des WebSocket-Protokolls wird zunächst wie eine ganz normale HTTP-Verbindung geöffnet. Im Header der HTTP-Anfrage wird allerdings die Anforderung an den Server gestellt, die geöffnete Verbindung im Anschluss an die Verarbeitung der HTTP-Anfrage offen zu lassen und in eine WebSocket-Verbindung zu konvertieren. In der Folge können Client und Server dann über das WebSocket-Protokoll binär über diese Verbindung kommunizieren.

Die Verwendung des WebSocket-Protokolls ermöglicht eine hochperformante Kommunikation zwischen Server und Client mit Realzeiteigenschaften und eignet sich daher gut für Realzeit-nahe Visualisierungen, Spiele und extensive Datenkommunikation.

Die Verwendung dieser neuen Kommunikationsmechanismen im Webbrowser setzen eine entsprechende Unterstützung im Webserver und die Ergänzung durch JavaScript-Bibliotheken auf der Clientseite voraus. Die für SSE- und WebSocket-Kommunikation notwendigen JavaScript-APIs sind im Rahmen der HTML5-Standardisierung bereits definiert und in den Browsern implementiert. Für asynchrone Kommunikation kann die bereits vorher existierende AJAX-Funktionalität genutzt werden.

3.3 Moderne Web-Entwicklungsplattformen und Frameworks

Aufgrund der Architektur des modernen Web besteht eine Webanwendung heute typischerweise aus Code, der auf Server(n) ausgeführt wird (Daten- und Geschäftslogik der Anwendung, Code zur Generierung der Webseiten der Anwendung) sowie den Webseiten selbst (eine Mischung aus HTML und zugehörigem JavaScript-Code), die auf den Clients im vom Nutzer verwendeten Browser interpretiert und ausgeführt werden und die Präsentations- und interaktive grafische Benutzungsoberfläche einer Webanwendung darstellen.

Eine serverseitige Infrastruktur, z.B. Webserver oder Applikationsserver mit integriertem Webserver (Web-Applikationsserver), stellt den Programmierern der Webanwendung dabei einerseits eine Kommunikationsinfrastruktur bereit, die die wichtigsten Kommunikationsprotokolle und -schnittstellen wie das HTTP-Protokoll oder den neueren Web-Socket-Standard implementiert, andererseits auch höherwertige Funktionalitäten, wie Frameworks für die einfache Implementierung von Web-Services, zum Zugriff auf Daten in Datenbanken, zum Zugriff auf und das Management von Dateien wie Mediendateien, JavaScript Bibliotheken, oder zum entfernten Zugriff auf andere Systeme, z.B. über Web-Service-Client Frameworks, bietet. Typische Web-Entwicklungsplattformen sind z.B. der Apache-Webserver unter Integration einer PHP-Laufzeitumgebung (hier stellt der Apache-Server die grundlegende Kommuni-

kationsinfrastruktur und PHP die grundlegenden Basisfunktionalitäten zum Zugriff auf Datenbanken, Dateien etc. bereit), das .NET-Framework zusammen mit dem Internet Information Server (IIS) als grundlegende Web-Kommunikationsinfrastruktur oder für Java-Programmierer die Java Enterprise Edition (JEE), in der sogenannte Java-Applikationsserver oder Web-Applikationsserver die grundlegende Kommunikations- und Laufzeitinfrastruktur für die Webanwendungen bereitstellen (vgl. Abb. 3.9).

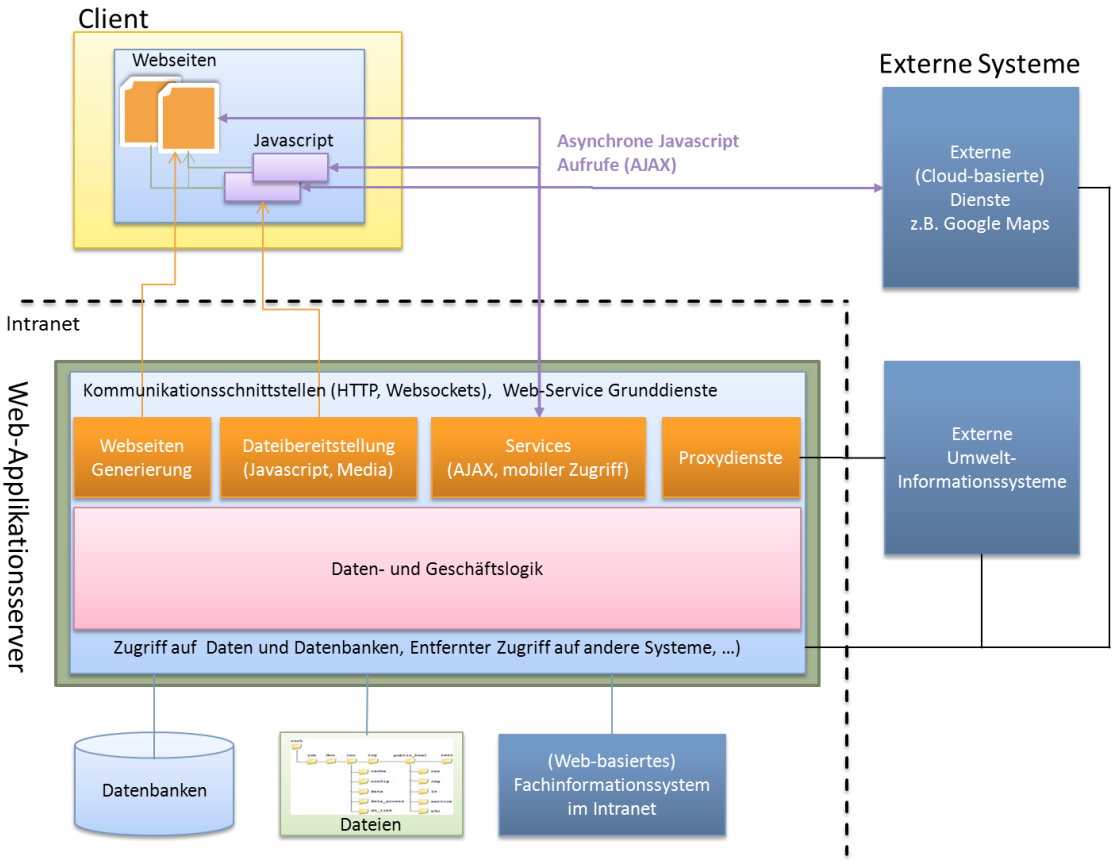


Abbildung 3.9: Zusammenspiel verschiedener Komponenten bei modernen Webanwendungen

Je nach Grundinfrastruktur enthält die Entwicklungsplattform dabei auch schon höherwertige Funktionalitäten zur Implementierung von Daten- oder Geschäftslogik innerhalb einer Anwendung, z.B. als modulare Daten- und oder Geschäftslogik-Komponenten oder zur Generierung der benötigten Webseiten über Webseiten-Auszeichnungssprachen und Schablonen (Templates).

Web-Applikationsserver und Applikationsserver gemäß der JEE-Spezifikation [66] nehmen den Programmierern eine ganze Reihe von Managementaufgaben ab, die man normalerweise auf der Serverseite manuell implementieren muss. So unterstützen sie ein sehr modulares Design der Webanwendung in Form von Komponenten-basierter Programmierung und kümmern sich dabei automatisch um die Verwaltung der Komponenten und deren Lebenszyklus (wann muss eine Komponente instanziiert werden, wie lange soll / darf sie im Server leben, um nicht zu viele Ressourcen zu verbrauchen, wie viel Instanzen benötigt man für eine optimale Performanz) sowie weiterer Ressourcen, die von den Komponenten benötigt werden. JEE-Server bieten zusätzlich weitere Dienste, wie das automatische Management von verteilten Transaktionen, asynchrone Nachrichtenübertragungsdienste oder Möglichkeiten zur Anbindung an Legacy-Anwendungen, wie Emaildienste, SAP etc. an. Die JEE-

Technologie eignet sich daher sehr gut für die Implementierung großer, serviceorientierter Webanwendungen, z.B. von Service-Oriented-Architecture-(SOA)-Anwendungen), an die bzgl. ihrer Funktionalitäten und Zuverlässigkeit große Anforderungen gestellt werden.

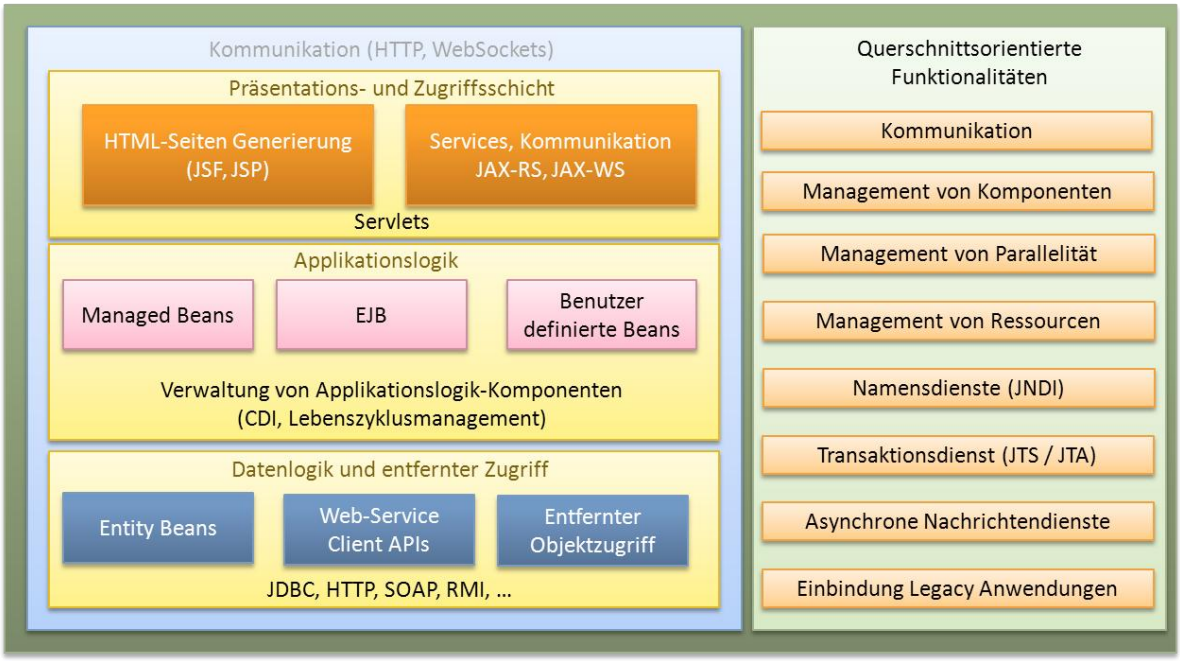


Abbildung 3.10: Funktionalitäten zur Programmierung von Webanwendungen in JEE

Je nach Einsatzbereich der JEE-Technologien bieten verschiedene JEE-Profile dem Programmierer mehr oder weniger ihrer Funktionalitäten an (s. Abb. 3.10). Für kleinere Webanwendungen bietet das JEE-Web-Profil zwar nur einen Teil der Gesamtfunktionalität (es fehlt z.B. die asynchrone Nachrichtenverarbeitung), dafür sind alle Funktionalitäten vorhanden, die man für Webanwendungen benötigt.

Um die Apache-PHP-Kombination auf den Funktionsstand der JEE-Umgebung zu bringen, muss man sie durch weitere Frameworks ergänzen, welche die entsprechende Funktionalitäten liefern, wobei längst nicht alle Funktionalitäten (z.B. asynchrone Nachrichtenverarbeitung) auch wirklich in PHP vorhanden sind. Typischerweise ergänzt man sein PHP-Portfolio für größere Webanwendungen zunächst einmal um ein Webanwendungsframework wie Symfony, das der Grundfunktionalität zumindest die Schablonen-basierte Generierung von Webseiten, einfache Formen der modularen Implementierung der Applikationslogik und vereinfachten Zugriff auf Datenbanken hinzufügt. Bei Bedarf muss man diese Funktionalitäten noch um weitere Frameworks ergänzen. Daher entwickeln sich PHP-basierte Webanwendungen schnell zu einem bunten Mix verschiedener Frameworks, wobei ein Versionsmanagement aller Komponenten nicht immer einfach ist. Hier bietet ein Gesamtkonzept wie der JEE-Standard deutliche Vorteile, da alles aus einem Guss ist und die einzelnen Bestandteile im Rahmen der Standardisierung mit jedem Release aufeinander abgestimmt werden.

Für die Generierung der Webseiten wurden in der Vergangenheit Schablonen-basierte Verfahren entwickelt, die der Model-View-Controller Architektur (auch Model-2-Architektur für Webanwendungen genannt) folgten. Bei diesen trennt man den HTML- und JavaScript-Code der eigentlichen Webseite so gut es geht von dem Code, der diese Webseiten erzeugt und dynamisch Informationen in diese integriert. Beim Schablonen-basierten Ansatz lagert man

den HTML- und JavaScript-Beschreibungsteil einer zu erzeugenden Webseite in eine Schablone (Seitenvorlage, Template) aus, die über eine zugehörige Schablonensprache (Template-Sprache) mit Platzhaltern versehen ist, in die automatisch bei der Erzeugung einer zugehörigen Webseite von der Kontrolllogik dynamisch Daten unter Nutzung der Datenkomponenten eingefügt werden. Hierzu ermittelt bei Eintreffen einer Webseitenanforderung die Kontrolllogik zunächst, welche Daten zur Beantwortung der Anfrage benötigt werden, extrahiert diese aus den Datenquellen, sucht dann die passende Schablone aus, mit der die Antwortwebseite erzeugt werden kann, und veranlasst dann die dynamische Integration der Daten in die Schablone, so dass eine vollständige Webseite entsteht, die dann an den Client ausgeliefert wird.

Schablonen (Templates) für Webseiten bestehen bei dieser Technologie in der Regel aus einem Mix von normalen HTML-Elementen, JavaScript-Elementen (wie Funktionen und Angaben zu Ereignishandlern spezifischer HTML-Elemente), die für die interaktiven Funktionalitäten der Webseite verantwortlich sind, und Codeschnipseln basierend auf der Templatesprache, die für die Einbettung dynamischer Datenelemente erforderlich sind.

Diese Vorgehensweise hat einige gravierende Nachteile: Während der serverseitige Applikationslogik-Code bei einer Model-2-Architektur einer Webanwendung sauber vom Präsentationscode (HTML, JavaScript) getrennt ist, und die dynamische Verknüpfung beider Elemente durch eine einfach zu verstehende und übersichtliche Templatesprache erfolgen kann, gestaltet sich der HTML- und JavaScript Code, der notwendig ist, um interaktive und funktional komplexe Oberflächenelemente (wie z.B. interaktive Tabellen) damit zu implementieren, als sehr komplex. Hinzu kommt noch, dass Webbrowser bzgl. der HTML-Standards und der Implementierung von JavaScript und der Schnittstellen von JavaScript zum DOM oftmals leichte funktionale Unterschiede und Inkompatibilitäten aufweisen, so dass der Code, der benötigt wird, um komplexere UI-Elemente zu implementieren noch für verschiedene Browser und Browserversionen unterschiedliche Implementierungen oder Workarounds enthalten muss, was die Sache für den Anwendungsprogrammierer noch komplexer macht.

Daher gibt es mittlerweile eine ganze Reihe von JavaScript-Bibliotheken (z.B. jQuery [36] [67] und MooTools), die das Ziel haben, oberhalb der HTML- und JavaScript-Standards des W3C vereinfachte Funktionalitäten zur JavaScript-Programmierung in Webanwendungen bereitzustellen, mit denen eine über verschiedene Browser hinweg kompatible Programmierung von Funktionalitäten möglich ist.

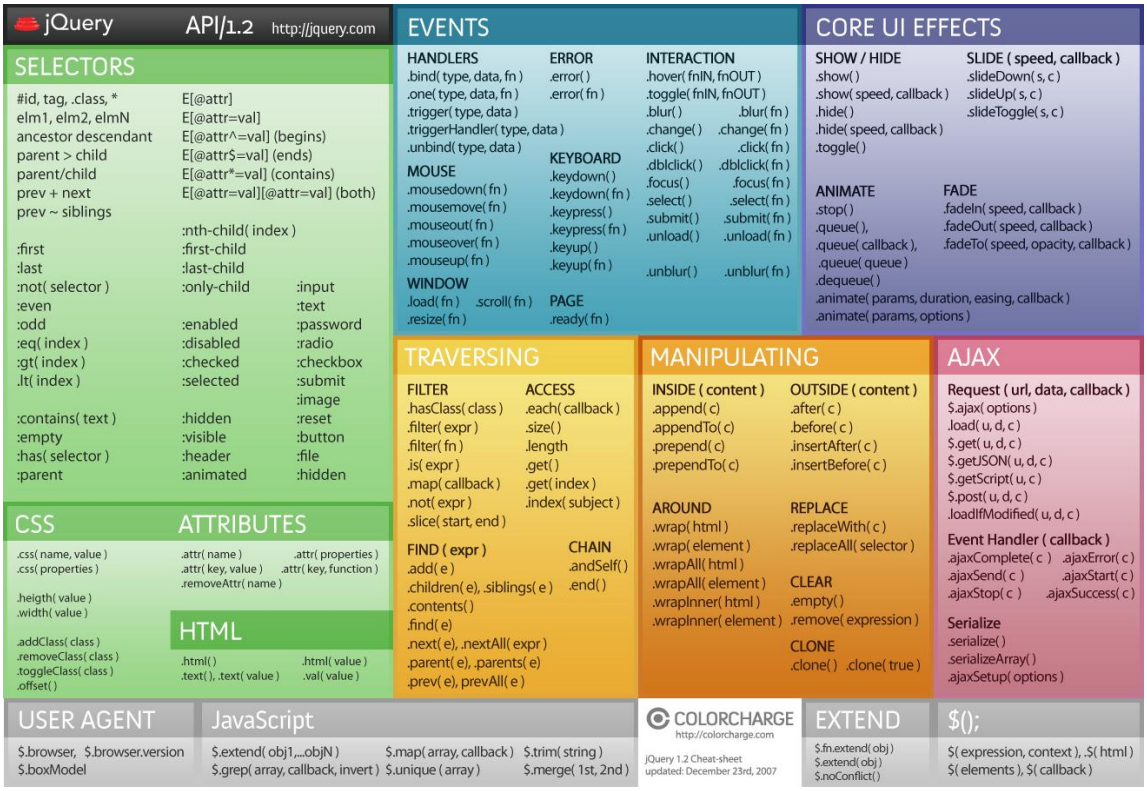


Abb. 3.11: Übersicht über wichtige Funktionen der jQuery-JavaScript-Library

Solche JavaScript-Libraries enthalten neben Funktionalitäten zur Manipulation des HTML-Dokument-Objektbaumes (in Abb. 3.11 MANIPULATING), wie Elemente hinzufügen oder löschen, Inhalte ersetzen auch Funktionen zum Ändern der Stilattribute des DOM-Baumes eines HTML-Dokumentes (CSS und ATTRIBUTES im Bild), um z.B. die Hintergrundfarbe eines Elements zu setzen. Weiter bieten die Libraries auch Funktionalitäten zum vereinfachten Suchen, Auffinden und Traversieren von DOM-Elementen (SELECTORS, TRAVERSING). Diese Funktionalitäten erlauben es auch, ganze Listen von Elementen gleichzeitig zu beeinflussen (z.B. die Schriftfarbe aller Überschriften eines bestimmten Typs zu ändern). Natürlich sind auch Funktionalitäten zur Definition von Ereignisbehandlungsroutinen (Interaktion mit dem Nutzer), für AJAX-Kommunikation mit dem Server und Effektbibliotheken vorhanden, die das Programmieren hochgradig interaktiver Komponenten erlauben.

Abb. 3.12: ThemeRoller Editor für Themes von jQuery-UI-basierten Widget(bibliotheken). Links sieht man die Kalenderkomponente in verschiedenen bereits vordefinierten Themes, rechts verschiedene jQuery-UI-Grundelemente im aktuell ausgewählten Theme

Das Programmieren komplexerer UI-Komponenten ist allerdings auch mit den Basisfunktionalitäten einer Bibliothek, wie jQuery, noch viel Arbeit. Daher gibt es oberhalb dieser Libraries weitere Bibliotheken, die bereits ausprogrammierte, komplexere UI-Komponenten (oft auch als Widgets bezeichnet), z.B. komplexe Tabellen-Widgets oder interaktive Kalenderwidgets zur Auswahl eines Datums, als Bibliotheken bereitstellen. Für jQuery gibt es als Zusatz die jQuery-UI-Bibliothek [68], die bereits einige solcher grundlegender Widgets bereitstellt und andererseits als Basisframework für weitere höherwertige Widgetimplementierungen dienen kann. Die jQuery-UI enthält weiter einen Mechanismus mit Namen „ThemeRoller“ (siehe Abb. 3.12), über den sich alle von der jQuery-UI-Widgetbibliothek abgeleiteten Widgets mit einem einfachen und einheitlichen Mechanismus bzgl. ihres Aussehens an ein neues Design anpassen lassen. Bei Einsatz von Widget-Bibliotheken, die auf der jQuery-UI basieren, können daher die verwendeten Widgets in Anwendungen sehr schnell auf verschiedene Corporate Identity Rahmendesigns angepasst werden. Viele höherwertige Widgetimplementierungen oder Toolkits bauen daher auf der jQuery- und der jQuery-UI-Bibliothek auf.

Unter Verwendung von modernen JavaScript- und darauf aufsetzenden Widget-Bibliotheken lassen sich auch unter Verwendung von Templates bereits Webanwendungen schreiben, deren Nutzerschnittstellen ein modernes Design mit einer Desktopanwendungs-ähnlichen Nutzererfahrung kombinieren.

Für den Programmierer ist allerdings die Kombination reinen HTMLs mit solchen JavaScript-Lösungen – wenn auch wesentlich komfortabler wie die Verwendung der direkten HTML- und JavaScript-Schnittstellen – nach wie vor mit einigem Aufwand verbunden, da in HTML-eingebundene Widgets noch manuell über JavaScript konfiguriert, Ereignisbehandlungsroutinen der Widgets aufgesetzt und mit zugehörigen AJAX-Serviceschnittstellen auf der Serverseite kombiniert werden müssen, und die Widgets oft von sich aus keine komplexere Zustandsverwaltung, Validierung von Eingaben oder ähnliche Funktionalitäten anbieten. Daher gibt es neuere Ansätze für das Schreiben von Präsentationsoberflächen von Webanwendungen, die eine kombinierte und abgestimmte Lösung sowohl der für komplexe Widgets notwendigen Server- als auch der clientseitigen Funktionalitäten vorsehen. Solche Frameworks ergänzen die reinen Oberflächenfunktionalitäten häufig durch Zusatzfunktionalitäten, wie automatische Mechanismen zur Zustandsverwaltung, zur Validierung von Eingabedaten, die automatische Umwandlung von Daten zwischen Datentypen, und eingebauten Mechanismen für die Programmierung neuer gekapselter Komponenten (Widgets), dem Aufbau interoperabler Widgetbibliotheken verschiedener Herkunft und zur Komposition von Widgets zu neuen komplexeren Oberflächenelementen (z.B. Wizards). Für solche kombinierten Ansätze gibt es prinzipiell zwei verschiedene Vorgehensweisen.

Bei Toolkits wie dem Google Web Toolkit (GWT) [69] besteht der Ansatz darin, dass grundsätzlich die Programmierung nur mit serverseitigen Komponenten erfolgt und aus dem serverseitigen Programmcode die HTML-Seiten inklusive der JavaScript-Funktionalität automatisch erzeugt werden. Der Web-Programmierer hat bei diesen Toolkits damit mit der Programmierung auf HTML-Ebene nur noch wenig zu tun. Der Nachteil dieses Ansatzes ist, dass er nur wenige Möglichkeiten des Eingriffs und zur Optimierung auf Ebene der Webseiten selbst (HTML und JavaScript) hat, da ja diese Ebene komplett generiert wird.

Ein weiterer Ansatz ist es, serverseitige und clientseitige, d.h. in einer Schablone für eine Webseite, stattfindende Programmierung auf „beste“ Art und Weise zu kombinieren, wie dies z.B. bei den Toolkits Vaadin [70] oder bei der JSF 2-Technologie [71] für die Sprache Java der Fall ist. Hierbei kann der Programmierer z.B. die verwendete Technologie auch durch Nutzung von JavaScript-UI-Bibliotheken ergänzen und an der einen oder anderen Stelle optimieren.

Die mit dem JEE 6 überarbeitete Version des JSF (Java Server Faces) Standards JSF 2 ist eine im Rahmen der Java Enterprise Standardisierung entwickelte Präsentationstechnologie, die die Vorteile des hybriden Ansatzes in nahezu optimaler Weise kombiniert. Sie unterstützt den Aufbau und die Nutzung großer Widgetbibliotheken mit nahezu beliebig komplexer Funktionalität bei einfachster Nutzung durch den Programmierer und weitgehender Kompatibilität verschiedener JSF-Widgetbibliotheken. JSF 2 Technologie enthält fortgeschrittene Funktionalitäten, wie Zustandsmanagement, automatische Validierung und Konvertierung von Datenfeldern, Unterstützung für die Navigation zwischen Webseiten und vieles mehr. JSF 2 basierte Widgetbibliotheken wie Primefaces [72] oder Richfaces [73] liefern dabei auch für komplexere Anwendungssituationen professionelle Komponenten, mit denen sich

die Oberflächen komplexer Fachanwendungen schnell und komfortabel erstellen lassen (siehe z.B. Abb. 3.13).

Abb. 3.13: JSF-Anwendung mit Primefaces-Widgets für Google Maps, Kalender, Menü- und Medieneinbindung, in einem dunklen ThemeRoller-Stil

Primefaces baut dabei auf den jQuery- und jQuery-UI-JavaScript-Bibliotheken auf. Mit Primefaces erstellte Oberflächen lassen sich daher über ThemeRoller auf einfache Weise bzgl. ihres Designs an verschiedene Corporate Designs anpassen (Unterstützung von Themes). Neben Primefaces gibt es auch eine hierzu kompatible Prime-UI-JavaScript-Bibliothek, die ebenfalls auf jQuery-UI aufsetzt und wesentliche Widgets der Primefaces-JSF-Bibliothek auch in JavaScript bereitstellt. Daher lassen sich hiermit auch beide Welten auf einfache Art und Weise miteinander kombinieren.

Des Weiteren sind jQuery und jQuery-UI und die darauf aufbauenden Technologien auch für mobile Anwendungen einsetzbar.

Als Fazit lässt sich sagen, dass moderne Konzepte zur Programmierung von Webanwendungen Präsentationsframeworks verwenden bzw. verwenden sollten, die Programmierern die Implementierung von Nutzeroberflächen mit Desktop-ähnlichen Komfort auf Basis von HTML5 und neueren JavaScript-Funktionalitäten erlauben, wobei die Performanz und Ergonomie solcher Anwendungen denen von Desktop-Anwendungen nahezu entsprechen kann. Für die Programmierer bieten moderne Frameworks wie die JSF-Technologien dabei einen Komfort, der immer mehr an den Komfort zur Erstellung von Desktopanwendungen heranreicht.

Kombiniert man eine solche Technologie mit Komponenten-basierten serverseitigen Entwicklungsplattformen, die auch das Schreiben von Datenlogik und Applikationslogik stark vereinfachen, lassen sich auch komplexe Fachanwendungen leicht als Webanwendungen mit zu Desktopanwendungen vergleichbarem Komfort realisieren.

Die Vorteile für Webanwendungen, gerade auch für Fachanwendungen, wie sie im Umweltbereich oft eingesetzt werden, sind aber enorm. Das Datenmanagement und Kern-Fachlogik kann zentralisiert und einzelne Funktionen können modular über Services für weitere Anwendungen bereitgestellt werden. Aufbauend auf serviceorientierten Daten- und Fachlogikdiensten, lassen sich mit heutigen Werkzeugen Webanwendungen oder Mobilanwendungen als Fachanwendungen schreiben, die Desktop-ähnlichen Komfort bieten, aber Daten und Fachlogik überall verfügbar machen. Die hierbei entstehenden Anwendungen lassen sich dabei auf eigenen Rechnerinfrastrukturen hosten oder auch teilweise oder vollständig auf externe Infrastrukturdienstleister oder Cloudservices outsourcen. In Zukunft wird der Trend sich immer weiter auf solche webbasierten Anwendungen konzentrieren.

3.4 Content-Management-Systeme, Portale und andere ausgewählte Webanwendungen

Gemäß der ursprünglichen Aufgabe des Web waren die ersten Webanwendungen Online-Redaktionssysteme (sogenannte Content-Management-Systeme, CMS) [1] [3] [4] zur Verwaltung der eigenen statisch erzeugten Webseiten einer Website. Hierbei wurden Webseiten zunächst als Ganzes am eigenen PC erzeugt und das Redaktionssystem übernahm nur die Verwaltung der Webseiten als Dateien auf dem Webserver.

Moderne CMS enthalten auf Webtechnologien basierende Online-Editoren [27], mit denen Webinhalte direkt über den Webbrowser in CMS-Systeme eingepflegt werden können. Diese Online-Editoren verfügen dabei auch über Funktionalitäten zum Hochladen von mit den Textinhalten zu verknüpfenden Medieninhalten, wie Bildern oder Videos, oder zur Integration von Download-Links für Dateien. Die Dateien werden dabei vom Content-Management-System über eine Datei-Inhaltsverwaltungskomponente (Asset- oder Dokumenten-Management) im CMS verwaltet und können über den Online-Editor für Webinhalte auf einfache Weise in Webseiten eingebettet oder über die Erzeugung eines Links (z.B. Download-Link) in Webseiten referenziert werden.

Die Dokumenten- und Medienverwaltung von manchen CMS unterstützt dabei die Versionierung der Dokument- und Medieninhalte. Andere CMS ergänzen die Bearbeitung und Verwaltung von Webinhalten, um Autoren-Workflows, bei denen die Erstellung von Inhalten durch Autoren erfolgt, eine zusätzliche Freigabe der Inhalte aber gesondert durch andere Personen (häufig Editor genannt), zu erfolgen hat, bevor die Inhalte tatsächlich innerhalb der Website für das Publikum sichtbar sind. Ein „Staging“-Prozess mit Versionierung von Inhalten sorgt dabei in manchen Systemen dafür, dass unter ein und derselben URL der Website eine bestimmte Inhaltsversion einer Seite für das Publikum sichtbar ist, während im Rahmen eines Autoren-Workflows bereits eine überarbeitete Version der Webseite im Entstehen, jedoch nur von Autoren und Editoren einsehbar ist.

Auch die Granularität von Web-Inhalten in Content-Management-Systemen ist von System zu System unterschiedlich. Während bei einer einfacheren Inhaltsorganisation, wie beim beliebten PHP-basierten CMS Joomla [74] oder auch bei WebGenesis [9], Web-Inhalte in ihrer Granularität einer vollständigen Webseite (außer den Inhalten, die durch das Rahmendesign definiert sind: Kopf- , Fuß, Navigation, etc.) entsprechen, verwalten kompliziertere Systeme (wie TYPO3 [11] [12] und siehe später auch Liferay [45]) Web-Inhalte als eigenständige, zu-

nächst von einer Webseite unabhängige, Bausteine, die dann auf im System als gesonderte Objekte verwalteten Webseiten als Inhaltsbausteine eingesetzt werden können. Dabei kann eine Webseite sowohl mehr als einen Web-Inhaltsbaustein enthalten, wie auch ein und derselbe Inhaltsbaustein auf verschiedenen Webseiten im System verwendet werden. Systeme, die auf dem Inhaltsbaustein-Prinzip basieren, bieten Autoren häufig auch die Auswahl von Layout-Templates für eine Webseite an. Ein Layout-Template gibt dabei mögliche Plätze für Inhaltsbausteine auf der Webseite strukturiert vor (vgl. Abb. 3.14).

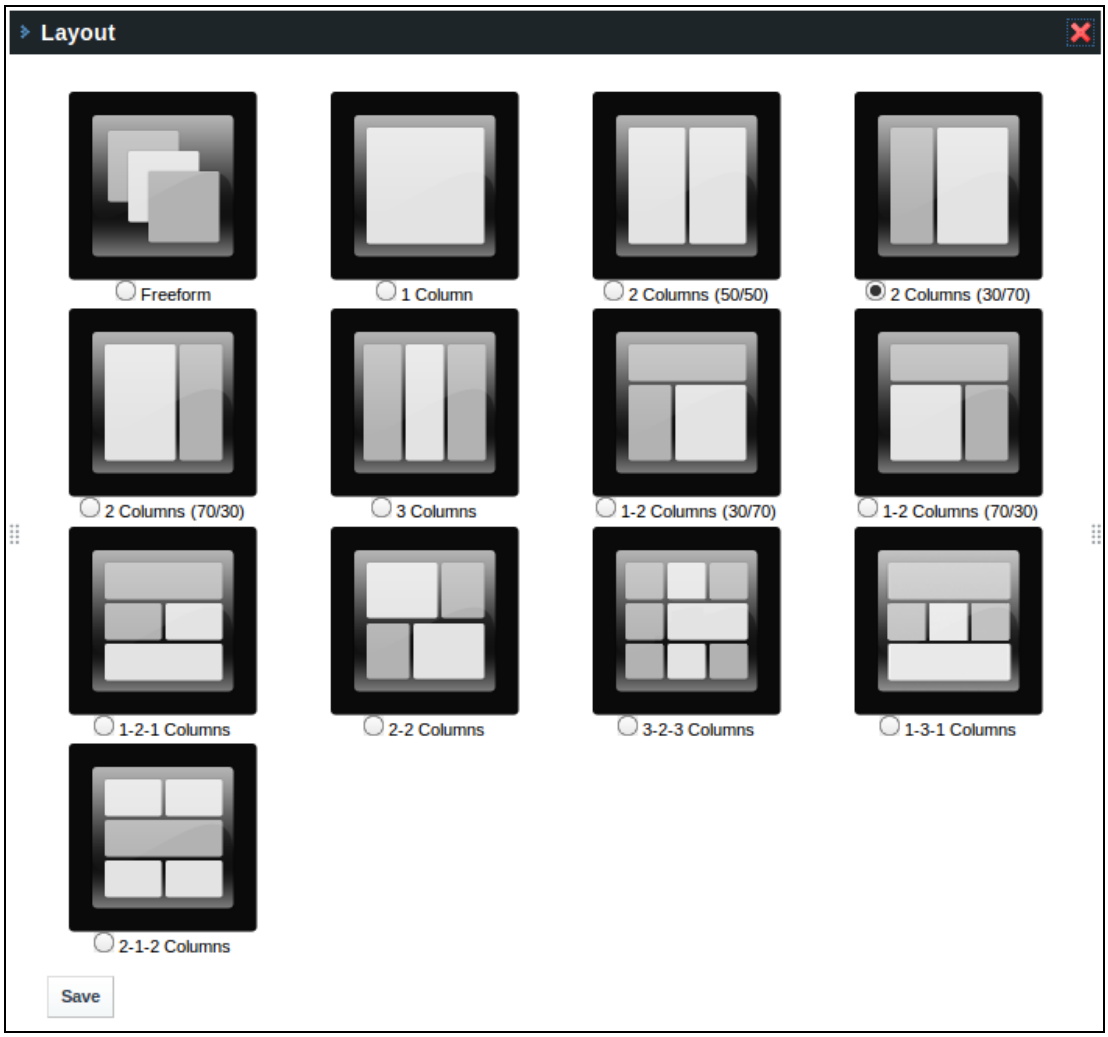


Abb. 3.14: Layout-Templates in Liferay. Autoren können ein Layout-Template für eine Webseite individuell auswählen und Inhaltsbausteine beliebig im Layout anordnen

Die für Autoren zunächst wesentlich komplexere Inhaltsorganisation, in der Inhaltselemente von Webseiten getrennte Bausteine darstellen, die dann zu Webseiten zusammengestellt werden müssen, bietet bei näherer Betrachtung viele Vorteile. So lassen sich Seiten flexibel aus verschiedenen Arten von Inhaltsbausteinen zusammensetzen (Bild+Text, Text, Bild, Video, oder Ansichten von Anwendungsbausteinen, Monatskalender, Wochenkalender, einzelner Termin, Datentabelle, Karte, usw.). Die Inhaltsbausteine sind dabei über verschiedene Seiten wiederverwendbar und dies reduziert die redundante Speicherung gleicher Inhalte innerhalb verschiedener Webseiten, was damit auch wartungsfreundlicher ist.

URLs, über die Inhalte letztendlich adressiert werden, sind bei einer solchen Inhaltsorganisation auch nicht mehr mit Inhalten, sondern nur noch mit den Webseiten assoziiert, die diese Inhalte darstellen. Daher lassen sich die Inhalte von Webseiten, die bestimmte Informationen enthalten sollen, nun durch neuere Inhalte (durch Austausch des jeweiligen Inhaltsbausteins z.B. durch eine neuere Version) aktualisieren, ohne dass sich die URL der umgebenden Webseite ändert. Dies löst ein zentrales Problem aller Systeme, die wie WebGenesis Web-Inhalte über deren ID adressieren, wodurch naturgemäß beim Austausch eines Inhalts eine neue URL entsteht – es sei denn, man überschreibt den Inhalt des alten Inhaltselementes.

3.4.1 Erweiterbarkeit von Webanwendungen

Schaut man auf die verschiedenen Arten von Inhaltselementen, die Nutzer auf ihren Webseiten platzieren möchten, so trifft man oft auch auf Inhaltselemente, die man als Informatiker lieber in einer gesonderten Anwendung verwalten möchte, da es potenziell viele dieser Inhaltselemente mit gleichartiger Struktur und Eigenschaften gibt, die alle auf die gleiche Art angezeigt werden sollen. Typische Vertreter hierfür sind z.B. Kalendereinträge (Termine), Aufgaben, Medien wie Videos oder Audioinformationen, Dokumente (z.B. PDF-Dateien) und Karten, die in Seiten referenziert werden sollen. All diese Elemente der jeweiligen Art lassen sich besser und für den Benutzer sinnvoller in einer gesonderten Datenmanagement-Applikation, wie Kalendersystem, Aufgabenverwaltung, Medienbibliothek, Dokumentenbibliothek (oder zusammen: Dokumenten- und Medienbibliothek) und Kartenanwendung verwalten. Trotzdem möchte man die jeweiligen Informationselemente natürlich innerhalb von Webseiten des eigenen CMS nutzen und anzeigen können.

Viele Content-Management-Systeme verfügen daher über Mechanismen zur Programmierung und Integration von Erweiterungen, die wie eigenständige Webanwendungen wirken (z.B. eine vollständig eigene Datenbankstruktur für ihre Daten besitzen und eigene optimierte Autorenoberflächen zum Editieren und zum Management der Daten zur Verfügung stellen), sich aber andererseits nahtlos in das umgebende CMS integrieren, so dass sich ihre Daten auf einfache Art und Weise auf vom CMS verwalteten Webseiten anzeigen lassen. In der CMS-Fachsprache bezeichnet man solche Erweiterungen häufig als *Komponenten*. Die Verwaltung der Daten durch eine Komponente kann dabei vollständig getrennt vom eigentlichen CMS erfolgen oder sogar über Serviceschnittstellen an eine entfernte Anwendung delegiert sein. So kann man z.B. in einem Liferay-Server die Verwaltung von Inhaltselementen an ein Fremdsystem, z.B. eine Alfresco-Installation (Alfresco ist ein eigenständiges Dokumenten-Management-System), delegieren.

Eine „Komponente“ ist gemäß unserem Verständnis also eine CMS-Erweiterung mit eigenem Datenmanagement und eigener Autorenoberflächen, die sich nahtlos in ein CMS integriert. Bei Systemen wie Joomla oder TYPO3 lassen sich dabei Komponenten zur Laufzeit des Systems über die Administrationsoberfläche nachinstallieren und können dann direkt genutzt werden. WebGenesis verfügt über ein eher rudimentäres Konzept von Komponenten. Man kann WebGenesis als Programmierer um eigene Inhaltskategorien erweitern, muss hierzu aber tief in das System eindringen. Neue Inhaltskategorien lassen sich auch nicht zur Laufzeit installieren, sondern müssen manuell in ein WebGenesis-System integriert werden.

Die Anzeige der Inhalte von Komponenten erfolgt bei Systemen wie Joomla und WebGenesis in der Regel als vollständige Webseiten, d.h. bestimmte URLs im umgebenden CMS sind

so konfiguriert, dass bei ihrem Aufruf der vollständige Nutzinhalt der angezeigten Webseite von der entsprechenden Komponente generiert wird. Über Konfigurationsparameter kann der Autor der Webseite dabei festlegen, wie der Inhalt der Webseite von der Komponente dargestellt werden soll. So kann eine Kalenderkomponente über Konfigurationsoptionen Ansichten ihrer Daten als Jahres-, Monatsansicht oder auch Liste von Terminen für spezielle Kalenderkategorien anzeigen.

Das Joomla-System verfügt neben dem Konzept der Komponenten noch über das Konzept von Modulen. Module sind ebenfalls Erweiterungen des Grundsystems, die über die Administrationsoberfläche dynamisch im laufenden System installiert werden können. Sie verfügen im Gegensatz zu Komponenten aber meist über kein eigenes Datenmanagement, sondern höchstens über ein Konfigurationsmanagement und erlauben es, in Teilbereichen des Rahmendesigns einer Website (diese Teilbereiche werden bei Joomla Modulpositionen genannt) Inhalte einzublenden, die aus dem Joomla-Grundsystem, von Komponenten oder z.B. von Fremdsystemen über serviceorientierte Schnittstellen (z.B. RSS-Feeds) bezogen werden. Komponenten lassen sich dabei häufig zusammen mit zu ihnen gehörigen Modulen installieren, welche die inhaltliche Darstellung von Komponenteninhalten dann auch an Modulpositionen im Rahmendesign ermöglichen. Bei Content-Management-Systemen wie TYPO3, die sowieso bereits mit inhaltlichen Bausteinen arbeiten, ist das Modulkonzept ein bereits im System integriertes Konzept. Hier bieten Komponenten dem Autor verschiedene Darstellungsmöglichkeiten von Inhalten, die als Inhaltsbausteine in Seiten integriert werden können. Auf diese Weise kann der Autor z.B. normalen Text mit einer Liste von Terminen aus der Kalenderkomponenten innerhalb einer Seite flexibel kombinieren.

Eine weitere Art von Erweiterungen für Content-Management-Systeme sind Inhaltsfilter, oft auch Plugins genannt. Ein Plugin wird vom CMS wie ein Filter auf den Inhalt einer zu erzeugenden Webseite angewandt und ersetzt innerhalb der Inhaltsmodule bestimmte Platzhalter durch dynamisch eingesetzte Inhalte. So kann z.B. bei Einsatz des allVideos Joomla-Plugins innerhalb des Textes eines Inhaltselementes z.B. ein String „{youtube}heu37ej3qs{/youtube}“ stehen, der dann vom Plugin automatisch durch einen an der Stelle des Platzhalters eingebetteten JavaScript-basierten YouTube-Videoplayer ersetzt wird. Auf diese Weise lassen sich durch Plugins spezielle Sonderinhalte an beliebigen Stellen innerhalb eines Textes integrieren. Ein weiteres Beispiel hierfür ist der Textstring

```
{mosmap width='500'|height='400'|lat='52.052312'|lon='4.447141'|zoom='3'|
zoomType='Large'|zoomNew='0'|mapType='Satellite'|showMaptype='1'|
overview='0'|text='svDWO'|tooltip='DWO'|marker='1'| align='center'}
```

der bei Nutzung des Google Maps-Plugins von Joomla eine Google Maps-Karte in die Webseite einblenden würde. Auch WebGenesis kennt diese Art der Inhaltsintegration über Tags mit bestimmter Escape-Syntax, die in WebGenesis „Autoformats“ genannt werden. Allerdings unterstützt WebGenesis nur eingebaute Funktionalitäten, z.B. für die Integration von Bildern an beliebiger Stelle im Text.

Damit Programmierer Erweiterungen für Content-Management-Systeme schreiben können, muss das CMS ihnen ein gut dokumentiertes, genormtes Erweiterungskonzept inklusive den notwendigen Programmierschnittstellen zum System bereitstellen. Je nach CMS ist die Erstellung von Erweiterungen dabei mehr oder weniger komfortabel und vom Aufwand her oftmals ebenso komplex wie das Schreiben vollständig eigenständiger Webanwendungen.

Manche Systeme bieten allerdings den Erweiterungsprogrammierern bereits ein Model View Controller (MVC)-nahes Entwicklungsframework an, das Grundfunktionalitäten zum Schreiben von Schablonen oder zum Zugriff auf Services, usw. bereitstellt oder die direkte Nutzung von z.B. jQuery in Anwendungen erlaubt, da diese JavaScript-Library bereits im Grundsystem bereitgestellt wird. In solchen Fällen reduziert sich der Entwicklungsaufwand für Erweiterungen leicht. Leider stellen bislang insbesondere die PHP-basierten Systeme keine einheitlichen Widget-Bibliotheken bereit, so dass Erweiterungen häufig vom Grunddesign her nur bedingt einheitlich stilisierbar sind. Für die Programmierung komplexer technisch-orientierter Erweiterungen bieten die CMS-Erweiterungsschnittstellen von PHP-basierten Systemen in der Regel wenig Unterstützung. Hier bieten Programmiersprachen wie Java wesentlich bessere Möglichkeiten.

Jedes CMS hat typischerweise sein eigenes Erweiterungskonzept, so dass Erweiterungen für gewöhnlich nicht von einem CMS auf ein anderes übertragen werden können, ohne dass die Erweiterung zu einem großen Teil neu geschrieben werden muss. Wie gleich noch gezeigt wird, bietet Java hier mit „Portlets“ ein Erweiterungskonzept an, das prinzipiell über verschiedene Systeme, die dem Portlet-Standard folgen, hinweg funktioniert.

Content-Management-Systeme verfügen in der Regel weiter über ein Konzept für Design-Schablonen (Site-Templates), die das Rahmendesign und die Corporate Identity einer Webseite vorgeben. Systeme wie Joomla und TYPO3 erlauben es dem Administrator einer Site, eine beliebige Anzahl von Site-Templates über die Administrationsoberfläche als Erweiterungen in das System hochzuladen. Nach der Installation eines Site-Templates lässt sich dann per Konfiguration festlegen, welche Teilmengen von Webseiten der gesamten Site durch ein zugeordnetes Site-Template gestylt werden sollen. Oft unterstützen die CMS dabei auch, dass man individuell in der Konfiguration jeder Webseite einstellen kann, welches Site-Design zum Anzeigen dieser Webseite verwendet werden soll.

Site-Templates sind HTML-Schablonen (meistens bestehend aus einer HTML-Seite und weiteren Hilfsdateien wie CSS-Dateien und Bildern), die HTML-Code und Platzhalter für den Hauptinhalt ($content) und weitere Inhaltselemente, wie Module (Modulpositionen), enthalten. Auf den Modulpositionen kann man je nach Konfiguration des CMS z.B. strukturelle Inhaltselemente, wie Navigationsmenüs, Breadcrumbs, Suchboxen, usw. positionieren. Eine vollständige Webseite wird dann so generiert, dass das zur Webseite zugehörige Site-Template geparst wird und dabei der Platzhalter $content durch die der Seite zugehörigen Inhaltselemente und die Platzhalter der Modulelemente durch die per Modulkonfiguration definierten Inhalte der Modulelemente ersetzt werden. Vor dem Einsetzen der eigentlichen Inhalte werden dabei gegebenenfalls bei Einsatz von Plugins noch die Tag-Platzhalter für Plugininhalte durch den jeweiligen Inhalt ersetzt. Content-Management-Systeme verwenden dabei für das „Programmieren“ von Site-Templates oft ihre eigene kleine Schablonensprache, mit der die Einbindung von Platzhaltern für Standardinhalte (Hauptinhalt und Modulinhalte) und evtl. auch direktem Inhalt aus dem CMS noch flexibler gehandhabt werden kann (z.B. das automatische Management von zusätzlichen Spalten je nach Inhalt programmierbar ist).

Die soeben beschriebenen Konzepte von Site-Templates, der zugehörigen CMS-spezifischen Schablonensprache und die Konzepte zur dynamischen Platzierung von Inhalten innerhalb der Site-Templates und deren Nutzungskomfort sind für Website-Ersteller sehr

wichtig. Sie bestimmen, wie schnell sich ein gewähltes CMS bzgl. seines Designs an eine vorgegebene Corporate Identity anpassen lässt und dann im Anschluss an die Design-Erstellung sich die gewünschte Funktionalität über den Einsatz bereits vorhandener Erweiterungskomponenten konfektionieren lässt.

PHP-basierte CMS wie Joomla, Drupal und TYPO3 bieten hier sehr viel Komfort, da ihre Konzepte zur Erzeugung neuer Site-Templates einerseits sehr leicht zu handhaben sind, andererseits aber die Anpassung selbst der kleinsten Bestandteile der erzeugten Webseiten (wie z.B. das Rahmendesign von bestimmten Inhaltselementen) über das Site-Template möglich sind. Weiter bieten diese Systeme sehr flexible dynamische Möglichkeiten zur Positionierung von Inhalten. Sie verfügen bereits über eine so große Menge an hochqualitativen funktionalen Erweiterungen, dass fast alle klassischen Anforderungen an CMS-Systeme, wie Webseitenerstellung, Terminkalender, integrierte Foren und Blogs, Integration mit sozialen Netzwerken, usw. durch Einsatz bereits vorhandener Erweiterungen gelöst werden können. Daher sind diese PHP-basierten Systeme bei Firmen, die Webseiten im Auftrag Dritter erstellen, sehr beliebt.

Die PHP-basierten Systeme enthalten zwar eine Menge Funktionalitäten und lassen sich schnell und komfortabel für die verschiedensten Anwendungsgebiete „out oft he box“ mit verfügbaren Erweiterungen konfektionieren, ihnen fehlen aber andererseits einige der Funktionalitäten, die für Enterprise Anwendungssituationen in größeren Firmen gewünscht werden. Dies wären z.B. Funktionalitäten, wie die Versionierung von Inhalten, die Unterstützung von Staging oder die Unterstützung komplexerer Autorenworkflows. Einer der wichtigsten Eigenschaften, die Enterprise Websysteme aufweisen müssen, ist die Integrierbarkeit des Systems mit internen Geschäftsanwendungen, sei es das interne Dokumentenmanagementsystem, Systeme zum Customer-Relationship-Management (CRM) oder interne betriebliche Anwendungen.

Die Integration von diversen innerbetrieblichen Anwendungen in ein Websystem, das diese in integrierter Sichtweise für verschiedene Zielgruppen im Internet bereitstellt (Kunden, Vertreter der Firma, Außendienstmitarbeiter, etc.), ist das Anwendungsgebiet von Portalen.

3.4.2 Integration von Webanwendungen und -inhalten

Ein *Portal* [6] [7] [8] ist eine direkt auf die Integration verschiedener Anwendungen spezialisierte Webanwendung, die verschiedenen Nutzergruppen einen hochgradig personalisierten und integrierten Zugriff auf alle Informationen einer Firma, die für diesen Nutzer bereitgestellt werden sollen, bieten kann. Ähnlich zu Content-Management-Systemen, die zunächst rein auf das Management von Webinhalten spezialisiert waren, dann aber auch Funktionalitäten zur Erweiterung und Einbindung fremder Applikationen bekamen, sind Portale zunächst einmal auf die Integration von Anwendungen spezialisiert, haben aber im Laufe der Zeit immer mehr Funktionalitäten zum Content-Management „out oft he box“ hinzubekommen. Daher sind Portalserver wie Liferay [45] bzgl. des Content-Managements durchaus mit klassischen Content-Management-Systemen vergleichbar. Ihre Spezialität liegt aber auf der Integration von Anwendungen und der personalisierten Bereitstellung von Informationen, die weit über die Funktionalitäten hinausgeht, die hierfür z.B. PHP-basierte Content-Management-Systeme üblicherweise bieten. Im Folgenden sollen die typischen Funktionalitäten von Portalservern am Beispiel des Java-basierten Liferay-Servers exemplarisch vorgestellt werden.

Portalserver, wie Liferay, besitzen ein ähnlich flexibles Inhaltskonzept für Webinhalte, wie z.B. das TYPO3 CMS, d.h. Web-Inhalte sind zunächst einmal Inhaltsbausteine, die unabhängig von einer Webseite sind. Generell werden Webseiten in einem Portalserver als eigenständige Objekte angelegt, denen über ein Layout-Template (Achtung: Layout-Templates sind dabei keine Site-Templates) dann eine bestimmte Struktur (z.B. ein dreispaltiges-Layout für Inhaltselemente) vorgegeben wird. Inhaltlich werden die Seiten dann gefüllt, indem interaktive Komponenten (sogenannte Portlets [75]) aus einer Portlet-Bibliothek, die das System bereitstellt, vom Autor per Drag and Drop auf Platzhalterpositionen im Layout der Seite gezogen werden. Dabei kann ein Layout-Template durchaus bereits einige Positionen im Layout mit Portlets vorkonfiguriert haben, so dass z.B. inhaltliche Module wie eine Seitennavigation oder Ähnliches bereits bei jeder neuen Seite vorhanden sind.

Zum Einfügen von Webinhalten in eine neue Webseite verwendet der Autor ein spezielles Portlet, das Web-Content-Portlet. Ein Web-Content-Portlet ermöglicht gerade die Darstellung und Editierung von Web-Inhaltsbausteinen. Hierzu kann man nach Einfügen eines Web-Content-Portlets in eine Seite entweder ein oder mehrere bereits existierende Web-Inhaltsbausteine in das Web-Content-Portlet laden oder über einen eingebauten Web-Text-Editor direkt einen Web-Inhaltsbaustein neu erstellen und diesen mit Text und zugehörigen Bildern füllen. Ein Web-Content-Portlet kann dabei mehrere Web-Inhaltsbausteine zeigen und vom Web-Content-Portlet kann man weiter beliebig viele Instanzen an verschiedenen Stellen im vorgegebenen Seitenlayout positionieren. Eine Portalseite kann aber nicht nur Web-Content-Portlets, sondern beliebige andere Portlets enthalten. Auf diese Weise lassen sich in einem Portalserver ähnlich zu TYPO3 beliebige andere Inhalte (z.B. Kalendereinträge, ganze Kalendermonatsansichten, etc.) in Portalseiten integrieren.

Administratoren des Portals können dabei für unterschiedlichste Gruppen von Nutzern des Portals bereits Webseiten mit festen und durch einen Nutzer frei ersetzbaren Portlets vordefinieren. Nach einem Login eines Benutzers konfiguriert sich dann der persönliche Arbeitsbereich dieses Nutzers (sein privater Web-Arbeitsplatz) aus den für ihn als Nutzer vorgegebenen Seiten und allen Webseiten, die Gruppen zugeordnet sind, denen der Nutzer angehört. Wie bereits erwähnt, können diese privaten Arbeitsseiten dabei je nach Konfiguration an bestimmten Stellen auch noch individuell durch Löschen, Austausch oder Ergänzen von Portlets an die Bedürfnisse eines Benutzers angepasst werden. Je nach Konfiguration des Portalservers kann ein Benutzer weiter öffentliche Webseiten besitzen, mit denen er sich ein öffentliches persönliches Profil für seine Person im Web anlegt. Die Funktionalität eines Portalservers, dass man maßgeschneiderte, hochgradig personalisierbare interne Arbeitsbereiche für Nutzer des Portals und / oder bestimmte Arbeitsgruppen anlegen kann, ist eine zentrale Funktion eines Portals gegenüber einem CMS. Content-Management-Systeme sind vornehmlich dazu da, Informationen nach außen einer allgemeinen Öffentlichkeit zu präsentieren. Portalserver sind an Benutzerbedürfnisse anpassbare Arbeitsplattformen für öffentliche Besucher, Kunden, Zulieferer, Innen- und Außendienstmitarbeiter einer Organisation, die als ein Internet-zugänglicher webbasierter Arbeitsplatz für das Arbeiten notwendige Informationen und die zugehörigen Anwendungen bereitstellt.

Hierbei kommt dann eine weitere wesentliche Funktionalität von Portalen zum Tragen, wie sie z.B. durch das Portlet-Konzept in Java manifestiert ist. Portlets entsprechen im Wesentlichen vollständigen Java-basierten Webanwendungen, die als zusätzliche Besonderheit noch mitbringen, dass sie sich verhaltensmäßig als Bausteine in ein Portal integrieren können,

d.h., sie verstehen Aufforderungen des Nutzers von einer Darstellung als kleiner Baustein innerhalb einer Webanwendung in eine Vollbildansicht zu gehen und sich ganz zu minimieren, um mehr Raum für andere Anwendungen zu lassen. Weiter können sie in der Regel über die Portaloberfläche konfiguriert werden. Ansonsten entsprechen sie aber gewöhnlichen Java-basierten Webanwendungen, die beliebig komplexe Applikationslogik implementieren können. Java-Webanwendungen lassen sich auch so programmieren, dass sie sowohl als Portlet und auch als von einem Portalserver unabhängige Anwendung arbeiten können. Daher lässt sich nahezu jede beliebige Java-basierte Webanwendung so umbauen, dass sie als Portlet in einem Portal arbeiten kann.

Ein Portal stellt den Portlets ebenfalls APIs bereit, um direkt auf Funktionalitäten im Portal zuzugreifen. Erfolgt das wie beim Liferay-Server auch über Web-Service-Schnittstellen, so kann eine Webanwendung aus der Entfernung mit dem Portal interagieren. Generell sind Portalserver gemäß ihrer Aufgabe bestens darauf ausgerichtet, über Web-Service-Schnittstellen mit anderen Webanwendungen, Portalen, CMS-Systemen oder betrieblichen Fachanwendungen zu kommunizieren. Portlets stellen dabei oft visuelle Benutzeroberflächen zum Zugriff auf Anwendungen im Portal bereit, die nur über Services auf die eigentliche Business- und Datenlogik einer entfernten (z.B. im Intranet einer Firma installierten) Geschäftsanwendung zugreifen. Hier manifestiert sich der Anspruch eines Portals, als Integrationsplattform für beliebige Anwendungen zu dienen.

Die Anzeige von Portlets in Portalseiten gemäß einem vorgegebenen Seitenlayout lässt sich komplett programmatorisch steuern. Dies macht Portale zu perfekten Umgebungen für dynamische Mashup-Anwendungen, bei denen Daten von verschiedenen Hintergrunddiensten dynamisch, z.B. basierend auf eine Suchanfrage eines Nutzers angefragt, und dann in geeigneten Portlets dynamisch angezeigt werden müssen. Hierbei kann man durchaus mit Portlets arbeiten, die die Integration der Daten clientseitig per asynchronem JavaScript oder serverseitig innerhalb der Portletanwendung selbst erzeugen.

Für die clientseitige Integration von Daten eignet sich dabei vor allem auch der Einsatz von Web-Widgets. Web-Widgets sind JavaScript- und HTML-Schnipsel, die automatisch beim Aufbau einer HTML-Seite im Browser des Anwenders dynamisch ihre HTML-Präsentation per JavaScript erzeugen (oft sogar erst asynchron, nachdem die Webseite bereits vollständig vom Browser geladen wurde). Hierzu besitzt der Code eines Web-Widgets häufig nur ein HTML-Platzhalter-Tag, wie ein -Tag mit einer bestimmten ID, in das dann die eigentliche HTML-Oberfläche des Widgets asynchron durch eine Laderoutine („onLoad“-Handler) nachgeladen wird. Dabei kann der Ladehandler die hierfür benötigten Daten ebenfalls durch Nutzung von AJAX von einem entfernten Datenserver nachladen, was für clientseitige Mashupbildung ein Standardverfahren darstellt. Web-Widgets lassen sich durchaus mit Portlets, die auch serverseitige Logik besitzen, kombinieren. Hierbei erzeugt die serverseitige Logik dann je nach Anforderung die clientseitige Oberfläche als Widget.

Gadgets sind Web-Widgets, die gemäß dem OpenSocial-Standard [76], über eine Konfigurationsdatei zusätzlich beschrieben werden können, damit sie in OpenSocial-konformen Servern automatisch installiert und als Komponenten verwaltet werden können. Der Server stellt ihnen dabei über die OpenSocial-API zusätzliche Hilfsdienste, z.B. zum transparenten Zugang auf REST-Services zur Verfügung. Außerdem lassen sich Gadgets oftmals über den Server beim Einsetzen in eine Webseite konfigurieren.

Liferay integriert einen OpenSocial-konformen Gadget-Container und stellt bereits ein universelles Portlet bereit, das beliebigen Gadget-Code innerhalb eines <iframe>-Tags in einem Portlet als Wrapper anzeigen kann. Mit diesem Portlet lassen sich beliebige Gadgets in Liferay anzeigen.

Gadgets spielen auch bei Portalplattformen, wie iGoogle, oder in sozialen Netzwerkplattformen wie LinkedIn, die auf den OpenSocial-Standard basieren, eine zentrale Rolle. Daher gibt es mittlerweile eine Vielzahl von Gadgets, die im Internet frei verfügbar sind und für eigene Portalserver genutzt werden. Eine typische Anwendung von Gadgets ist die Integration mit sozialen Netzwerken. So stellen alle sozialen Netzwerke Gadgets bereit, um die eigene Homepage in einem CMS oder den eigenen Portalserver mit dem jeweiligen Netzwerk zu vernetzen (Like-Button, Google+ +1, social Badgets).

Auch bei der Integration fremder Anwendungen zeigen sich stärkere Unterschiede zwischen Content-Management-Systemen und Portalservern. CMS können zwar durch Erweiterungen funktional ergänzt werden, konzentrieren sich aber eher auf die Erweiterung der Präsentationsmöglichkeiten für Informationen unterschiedlicher Arten. Kerngeschäft von Portalservern ist dagegen die Integration externer Anwendungen: Sie sind optimiert auf die Integration beliebig komplexer Anwendungen mit beliebig komplexen externen Anwendungen über Serviceschnittstellen. Dabei unterstützen sie auch durchgängig die Bereitstellung eigener Funktionalitäten wiederum über Services, die von externen Anwendungen genutzt werden können.

3.4.3 Spezialisierte Webanwendungen

Neben Content-Management-Systemen und Portalservern gibt es noch eine ganze Reihe anderer Webanwendungen, z.B. für den Betrieb von Blogs, Wikis oder Foren. Eine Blogsoftware kann man als Sonderfall eines CMS ansehen, wobei sich ein Blog von einem CMS durch Eigenschaften wie eine geringe Anzahl von Autoren und eine sehr einfache Inhaltsorganisation abgrenzt. Bloginhalte sind in der Regel von Autoren geschriebene Artikel, die von der Blogsoftware automatisch auf einer Blogseite chronologisch mit einem Aufmacher angeordnet sind. Der Artikel ist dabei häufig in den Aufmacherteil und den weiteren Inhalt („mehr…“) unterteilt, wobei die Blogseite zunächst nur den Aufmacher jeden Artikels zeigt. Die bekannteste Blogsoftware Wordpress erlaubt es, neben automatisch verwalteten Artikeln auch einige statische Webseiten (z.B. für ein Impressum) anzulegen und enthält auch sehr gute Mechanismen für die Verwaltung und Einbindung von Medien in Artikel. Blogs werden häufig für eine persönliche Seite mit privaten Inhalten wie Reiseberichte, Mitteilungen über ein Hobby, wie Fotografie, oder ähnliches genutzt.

Ein Wiki dient demgegenüber als webbasiertes kollaboratives Dokumentationssystem: Die Besonderheit des Wikis ist, dass viele Personen gemeinsam an Inhaltsseiten schreiben können und die Inhalte der Seiten versioniert werden, so dass man schnell nach Änderungen, die nicht gewünscht sind, auf eine ältere Version zurückgehen kann. Eine vereinfachte Seitenauszeichnungssprache ermöglicht bei Wikiseiten eine Formatierung der Seiten ohne Einsatz eines GUI-basierten Online-Editors.

Foren dienen zur Diskussion verschiedener Thematiken zwischen Nutzern. Eine Forumsoftware erlaubt es hierzu, Foren zu verschiedenen Themenbereichen anzulegen, in

denen Nutzer dann einzelne thematische Artikel anlegen können (Forum-Thread), die dann von anderen kommentiert und erwidert werden können. Hierdurch entsteht eine Diskussion entlang eines Themas. Moderatoren können dabei häufig als Schiedsrichter fungieren und in eine bestehende Diskussion eingreifen, Teilnehmer von der Diskussion ausschließen oder Themen schließen, wenn die Diskussion nicht mehr zielführend ist.

Anwendungen, wie Blogs, Wikis und Foren sind typische Anwendungen, die in größeren Systemen wie CMS oder Portalen als Erweiterungen integriert werden können. Für größere Firmen gibt es daher grundsätzlich wenig Bedarf, solche Software gesondert zu installieren, wenn es Möglichkeiten gibt, diese in ein zentrales CMS oder Portal zu integrieren. Für einzelne Benutzer oder kleinere Benutzergruppen werden Softwaresysteme, wie Blogsoftware, Wikis oder Foren bereits oft als Cloud-basierte Services (siehe Kap. 4.3) bereitgestellt. Anstatt die Software also selbst bei einem Provider zu installieren, kann man sich deshalb seinen Blog z.B. gleich bei blogger.com oder wordpress.com in der Cloud erstellen. Dies vereinfacht die Verwaltung und das „Handling“ beträchtlich.

3.4.4 Standalone-Anwendung vs. Serviceorientierung

Neben diesen eher generischen Webanwendungen gibt es eine ganze Reihe von Webanwendungen, die so speziell sind, dass sie entweder nur für spezielle Nutzergruppen interessant sind, oder aber sogar für eine Firma oder Institution dediziert entwickelt werden. Hierzu gehören im Umweltbereich die webbasierten Fachanwendungen. Diese Anwendungen erfordern oft die Implementierung spezieller Fachlogik oder müssen mit großen Datenmengen umgehen können. Sie stellen daher an die Programmierung große Anforderungen, weshalb diese Anwendungen typischerweise als „Standalone“-Webanwendungen konzipiert und genutzt werden. In der LUBW werden solche Anwendungen gemäß der UIS-Forschungskooperation häufig als Java-basierte Webanwendungen entwickelt. Für diese Art von Fachanwendungen wird man in den wenigsten Fällen passende Content-Management-Systeme oder Portallösungen finden, die man „out of the box“ als Erweiterungen einbinden kann.

Trotzdem ist es oftmals sinnvoll, Teilaspekte solcher Anwendungen, z.B. bestimmte Präsentationen von Daten, in CMS oder Portale zu integrieren oder sogar für mobile Anwendungen zugreifbar zu machen. Bei komplexeren dedizierten Webanwendungen ist es daher üblich, größere Teile der Funktionalität über Web-Services für andere Anwendungen bereitzustellen. Dies ermöglicht die Entwicklung von CMS- oder Portalerweiterungen, die Teilfunktionalitäten der komplexen Fachanwendung innerhalb des CMS oder Portalservers als Erweiterung bereitstellen. Weiter ermöglichen solche Web-Service-Schnittstellen die Nutzung von Fachanwendungssemantik und Daten innerhalb von mobilen Anwendungen (Apps) als Frontend zur Fachanwendung.

Schließlich lässt sich die vollständige IT-mäßig erschlossene Business-Funktionalität einer Organisation im Sinne einer Service Oriented Architecture (SOA) [77] als modulare Komponenten mit zugehörigen Serviceschnittstellen erschließen und dann in verschiedenen (webbasierten) Endanwendungen optimal nutzen. Fachbasierte Webanwendungen, zugehörige Erweiterungen für CMS und Portale sowie mobile Apps stellen in einer solchen Architektur nur noch GUI-Frontends zur serviceorientiert bereitgestellten Business-Logik dar, die die

Businessfunktionalität für den Nutzer abhängig von Gerät und Ort, an dem er arbeitet, optimal erschließen.

3.5 Serviceorientierte Webdienste

In sich geschlossene Businessfunktionalitäten werden im Kontext des Web heutzutage im Sinne einer SOA [77] als eigenständig nutzbare Dienstleistungen bereitgestellt, sodass sie über Serviceschnittstellen in andere (Web-)Anwendungen integrierbar sind. Dies kann im Kontext einer Firma innerhalb der eigenen Softwarelandschaft erfolgen, aber auch global über den Firmenkontext hinweg. Letzteres ermöglich das Schreiben von Webanwendungen, die Businessfunktionalitäten verschiedener Organisationen in einer Anwendung zusammenführen: Ein typische Beispiel hierfür ist die Webanwendung eines Tourismusanbieters, über die ein Kunde eine Urlaubsreise buchen kann, bei der Flug, Hotel, Mietauto und verschiedene Veranstaltungen vor Ort als Gesamtpaket individuell zusammengestellt werden können. An einer solchen Anwendung sind Services von Hotelanbietern, Fluglinien, Autovermietungen und Veranstaltungsagenturen beteiligt, die über Serviceschnittstellen in die Gesamtapplikation integriert werden.

Während das geschilderte Szenario eher eine Anwendung von Services im Kontext einer Business-to-Business-(B2B)-Lösung ist, bieten mittlerweile sehr viele Internetfirmen auch in sich geschlossene Services zur generellen Nutzung im Internet an. Solche Services lassen sich in eigene Webanwendungen einbinden, von Nutzern direkt über Webinterfaces und/oder mobilen Apps nutzen oder mit dem eigenen Desktoprechner integrieren. Die Geschäftsmodelle der Anbieter reichen hier von einer kostenfreien, oft werbefinanzierten Nutzung bis hin zu gebührenpflichtigen Angeboten, die gegenüber der kostenfreien Variante häufig mit erweiterter Funktionalität verbunden sind.

Typische Vertreter solcher allgemein nutzbarer Services sind Dienste zum Management und zur Bereitstellung von Dateien im Internet (dropbox, Google Drive), zum Management und zur Bereitstellung von Medien (YouTube für Videos, Picasa, Flickr, 5oopx für Bilder), zur Verwaltung von Aufgaben und Terminen (Google Calendar und Tasks, Toodledo), Email-Dienste (Google Mail, Microsoft Outlook Email Services, GMX, web.de), soziale Bookmarkingsysteme (dig), Internet-basierte Kartendienste (Google Maps, Bing Maps, Yahoo Maps) oder Büroanwendungsangebote (Google Docs – jetzt in Google Drive integriert, Microsoft Office 365) sowie Kommunikationsdienstleistungen, wie Videotelefonie (Skype), Videokonferenzen (GotoMeeting), Chat, etc.

Größere Anbieter wie Google oder Microsoft setzen dabei zunehmend darauf, Kunden ein Gesamtangebot an serviceorientierte Diensten anzubieten, das es ihnen erlaubt, einen großen Teil ihres privaten oder geschäftlichen IT-Bedarfs über den jeweiligen Dienstanbieter vollständig unter Nutzung Internet-basierter Services für die Datenhaltung, Businesslogik, Webanwendungen sowie Apps als GUI-Frontendanwendungen abzuwickeln.

Als Beispiel für solche Komplettangebote sollen im Folgenden einige Dienstleistungen der Google-Dienste vorgestellt werden.

3.5.1 Google-Business-Dienste

Ein Gesamtpaket von Google-Diensten für Geschäftskunden sind die „Google Business Services“ [78]. Dieses Paket soll für Kunden die klassische Microsoft-basierte Bürolösung ablösen und ein Internet-basiertes, serviceorientiertes Dienstleistungspaket bereitstellen, das Verwaltung von Nutzern und Gruppen innerhalb der eigenen Google-Business-Domain, Kalender- und Aufgabenmanagement, Kontakt- und Adressverwaltung, Email- und Kommunikationsdienste, Groupware- und Office-Funktionalitäten, Datei- und Medienmanagement, soziale Netzwerkfunktionalitäten und Social Media sowie Webpräsentation über Blogs und vereinfachte CMS-Funktionalitäten in einem Paket zusammenfasst. Bis vor kurzem gab es eine kostenlose Version der Google-Business-Dienste, die bis zur Anzahl von 10 Benutzern kostenlos genutzt werden durfte. Andere Kunden können kommerziell eine Google-Business-Domain erwerben, die nach Anzahl aktiver Nutzer abgerechnet wird.

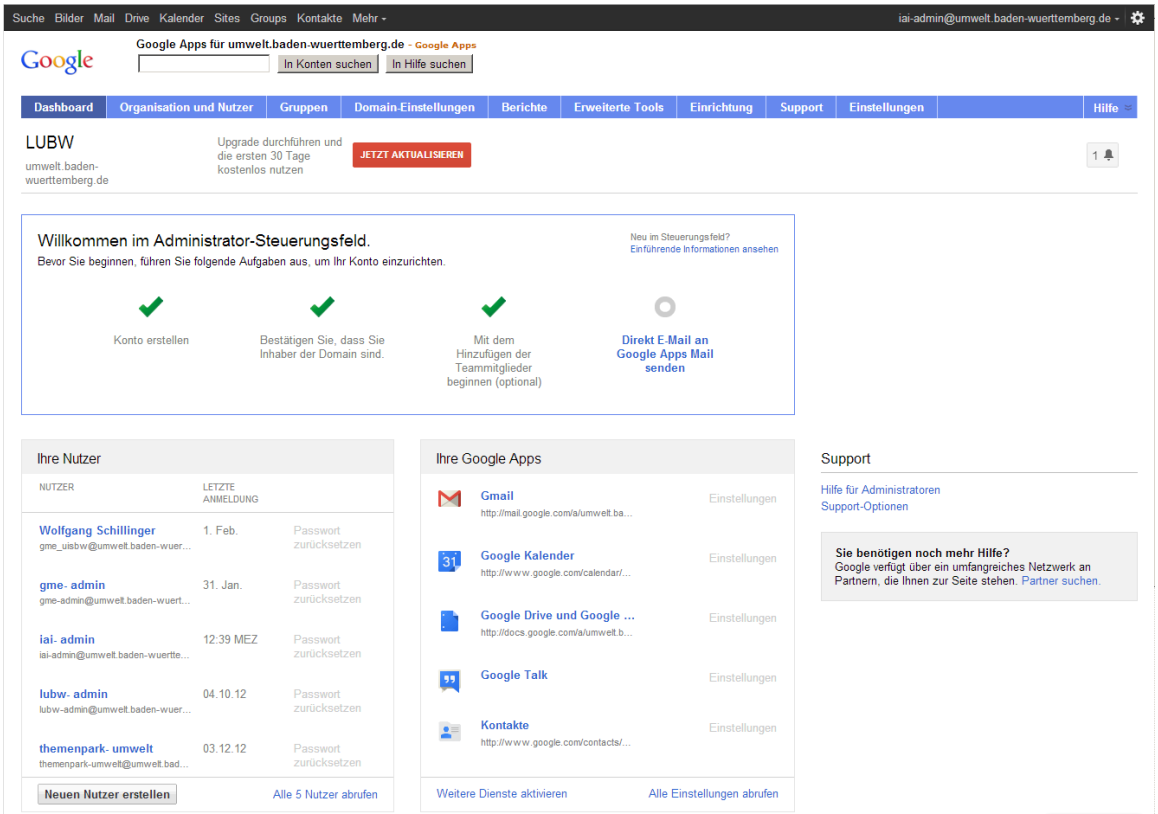


Abb. 3.15: Administrationsoberfläche einer Google-Business-Domain (in diesem Fall der Google-Business-Domain der LUBW für die Evaluation von Google-Diensten)

Das Gesamtpaket bietet dabei für Business-Administratoren und -Nutzer ein durchgängiges Gesamtkonzept, wie man es von integrierten Büronetzwerken kennt. Jeder Benutzer hat nur einen Account, mit dem er sich bei allen Diensten autorisiert, ohne dass er seine Accountdaten immer wieder neu eingeben muss (Single-Sign-On). Die bereitgestellten Funktionalitäten sind dabei an jedem Ort mit fast jedem Gerät verfügbar (über Webbrowser, Desktopanwendungen, über Apps auf mobilen Geräten oder sogar auf einem Internet-fähigen TV). Eine Internetverbindung ist bei vielen Anwendungssituationen nicht mehr zwingend erforderlich,

da Anwendungen wie Gmail, Google Docs oder Google Drive auch offline genutzt werden können.

Abb. 3.15 zeigt die Übersichtsseite der Administrationsoberfläche einer Google-Business-Domain (in diesem Fall der LUBW für die Evaluation der Google-Business-Dienste und der Google Maps-Engine – siehe die folgenden Seiten). Über die Administrationsoberfläche kann man zunächst wesentliche Einstellungen der Domain sowie der einzelnen Dienste, wie Gmail, Kalender, etc. verwalten. Hiermit kann z.B. auch die Google-Business-Domain mit weiteren Dienstservern auf dem Internet, wie einem getrennten Emailserver und damit einer externen Maildomain für die eigene Organisation oder einem externen Webserver und einer eigenen Internetdomain verknüpft werden.

Wie Abb. 3.16 zu entnehmen, kann man über die Administrationsoberfläche Unterorganisationseinheiten, Nutzergruppen und Nutzer anlegen, die dann Organisationseinheiten zugeordnet werden können. Das Benutzungsrechtesystem einer Google-Business-Domain erlaubt es dabei später, Nutzern je nach Zugehörigkeit zu Organisationseinheiten oder Gruppen entsprechende Rechte zur Nutzung bestimmter Dienste zu geben. Die Administrationsoberfläche besitzt auch Funktionalitäten, um Nutzer in einem Batchprozess aus einem externen Nutzerverwaltungssystem zu importieren bzw. zu exportieren. Zugehörige Service-APIs ermöglichen weiter die programmatorische Synchronisation von Nutzeraccounts der Google-Business-Domain mit Accounts von externen Systemen, z.B. einem Liferay-Portalserver. Hier stehen Mechanismen wie Active Directory Sync (mit Active Directories von Windows Domänen oder LDAP-Servern), SAML-basiertes Single-Sign-On zwischen verschiedenen Systemen oder die Unterstützung entfernter Autorisierung über OAuth- und OpenID-Accounts zur Verfügung. Die Administrationsoberfläche bietet weiter Funktionalitäten zur Überwachung von Nutzungsaktivitäten und zum Reporting der Nutzung von Diensten durch die Benutzer einer Business-Domain.

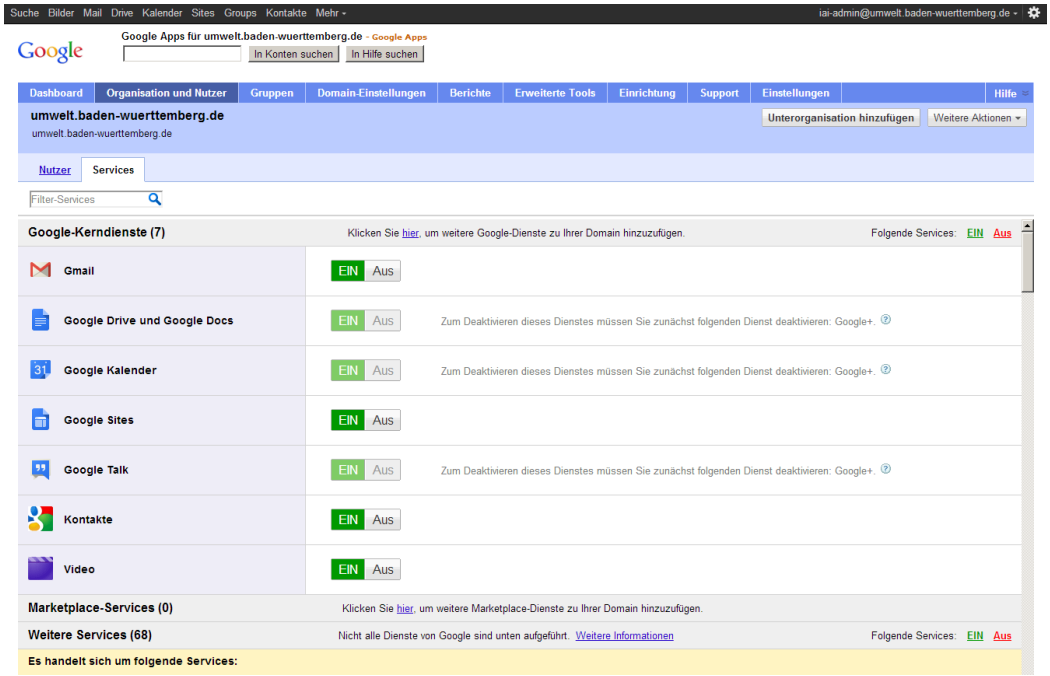
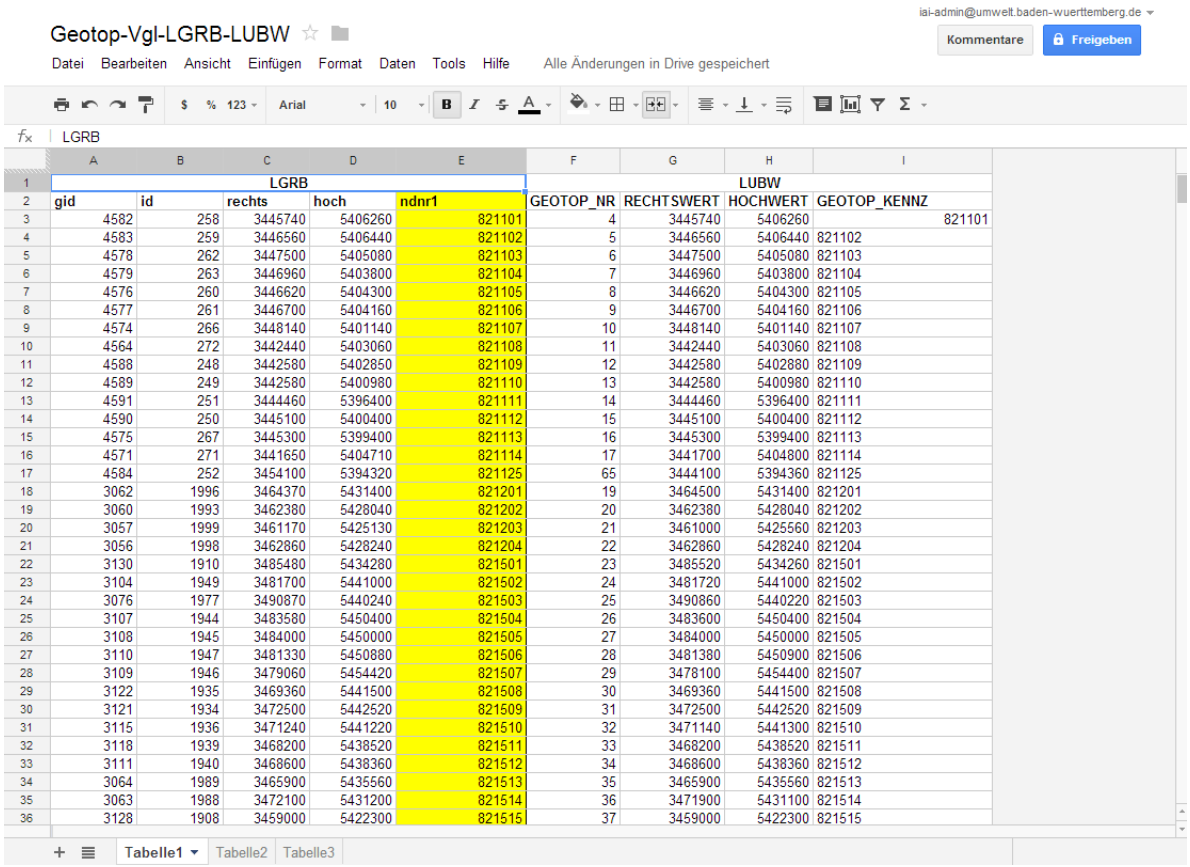


Abb. 3.16 : Administrationsbereich einer Google-Business-Domain zur Aktivierung oder Deaktivierung von Diensten

Über das Dashboard kann der Benutzer in einen Konfigurationsbereich gelangen, in dem er Basisdienste der Google-Business-Domain für eine Organisation oder Unterorganisationen aktivieren oder deaktieren kann. Standardmäßig sind hier Kerndienste wie Gmail, Google Drive und Google Docs, Google Kalender, Google Talk, Kontakte, Video (YouTube), Google+, usw. aktiviert. Darüber hinaus gibt es hier viele weitere Dienste, die aktiviert werden können, und ein Marketplace-Zugang erlaubt dem Administrator sogar das Hinzufügen neuer, frei verfügbarer oder kommerzieller servicebasierter Webapplikationen (z.B. Werkzeuge zur Diagrammerstellung oder zum Projektmanagement).

Anwendungen sind dabei Webanwendungen, die sich mit ihren Oberflächen in den Browser integrieren, oftmals aber zur Datenhaltung Services im Hintergrund verwenden, die auch von anderen Clients, wie Apps auf mobilen Geräten, verwendet werden können, so dass Applikationen innerhalb einer Google-Business-Domain nicht nur über einen Browser, sondern auch über andere Geräte genutzt werden können.



	A	B	C	D	E	F	G	H	I
1			LGRB					LUBW	
2	gid	id	rechts	hoch	ndnr1	GEOTOP_NR	RECHTSWERT	HOCHWERT	GEOTOP_KENNZ
3	4582	258	3445740	5406260	821101	4	3445740	5406260	821101
4	4583	259	3446560	5406440	821102	5	3446560	5406440	821102
5	4578	262	3447500	5405080	821103	6	3447500	5405080	821103
6	4579	263	3446960	5403800	821104	7	3446960	5403800	821104
7	4576	260	3446620	5404300	821105	8	3446620	5404300	821105
8	4577	261	3446700	5404160	821106	9	3446700	5404160	821106
9	4574	266	3448140	5401140	821107	10	3448140	5401140	821107
10	4564	272	3442440	5403060	821108	11	3442440	5403060	821108
11	4588	248	3442580	5402850	821109	12	3442580	5402880	821109
12	4589	249	3442580	5400980	821110	13	3442580	5400980	821110
13	4591	251	3444460	5396400	821111	14	3444460	5396400	821111
14	4590	250	3445100	5400400	821112	15	3445100	5400400	821112
15	4575	267	3445300	5399400	821113	16	3445300	5399400	821113
16	4571	271	3441650	5404710	821114	17	3441700	5404800	821114
17	4584	252	3454100	5394320	821125	65	3444100	5394360	821125
18	3062	1996	3464370	5431400	821201	19	3464500	5431400	821201
19	3060	1993	3462380	5428040	821202	20	3462380	5428040	821202
20	3057	1999	3461170	5425130	821203	21	3461000	5425560	821203
21	3056	1998	3462860	5428240	821204	22	3462860	5428240	821204
22	3130	1910	3485480	5434280	821501	23	3485520	5434260	821501
23	3104	1949	3481700	5441000	821502	24	3481720	5441000	821502
24	3076	1977	3490870	5440240	821503	25	3490860	5440220	821503
25	3107	1944	3483580	5450400	821504	26	3483600	5450400	821504
26	3108	1945	3484000	5450000	821505	27	3484000	5450000	821505
27	3110	1947	3481330	5450880	821506	28	3481380	5450900	821506
28	3109	1946	3479060	5454420	821507	29	3478100	5454400	821507
29	3122	1935	3469360	5441500	821508	30	3469360	5441500	821508
30	3121	1934	3472500	5442520	821509	31	3472500	5442520	821509
31	3115	1936	3471240	5441220	821510	32	3471140	5441300	821510
32	3118	1939	3468200	5438520	821511	33	3468200	5438520	821511
33	3111	1940	3468600	5438360	821512	34	3468600	5438360	821512
34	3064	1989	3465900	5435560	821513	35	3465900	5435560	821513
35	3063	1988	3472100	5431200	821514	36	3471900	5431100	821514
36	3128	1908	3459000	5422300	821515	37	3459000	5422300	821515

Abb. 3.17: Google Drive Tabellenkalkulation mit Vergleichsdaten zu Geotopen. Die Daten wurden dabei aus verschiedenen Excel-Dateien für den Vergleich nach Google Drive importiert

Abbildung 3.17 zeigt als Beispiel für eine Google-Business-Browser-Anwendung die Oberfläche von Google Docs bei der Bearbeitung einer Tabellenkalkulation, die auf Google Drive gespeichert ist, im Browser. Man sieht, dass diese Anwendung eine für Tabellenkalkulationsprogramme typische Oberfläche bietet. Dabei sind die gängigsten Funktionalitäten von Tabellenkalkulationsprogrammen, aber durchaus nicht die kompletten Funktionen von Programmen wie Excel, implementiert. Neben Tabellenkalkulation enthält Google Docs weitere Webanwendungen für die Erstellung und das Management von Präsentationen (wie Power-

point), Textdokumenten (wie Word), Vektorgrafik (grafische Diagramme). Google Docs lässt sich beliebig durch Webanwendungen erweitern, die über den Google Marketplace bezogen werden können. Solche Anwendungen legen dann häufig ihre Daten wieder in Google Drive innerhalb der Business-Domain ab. Hierzu gehören Diagrammtools, PDF-Bearbeitungswerkzeuge, Fotoeditoren, Videobearbeitung und selbst Entwicklerprogramme, z.B. zur Erstellung von Mockups für Benutzeroberflächen und vieles mehr.

Ein eingebauter Formulareditor und eine Script-Ausführungsmaschine erlaubt es Entwicklern, auf Basis von Google Docs und Google Drive schnell maßgeschneiderte Erfassungsanwendungen für Daten zu erzeugen, die von Nutzern der Google-Business-Domain verwendet werden können. Informationen, die mit Google Docs-konformen Anwendungen erstellt wurden und auf Google Drive gespeichert werden, lassen sich auch kontrolliert mit externen Anwendern teilen und damit Nutzern außerhalb der Google-Business-Domain zugänglich machen. Das funktioniert auch für mit Appscript erstellte Anwendungen, die daher z.B. für die Erfassung von Daten über Dritte genutzt werden können. Mit Inhalten, die man ausgehend von Google Drive z.B. auf Google+ im sozialen Netzwerk von Google teilt, können externe Nutzer direkt interagieren. Ein Formular für eine Umfrage kann direkt ausgefüllt werden, eine Google Docs-Präsentation oder eine Google Docs-Tabellenkalkulation direkt im Browser angezeigt werden.

Die so durch eine Google-Business-Domain gegebene Büroumgebung lässt sich, wie bereits angesprochen, beliebig um Anwendungen von Drittanbietern und auch durch eigenentwickelte Webanwendungen erweitern. Für die Entwicklung solcher eigener Webanwendungen, die Google-Business-Dienst kompatibel sind, kann dabei die Google Apps-Engine als Cloud-Entwicklungsplattform genutzt werden.

Der soziale Teil der Bürokommunikationsplattform einer Google-Business-Domain wird durch die soziale Netzwerksoftware Google+ definiert. Google selbst bezeichnet Google+ weniger als soziales Netzwerk sondern als Integrationsplattform für alle Google-Dienste. Google+ fügt einer Business-Domain die Fähigkeit hinzu, dass Benutzer ein Profil mit eigener Pinnwand besitzen, über das sie Nachrichten mit anderen Benutzern der Domain austauschen können. Ähnlich zu Facebook kann ein Google-Business-Nutzer dabei seine Kollegen zu Gruppen (den sogenannten Circles) zusammenfassen und Nachrichten an all diese Personen schicken. Die Funktionalität der Circles ist dabei eng mit derjenigen der Google-Business-Gruppen synchronisiert, so dass Nutzergruppen in der Business-Domain spezielle Circles sind und umgekehrt. Auf diese Weise lässt sich eine Diskussion über Gruppen von Nutzern führen. Eingebaute Funktionalitäten zur Durchführung von Videokonferenzen (sogenannte Hangouts), in denen Circles oder Gruppen beteiligt sind, oder zur Durchführung gruppenbasierter Projektsitzungen bieten dabei alternative Formen zur kollaborativen Diskussion in Gruppen.

Hangouts sind dabei besonders interessant, da sie es, ähnlich zu Anwendungen wie GotoMeeting, erlauben, in einer Videokonferenz Ressourcen wie den Computer-Desktop eines Teilnehmers, ein Dokument aus Google Drive oder eine beliebige Google Docs-konforme Webanwendung innerhalb des Hangouts als Hauptbildschirm zu teilen. Hier funktionieren sogar interaktive Anwendungen, über die man dann gleichzeitig an Dokumenten, z.B. einer Mindmap, einer Tabellenkalkulation, etc. arbeiten kann. Abb. 3.18 zeigt als Beispiel ein Hangout einer Softwareentwicklungsgruppe, die über ein webbasiertes Diagramm-

tool, welches im Google-Business-Dienste-Appstore vorhanden ist, kollaborativ ein Softwareablaufdiagramm erstellt und diskutiert.

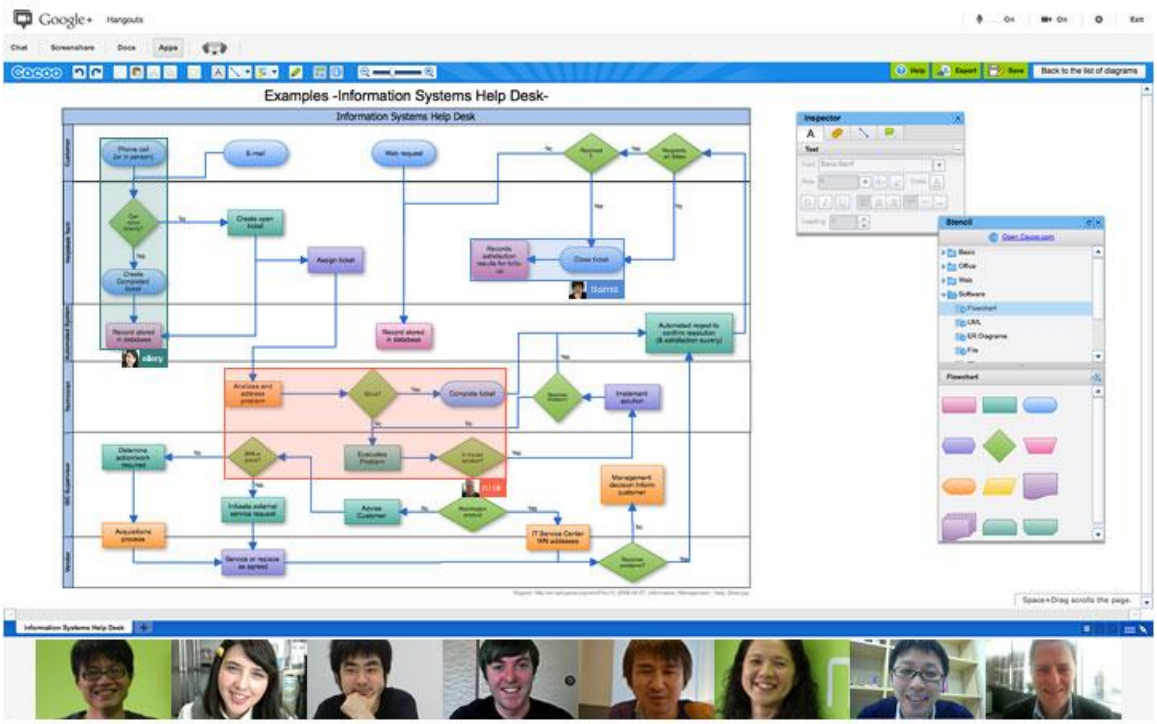


Abb. 3.18: Google+ Hangout[1] einer international verteilten Gruppe von Informatikern zur Erstellung eines Ablaufdiagrammes für eine Software. Das Diagrammtool im oberen Teil des Bildschirms ist eine kollaborative Websoftware, d.h. jeder Teilnehmer der Videokonferenz kann das Diagramm modifizieren

Prinzipiell lassen sich alle Informationen in Google+ mit anderen Personen teilen, die als Informationen in einer Google-Business-Domain verfügbar sind. Administratoren der Domain können dabei einstellen, dass Inhalte, die nur innerhalb von privaten Nutzerkreisen der Domain geteilt werden sollen, auch für Gruppen, an denen externe Benutzer beteiligt sind, verfügbar sind. Prinzipiell erlaubt es eine Google-Business-Domain natürlich, dass spezielle Kalender, Dokumente oder andere Informationen und Anwendungen auch größeren Benutzerkreisen, im Extremfall sogar der allgemeinen Öffentlichkeit, zur Verfügung gestellt werden. Auf diese Weise lassen sich z.B. Dokumente, aber auch interaktive Anwendungen mit externen Nutzern austauschen oder für diese bereitstellen.

3.5.2 Google Maps-Engine und Google Maps-API

Für Firmen, die professionelle Karten im Internet bereitstellen müssen, ist die Google Maps-Engine (GME) [79] eine willkommene Erweiterung der Google-Business-Dienste. Zur Nutzung der GME ist dabei eine Google-Business-Domain zur Verwaltung der nötigen Nutzer und Accounts zwingend erforderlich, auf eine Nutzung der zugehörigen Services, wie Google Drive, kann allerdings verzichtet werden. Verzichtet man auf Google Drive, können allerdings eine Reihe von kartenbezogenen Diensten nicht genutzt werden, da z.B. Google Fusion Tables darauf basieren.

[1] Screenshot von mygoogleplus.de aus Beitrag über Nutzung von Hangouts auf deren Homepage

Die Google Maps-Engine-Webanwendung erlaubt als Basisfunktionalität zunächst das Hochladen Karten-relevanter Daten in die Google Cloud (Anlegen von Datenquellen). Hierbei können Vektor-basierte Daten wie Esri Shapefiles, CSV-Dateien mit assoziierter GDAL-Datei (Geospatial Data Abstraction Library-Format), KML- oder auch MapInfo-TAB-Dateien importiert werden. Eine Vektordatei kann dabei bis zu einer Million Features haben. Größere Datenmengen müssen entsprechend in separate Datenquellen aufgesplittet werden. Neben Vektorgrafikdaten, können auch georeferenzierte Rastergrafiken hochgeladen werden. Hier sind Daten im JPEG2000, GeoTIFF, JPEG, TIFF und MrSID Format erlaubt. Die GME unterstützt zurzeit allerdings nicht das BigTIFF-Format. Bildbasierte Daten werden zunächst in Kacheln zerlegt und dann in einem Tilecache der Cloud hinterlegt. Die Größe von Bilddaten ist dabei auf 10 GB pro Datenquelle beschränkt. Größere Datenmengen müssen entsprechend auf mehrere Datenquellen aufgeteilt werden.

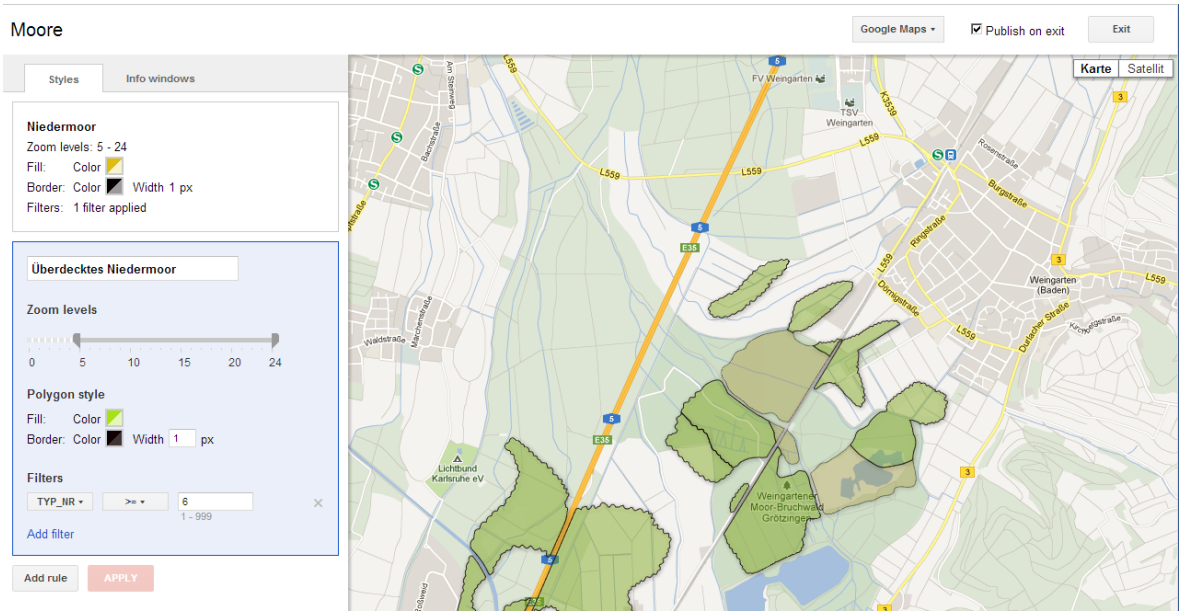


Abb. 3.19: Editor zur Definition von Stilklassen eines Geodatenlayers in der Google Maps-Engine. Hier werden zwei Typen von Mooren (Niedermoor und „überdecktes Niedermoor“ mit unterschiedlichen Grüntypen) definiert

Nach dem Erstellen von Datenquellen können daraus Layerobjekte erzeugt werden. Innerhalb eines Layers kann man durch Definition von Filtern die gesamten Daten in Klassen aufteilen (z.B. alle Moore in Klassen von Mooren, wie Hochmoore, Niedermoore, etc.) und diesen Klassen von Geoobjekten verschiedene Stileigenschaften wie Farben, Transparenz, etc. zuweisen (vgl. Abb. 3.19). Dabei kann man auch definieren, wie eventuell vorhandene Metadaten von Geodatenobjekten in einem Balloon-Popup auf einer Karte angezeigt werden, wenn das Objekt in einer Karte angeklickt wird. Hier lässt sich beliebiger HTML-Code einbinden, der Bezug auf etwaige Metadatenfelder nimmt.

Schließlich lassen sich die einzelnen Layer in der GME-Webanwendung zu kompletten Karten zusammenbauen, wie dies Abb. 3.20 verdeutlicht.

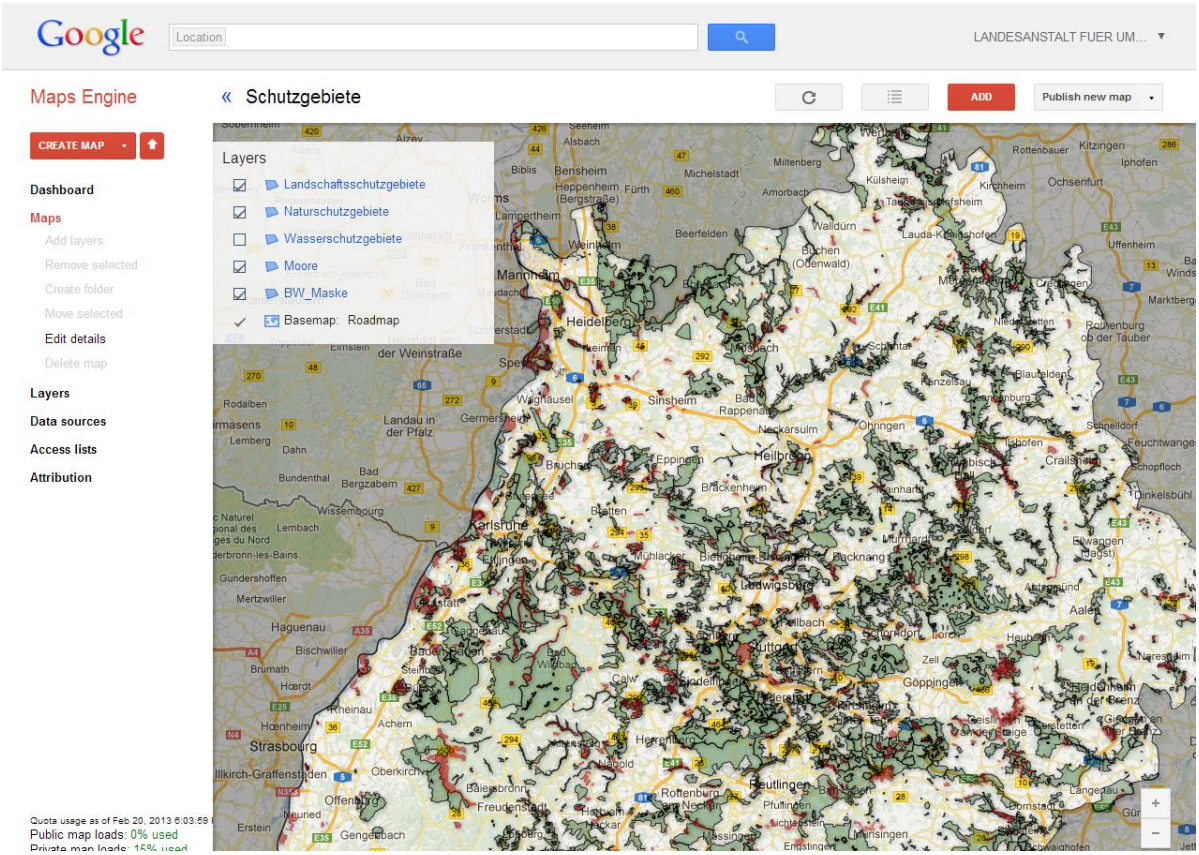


Abb. 3.20: Ansicht des Karteneditors der Google Maps-Engine. Im Bild sieht man eine bereits aus verschiedenen Layerobjekten zusammengesetzte Karte mit ihrer Legende

Innerhalb der Google Maps-Engine (GME) erstellte Karten können zur Nutzung freigegeben („Publish new map“ in der Abbildung) und auf eigenen Webseiten oder Webanwendungen verwendet werden. Hierzu benötigt man nur die URL der Karte, die beim Veröffentlichen generiert wird. Über diese URL lässt sich z.B. die fertige interaktive Karte direkt anzeigen oder über einen IFrame in eine eigene Webseite einbinden. Ein Zugriffsschutzmechanismus erlaubt es dabei festzulegen, auf welchen Domains die so entstandene Karte genutzt werden darf. Zur Nutzung von GME Karten in den verschiedensten GIS-Anwendungen wird für jede Karte eine WMS-URL bereitgestellt.

Veröffentlichungs-Links: Google Earth | Google Maps | ***WMS*** | Karten-Widget | Karten-ID

Leider lassen sich mit dem aktuellen Implementierungsstand der Google Maps-Engine nur rudimentär dynamische Layer, die von externen anderen Systemen stammen, innerhalb vorkonfigurierter Karten nutzen. Die Einbindung solcher per Link referenzierter Layer ist nur für externe Services erlaubt, die das KML-Format zurückgeben. Außerdem können solche dynamischen KML-Layer zurzeit nur in Google Earth, nicht aber in einem Google Maps-Client angezeigt werden. Externe Quellen, die KML produzieren, lassen sich jedoch statisch einbinden. Hierzu muss die gesamte KML-Datei als Datenquelle in die GME hochgeladen werden. Anschließend lässt sich daraus ein GME-Layer erzeugen. Die hierbei hochgeladenen

Daten lassen sich jedoch nachträglich nicht mehr dynamisch ändern (außer durch erneutes Hochladen).

Daher lassen sich zum momentanen Zeitpunkt leider auch nicht sogenannte Fusion Tables oder generell tabellarische Daten, die georefenziert und in Google Drive gespeichert sind, dynamisch als Layer in statische GME- Karten einbinden. Die Erzeugung solcher Karten mit Layern, deren Daten dynamisch geändert werden können, ist nur programmatorisch unter Nutzung der Google Maps-JavaScript-API möglich.

Über die Google Maps-API lassen sich alle Layer und Karten, die innerhalb der GME konfiguriert sind, in dynamisch erzeugte Karten einbinden. Die GME eignet sich in diesem Kontext im Wesentlichen dazu, bestimmte Layer und Karten statisch als Basiskarten für darauf aufbauende Karten vorzukonfigurieren, auf die dann weitere dynamische Layer aufgeblendet werden können.

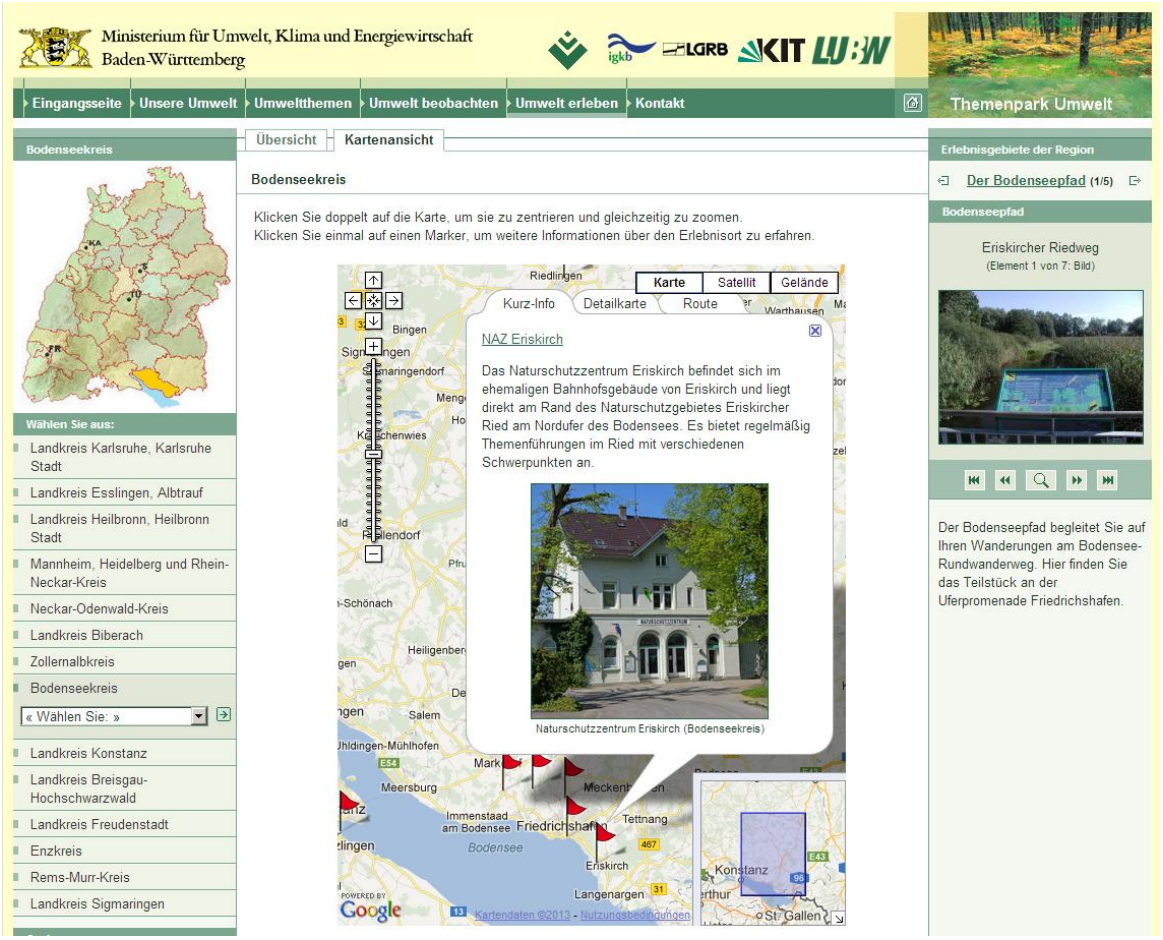
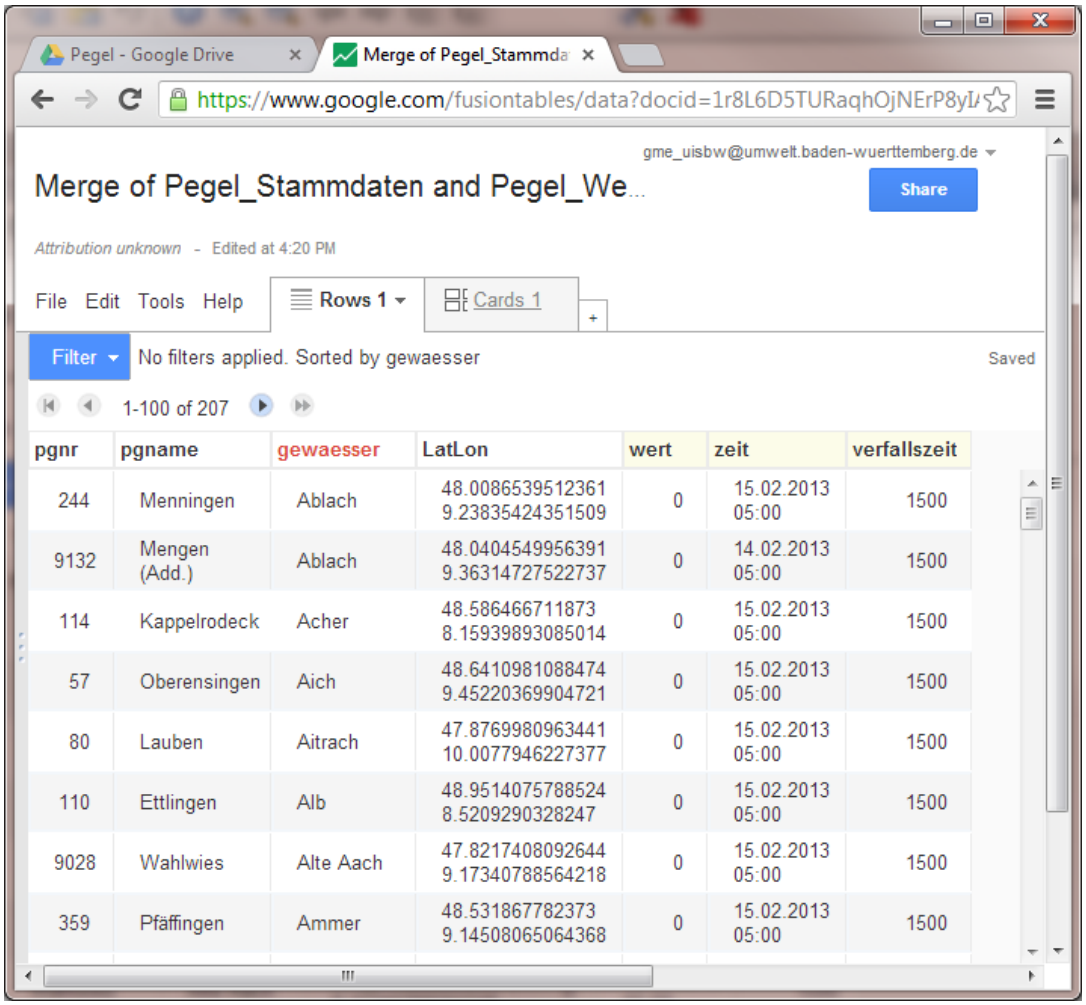


Abb. 3.21: Die Google Maps-Erlebniskarte im Themenpark ist mit der Google Maps-API implementiert und ruft die angezeigten Objekte als GeoRSS-Feed vom Themenpark ab. Der Feed kann dann direkt als Layer von Markern auf einer Google Maps-Grundkarte angezeigt werden

Über Basislayer, die aus der GME kommen, lassen sich mit der Google Maps-API dynamisch georeferenzierte Daten als weitere Layer anzeigen. Die Google Maps-API unterstützt dabei sowohl das Generieren von Layern (Overlays), die aus einzelnen georeferenzierten Vektorzeichnungsobjekten bestehen (z.B. Liste von georeferenzierten Punktobjekten, Polygonzügen, Kreisen oder Rechtecken, siehe auch Abbildung 3.21), als auch die Generierung

von Layern aus dynamischen Datenquellen in Formaten, die Daten georeferenziert liefern (GeoRSS oder KML).

Eine sehr flexible und einfache Methode, um größere, georeferenzierte Datenmengen auf einer Google Maps-Grundkarte abzubilden, ist die Nutzung sogenannter *Fusion Tables*. Eine Fusion Table ist eine spezielle Form von Tabellen, die in Google Drive als tabellenartiges Dokument gespeichert werden und größere Mengen von Datenobjekten enthalten kann, die jeweils durch eine Tabellenzeile beschrieben sind. Ein Tabelleneintrag beschreibt dabei typischerweise ein Objekt, z.B. einen Messwert zur Luftqualität mit seinen Attributen (Zeitpunkt der Messung, Messstation, an dem Messwert aufgenommen wurde, Messgröße, z.B. Ozon und Messwert). Mehrere Tabellen können dabei zu einer größeren Tabelle zusammengefasst werden, z.B. eine Tabelle, die die Stammdaten einer Messstation enthält und eine, welche die aktuellen Messwerte enthält, zu einer integrierten Tabelle (vgl. Abb. 3.22). Zur Fusion Tables-Funktionalität gehört eine REST-basierte Serviceschnittstelle, mit der Fusion Tables angelegt, modifiziert und ausgelesen werden können. Die API unterstützt dabei zum Lesen der Daten eine mächtige SQL-ähnliche Abfragesprache. Mit Fusion Tables lassen sich daher sehr einfach Webapplikationen erstellen, die Daten in einer Tabelle sammeln, filtern und auslesen und dann z.B. in Form einer Tabelle oder Grafik innerhalb einer Webseite darstellen.



pgnr	pgname	gewaesser	LatLon	wert	zeit	verfallszeit
244	Menningen	Ablach	48.0086539512361 9.23835424351509	0	15.02.2013 05:00	1500
9132	Mengen (Add.)	Ablach	48.0404549956391 9.36314727522737	0	14.02.2013 05:00	1500
114	Kappelrodeck	Acher	48.586466711873 8.15939893085014	0	15.02.2013 05:00	1500
57	Oberensingen	Aich	48.6410981088474 9.45220369904721	0	15.02.2013 05:00	1500
80	Lauben	Aitrach	47.8769980963441 10.0077946227377	0	15.02.2013 05:00	1500
110	Ettlingen	Alb	48.9514075788524 8.5209290328247	0	15.02.2013 05:00	1500
9028	Wahlwies	Alte Aach	47.8217408092644 9.17340788564218	0	15.02.2013 05:00	1500
359	Pfäffingen	Ammer	48.531867782373 9.14508065064368	0	15.02.2013 05:00	1500

Abb. 3.22: Fusion Table mit Daten zu Pegeln und Pegelmessstellen: Die Tabelle wurde aus zwei Datenquellen (den Pegelstammdaten und den eigentlichen Messdaten zusammengefügt)

Das Besondere an Fusion Tables in Verbindung mit der Google Maps-API ist, dass eine Fusion Table auch Spalten enthalten kann, die das zugehörige Objekt georeferenzieren. Das können ein oder mehrere Spalten sein, die z.B. Longitude- und Latitude-Koordinaten des Objektes oder seine Adresse enthalten, oder aber Felder, die dem Objekt vollständige Shapes (also eine oder mehrere Flächen ausgedrückt als KML) zuordnen. Die Semantik der Georeferenzierung kann man dabei über die Konfiguration einer Fusion Table zuordnen. Sind die Objekte einer Fusion Table georeferenziert, so können diese automatisch als „Fusion Table Layer“ einer Google Maps-Karte hinzugefügt werden. Eine Filterabfrage bestimmt dabei, welche Objekte auf der Karte tatsächlich angezeigt werden sollen. Durch Stilangaben kann man weiter festlegen, wie bestimmte Klassen von Objekten auf der Karte visuell dargestellt werden sollen.

Klickt man auf eines der Objekte im Fusion Table Layer, können die Sachdaten dieses Objektes in einem Overlay (Balloon-Popup) angezeigt werden. Mit dieser Methodik können daher Inhalte von Fusion Tables auf einfache Art und Weise räumlich über Karten repräsentiert werden. Abb. 3.23 zeigt als Beispiel für eine Fusion Table eine Google Maps-Karte, die Messstationen für Flusspegel-Messungen als Layer einer Fusion Table anzeigt. Klickt man eine einzelne Messstation an, erhält man ein Overlay, das u.a. den aktuellen Messwert und den Namen der Station anzeigt.

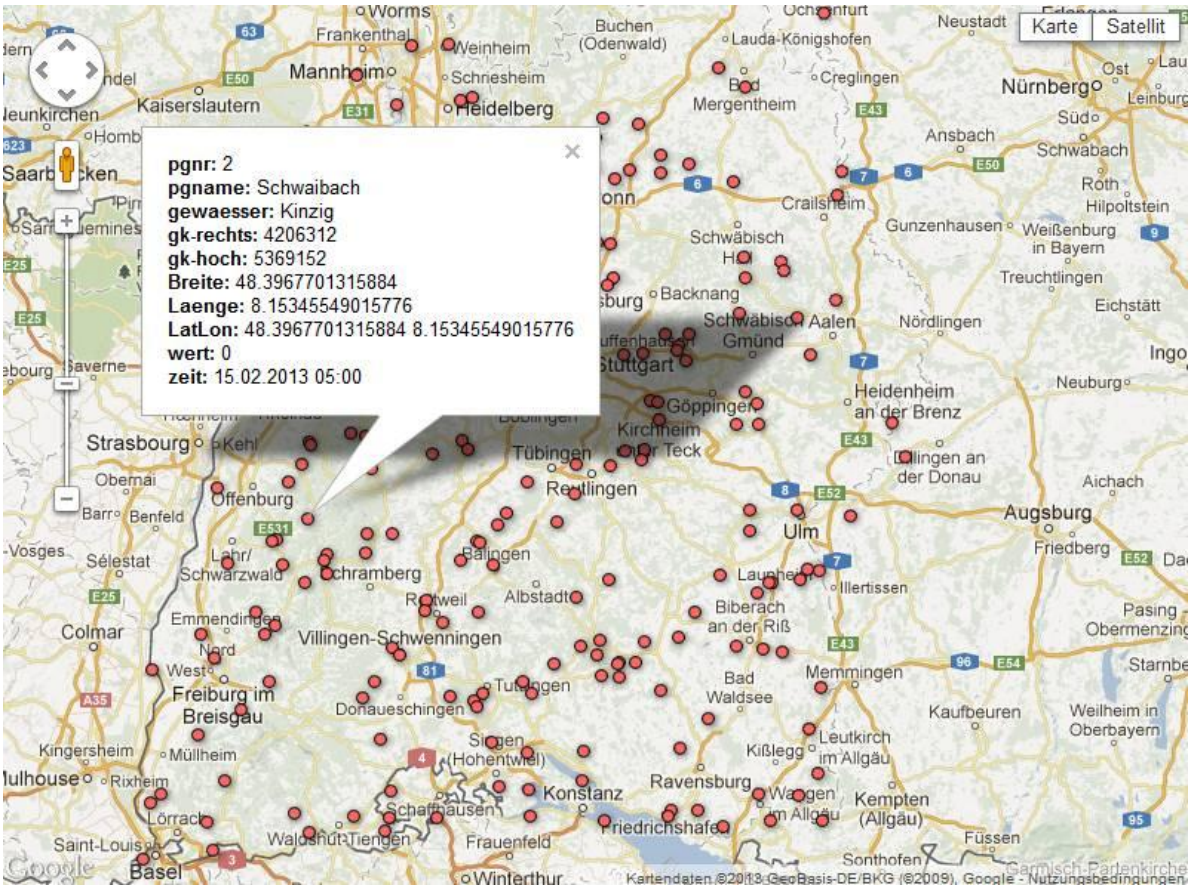


Abb. 3.23: Aus der Fusion Table mit Pegelwerten erstellte interaktive Google Maps-Karte. Die einzelnen Punkte sind die anklickbaren Messstellen. Daten zur Messstelle und Messwert erscheinen im dynamischen Popup.

3.6 Mobiler Zugang und Services

Mit dem Aufkommen neuerer Gerätetechnologien wie Smartphones und Tablet-PCs zeigen Statistiken der Webnutzung den Trend, dass immer weniger Zugriffe auf Websysteme über Desktopbrowser erfolgen. Die Nutzung des Internets verlagert sich vielmehr auf die Nutzung von Mobilgeräten, aber auch auf größere Systeme wie z.B. Internet-basierte TV-Geräte.

Während all diese neuen Gerätetypen auch mit Webbrowsern ausgestattet sind, und damit Informationen von Websystemen als Webseiten abgerufen werden können, erfolgt der Informationszugriff jedoch häufig nicht mit dem Webbrowser, sondern mit dedizierten Anwendungen (Apps), die die Informationen serverseitiger Informationssysteme über serviceorientierte Schnittstellen abrufen.

Abb. 3.24: Responsives Web-Design: Webseite des „The Boston Globe“ auf Notebook, IPad und Smartphone

Dies führt dazu, dass webbasierte Informationssysteme nicht nur bereit sein müssen, Webseiten mit flexiblem Layout an Webbrowser auf eine Vielzahl von Geräten mit ganz unterschiedlichen Bildschirmgrößen zu liefern (responsives Web-Design [34] [35], siehe Abb. 3.24), sondern dass sie zusätzlich die Möglichkeit bieten müssen, dieselben Informationen über serviceorientierte Schnittstellen direkt an Anwendungen (z.B. Apps) so flexibel auszuliefern, dass Anwendungsentwickler diese Serviceschnittstellen dazu nutzen können, diese Daten für den Benutzer möglichst ergonomisch bereitzustellen.

Moderne webbasierte Informationssysteme stellen daher Informationen nicht mehr primär über Webseiten, sondern zunächst über Services maschinenlesbar bereit. Diese Serviceschnittstellen werden dann sowohl von Webanwendungen zur dynamischen Erzeugung von zugehörigen Webseiten genutzt, als auch als Datenlieferant für mobile Anwendungen.

Eine Serviceorientierung der eigenen Informationsdienste ist hierbei von zentraler Bedeutung.

3.7 Zusammenfassung

Das klassische webbasierte Internet wandelt sich immer mehr hin zu einer weltweiten Infrastruktur von miteinander interagierenden Diensten, die über die Prinzipien REST-basierter Services und des Semantic Web miteinander zu neuen Anwendungen verknüpft werden können. Die Dienste bilden ähnlich zu den Webseiten ein durch Hyperlinks verknüpfbares Netz von Informationsressourcen und Softwaredienstleistungen, das von anderen Softwareanwendungen in immer neuen Kontexten verknüpft und hierbei zu neuen Anwendungen kombiniert werden kann.

Große Anbieter wie Google stellen dabei ihren Nutzern eine immer größere Anzahl von hochintegrierten Diensten bereit, die in ihrer Gesamtheit bereits jetzt vollständige Bürokommunikationslösungen für Anwender ersetzen können. So gibt es bereits jetzt ganze Schulen, Universitäten und auch kleine und mittelständische Firmen, die die Google-Business-Dienste als integrierte Bürokommunikationsplattform nutzen.

Ein Nutzer nutzt die Informationen und Dienstleistungen eines solchen Servicenetzes entweder durch dessen integrierte Webanwendungen, die ihm die benötigten Informationen und Dienstleistungen über Webseiten zugänglich macht, oder kann auch über mobile Anwendungen oder sogar Desktopanwendungen auf die bereitgestellten Funktionalitäten zugreifen. Auf diese Weise werden alle angebotenen Dienste und Informationen an allen Orten verfügbar, wo ein Internetzugang besteht, weitgehend unabhängig vom vorhandenen Gerät. Dabei werden insbesondere von den mobilen Apps und Desktopanwendungen Serviceschnittstellen benutzt, um mit der Dienstleistungsinfrastruktur des Dienstleisters zu interagieren.

Auf die gleiche Art und Weise lassen sich serviceorientierte Dienstleistungsangebote von Firmen oder Behörden aufbauen, um den Mitarbeitern und Nutzern Dienste universell bereitzustellen.

Content-Management-Systeme stellen in dieser Welt primär Öffentlichkeits-nahe Informationen als Webseiten für einen Browser-basierten Zugang bereit (dienen also nur der Verbreitung von Informationen wie Pressemitteilungen, Blogposts, Social Media), während Enterprise-Portale oder webbasierte Fachanwendungen das Arbeiten im Webbrowser mit komplexen Fachanwendungen ermöglicht, die in ihrer Ergonomie mittlerweile Desktopanwendungen kaum noch nachstehen. Portale dienen dabei als Integrationsplattformen, die nicht nur dem Webanwender mit Browser, sondern auch Nutzern von mobilen Anwendungen einen integrierten Zugriff auf Informationen und das Arbeiten mit serverbasierten Diensten und Anwendungen erlaubt. Die Zielgruppe solcher Portale oder Fachanwendungen ist dabei häufig nicht der allgemeine Bürger, sondern der Fachanwender, seien es der eigene Mitarbeiter oder externe Fachleute.

Moderne Webtechnologien machen es dabei heutzutage möglich, Öffentlichkeits-nahe Informationssysteme und auch webbasierte Fachanwendungen und Portale mit hochgradig interaktiven und ergonomischen Nutzeroberflächen auszustatten, die bei der Benutzung einen ähnlichen Komfort wie Desktopanwendungen aufweisen. Daher lässt sich in der Praxis beobachten, dass kaum noch Softwareentwickler für Desktopanwendungen gesucht werden, sondern die Investitionen in der Softwareindustrie fast nur noch in die Entwicklung von webbasierten und mobilen Anwendungen mit webbasierten Services als Grundinfrastruktur ge-

hen. In näherer Zukunft ist zu erwarten, dass dieser Trend noch zunimmt und auch Softwaregiganten wie Microsoft dazu übergehen werden, ihre Officeanwendungen nur noch als Webanwendungen in der Cloud anzubieten.

Webbasierte serviceorientierte Infrastrukturen lassen sich dabei sowohl im Internet als auch im Intranet nutzen und sogar mit externen Dienstleistungsinfrastrukturen kombinieren, um eigene Funktionalität durch externe Funktionalitäten in geeigneter Weise zu ergänzen. Auf lange Sicht werden daher viele Organisationen dazu übergehen, einen großen Teil ihrer Softwareinfrastruktur komplett über Services und Web abzubilden.

4 Anforderungen an die zukünftige Web-UIS Infrastruktur

In diesem Kapitel werden die Anforderungen an eine künftige Web-UIS Infrastruktur beschrieben. Selbstverständlich müssen bestehende Webangebote, Webanwendungen und Funktionalitäten für die Autoren erhalten bleiben oder in verbesserter Form zur Verfügung stehen.

Eine wesentliche grundsätzliche Randbedingung an das neue Konzept ist, dass die bestehende Web-UIS Infrastruktur evolutionär in inkrementellen Schritten zur neuen Infrastruktur ausbaubar sein muss, d.h., in einer Übergangsphase müssen alte Systeme und Inhalte noch genutzt werden können und inkrementell verbessert bzw. auf neue Systeme migriert werden können. Dabei sollten die technischen Systeme natürlich so gut es geht auf den in Kapitel 3 beschriebenen aktuellen Stand der Technik gebracht, und in Zukunft verstärkt auf HTML5 als HTML-Version für die Webseiten gesetzt werden.

Wie bereits in der Einleitung beschrieben, lassen sich die Anforderungen in drei grobe Klassen unterteilen: Anforderungen für typische CMS-Anwendungssituationen, Anforderungen in Bezug auf personalisierbare Portale und Web-basierte Arbeitsumgebungen für Fachanwender und Anforderungen an eine Architektur und technische Infrastruktur für die Implementierung von Web-basierten Fachanwendungen.

Anforderungen der ersten Kategorie oder zweiten Kategorie übertragen sich dabei durchaus auch auf die anderen Kategorien, so dass die folgende Auflistung nicht ausschließlich zu sehen ist.

4.1 Content-Management

In diesem Kapitel werden Anforderungen beschrieben, die sich für alle CMS-Anwendungen, aber auch für Portale und teilweise auch für Web-basierte Fachanwendungen als Grundanforderungen ergeben.

4.1.1 Benutzer- und Rechteverwaltung

Die Benutzer- und Rechteverwaltung stellt einen wichtigen Bestandteil eines CMS, Portals oder auch einer Fachanwendung dar. Es müssen konkrete Berechtigungshierarchien für Benutzer definiert werden können, so dass verschiedenen Nutzern oder Nutzergruppen unterschiedliche Rechte auf verschiedenen Inhaltsbereichen zugeordnet werden können. Um dabei diese Rechte nicht jedem einzelnen Inhaltsobjekt gesondert zuweisen zu müssen, sollte es Möglichkeiten geben, Rechte z.B. hierarchisch über Inhaltsbereiche zu vererben.

Ein im Web-UIS 3.0 als CMS oder Portal genutztes technisches System sollte daher die Möglichkeit bieten, Nutzer und Nutzergruppen zu definieren, denen unterschiedliche Zugriffsmöglichkeiten („Rollen“) auf das System für unterschiedliche Inhaltsbereiche und Inhaltsobjekte zugeordnet werden können.

Beispiele für mögliche Rollendefinitionen:

- „Super-Administrator“
  - (alle Zugriffsrechte in allen Bereichen des Systems)
- „Light-Administrator“
  - (alle Zugriffsrechte für einen bestimmten Themenbereich des Systems)
- „Autor“ / „Heavy-User“
  - (kann Inhalte für einen bestimmten Bereich erstellen sowie verändern)
- „Light-User“
  - (Lesezugriff auf bestimmte vordefinierte Themenbereiche)
- „Any-User“ (Lesezugriff auf allgemeinzugängliche Bereiche des Systems)

Beispiele für mögliche Rollen- bzw. Gruppenzuordnungen:

- Max Mustermann erhält als Mitarbeiter der IT-Abteilung automatisch alle Zugriffsrechte; sein Benutzeraccount wird der Gruppe der „Super-Administratoren“ zugeordnet.
- Moritz Mustermann ist neuer Mitarbeiter im Bereich „Luftmesswerte“. Da er den Seitenbereich seiner Abteilung als Autor verwalten soll, wird ihm die Rolle eines „Autor für den Bereich Luftmesswerte“ durch Einfügen in die entsprechende Gruppe zugeordnet.
- Manfred Mustermann wird Stellvertreter von Moritz Mustermann. Sein Benutzeraccount wird daher der Gruppe „Autor für den Bereich Luftmesswerte“ zugeordnet und hat nun entsprechende Zugriffsrechte.

4.1.2 Trennung von Life-Inhalten und Inhalten in Bearbeitung, Versionierung von Inhalten

Inhalte, die für Endnutzer bereits veröffentlicht bereitgestellt worden sind, sollten in zukünftigen CMS-Anwendungen oder Portalen des Web-UIS sauber von Inhalten, die noch in Bearbeitung sind, getrennt werden. Damit soll ein möglicher Datenfluss von nicht veröffentlichten oder nicht vollständig freigegebenen Seiten bzw. Inhalten ausgeschlossen werden. Hierbei muss es auch möglich sein, dass der Inhalt einer Webseite überarbeitet wird, während die noch aktuelle freigeschaltete Version mit älterem Inhalt noch online verfügbar bleibt. Bei solchen Änderungen sollte nach Möglichkeit auch die URL der Seite gleich bleiben.

In diesem Zusammenhang ist auch die Versionierung von Inhalten eine sinnvolle Ergänzung. Gibt es eine Versionierung, dann kann in der Regel eine bestimmte Version eines Inhaltes als Life-Inhalte online veröffentlicht sein, während an anderen Versionen noch gearbeitet wird oder diese nur noch als archivierte ältere Versionen dienen. Das erlaubt es auch, Änderungen an einem bestimmten Artikel in der Historie bis zu einem gewissen Grad nachzuverfolgen. Ferner können ältere Versionen von Inhalten – zumindest in definierten zeitlichen Grenzen – als Backup aktueller Inhalte dienen.

Für die technische Weiterentwicklung und zum Testen größerer organisatorischer Änderungen einer Website sollte neben der Produktions-Website noch eine Entwickler-Website im

Intranet verfügbar sein. Letztere kann auch als Testumgebung für neu entwickelte Softwarekomponenten für die Website dienen. Die eigentliche Entwicklung von Software erfolgt dagegen auf speziellen Entwicklungssystemen der Entwickler selbst.

4.1.3 WYSIWYG-Editor

In CMS-Anwendungen oder Portalen stellt der ggf. enthaltene Inhaltseditor mit der damit verknüpften Veröffentlichungslogik eine zentrale Funktion hinsichtlich der Erstellung von Webinhalten dar.

Von der Leistungsfähigkeit eines solchen Editors und seinen Funktionalitäten hängt entscheidend ab, wie leicht mit Internettechnologie nicht vertrauten Nutzern die Arbeit als Autor fällt. Die Mehrzahl der Web-Autoren im Web-UIS Umfeld wird i.d.R. nicht über eingehende Programmierkenntnisse (z.B. HTML) verfügen und entsprechende Schulungen einen nicht unbeträchtlichen zeitlichen und finanziellen Aufwand darstellen.

Daher ist die Bereitstellung eines WYSIWYG-Editors durch ein einzusetzendes CMS oder Portal zwingend erforderlich.

Als Beispiel sei das bislang eingesetzte WebGenesis System erwähnt. Dieses bietet in der aktuell laufenden Version bereits einen solchen Web-Editor, der umfangreiche Funktionalitäten zur Generierung von Inhalten ohne Programmierkenntnisse bietet (siehe Abb. 4.1). Die Auswahlmöglichkeiten im Editor wie Fonts, Schriftgrößen, Icons für Links, spezielle Paragraphtypen für Überschriften, etc. wurden speziell auf das Landeslayout Baden-Württemberg angepasst. Das sollte auch für die Online-Editoren im zukünftigen Web-UIS möglich sein.

Der Online-Web-Editor sollte auch Möglichkeiten zum Einfügen von Bildern, zur Generierung von HTML-Links und Verweisen auf andere Seiten der Website beinhalten.

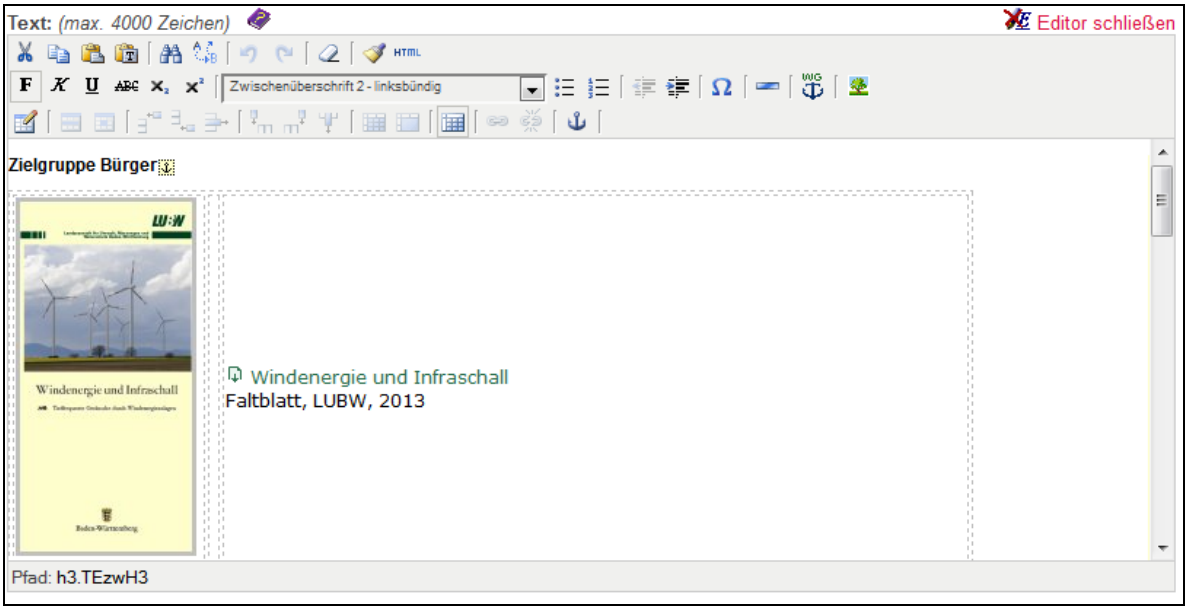


Abb. 4.1: Auf tinyMCE [27] basierender WYSIWYG-Online-Editor für HTML-Inhalte in einem WebGenesis-System

4.1.4 Metadaten (Metatags)

Zur Optimierung der Auffindbarkeit von Informationen und Webseiten ist die Zuordnung geeigneter Metadaten zu Seiten oder Inhaltselementen von Bedeutung. Dabei macht es auch Sinn, einige Informationen, wie „Titel“ des Inhaltselementes, Autorangabe, Veröffentlichungsdatum, oder evtl. eine Kurzbeschreibung dem eigentlichen Inhalt getrennt zuzuordnen und das Inhaltselement oder die Seite durch freie Schlagworte bzw. Tagging als „Meta-Keywords“ oder durch eine Kategorisierung weiter zu klassifizieren.

Das technische System sollte dann diese Metadaten auf entsprechende HTML-Konstrukte in der generierten Webseite abbilden. Bei Verwendung von HTML5 als HTML-Standard wäre es insbesondere sinnvoll, wenn das technische System Inhalte durch Verwendung solcher Metainformationen dann auch durch Microdata-Formate im Sinne von „Schema.org“ semantisch annotieren könnte (siehe hierzu auch Kapitel 3.1.1).

Um die Verschlagwortung zu erleichtern, sollte das System, um dies zu unterstützen, dem Autor bereits einen Katalog vorhandener Schlagworte (z.B. Umweltthesaurus) anbieten.

Jedes CMS bzw. Portal, das im Rahmen des Web-UIS eingesetzt wird, sollte solche Möglichkeiten zur Metadatenbeschreibung von Inhalten besitzen.

4.1.5 Formular-gestützte Erstellung von Inhalten

Die Ergänzung der reinen Eingabe von Webinhalten über einen WYSIWYG-Web-Editor durch weitere Metadatenfelder-Eingabefelder ist bereits eine spezielle Form der Formular-gestützten Erstellung von Inhalten.

Bei dieser wird der eigentliche Webinhalt noch stärker in verschiedene Felder strukturiert und die Eingabe des Inhaltes erfolgt in strukturierter Form dann über ein entsprechendes Formular. Solche Formulare können von Administratoren des CMS oder Portalspezielle mit einem hierfür vorgesehenen Formulareditor erstellt oder über eine Konfigurationssprache definiert und bestimmten Inhaltkategorien zugeordnet werden. Möchte ein Autor einen Inhalt einer solchen Kategorie erstellen, wird ihm dann das entsprechende Inhaltsformular zum Erstellen angeboten.

Über ein zugehöriges Layout-Template wird dann daraus die eigentliche HTML-Repräsentation generiert. Die Felder des Formulars sollten dabei auch gemäß der vorgegebenen Struktur zerlegt in der Datenbank gespeichert werden.

Dieser Mechanismus eignet sich gut für die Erstellung von Inhalten, bei denen Wert auf eine feste Strukturierung von Information gelegt wird und ist durchaus ein nettes Feature zur Vereinfachung der Eingabe mancher Informationen in ein CMS oder Portal. Eine Anwendungssituation wäre z.B. eine Veranstaltungsankündigung

Man sollte so ein Formular aber nicht mit dem Formular einer speziellen Softwarekomponente, wie z.B. einem vollständigen Kalendersystem verwechseln. Die Formular-gestützte Erstellung von Inhalten eignet sich nicht als vollständiger Ersatz für solche in sich geschlossenen Datenbankanwendungen.

4.1.6 Interaktive Eingabeformulare für Endnutzer

Der Einsatz von Formularen ist aber nicht nur sinnvoll zur strukturierten Erfassung von Inhalten der Autoren eines CMS oder Portals, sondern auch zur Bereitstellung von Eingabeformularen für den Endnutzer des CMS bzw. Portals. Typische Anwendungen sind hier z.B. Formulare wie ein Kontaktformular, das Nutzer des Systems ausfüllen, damit z.B. eine Kontaktanfrage in strukturierter Form als Email an eine Kontaktperson weitergeleitet werden kann. Im Umfeld von Portalen und Arbeitsumgebungen werden solche Formulare natürlich auch häufig zur Erfassung komplexerer Daten genutzt, die dann nicht als Email verschickt, sondern direkt in eine Datenbank eingetragen werden.

Die Eingabemasken solcher Formulare bestehen häufig aus einer unbestimmten Anzahl unterschiedlicher Eingabefelder (Textfelder, Auswahllisten, Checkboxen, Buttons …). Wird das Formular per Knopfdruck abgeschickt, wird dann vom System entweder eine E-Mail generiert und an vorkonfigurierte Empfänger weitergeleitet, oder aber die erfassten Daten werden gleich strukturiert in eine Datenbank geschrieben.

Die in Zukunft eingesetzten CMS, Portale und Entwicklungsumgebungen für Web-basierte Fachanwendungen sollten nach Möglichkeit eine einfache Erstellung solcher Nutzer-orientierten Formulare durch Administratoren ohne größeren Programmieraufwand unterstützen. Idealerweise sollte ein Formular-Konfigurator enthalten sein, über den sich einzelne Eingabefelder bzw. bei Bedarf auch ganze Eingabemasken erstellen lassen. Ferner sollten damit verbundene tiefergehende Festlegungen vorgegeben werden können – z.B. wohin eingegebene Daten gespeichert werden sollen (in welche Datenbank) und in welchem Format das geschehen soll (z.B. XML, SQL usw.).

Der folgende Screenshot (Abb. 4.2) zeigt am Beispiel des „WebForm“-Konfigurators von Liferay, wie eine Definitionsoberfläche für Formulare aussehen könnte.

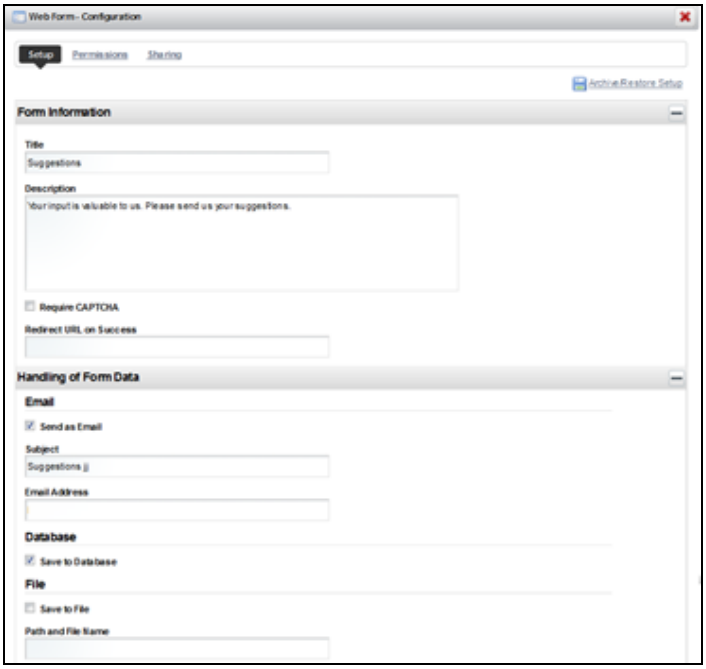


Abb. 4.2: Screenshot des „WebForm“-Konfigurators der Portalsoftware Liferay als Beispiel für die Online-Erstellung von Nutzerformularen

4.1.7 Unterstützung für redaktionelle Workflows

Redaktionelle Workflows [80] dienen in einem CMS zur Steuerung des Veröffentlichungsprozesses von Webseiten oder anderen Inhalten. Dies ist in Behörden eine durchaus wünschenswerte Funktion für ein CMS, da bestimmte Inhalte von Autoren, wie offizielle Pressemitteilungen, erst dann veröffentlicht werden dürfen, wenn sie z.B. durch Mitarbeiter der Abteilung Öffentlichkeitsarbeit genehmigt wurden. Im einfachsten Fall reicht ein redaktioneller Workflow, bei dem Autoren Artikel für die Webseite schreiben dürfen, die von anderen Personen (häufig Editor oder Publisher) genannt gegengelesen und dann freigegeben werden.

Durch die Vergabe von Rollen können die unterschiedlichen Aktionen des Workflows, z.B. „Schreiben“ und „Freigeben“ auf bestimmte Personenkreise eingegrenzt werden.

Manche Content-Management-Systeme bieten die Möglichkeit, beliebige betriebliche Abläufe bezogen auf die Entscheidungs- und Genehmigungsprozesse in einen redaktionellen Workflow abzubilden. Beispielsweise wäre es möglich, Workflows zu generieren, in denen festgelegt ist, dass z.B. die erstellten Inhalte bestimmter Nutzergruppen automatisch zunächst deren jeweiligen Vorgesetzten vorgelegt werden. Nach Freigabe der betreffenden Inhalte werden diese dann an die für die Veröffentlichung zuständigen Stellen weitergegeben. Bei Verweigerung der Freigabe müssten die jeweiligen Autoren der betreffenden Inhalte diese nochmal bearbeiten.

Zusammenfassend lässt sich sagen, dass hierüber die unternehmensinternen Hierarchien viel besser abgebildet sowie wie damit verbundenen Freigabeprozesse für CMS-Inhalte einfacher verwaltet werden können.

Für die Content-Management- und Portalsysteme im zukünftigen Web-UIS wäre zumindest die Unterstützung einfacher redaktioneller Workflows eine sinnvolle Ergänzung. Komfortabel wäre es dabei natürlich, wenn die Workflows dabei flexibel frei definiert werden könnten.

4.1.8 Sprechende URLs

In vielen Content-Management-Systemen werden einzelne Seiten zunächst über Seiten-ID´s referenziert. Dabei haben diese URLs häufig eine flache, nicht-hierarchische Struktur, so dass sich aus der URL nicht herauslesen, welcher Inhalt sich hinter der URL verbirgt bzw. zu welchem Inhaltsbereich der Inhalt gehört. So kann z.B. eine neu angelegte Seite im Bereich „Klimaschutz“ die ID 4711 haben und aus der URL ist weder erkenntlich, um welche Seite es sich genau handelt, noch, dass die Seite zum Inhaltsbereich „Klimaschutz“ gehört.

Sprechende URLs sind URLs, bei denen IDs durch „sprechende“ URL-Kürzel ersetzt und damit die gesamte URL für Menschen und auch Software, etwa Suchmaschinen, besser lesbar wird [81]. Häufig bildet die sprechende URL dabei auch die Hierarchie der Inhaltsbereiche ab.

Die URL

http://www.lubw.baden-wuerttemberg.de/erneuerbare-energien/

könnte als sprechende URL z.B. für den Inhaltsbereich „Erneuerbare Energien“ stehen, während

http://www.lubw.....de/erneuerbare-energien/wasser/

dann den Unterbereich des Inhaltsbereiches „Erneuerbare Energien“ adressiert, der die erneuerbaren Energien, die auf Wasser basieren, beinhaltet. Schließlich wäre dann

http://www.lubw.....de/erneuerbare-energien/wasser/wasserkraftwerk.html

dann evtl. eine zu diesem Bereich gehörige Webseite.

Sprechende URLs sind auch für Webstatistik sehr interessant, da man hier Zugriffe auf verschiedene Inhaltsbereiche unterscheiden können möchte. Das geht umso besser, je besser sich die Inhaltshierarchie in der zugehörigen hierarchischen Struktur einer sprechenden URL widerspiegelt.

4.1.9 Bilder, Bildergalerie, Videos und andere Medien

Jede Website muss auch die Nutzung von Bildern oder auch anderer Mediadateien, wie Videos und Audiodateien, in Webseiten ermöglichen. Hierzu benötigt ein CMS eine Verwaltung der zugehörigen Bilder und Mediendateien. Bilder sollten dabei redundanzfrei (bis auf die evtl. Bereitstellung verschiedener Größen) im System gespeichert und auf verschiedenen Webseiten wiederverwendet werden können. Da es potenziell viele Bilder und Mediendateien im System geben kann, sollte ein effizientes Blättern und Suchen nach Bildern im System, z.B. unter Nutzung von Thumbnails und mit der Einbeziehung von Vorschaubildern möglich sein, wie es z.B. Abb. 4.2 angedeutet wird.

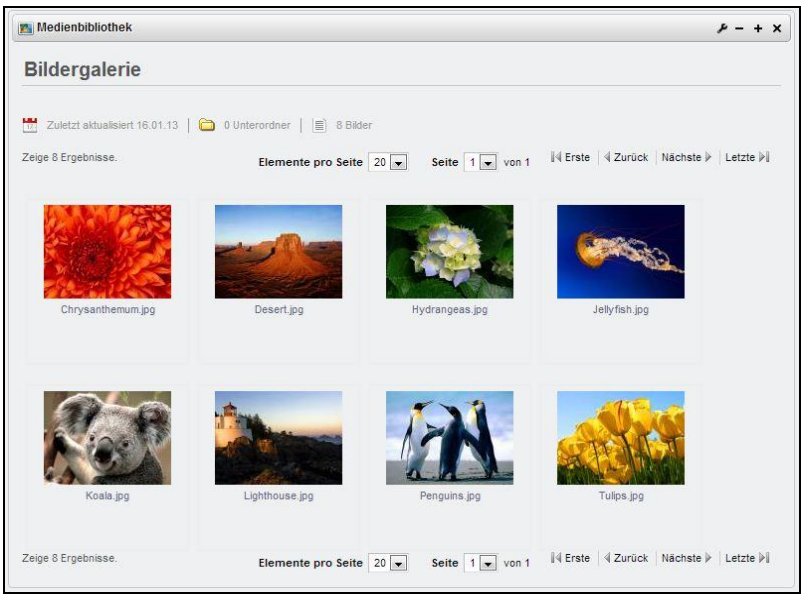


Abb. 4.2: Fenster zum Durchblättern der Medienbibliothek eines CMS

Eine weitere Funktionalität in Bezug auf Bilder, die bei einem CMS benötigt wird, betrifft die Darstellung mehrerer (evtl. sogar vieler Bilder, z.B. eines Fotoalbums) in einer Webseite. Hier wünscht man sich, dass die Bilder dabei in Form einer Bildergalerie zunächst ebenfalls als Thumbnail und bei Klicken auf ein Bild z.B. als Slideshow in Groß gezeigt werden. Dabei verwenden moderne Slideshow-Komponenten heutzutage ein HTML-Konzept namens „Lightbox“ [82], um die im Hintergrund liegende Webseite abzudunkeln und so den Benutzer auf den darüber dynamisch als Overlay dargestellten Slideshow-Viewer zu fokussieren, wie dies in Abbildung 4.3 dargestellt wird.

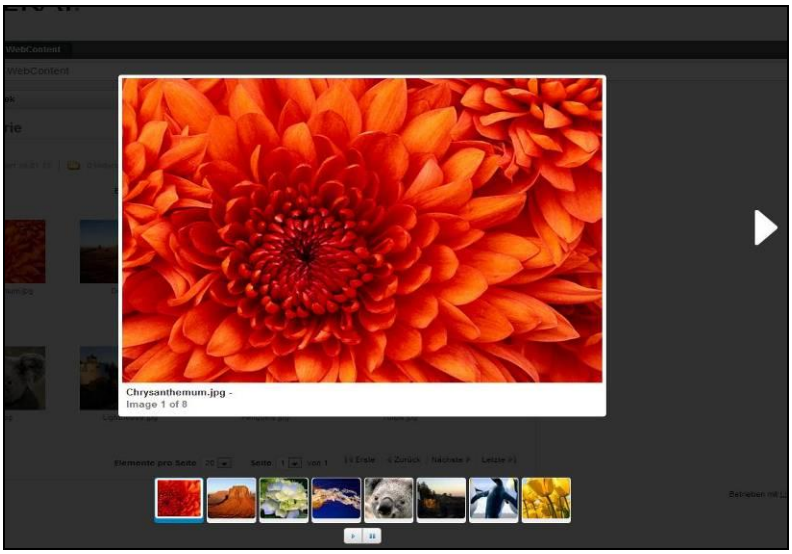


Abb. 4.3: Dynamischer Slideshow-Viewer in einer Lightbox

Die Bilder können dann im Slideshow-Viewer wahlweise manuell oder automatisch durchgeblättert werden. Technisch verbirgt sich dahinter auch ein dynamisches Nachladen von Bildern über asynchrone JavaScript-Aufrufe, wodurch nicht die ganze Seite *„refreshed“* werden muss und der Übergang fließender wirkt.

Die Einbindung von Videoplayern zum Abspielen von Videodateien muss ebenfalls möglich sein, derzeit ist das relativ kompliziert, d.h. ein Redakteur kann nur mit Hilfe von Experten Videos integrieren. Wünschenswert wäre eine einfache Integration, die auch Redakteure beherrschen.

4.1.10 Einbindung von Karten

Ein wesentliches Element bei der Bereitstellung von Umweltinformationen über das Internet ist die Darstellung der Umweltdaten mittels Karten. Jeder soll sich ein Bild über seine eigene Umweltsituation machen und Rückschlüsse ziehen können („What’s in my backyard?“). Über ein entsprechendes Umweltportal soll sich z.B. der Bürger über seine Hochwassergefährdung oder Eignung seines Hauses für Photovoltaik schnell und einfach informieren können. Hierbei spielt der Raumbezug eine entscheidende Rolle. Die Kartensichten sollen einfach interpretierbar, schnell im Zugriff sein und auch über mobile Endgeräte aufgerufen werden können. Als Marktführer bei den sog. Online-Karten und webbasierten Routing-Programmen

hat sich in den letzten Jahren Google mit seinen Produkten etabliert (Google Maps / FusionTables, siehe Kapitel 3.5.2).

Im UIS wird für Kartenanwendungen mit Web-Client UDO / Cadenza eingesetzt, das in einer eigenen Umgebung läuft und per Link (parametrisierbare URL's) von den verschiedenen Webangeboten thematisch und funktional gefiltert aufgerufen wird. Karten werden aber auch für einfache Webseiten produziert und auf vielfältige Weise in Webumgebungen eingebunden. Sie werden beispielsweise als statische Bilder, clickable images, animierte Gifs oder in Ausnahmefällen auch Flash-Animationen eingebunden und es werden interaktive Karten per <iframe>-Tag vom Kartenserver der LUBW integriert. Für kartographisch anspruchsvollere Kartendarstellungen und spezialisierte Geofachanwendungen kommen verschiedenste Lösungen des GIS-Herstellers Esri zum Einsatz. Diese Möglichkeiten sollten selbstverständlich weiterhin gegeben sein.

Die Möglichkeiten, speziell mit Google generierte Karten, aber auch Karten aus UDO / Cadenza in Websites des UIS einzubinden, sind zurzeit relativ gering. Zum einen können solche Karten zwar ohne großen Aufwand per <iframe>-Tag eingebunden werden. Allerdings sind diese Karten zunächst eher statischer Natur und lassen sich nicht im einzubindenden Websystem frei konfigurieren. Die Konfiguration muss stattdessen extern vorgenommen werden (über JavaScript) und auch dies wird vom integrierenden Websystem nur unzureichend unterstützt.

Seit Längerem besteht mit Legato, einem JavaScript-basierten Web-Mapping-Client, eine weitere Möglichkeit, Karten in andere Websysteme einzubinden. Durch eine in einem Web-Genesis-System abgelegte Konfigurationsdatei im xml-Format für eine solche Legato-Karte hält sich der Konfigurationsaufwand zur Integration z.B. in Grenzen. Da der Pflegeaufwand dieser Seiten, bedingt durch Änderungen an Legato aber relativ hoch war bzw. ist, kommt diese Lösung nur an wenigen Stellen im Einsatz.

2012 wurde als weitere Möglichkeit der Einbindung von Karten in die Webumgebung die Nutzung von Google Maps erprobt. Nach dem Kauf einer Lizenz (zunächst für ein Jahr) sowie ersten Tests mit der Lizenzversion waren die rechtlichen Bedenken beim Einsatz der Google-Karten ausgeräumt, wobei die Erprobungsphase noch nicht ganz abgeschlossen ist. Künftige Webstrategien sollten die Einbindung von Google Maps berücksichtigen, da die Erprobung dieser Technologien bereits jetzt sehr zufriedenstellende Ergebnisse liefert.

Ziel sollte dabei sein, für die zukünftigen Websysteme wiederverwendbare Softwaremodule zu schaffen, welche die Einbindung externer, aber auch interner Dienste zur Karten- und in Zukunft auch Diagrammdarstellung von Daten erlauben. So ist z.B. die Einbindung von Kartendarstellungen zur Anzeige eines Orts und entsprechender Routinginformationen über eine Google-Maps-Softwarekomponente, wie sie schematisch in Abbildung 4.4 dargestellt wird, in vielen CMS möglich.

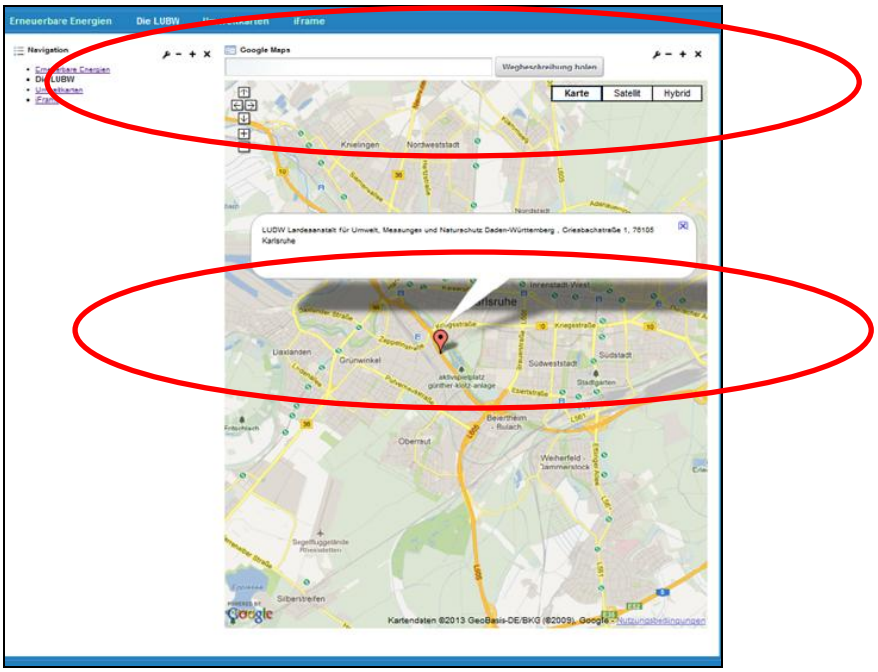


Abb. 4.4: Integration einer Orts- oder Anfahrtsbeschreibung über eine Google Maps Karte

Auch die Integration beliebiger Inhalte über eine <iframe>-Komponente sollte immer möglich sein. Diese erlaubt es dann z.B., durch andere Systeme bereitgestellte Karten einfach in ein Webangebot zu integrieren, wie dies Abbildung 4.5 verdeutlicht.

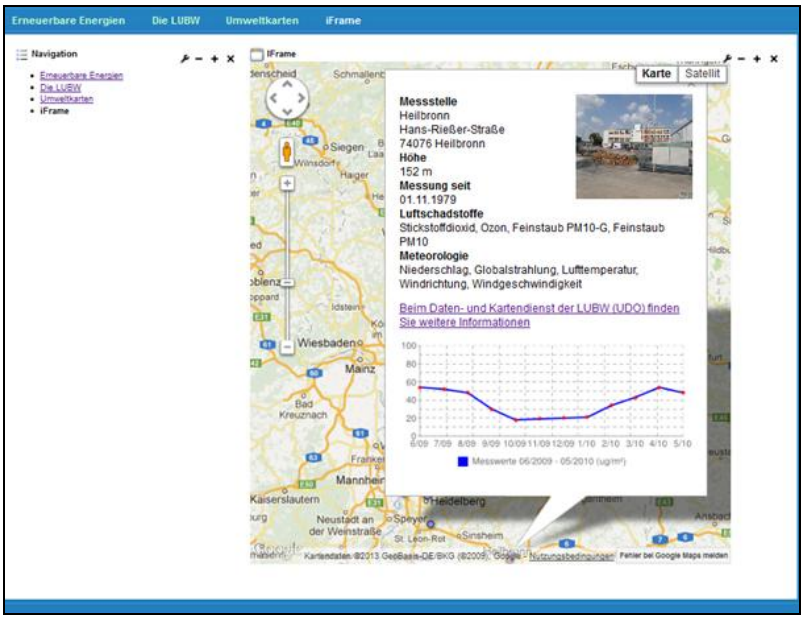


Abb. 4.5: Integration einer Fachkarte, die durch ein anderes Web-basiertes Fachinformationssystem bereitgestellt wird, über die Nutzung einer generischen <iframe>-Lösung

4.1.11 Kalender / Termine

Öffentlichkeits-orientierte CMS oder Portale benötigen typischerweise auch Softwarekomponenten zur Verwaltung von Terminen, wie öffentlichen Veranstaltungen und zugehörigen

Autorenformularen zur Eingabe von Terminen bzw. Anzeigemodulen, die Termine als Listen oder in einer Kalenderansicht anzeigen.

Ein Kalender kann in einem Portal, das als Arbeitsumgebung dient, natürlich noch weitere Aufgaben übernehmen, wie dies in 4.3.2 beschrieben ist.

4.1.12 RSS-Feeds

RSS-Feeds [83] sind ein Kernelement der Web 2.0-Technologie und beschreiben über das Web angebotene Nachrichtenströme zu bestimmten Themen. Nachrichten werden dabei als Artikel veröffentlicht, die i. d. R. einen weiterführenden Link zur eigentlichen Nachricht enthalten.

Ein CMS im Web-UIS sollte sowohl RSS-Feeds aus Webseiten erzeugen, als auch RSS-Feeds von Fremdsystemen darstellen können. Dabei sollte das CMS die Klassifikation von Artikeln innerhalb von Feeds und die Bildung von spezifischen RSS-Nachrichtenkanälen für bestimmte Klassen von Inhalten unterstützen.

Bzgl. des Feed-Formats sollten RSS 1.0 und RSS 2.0 sowie ATOM 1.0 Feeds unterstützt werden.

4.1.13 Integration mit sozialen Netzwerken und Social Media Diensten

Mit der Verbreitung des Web 2.0 und der zunehmenden Sozialisierung des Internets durch Blogs, Broadcasts, soziale Netzwerke und Social Media Dienste entstand auch eine Notwendigkeit, die Unternehmensportale an diese Entwicklung anzupassen. Der immer noch anhaltende Hype macht es für Webseiten und Portale fast schon zu einem Muss, eine Integration mit sozialen Netzwerken und Social Media Diensten anzubieten. Neben Links zu den Seiten des Unternehmens in den Social-Networks sind auch direkte Integrationen, wie Newsstreams und das Teilen von Artikeln auf der eigenen Pinnwand mittlerweile ein essenzieller Bestandteil vieler Webseiten.

Die Anzeige der Newsfeeds aus sozialen Netzen erfolgt dabei typischerweise wieder unter Nutzung der bereits beschriebenen Feed-Technologien.

Im Web-UIS Umfeld wird zurzeit nur ein Twitter-Newsstream in die UM-Homepage eingebaut, der durchgehend die Tweets des zugehörigen Twitterkanals des UM auf der Homepage analog zu einem RSS-Newsfeed darstellt. Dieser basiert auf JavaScript und kann somit in nahezu jeder Webseite implementiert werden. Weitere Netzwerke wie Facebook, Google+ oder LinkedIn werden momentan nicht unterstützt.

Des Weiteren unterhält das UM im YouTube Social Media Dienst einen Videokanal, dessen Videos sich über den JavaScript-basierten JWPlayer in Webseiten einbinden lassen. Der Themenpark Umwelt verfügt ebenfalls über eine solche Integration von YouTube-Videos und anderen Life-Webkamera-Videos basierend auf JavaScript.

Wünschenswert für eine zukünftige Web-UIS Infrastruktur wären auf Dauer gute Integrationsmöglichkeiten sowohl von Social Media Dateien, wie Videos aus YouTube oder Bildern aus FlickR, als auch Module zur Integration mit diversen sozialen Netzwerken (Twitter, Facebook, Google+, Tumblr.), sofern das Land offiziell auf diesen Netzwerken präsent sein will.

Ebenso wünschenswert wären auch Funktionen für mehr Bürgerbeteiligung und Interaktion mit Nutzern, wie Umfragemodule, Gästebücher, Diskussionsforen und Unterstützung bei der Erstellung von Wikis und Blogs (siehe auch die folgenden Unterkapitel).

4.1.14 Abstimmungen

Ein anderes Element des interaktiven Web 2.0 sind einfache Abstimmungen (Polls). Hierbei lässt man die Nutzer einer Website zur Meinungsfindung über ein bestimmtes Thema durch Auswahl verschiedener Alternativen abstimmen.

Solche Abstimmungen können natürlich auch für Portale und Arbeitsumgebungen genutzt werden.

4.1.15 Umfragen

(Online-)Umfragen sind ebenso ein „kollaboratives“ Web 2.0-Element. Im Gegensatz zu den oben beschriebenen Abstimmungen, die nur einfache Antworten auf eine Frage erlauben, können Umfragen durchaus komplex sein und aus mehreren eventuell in Beziehung miteinander stehenden Fragen bestehen. Auch die Antworten können teilweise komplex sein, da sie unterschiedliche Datentypen als Rückgabe haben können (z.B. bei Erfassung der demographischen Daten einer an einer Umfrage teilnehmenden Person wird die Altersangabe als Integer, das Geschlecht als Aufzählungswert, entweder männlich oder weiblich, oder das Geburtsdatum als Datum erfasst).

Eine typische Umfrage wäre eine Meinungsumfrage zum Thema „Wie gefällt euch unsere neue Webseite?“. In einzelnen Fragen kann dann die Meinung zum Design, zur Ergonomie der Nutzerführung, zu vorhandenen Inhaltskategorien, etc. ausdifferenziert werden. Dabei soll sowohl die Zielgruppe vorgegeben werden können (z.B. im Intranet alle Mitarbeiter des Unternehmens) wie auch die Art der Fragestellung (wahlweise Möglichkeit zu offenen Antworten bzw. zur Auswahl zwischen vordefinierten Antworten).

Das Komplizierte bei Umfragen ist die Erstellung des Umfrageformulars, d.h. die Definition der Fragen und der möglichen Antworten. Dies sollte natürlich über eine Weboberfläche mit einem komfortablen Editor erfolgen können. Ferner sollte es bei erstellten Umfragen möglich sein, die erhaltenen Umfrageergebnisse automatisiert auswerten zu können (z.B. „Wie viel Prozent der Umfragebeteiligten haben bzgl. einer bestimmten Frage mit ‚Ja‘ oder mit ‚Nein‘ gestimmt?“).

Damit die Umfragen auch über einen längeren Zeitraum hinweg ausgewertet und somit verglichen werden können („Wie waren die Umfrageergebnisse 2011?“, „Und wie die aus 2012?“) sollten die Ergebnisse idealerweise innerhalb einer Datenbank gespeichert werden (können) und das eingesetzte CMS entsprechende Schnittstellen bieten.

Dies erfordert aber bereits eine funktional reiche und sehr komplexe Software, um Umfragen auch komfortabel und für jeden Anwendungsfall erstellen zu können. Deshalb kann nicht davon ausgegangen werden, dass diese Funktionalität für jedes CMS zur Verfügung steht.

Umfragen lassen sich natürlich auch für Portale und kollaborative Arbeitsumgebungen gut einsetzen.

4.1.16 Barrierefreiheit

„Barrierefrei“ bedeutet in erster Linie, dass Menschen mit Behinderung die Bedienung von Hard- bzw. Software durch unterstützende Technologien ermöglicht werden soll. Dies kann beispielsweise durch den Einsatz von Screenreadern geschehen wie beispielsweise auch durch die Festlegung von Tastenkürzeln, anhand derer sich einzelne Befehle verkürzt und ohne lange Klickpfade ausführen lassen.

Die Barrierefreiheit wird bereits von den jetzigen Systemen im Web-UIS unterstützt. Die Navigation kann beispielsweise alternativ per Tastenkombination ausgeführt werden. Jedes weitere CMS, das in absehbarer Zeit eingesetzt werden wird, muss zwingend umfassende Funktionalitäten (bzw. entsprechende Schnittstellen dazu) bieten, um die Entwickler- sowie die Anwenderseite so barrierefrei wie möglich zu gestalten.

Folgende Empfehlungen sollten in Zukunft für barrierefreie Webseiten beachtet werden:

- Modernes Struktur-beschreibendes HTML / CSS (z.B. HTML5 Strukturelemente) nutzen
- Keine Tabellenlayouts (table), keine Frames
- Überschriften und -Ebenen sowie Tab-Reihenfolgen beachten
- JavaScript nur als Ergänzung, nicht als Notwendigkeit (z.B. bzgl. Navigation)
- Kein Flash für Inhalte, die sich anders darstellen lassen (z.B. Texte)
- Alternativtexte für Nicht-Text-Elemente verwenden (z.B. Bilder, Links)
- Menüs / Navigation eindeutig auszeichnen und für Blinde navigierbar machen
- Logische URL- und Link-Strukturen verwenden
- Web-Design verwenden, das bzgl. Fontgrößen skalierbar ist (für Menschen, die schlecht sehen)
- Kontrastreiches Design verwenden (auch für bessere Lesbarkeit draußen sinnvoll)

4.1.17 Unterstützung für die Optimierung der Webseiten für mobile Geräte

Aus der zunehmenden Verbreitung internetfähiger mobiler Endgeräte und deren Rolle als fast adäquater Ersatz von stationären Rechnern ergibt sich die Notwendigkeit, Internetangebote hinsichtlich ihrer Darstellung für solche mobile Geräte zu optimieren.

Ein CMS sollte in diesem Zusammenhang umfassende Dienste zur Erkennung der Endgeräte, von denen aus eine Seite aufgerufen wird, bieten und möglichst automatisiert eine Anpassung der Webseiten einer Website an die Gegebenheiten einer bestimmten Geräteklasse unterstützen.

Dies ist bei momentanen Webanwendungen im UIS nicht der Fall. Abbildung 4.6 zeigt z.B. die Homepage der LUBW im Browser hochkant auf einem iPhone. Man sieht, dass sich die Inhalte aufgrund der zu geringen Breite übereinander gelegt haben.

Abb. 4.6: Webseite der LUBW auf einem iPhone. Die Inhalte liegen aufgrund der zu geringen Breite übereinander

Wünschenswert wäre die Unterstützung eines Responsive Webdesigns [34] [35], bei dem die Seiten auf den verschiedenen Ausgabemedien unterschiedlich aussehen, weil sie sich anpassen. Das Layout könnte, wenn viel Platz verfügbar ist, mehrspaltig sein und bei wenig verfügbarem Platz - wie auf einem Smartphone - einspaltig.

Es ist nicht geplant, unterschiedliche Webseiten für die verschiedenen Endgeräte anzulegen und je nach Gerät/Betriebssystem zu verzweigen, es sei denn, es lassen sich automatisiert mobil optimierte Webseiten erzeugen.

4.2 Integration wichtiger externer Softwaresysteme

4.2.1 Web-Statistik

Webstatistik-Programme werden zur Messung des Nutzerverhaltens eingesetzt. Solche Tools halten u.a. fest, woher ein Besucher kam, welche Bereiche auf einer Seite aufgesucht wurden und wie oft oder wie lang der Benutzer auf dieser Seite war. Durch dieses Monitoring kann das Benutzerverhalten analysiert und anschließend können die Webseiten auf den Nutzer zugeschnitten und komfortabler gestaltet werden. Des Weiteren bieten diese Methoden den Administratoren einen Überblick über die Benutzer, um die aktuelle Auslastung der

Webserver und Seiten abschätzen zu können. Der Einsatz solcher Tools ist in Deutschland aus Datenschutzgründen seit Jahren umstritten und muss – wenn er erfolgt – dem Besucher explizit mitgeteilt werden.

In WebGenesis gibt es eine eigene integrierte Webstatistik, die aber nur über wenige Funktionen und Ausschlusskriterien verfügt. Stattdessen betreibt das Land Baden-Württemberg eine Webstatistik mit einer zentralen Installation des Produktes „Wiredminds“ [84]. Da es sich um ein pixelbasiertes Tool handelt (die einfacheren, älteren Tools werten nur Logfiles aus), muss in jeder Webseite ein Snippet (Java-Script-Code) eingebaut werden. Diese Form der Nutzung externer Statistiktools sollte auf einfache Art von zukünftigen Webanwendungen im UIS unterstützt werden.

4.2.2 Interne Suche und Einbindung der GSA-Suche

Damit die in den jeweiligen Content-Management-Systemen enthaltenen Informationen für sämtliche Nutzer verfügbar sowie zugänglich sind und auch jederzeit aufgefunden werden können, setzen viele moderne CMS und Portale eine intern implementierte Suchmaschine mit gängigen Funktionen ein.

In jedem Fall muss ein künftig eingesetztes CMS standardmäßig eine umfassendere interne Suche bieten. Zur Integration mit externen Suchmaschinen, wie der im Web-UIS Umfeld bereits benutzten GSA (**G**oogle **S**earch **A**ppliance) [37], sollte das System aber auch die Möglichkeit bieten, die interne Struktur auch für externe Suchmaschinen, z.B. über bereitgestellte Services, als Sitemap bereitzustellen und die Semantik innerhalb der einzelnen Webseiten für Suchmaschinen-Crawler über die Instrumentierung des HTML durch Metadaten und Microdata-Formate optimal zu erschließen.

Portale und CMS im Web-UIS können andererseits die GSA-Suche dazu nutzen, als Frontendsysteme den Anwendern nicht nur Suchergebnisse bezogen auf das eigene System, sondern auch von nachgeschalteten Webfachanwendungen zu liefern. Dabei unterstützt die Integration der GSA in diese Systeme auch Suchfunktionalitäten, wie:

- Boolesche Suche
- Fuzzy Suche
- Proximity Search
- Verstärkungsfaktor
- TagCloud (s. u.)

Daher sollte jede CMS oder Portalplattform auch Möglichkeiten zur Integration externer Services, wie in diesem spezifischen Fall der GSA-Suche, bieten.

Bisher werden GSA-Suchergebnisse aus Fremdsystemen (z.B. Bibliothek, Datenbank des Statistischen Landesamtes) über OneBoxen dargestellt, d. h., die GSA durchsucht die angebundenen Fremdsysteme nicht selbst und direkt nach möglichen Ergebnissen, sondern nutzt deren eigene Suchfunktionalitäten. Die jeweiligen Ergebnisse werden dann über XML ausgelesen und in OneBoxen dargestellt.

Jedes CMS oder Portal, das bei der LUBW eingesetzt werden wird, sollte über Funktionalitäten verfügen, um solche OneBoxes einfach in eigene Webseiten integrieren zu können. Das kann z.B. auch über die Nutzung von <iframe>s geschehen.

4.2.3 Einbindung von Fremdsystemen über <iframe>-Tags

In vielen Fällen kann man Inhalte von Fremdsystemen einfach über <iframe>-Tags in ein CMS oder Portal einbinden und anzeigen.

In Kapitel 4.1.6 wurde bereits auf die Einbindung von Google-Karten eingegangen. Dies ist eine typische Anwendungssituation einer Integration über <iframe>-Tags. Viele der von Google über Dienste angebotenen Funktionalitäten, wie Karten, Kalender, etc. werden von Google auch über Web-Widgets (Schnipsel von JavaScript-Code plus und HTML-Tags als Container für den Inhalt des Widgets, siehe auch Kapitel 3.3) angeboten. Kopiert man diesen Code-Schnipsel in ein <iframe>-Tag, wird der zugehörige Inhalt in diesem <iframe> angezeigt.

Als Weiteres können auf diese Weise auch YouTube-Videos zu Umweltinhalten per <iframe> in bestimmte Seiten eingebunden werden, sofern der rechtliche Aspekt hierbei geklärt ist.

Solche <iframe>-Lösungen lassen sich auf der HTML-Source-Ebene bereits „zu Fuß“ erstellen: Manchmal bietet auch ein WYSIWYG-Online-Web-Editor die Möglichkeit, solche <iframe>-Tags in einen eigenen Inhalt zu integrieren.

Komfortabler geht es, wenn das CMS oder Portal eine spezielle Funktionalität bereitstellt, solche <iframe>-Elemente direkt in Webseiten (z.B. als eigenständige Webinhalte) zu integrieren. Softwarekomponenten, die <iframe>-Integration dediziert unterstützen, erlauben es oftmals noch Authentifizierungsdaten über einen Konfigurationsdialog anzugeben. Der in das <iframe> zu ladende Inhalt wird dann nur noch durch eine URL auf ein vom Fremdsystem bereitgestelltes Web-Widget spezifiziert. In diesem Fall authentifiziert sich das CMS oder Portal zunächst bei dem Fremdsystem mit dem angegebenen Account, bevor es die <iframe>-Daten abruft. Durch diese Lösung kann man dann auch Daten von Fremdsystemen integrieren, wenn diese Daten durch einen Account geschützt sind.

Zukünftige CMS oder Portalsoftware sollten beide Lösungen zur Integration von <iframe>-Inhalten bereitstellen. Hierdurch lassen sich bereits sehr viele Fremdsysteme auf generische Art einbinden.

4.2.4 Integration über Erweiterungsprogrammierung

Eine andere Form der Integration von Fremdsystemen bietet ein CMS, Portal oder Webanwendungsframework durch Unterstützung zur Programmierung von „Erweiterungen“. Dies sollte bei den im Web-UIS verwendeten Systemen auch möglich sein.

Die hierfür benötigten Funktionalitäten werden in diesem Dokument unter Kap. 4.4 („Zukünftige Anforderungen an die Entwicklungsplattformen für Web-Anwendungen“) beschrieben.

Wichtig ist, dass sich einmal geschriebene Erweiterungen dann nahtlos durch Administratoren oder Autoren als Inhalte in das System integrieren lassen.

4.3 Anforderungen an personalisierbare Portale und zukünftige Web-basierte Arbeitsumgebungen

Bei der zukünftigen Entwicklung des Web-UIS soll verstärkt der personalisierte Zugang zu Informationen sowie die Nutzung von Portalen im Intranet und Extranet als kollaborative und soziale Arbeitsumgebungen unterstützt werden.

Die zugehörigen technischen Plattformen müssen hierzu Funktionalitäten zur Personalisierung und zur Integration von Web-basierten Fachanwendungen sowie zur kollaborativen Zusammenarbeit über das Portal bieten.

4.3.1 Personalisierung

Für einen personalisierten Zugang [85] sollte das System nach der Anmeldung eines Nutzers am System diesem spezifische Webseiten anbieten, die auf ihn als Nutzer oder auf seine Rolle(n) im System zugeschnitten sind. Ist er z.B. Editor für einen gewissen Webbereich, dann könnte eine spezielle Webseite ihm eine genaue Übersicht darüber geben, was für Änderungen sich in dem Inhaltsbereich ergeben haben, für den er verantwortlich ist. Im Sinne einer Arbeitsumgebung würde das Portal ihm dann evtl. auch seine aktuell vorliegenden Aufgaben oder z.B. Termine von Veranstaltungen anzeigen, die für ihn relevant sind.

Personalisierung kann aber auch bedeuten, dass dem Nutzer nach der Anmeldung Webseiten als persönliche Homepages zur Verfügung stehen, die er nach seinen persönlichen Interessen mit Inhalten konfigurieren kann. Dies funktioniert allerdings nur in einem System, das die dynamische Platzierung von Inhalten auf Webseiten durch den Endnutzer erlaubt (eine typische Eigenschaft von Portalsoftware).

Um eine wirkliche Arbeitsumgebung für Fachanwender zu bekommen, sollte eine solche web-basierte Arbeitsumgebung auch die Integration von web-basierten Fachanwendungen erlauben, die für Endanwender bereitgestellt werden können. Da sich ein Fachbenutzer bei solchen Fachanwendungen häufig für das Bearbeiten von Fachdaten autorisieren muss, sollten solche Portalserver dann auch Single-Sign-On-Funktionalitäten bieten, bei denen die Autorisierung bei Fremdsystemen über den Account eines Nutzers im Portal erfolgen kann.

Dabei ist es auch wünschenswert, dass man sich als interner Mitarbeiter in Portale über den eigenen Account in der Office Domain der Behörde in ein Portal einwählen kann. Hierzu sollten solche Portale als Arbeitsplattformen für Mitarbeiter Verfahren zur Fremdauthentifizierung, z.B. LDAP-Unterstützung, bieten und das Autorisierungssystem der technischen Plattform muss dann auch die Rechtevergabe an Accounts unterstützen, die durch entfernte Nutzeraccounts beschrieben sind.

4.3.2 Kalender zur Kollaboration

Kalender können in Portalsystemen verschiedene Funktionen einnehmen, z.B. als „Werkzeug“ zum Hinweis auf bestimmte Informationsveranstaltungen. Im Kollaborationsbereich (Intranet) sind Kalender dagegen essenzielle Werkzeuge, um die Zusammenarbeit und die Terminabstimmungen für Arbeitsgruppen besser zu gestalten.

Deshalb sollte ein Kalender in einem Portal, das als personalisierbare und kollaborative Arbeitsplattform dient, sowohl für die Koordination und Bekanntgabe interner Termine (z.B. Personalversammlung) innerhalb des Unternehmensintranets genutzt werden können, wie auch für die Bekanntgabe „externer“ Termine innerhalb des Internetauftritts (z.B. Bekanntgabe von Umweltveranstaltungen oder Umweltmessen).

Ferner sollten die im Portal gespeicherten Termine mit entsprechenden Werkzeugen mit anderen Kalendersystemen, wie z.B. einem Outlook-Kalender innerhalb der Office-Welt der Behörde synchronisierbar sein. Innerhalb des Intranetbereichs wäre beispielsweise in Zukunft auch die Einbindung eines automatisierten Workflows denkbar, der bei die Allgemeinheit betreffenden Terminen (z.B. Personalversammlung) diese automatisch in die Kalendertools der Mitarbeiter schreibt. Wahlweise könnte dies auch als eine Art „Kalender-Abo“ gestaltet werden.

Die Möglichkeiten der „Kollaboration“ stehen hierbei im Vordergrund.

4.3.3 Dokumentenmanagement

Ein Portal als Arbeitsumgebung im Web-UIS sollte auch Zugriff auf ein über das Portal zugreifbares Dokumentenmanagementsystem (DMS) [50] bieten. Solche besitzen in der Regel eine frei definierbare hierarchische Ablage von Dateien und Ordnern, wie man sie von Dateisystemen kennt, ergänzen diese aber häufig durch ein Metadatensystem, mit dem man Dokumente durch zusätzliche Metadaten beschreiben oder inhaltlich kategorisieren und mit Schlüsselworten taggen kann (vgl. Abb. 4.7).

Ein solches DMS sollte direkt in einem Portalserver eingebaut sein, der als Arbeitsumgebung dienen soll. Ein solcher Portalserver sollte aber auch die Verknüpfung mit einem externen DMS zulassen. Letzteres hat den Vorteil, dass ein zentral im Intranet bereitgestelltes DMS-System als DMS von mehreren Portalen oder CMS genutzt werden kann und evtl. sogar von der Office-Welt aus z.B. als Sharepoint-Server zugreifbar ist.

Ein Portalserver, der als Grundlage für eine kollaborative Arbeitsumgebung genutzt werden soll, sollte daher sowohl ein internes DMS als auch die Anbindung an externe DMS unterstützen. Möglichkeiten zur Versionierung der gespeicherten Dokumente wären dabei ebenfalls sehr willkommen.

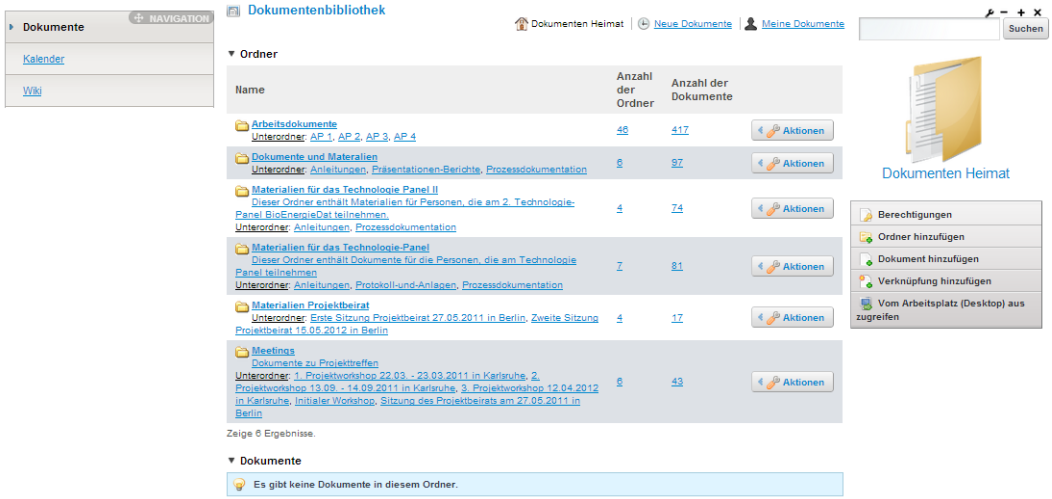


Abb. 4.7: Integrierter Dokumentenmanagementbereich für eine Arbeitsgruppe

4.3.4 Forum / Schwarzes Brett

Diskussionsforen und Schwarze Bretter (Informationsseiten) dienen als virtueller Platz zum Austausch und Bekanntmachen von Informationen, Meinungen oder Erfahrungen. Die Kommunikation erfolgt asynchron, was sich dadurch äußert, dass ein Eintrag nicht sofort, sondern zeitversetzt beantwortet werden kann.

Elektronische Foren und Schwarze Bretter sind ein guter Ersatz für die Pinnwände und Aushängebereiche einer Behörde im Sinne eines papierlosen Büros. Daher sollte ein Portal als Arbeitsumgebung komfortable Möglichkeiten zur Einrichtung solcher Foren / Schwarzer Bretter bieten.

Thematische Threads sind in Foren normaler Weise beliebig durch andere Nutzer kommentierbar. Soll ein Forum als Schwarzes Brett nur als Mitteilungsplattform benötigt werden, sollte es möglich sein, Kommentierfunktionen bei Foren zu deaktivieren.

Idealerweise sollten auch virtuelle Schwarze Bretter sowie ausgewählte Inhaltsseiten (z.B. im Intranet) bei Bedarf mit Kommentarfunktionen oder zumindest mit Verweisen auf ein dazugehörendes Forum versehen werden können.

4.3.5 Wiki

Ein Wiki ist ein Web-basiertes Dokumentationssystem, das eine kollaborative Zusammenarbeit vieler Personen an gemeinschaftlich erstellten Webseiten ermöglicht. Es eignet sich daher hervorragend zur kollaborativen Erstellung aller möglichen Formen von Dokumentationen. Ein Wiki ist daher ein sehr sinnvoller Baustein einer kollaborativen Arbeitsplattform.

Manche Wikis erlauben es dabei, die Informationen auf den einzelnen Webseiten durch semantische Beschreibungen anzureichern (Semantisches Wiki). Solche Wikis werden von Unternehmen auch dazu genutzt, eine zentrale, unternehmensspezifische Wissensdatenbank aufzubauen (analog zu Wikipedia), an der gemeinschaftlich gearbeitet werden kann. Ein solches System kann auch als „Kollektive Intelligenz“ bezeichnet werden, da die Fach-

kenntnisse einzelner Personen mit einander kombiniert werden. Ferner wird zumindest ein Teil des im Unternehmen vorhandenen Wissens - sofern es sich innerhalb eines Wikis dokumentieren lässt und auch tatsächlich dokumentiert wird – innerhalb des Unternehmens gehalten,. d. h., ein Mitarbeiter wird, sofern er (z.B. altersbedingt) das Unternehmen verlässt, nicht sein ganzes Wissen mitnehmen.

Ein Wiki sollte natürlich in Bezug auf Erstellung und Verwaltung von Einträgen einfach zu bedienen sein. Zur Erstellung der Seiten kann man eine Art Wiki-Auszeichnungssprache oder analog zur Erstellung von Seiteninhalten eine Art WYSIWYG-Wiki-Editor benutzen. Wikis besitzen weiter ein sehr gutes Versionsmanagement für Seiten, so dass Änderungen an Seiten nachverfolgt und gegebenenfalls sehr schnell wieder rückgängig gemacht werden können.

Die folgenden Screenshots (Abb. 4.8) zeigen typische Webseiten von Wikis, wie sie z.B. in Portalservern implementiert sind.

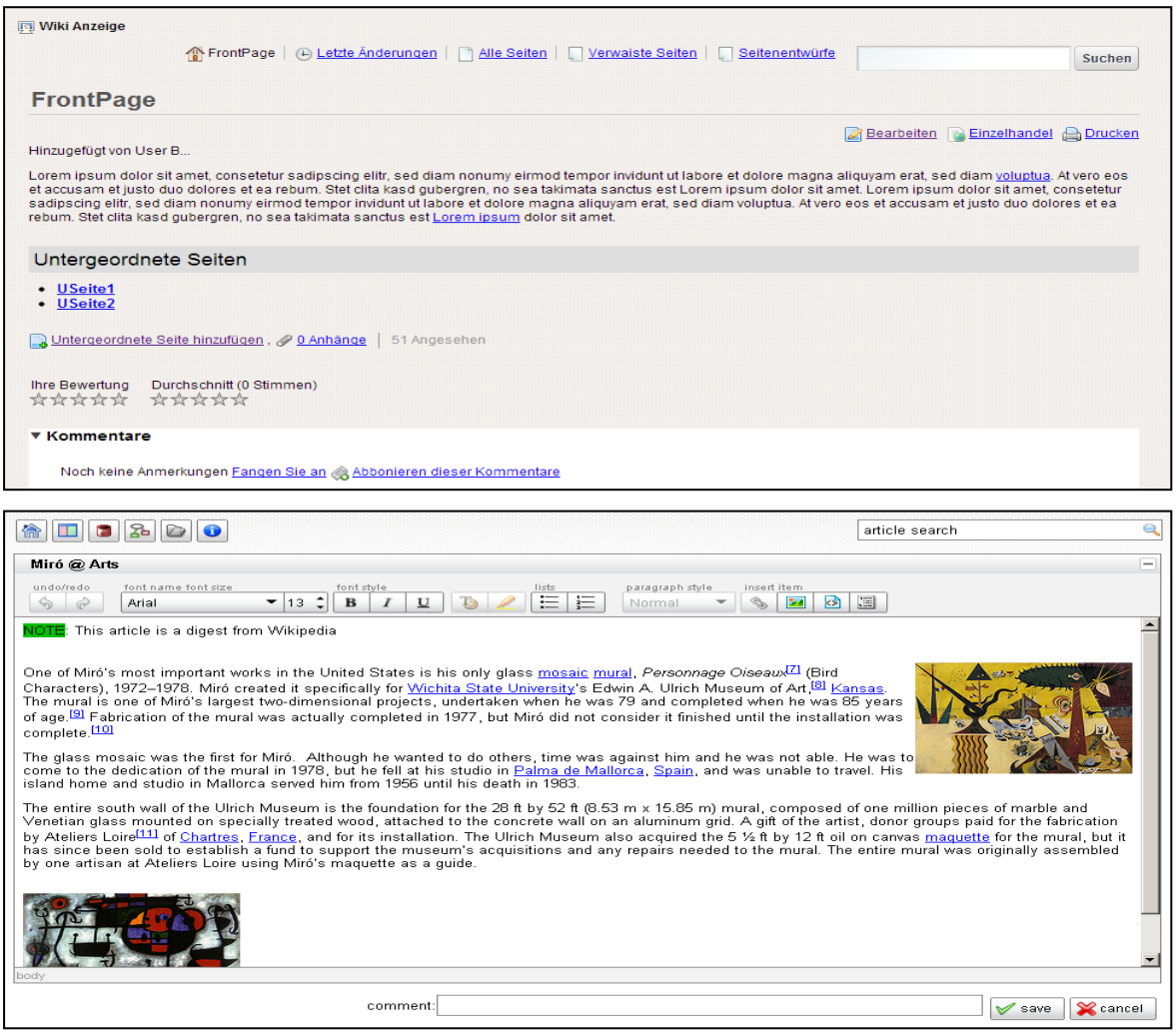


Abb. 4.8: Typische Seiten von Wikis, wie sie in Portalen integriert sein können. Wikiseiten sind dabei typischerweise über Hyperlinks miteinander verlinkt. Manche Wiki-Implementierungen pflegen eine Ober- und Unterbeziehung zwischen Wikiseiten (oberer Screenshot)

4.4 Zukünftige Anforderungen an die Entwicklungsplattformen für Web-Anwendungen

4.4.1 Die Rolle von HTML5

Die Beschreibungssprache HTML für Inhalt und Struktur von Webseiten wurde im Rahmen der HTML5-Standardisierung um einige wesentliche Funktionalitäten erweitert. Auch andere neue mit HTML5 assoziierte Standards wurden in diesem Zusammenhang neu spezifiziert oder aktualisiert (siehe Kapitel 3.1). Die neuen Funktionalitäten bieten essentielle Verbesserungen für den Betrieb von Webanwendungen mit einem responsiven Design, das sich sowohl für den Zugriff vom Desktop-Computer aber auch von Mobilgeräten eignet und sie verbessern die Bedingungen zum Schreiben moderner Webanwendungen mit Desktopanwendungs-naher Ergonomie.

Daher sollten CMS-Anwendungen, Portale und Web-basierte Fachanwendungen auf Dauer auf die Nutzung von HTML5 migriert werden. Die Möglichkeit hierzu wird sich z.T. auch dann bieten, wenn diese Systeme auf die neuen Layoutvorgaben des Landes für untergeordnete Behörden migriert werden müssen.

Die verwendeten technischen Systeme sollten daher auf Dauer auch darauf ausgelegt sein, HTML-Code in HTML5-kompatibler Weise zu generieren.

4.4.2 Java als Programmiersprache für Webanwendungen

In den letzten Jahren ging die Entwicklung von Webseiten zunehmend von der statischen Darstellung von Inhalten hin zur dynamischen Webseitengenerierung.

Auch die von der LUBW verwendeten technischen Systeme für Webanwendungen müssen daher darauf ausgelegt sein, in mehr oder weniger großen Umfang die Programmierung von Erweiterungen oder vollständiger, komplexerer Web-basierter Fachanwendungen zu unterstützen. Dabei sind solche Programmierarbeiten weniger bei einem CMS „von der Stange“ notwendig, das bereits über Standardmodule alle wichtigen Grundfunktionalitäten bereitstellt. Für personalisierbare Portale und Web-basierte Arbeitsumgebungen oder Systeme zur Entwicklung dedizierter Web-basierter Fachanwendungen sind die bereitgestellten Funktionalitäten zur Programmierung jedoch von entscheidender Bedeutung.

Dabei wurde Java in den letzten Jahre als wichtigste Entwicklungssprache für Erweiterungen der CMS-Anwendungen auf Basis von WebGenesis, für Erweiterungen der Portale und für die Programmierung dedizierter Fachanwendungen verwendet.

Durch Nutzung von Java als Programmiersprache konnten spezielle Funktionalitäten für Portale wie FADO, Themenpark und Energieportal problemlos implementiert werden. Auch die für die WebGenesis-Systeme entwickelten Suchmaschinen-OneBoxes sind in Java geschrieben. Der hierbei entwickelte Code sollte auch in der Zukunft weiterverwendet werden können. Daher ist eine wichtige Anforderung an zukünftige Portale oder Entwicklungsumgebungen für kompliziertere Web-basierte Fachanwendungen, dass sich die bereits in Java implementierten Softwarebausteine weiter nutzen lassen. Dies geht besonders leicht, wenn

die zugrunde liegende technische Plattform ebenfalls Java-basiert ist und dem JEE-Standard folgt (siehe auch Kapitel 3.3).

Über die Webanwendung „Umwelt-Datenbanken und -Karten Online“ (UDO) werden die Umweltdatenbanken der Umweltverwaltung Baden-Württemberg im Internet verfügbar gemacht. Diese Systeme basieren auf der Software disy Cadenza, die ebenfalls mit der Java-Sprache geschrieben ist. Künftig sollten Portale eine bessere Integration von UDO / Cadenza Inhalten ermöglichen. Dies lässt sich ebenfalls leichter realisieren, wenn beide technologischen Systeme nach Möglichkeit auf der gleichen Programmiersprache (hier Java) aufbauen.

4.4.3 Erweiterungsentwicklung und Hot Deployment

Hot Deployment bezeichnet die Fähigkeit eines CMS, Portals oder einer anderen Webanwendung Erweiterungen (wie bei Java-basierten Portalservern z.B. Portlets) während der Laufzeit des Servers hinzuzufügen oder zu entfernen. Die Vorteile solcher zur Laufzeit installierbarer Softwarekomponenten als Erweiterungen eines Systems wurden bereits in 3.4 ausgiebig beschrieben und sollen an dieser Stelle nicht noch einmal wiederholt werden.

Wünschenswert ist auf jeden Fall, dass eine technische Grundinfrastruktur für Portale oder zur Programmierung Web-basierter Fachanwendungen gute Möglichkeiten zum Schreiben von funktionalen Erweiterungen besitzt. Für Portale ist dabei auf jeden Fall die Unterstützung von „Hot Deployment“ vorteilhaft, da bei solchen Systemen in Zukunft immer mal wieder neue Web-basierte Fachanwendungen von verschiedenen Erstellern installiert werden müssen. Idealerweise wird dabei die Installation solcher Erweiterungen direkt über die Administrationsoberfläche ermöglicht.

Bei Web-basierten Fachanwendungen, die aus einem „Guss“ sind, stehen dagegen die Programmiermöglichkeiten zur Erweiterung des Grundsystems und nicht so sehr das Hot Deployment im Vordergrund.

4.4.4 Nutzung von Diensten

Innerhalb des Web-UIS werden bereits eine Reihe von generischen Diensten, vor allem im GIS-Bereich, genutzt, um anderen Systemen Hilfsfunktionalitäten, wie die Ermittlung der Bounding-Box eines Schutzgebietes, zu ermöglichen. Die Verwendung von dedizierten, in sich abgeschlossenen Diensten, die per Services erreichbar sind, bietet generell Vorteile innerhalb einer Web-basierten Softwarelandschaft, da hierdurch Redundanzen bei der Implementierung gleichartiger Funktionalitäten vermieden werden und teilweise komplexe zu implementierende Funktionalitäten für andere Software auf leicht zu bedienende Art über das Netz entfernt bereitgestellt werden (siehe auch Kapitel 3.6).

In einer Dienste-orientierten Softwarearchitektur sind aber nicht nur interne Dienste von Bedeutung, sondern auch das Internet selbst stellt heutzutage bereits sehr viele, oftmals kostenlose, Dienste (z.B. Gazetteer-Services zur Ermittlung der räumlichen Position von Örtlichkeiten) bereit, die allgemein genutzt werden können.

Ein zukünftiges Konzept für das Web-UIS sollte daher die Nutzung von Diensten und eine Dienste-orientierte Grundarchitektur stärker berücksichtigen.

So können auch mobile Apps, die im Rahmen des UIS entwickelt werden, die in der Webumgebung bereitgestellten Dienste nutzen. Ebenso kann man sich vorstellen, dass die eigenen Server für sichere Cloud-Dienste im Web genutzt werden, z.B. zum Speichern von Ergebnissen oder Fotos, oder dass Karten, Messdaten und andere Inhalte im Web für die Apps nutzbar bereitgestellt werden.

4.5 Anforderungen aus Betriebssicht

Die Anforderungen des Betriebs ergeben sich aus organisatorischen und technischen Gesichtspunkten. Die Rahmenbedingungen für den Betrieb von Anwendungsumgebungen haben dazu geführt, dass die Verfügbarkeit von Anwendungsbetreuern schon während der Kernarbeitszeit nicht mehr kontinuierlich über das ganze Jahr gewährleistet ist. Die Situation wird sich durch neue Sparmaßnahmen in den nächsten Jahren weiter verschärfen. Um trotzdem den Betrieb der Anwendungsumgebung für die Webpräsentation mit wirksamen Vertretungsregelungen wenigstens während der üblichen Arbeitszeit sicherstellen zu können, ist eine Standardisierung zwingend notwendig. Als „Bausteine“ der Anwendungsumgebung sollten deswegen Windows als Betriebssystem, Oracle als Datenbank, Apache als Webserver und Tomcat als Laufzeitumgebung für Java unterstützt werden. Das übergreifende bzw. „bausteinspezifische“ Wissen und die Werkzeuge zur Fehlersuche, Performance-Analyse und zum Monitoring könnten so weitreichender und damit wirtschaftlicher eingesetzt werden.

Aus dem Blickwinkel des Betriebs gehört zu den technischen Anforderungen weiter, dass die Anwendungsumgebung über eine nachgewiesene Systemstabilität in einer größeren Umgebung verfügt. Wichtige Aspekte sind die einfache Skalierbarkeit und Gewährleistung der Ausfallsicherheit, um auch die hohe Systemlast während eines Hochwassers mit festlegbaren Antwortzeiten abdecken zu können.

Die Architektur sollte weiter die Umsetzung von Sicherheitsrichtlinien ermöglichen. Auf weitere separate Middleware (bis auf Tomcat oder einen äquivalenten Applikationsserver) sollte verzichtet werden, um die Komplexität der Gesamtarchitektur beherrschbar zu halten. Um eine nachvollziehbare Überführung von der Entwicklung zum Betrieb zu ermöglichen, ist eine integrierte Versionsverwaltung notwendig. Aus Sicht der Anwender wäre ein feingranulares Backup und Restore wichtig, um gezielt und zeitnah Daten zurückspielen zu können. Für die Authentifizierung ist der Einsatz von Zertifikaten erforderlich. Um die notwendige Rechtevergabe zu erleichtern, wäre die Anbindung an ein Active Directory hilfreich.

Die Weiterentwicklung für die bisherige Anwendungsumgebung ist dadurch gekennzeichnet, dass unmittelbar nach dem Entwicklungsende der Produktionsbeginn erwartet wird. In vielen Fällen werden dem Betrieb nur kurzfristig Termine der Entwickler angekündigt, der Produktionsbetrieb soll allerdings sofort reagieren. Das sollte sich in Zukunft ändern.

Aufgrund der eingeschränkten Belastbarkeit der Anwendungsumgebung werden bisher zahlreiche Einzelsysteme betrieben, für die auch noch eine ständige Betreuung durch die Entwickler notwendig ist. Hierzu mussten zahlreiche Rechte auf den jeweiligen Rechnersystemen vergeben werden, damit die Entwickler an ihren eigenen Systemen selbst die neue

Funktionalität in Produktion nehmen können. Zum einen führt es dazu, dass der Support von einem System hauptsächlich von den jeweiligen Entwicklern geleistet werden muss. Zum anderen ist der Überblick über die verfügbare Funktionalität für den Betrieb nur mit hohem Aufwand nachvollziehbar. Bevor eine neue Anwendungsumgebung eingeführt wird, ist es dringend notwendig, dieses „ad hoc“-Vorgehen durch einen nachvollziehbaren Prozess zu ersetzen.

Die Betreuung zukünftiger Systeme sollte daher in Zukunft bevorzugt nicht mehr auf Betriebssystemebene, sondern über die Systeme selbst möglich sein.

5. Analyse bestehender Systeme

Wie bereits in Kapitel 4 beschrieben wurde, entsprechen die unter Kapitel 2 aufgeführten Systeme nur teilweise den heutigen Anforderungen von Webnutzern, Webautoren und Administratoren und sind auch aus Sicht des aktuellen Entwicklungsstandes (siehe Kapitel 3) nicht mehr „State of the Art“. Zur genauerer Analyse dieser Tatsache werden im Folgenden mehrere Aspekte von Webanwendungen unterschieden: Das Webdesign und die zugehörige technische Implementierung der Design-Vorlage als HTML-Code für ein bestimmtes technisches System (Design-Template genannt), das zu Grunde liegende technische System selbst als CMS bzw. Portalsoftware (für die bisherigen Homepages und Portale WebGenesis) und die im Web-UIS zur Zeit angewandten Konzepte und Entwicklungsplattformen zur Programmierung von Webanwendungen.

5.1 Web-Design und HTML-Code

Das visuelle Design einer Website ist zu großen Teilen Geschmackssache und damit oft Generations- und Zeit-abhängig. Das aktuell im Land Baden-Württemberg verwendete Design für die landeseigenen Websysteme ist daher in die „Jahre gekommen“ und wurde von der neuen Landesregierung als alt empfunden, so dass für das neue Landesportal ein grundlegend neues Design entwickelt wurde. Ein gutes Design hat allerdings nicht nur visuelle, sondern auch strukturelle Aspekte.

Das bisher verwendete Design, wie es zur Zeit für die Webauftritte des UM und der LUBW noch aktuell verwendet wird, verwendet vom Grundprinzip her ein sich in der Breite nur zu einem gewissen Grad anpassbares Layout (Design mit variabler Breite), wobei die beiden linken und rechten Spalten konstante Breite behalten, aber die Breite der mittleren Hauptinhaltsspalte entsprechend angepasst wird. Die Gesamtbreite der Bilder im Kopfbereich (Logo + Schriftzüge ergeben dabei eine natürliche Schranke für die minimale Breite, bis zu der das Design noch skaliert. Diese minimale Breite liegt aber weit oberhalb der Nutzbreite, die ein Smartphone im Hochformat besitzt. Ein solches Web-Design eignet sich daher prinzipiell nicht dazu, Inhalte ohne horizontalen Rollbalken sowohl in Desktopbrowsern als auch auf Smartphones anzuzeigen.

Von einem modernen responsiven Design erwartet man, dass bei Schrumpfen der Breite nicht nur der mittlere Inhaltsbereich bis zu einer vorgegebenen Breite verkleinert, sondern ab gewissen Breiten auch die Lage der drei Spalten zueinander entsprechend angepasst wird,

damit das Design auf Geräten mit unterschiedlich breiten Bildschirm immer geeignet dargestellt wird. Hierzu kann man entweder die Spalten automatisch ab gewissen Breiten untereinander statt nebeneinander darstellen oder z.B. die Spalten mit JavaScript auf- und zuklappbar oder auf andere Art navigierbar machen. Dabei müssen auch Kopf- und Fußbereich den neuen Größenverhältnissen gerecht angepasst werden (z.B. zu breite Navigationsmenüs umsortiert und in kleinere, aufklappbare Navigationsmenüs zusammengefasst und zu große Logos und Schriftzüge durch kleinere ersetzt oder anders angeordnet werden).

Das neue Landesdesign unter http://www.baden-wuerttemberg.de implementiert im Prinzip solch ein „Responsive Design“, bei dem ab einer gewissen Breite vorher nebeneinander liegende Spalten nun untereinander dargestellt werden. Die Implementierung funktioniert allerdings noch nicht in allen Situationen vorbildlich. So erhält man im Desktopbrowser bei gewissen Breiten ein abgeschnittenes Bild, da Spalten erst ab einer gewissen Breite untereinander geschoben werden. Die Breite, ab der dies passiert, ist dabei deutlich kleiner als die Breite, ab der alle Spalten nicht mehr nebeneinander in den Browser passen. Daher kommt es bei Zwischenbreiten dazu, dass nicht mehr die gesamte Breite der Webseite angezeigt wird, ohne dass man das Browserfenster größer macht.

Trotzdem kann man von der Umstellung der bestehenden Systeme auf das neue Landesdesign auf die Dauer erwarten, dass sich damit strukturelle Verbesserungen hin zu einem „Responsiven Design“ ergeben, so dass die Webangebote dann auch gut auf Geräten mit unterschiedlichen Bildschirmgrößen anzeigbar sind. Dabei sollten allerdings die zurzeit noch bestehenden implementierungstechnischen Probleme behoben sein.

Leider hält die Implementierung des neuen Landeslayouts für das TYPO3-System auch an vielen anderen Stellen nicht, was das Design indirekt verspricht. So enthält das System einen Link „Gebärdensprache“ zum barrierefreien Zugang von Taubstummen zu Videos; das System ist aber nicht wirklich mit einem Browser für Blinde navigierbar, wie bei einem kurzen Test festgestellt wurde. Des Weiteren wird im Kopf der Webseiten formal angedeutet, dass es sich bei der HTML-Implementierung um eine HTML5-konforme Webseite handelt, HTML5-spezifische Funktionalitäten werden allerdings nur von einigen TYPO3-Komponenten vornehmlich zur Erzeugung dynamischer Inhalte und für Medien genutzt, aber z.B. die HTML5-eigenen Strukturierungselemente überhaupt nicht für die Implementierung des Grunddesigns der Webseite verwendet. So sind z.B. Navigationsbereiche nicht HTML5-konform ausgezeichnet und damit gerade nicht durch Browser für Blinde unmittelbar erkennbar oder von speziellen Browsern auf Mobilgeräten für einen intelligenten Umgang mit dem Layout interpretierbar.

Diese Defizite in der Implementierung des neuen Designs sollte man bei der Übertragung auf andere technische Systeme in Betracht ziehen und gegebenenfalls das visuelle Design auf HTML5-Ebene auch wirklich unter Nutzung von reinem HTML5 richtig implementieren. Ansonsten verschenkt man doch eine ganze Reihe der Möglichkeiten, die HTML5 bietet.

Im Fazit lässt sich sagen, dass manche aktuellen Probleme mit den Webangeboten im UIS weniger mit den technischen Systemen, wie WebGenesis, zu tun haben, sondern dem veralteten Rahmendesign und der gegebenenfalls veralteten HTML5-Implementierung der Rahmentemplates zuzuschreiben sind. Hier kann man eine deutliche Verbesserung erwarten, wenn man die Systeme bzgl. des Rahmendesigns auf die neuen Landeslayout-Vorgaben

aktualisiert. Dabei sollte man aber nach Möglichkeit die HTML-Implementierung, wie sie von den Designern kommt, genau anschauen und gegebenenfalls an moderne HTML5-Konzepte anpassen, um alle Vorteile der Umstellung auch nutzen zu können.

Der HTML-Code kommt allerdings nicht zu 100% vom Rahmendesign, sondern wird auch dynamisch von dem zugrunde liegenden technischen System generiert. Der aus dem TYPO3-System heraus generierte Code der neuen Baden-Württemberg Homepage zeigt z.B. keine Hinweise auf eine evtl. Nutzung von Microdata-Formaten, um die Semantik der Informationen innerhalb der Seiten besser darzustellen, denn dies wird von dem TYPO3-System zurzeit wohl auch noch nicht unterstützt. Hier sollte man auf Dauer darauf achten, dass nicht nur der HTML-Code des Rahmendesigns, sondern auch der HTML-Code, den das zugrunde liegende technische System generiert, moderne HTML5-basierte Konzepte verwendet. Für CMS, wie Drupal und Joomla, gibt es mittlerweile Plugins, die das Generieren von Microdata-Beschreibungen unterstützen. Dies ist aber bei den meisten bis dato eingesetzten Systemen, selbst bei dem von Land eingesetzten TYPO3, bislang nicht der Fall. Außerdem benutzen die meisten CMS oder Portale auch noch nicht die neuen HTML-Tags für die Auszeichnung von Strukturen, wie Navigationselemente, Sidebars oder Artikel.

5.2 Analyse momentan eingesetzter technischer Systeme

Für viele der unter Kapitel 2 aufgeführten Systeme wird das WebGenesis-System des Fraunhofer IOSB als technische Plattform eingesetzt. WebGenesis übernimmt dabei zweierlei Funktion im Web-UIS. Es dient den Forschungspartnern der Forschungskooperation MAF-UIS (Moderne anwendungsorientierte Forschung und Entwicklung für Umweltinformationssysteme) sowohl als Java-basierte Implementierungsplattform für Webanwendungen und Portale als auch den Umweltbehörden als Standard-CMS. Im Folgenden sollen die Ergebnisse der Analyse des WebGenesis-Systems in Bezug auf diese Rollen kurz vorgestellt werden. Dabei ist zu beachten, dass sich die Analyse auf die Versionen 7.x von WebGenesis beziehen, die derzeit im Web-UIS im Einsatz sind. Neuere Versionen (8.x) sind im Web-UIS nicht im Einsatz und werden daher hier nicht betrachtet (siehe hierzu auch das Kapitel 6).

Ein anderes technisches System, das im Umfeld der Behördenauftritte des Landes Baden-Württemberg für mehrere Behörden eingesetzt wird, ist das Enterprise Content-Management-System pirobase der Imperia AG.

Diese beiden Systeme sollen daher mit TYPO3 als System für den neuen Landesauftritt und Liferay als Kandidat für einen Open Source Enterprise Portalserver, der ebenfalls für Behördenauftritte empfohlen wird, verglichen werden. Beim Vergleich unterscheiden wir dabei zwischen der Nutzung der Systeme in Richtung reiner CMS-Anwendung, der Portalnutzung sowie der Nutzung als Entwicklungsplattform für Web-basierte Fachanwendungen.

5.2.1 WebGenesis und pirobase als Laufzeitplattform für CMS-Anwendungen

Laut Aussage des IOSB ist WebGenesis eher als Entwicklungsplattform für Webanwendungen und weniger als CMS „von der Stange“ positioniert. WebGenesis bietet daher „out of the

box“ nur Grundfunktionalitäten eines CMS, und eine Reihe weiterer dringend benötigter Funktionalitäten mussten im Web-UIS Umfeld zunächst als zusätzliche Funktionalitäten für LUBW und UM eigens programmiert werden.

pirobase ist wie WebGenesis und Liferay in Java programmiert, verwendet wie Liferay die Template-Sprache Velocity als bevorzugte Sprache für Templates von Webseiten und lässt sich wie WebGenesis oder Liferay durch Java-Programmierung unter Nutzung der bereitgestellten API-Schnittstellen erweitern. Im Gegensatz zu Liferay ist pirobase als reines Enterprise CMS und WDMS und nicht als Portal positioniert, und enthält deshalb anstelle von Portalfunktionalitäten nur Serviceschnittstellen, um sich mit Portalen, wie Liferay, zu integrieren (siehe auch Bemerkungen zu Alfresco als DMS und WCMS). Eine Anbindung an DMSs und DAMs ist ebenfalls möglich. Der Fokus von pirobase liegt dabei auf der Erstellung, Verwaltung und Bereitstellung großer Mengen modularer multimedialer Webinhalte (daher Enterprise CMS) sowie der zielgruppen-spezifischen Anzeige dieser Inhalte in Webseiten oder der Bereitstellung der Inhalte für andere technische Systeme. Hierzu stellt pirobase gängige Funktionalitäten zur Erstellung und modularen Anzeige der Inhaltsbausteine in Webseiten bereit. Als Enterprise CMS enthält es natürlich ausgefeilte interne Funktionalitäten zur Verwaltung von Web-Content oder auch binären Inhalte im Sinne eines WDMS.

Die folgende Tabelle 5.1 gibt einen ungefähren Überblick über in WebGenesis und pirobase bereits existierende und nicht existierende Funktionalitäten im Vergleich zu dem vom Land verwendeten CMS TYPO3 und Liferay als Beispiel für einen typischen Portalserver. Dabei wird in der Tabelle vor allem auf Funktionalitäten eingegangen, wie sie in Kapitel 4 als Anforderungen formuliert und auch für personalisierbare Portale und Web-basierte Arbeitsumgebungen definiert sind.

Funktionalität	WebGenesis	pirobase	TYPO3	Liferay
CMS-Basisfunktionalitäten				
Beziehung Webinhalt zu Seite	Inhalte sind in sog. Einträgen enthalten. Diese können mit Hilfe von Relationen einer Ontologie mit einer Seite verknüpft werden.	Inhaltsbausteine (Content), die vom Autor frei platziert in Webseiten-Schablonen positioniert werden können.	Inhaltsbausteine können frei auf Seite eingefügt werden	Inhaltsbausteine können frei auf Seite eingefügt und positioniert werden
Versionierung von Inhalten	Nein, indirekt über sog. Meilensteine	Ja	Ja	Ja
Unterstützung für redaktionellen Workflow	Selbstdefinierte Workflows mit Hilfe der Workflow-Engine von WebGenesis, beliebig erweiterbar bzgl. Rollen, Zustände,	Über Workflow Engine	Über Erweiterung	Ja, beliebig über Administrationsoberfläche definierbare Workflows

	Aktionen inkl. Anbindung externer Dienste			
Staging: Zweistufige Inhaltserstellung; Unterscheidung zwischen Life-Inhalten und „Inhalten in Bearbeitung“	Nein	Nein	Nein	Ja
Unterstützung für mehrere Sites und mehrere Domainen in einer Instanz	Ja, über Baukasten für UM und LUBW in Verbindung mit Apache-Tomcat	Ja	Ja	Ja
Authentifizierung	Integriertes User Management, LDAP-Import, erweiterbar	Integriertes User Management, LDAP (ADS)	Integriertes User Management, LDAP-Integration, erweiterbar	Integriertes User Management, LDAP-Integration, OpenID-Accounts, Facebook Login
Möglichkeiten für Single Sign On (SSO)	nein	Ja, Kerberos, NTLM, SAP® Logon Ticket	nein, evtl. über Erweiterung	Ja, Unterstützung für CAS, NTLM, OpenSSO, Oauth-Authorisierung
Authorisierung / Sicherheit	User- und Gruppen-basierte Authorisierung	Ja, differenziertes User, Gruppen, Rollen und Rechtekonzept	User, Gruppen, flexibles Rollenkonzept, Beschränkung bzgl. IP-Adressen, Restriktion auf Domain	User, Gruppen, flexibles Rollenkonzept, erweiterbar
Leistungsfähiger Online-Webeditor	Ja	Ja	Ja	Ja
Medien- und Dokumentenmanagement	Keine eigenständige Verwaltung	Ja; Integration mit externen DMS und DAM möglich	Ja; Integration mit DAM (Digital Asset Management) möglich	Ja; Integration mit DMS (Dokument Management Systemen) und DAM möglich
Sprechende URLs	rudimentär, über Alias	Ja	Ja	Ja
Formulargenerierte Inhaltserstellung	Ja; WebGenesis Formulareditor auf Basis von XML / XForms	Ja	Formulare über Erweiterung in TYPO3 erstellbar	Ja; Autoren-Formulare in Liferay erstellbar

Editierung von Metadaten	Ja	Ja	Ja	Ja
Konzept von Design-Templates	Nein	Nein	Ja	Ja
Kollaborative Eigenschaften / Personalisierung				
Explizites Konzept für Personalisierung	Nein	Ja	Bedingt: eigene Gruppenseiten und Nutzerseiten möglich; beschränkt konfigurierbar durch Nutzer	Volle Unterstützung für personalisierbare Seiten für Nutzer, Nutzergruppen
Interne Social Funktionalitäten	Nein	Nein	Über Erweiterungen (Blogs, Foren, ToDo-Verwaltung, Kalender, Chat, private Messages)	Social Office Erweiterung (Personal Blogs, Foren, ToDo-Verwaltung, gemeinsamer Arbeitskalender, Shared Dokumentbereiche, Wikis, Chat, private Messages, …)
Integration mit Office-Welt	Nein	Sharepoint Integration, WebDav	Nein	Module zum Darstellen von Office-Dokumenten; Integration mit Windows-Systemen; Sharepoint Connector
Erweiterungen / Module für gängige Funktionalitäten				
Kalender	Ja	Nein	Ja	Ja
Aufgabenmanagement	Ja (einzelne Projekte)	Nein	Ja	Ja
Forum / Schwarzes Brett	Ja	Ja	Ja	Ja
Wiki-Integration	Nein	Nein	Ja	Ja
Social Integration Twitter / Facebook	Nein	Ja	Ja	Ja
Umfragemodule	Nein	Nein	Ja	Ja

Social Media (z.B. YouTube), Medienplayer	Nein	Interner Video-player	Ja	Ja
Bildergalerien	Ja, eingeschränkt	Ja, einge-schränkt	Ja	Ja
Kartendar-stellung	Weitestgehende Integration mit OGC-Standards (auf Grundlage von OpenLayers)	Nein	Google Maps Module	Google Maps Module
Feed-Unterstützung (RSS / Atom)	Ja	Ja	Ja	Ja
Kontakt- und Adressenver-waltung	Nein	Ja, über LDAP oder ADS Integ-ration	Ja	Ja

Tabelle 5.1: Übersicht über wichtige CMS-Funktionalitäten

Wie Tabelle 5.1 zeigt, bietet WebGenesis gegenüber anderen spezialisierten CMS-Systemen oder Portalservern zurzeit nur eingeschränkte CMS- und auch Portal-Funktionalitäten und ist daher für eine zukünftige „Out of the box"-Nutzung als CMS, personalisierbares Portal oder personalisierbare Web-basierte Arbeitsumgebung nur dann geeignet, wenn auf bestimmte fehlende Funktionalitäten verzichtet werden kann oder diese für die zukünftige Verwendung entwickelt.

Es lassen sich aber durchaus einige der fehlenden Funktionalitäten mit geringem Aufwand für WebGenesis entwickeln, andere (wie z.B. Portalfunktionalitäten, wie Personalisierbarkeit und Integration von neuen Anwendungsmodulen über Hot Deployment) würden dagegen größere Entwicklungsarbeiten nach sich ziehen.

pirobase besitzt gegenüber WebGenesis bessere „Out of the box"-Funktionalitäten zum Management von Inhalten (dies ist gerade die Stärke eines Enterprise CMS), wie z.B. leistungsfähiger Bildeditor und WYSIWYG-Editor mit der Unterstützung verschiedener Autorenprofile für unterschiedliche Autorengruppen (pirobase smart editor), Inhaltsversionierung und Archivierungsfunktionalitäten. Auch der flexible Einsatz von Inhaltskomponenten in Webseiten wird mit Mechanismen zum automatischen Filtern, regelgestützte Aggregation von News- und Inhaltsübersichten, etc. sehr gut unterstützt.

pirobase fehlt aber wie WebGenesis ein flexibles Konzept der Erweiterbarkeit durch wiederverwendbare Komponenten von Fremdanbietern, wie z.B. das Portlet-Konzept von Enterprise Java Portalservern oder das Extension-Konzept von TYPO3. Das bedeutet, dass für eine ganze Reihe von Standardkomponenten, wie Kalender, Wiki-Integration, Youtube-Video oder Kartenintegration, Entwickler bei dem Aufbau eines pirobase-Systems für einen Kunden solche Funktionalitäten in der Regel nachprogrammieren müssen. Hier bietet gerade für CMS-Funktionalitäten ein System wie TYPO3 deutlich mehr Standardkomponenten über einen Katalog von Komponenten von Drittanbietern an.

Beide Systeme kennen ebenfalls kein direktes Erweiterungskonzept, was den Austausch des Corporate-Designs über Hochladen eines anderen Design-Templates für alle Webseiten des Systems erlaubt. Hier müssen vielmehr für jedes neue Design die Templates des Gesamtsystems komplett überarbeitet werden, was umfangreiche Arbeiten und eine lange Migrationsdauer für die Umstellung der pirobase- oder WebGenesis-Instanzen auf das neue Landesdesign nach sich zieht.

5.2.2 WebGenesis und pirobase als Entwicklungsplattform

WebGenesis wie auch pirobase basieren als Java-basierte Entwicklungsumgebung für Webanwendungen auf dem Java Enterprise Edition (JEE) Standard und bieten daher eine gute Grundlage für Java-basierte Webanwendungen (siehe auch Kapitel 3.3). Die hierbei im Web-UIS verwendete WebGenesis-Laufzeitplattform ist allerdings typischerweise kein voller JEE-Applikationsserver, sondern ein reiner Webcontainer (Tomcat), so dass bei einer Standardinstallation von WebGenesis nicht alle JEE-Funktionalitäten (nicht einmal die des sogenannten JEE-Web-Profils) zur Verfügung stehen. Dies gilt für pirobase ebenfalls. pirobase setzt intern das Spring-Framework ein und empfiehlt Spring MVC und Velocity für die Entwicklung von Oberflächen. Dies ermöglicht eine etwas modularere Erweiterbarkeit als WebGenesis, löst aber nicht das grundsätzliche Problem mit diesem Ansatz (siehe weiter unten).

Außerdem kapseln die WebGenesis- und pirobase-Umgebung die JEE-Grundfunktionalität vollständig und stellen ihre eigenen Mechanismen zur Erweiterungsprogrammierung bereit. So nutzen Erweiterungen von beiden Systemen zur Definition web-basierter Nutzeroberflächen nicht die gemäß dem Standard bereitgestellte JSP- oder als bessere Alternative JSF-Technologie, sondern WebGenesis stellt für die Weboberflächenprogrammierung seine eigene Templatesprache Embedded Java (EJava) bereit, während pirobase hier den Einsatz von Velocity empfiehlt. Template-Sprachen, wie EJava und auch Velocity als besssere Variante, fehlen aber viele Fähigkeiten moderner Java-basierter Weboberflächen-Frameworks, wie eingebaute AJAX-Funktionalitäten, State- und Lebenszyklusmanagement, Möglichkeiten zur Programmierung wiederverwendbarer Oberflächenkomponenten oder Tools zur automatischen Verifikation und Konvertierung von Eingabedaten. Solche Funktionalitäten müssen dann in Anwendungen, wie WebGenesis oder pirobase, auf andere Weise bereitgestellt werden, fehlen aber in einer WebGenesis-Grundversion.

In Standardinstallationen von WebGenesis oder pirobase fehlen teilweise auch weitere JEE-Funktionalitäten, wie z.B. ein JAX-RS konformes Framework zum Umgang mit REST-Services. Diese Defizite lassen sich aber leicht beheben, da im Rahmen von Projekten, wie dem Themenpark Umwelt, bereits gezeigt wurde, dass sich solche fehlenden Funktionalitäten in WebGenesis und wohl auch pirobase leicht nachinstallieren lassen und meistens auch kompatibel zum Rest von WebGenesis genutzt werden können.

Die folgende Tabelle 5.2 gibt einen Überblick darüber, welche Funktionalitäten zur Anwendungsprogrammierung in WebGenesis bereits „out of the box“ vorhanden sind, welche einfach nachinstalliert und dann genutzt werden können, welche evtl. nur mit größeren Modifikationen von WebGenesis nutzbar sind und welche komplett fehlen und gegebenenfalls auf andere Weise integriert werden müssten. Um einen Vergleich mit anderen JEE-basierten Entwicklungsplattformen zu bekommen, ist zusätzlich in der Tabelle ein Vergleich mit Liferay als typischem Enterprise-Portalserver enthalten.

Funktionalität	WebGenesis	pirobase	Liferay
Grundlage JEE-basierter Webcontainer (Servlet 2.x/3.x Spezifikation)	Ja	Ja	Ja
Volles JEE 6 Webprofil	Nein	Nein	Ja
Unterstützung der Web-Service Standards JAX-RS, JAX-WS	Ja, prinzipiell möglich JAX-WS Support über Axis. JAX-RS Support lässt sich nachinstallieren; keine eingebaute Mechanismen zur Nutzung von Services	Nicht als offizielle APIs, lässt sich einfach nachinstallieren. Keine eingebaute Mechanismen zur Nutzung oder Bereitstellung von Services	Ja, alles bereits an Bord. Eingebaute Mechanismen zur Nutzung von Services
Getrennte Design-Templates und/oder Layout-Templates	WebGenesis-eigene EJava Template-Sprache. Kein separiertes Konzept für Design-Templates. WebGenesis-Einträgen können wahlweise unterschiedliche Layout-Templates von Autoren zugeordnet werden.	Verwendet Velocity und Spring MVC für Vorlagen (Templates) von Webseiten oder Inhaltsbausteinen	Unterstützung von Design- und Layout-Templates. Diese basieren auf Velocity; eigene Alloy-JavaScript-Library für Widget-Integration in Design-Templates Templates und Layout-Templates lassen s. als Komponenten im System installieren
Unterstützung von JSF als JEE-Standard	Nein, Integration erfordert einiges an Arbeit	Nein	Einsatz von JSF für Portletprogrammierung möglich
Konzept(e) für Erweiterungsprogrammierung	Ja. Das Schreiben eigener Applikationsservices ist möglich, eigene Templates, Programmierung eigener Inhaltskategorien, Modifikation von Templates des Grundsystems. Das Ganze erfolgt jedoch eng verzahnt mit dem Grundsystem.	Smart View Plugin Mechanismus, aber keine expliziten Konzepte über eine Administrationsoberfläche installierbarer Komponenten sichtbar	Ja **explizite Konzepte** für verschiedene Erweiterungen: Services, Portlets, Hooks, Extensions. Erweiterungen auf verschiedenen Ebenen bis zum Überschreiben interner Funktionalitäten oder der Erweiterung der Core-Funktionalitäten durch Hooks und Extensions möglich.
Über die Administrationsoberfläche installierbare Erweiterungen; eigenständi-	Nein	Nein	Ja, Erweiterungen sind über Oberfläche installierbar. Portlets als Java-Standard für eigen-

ge Komponenten			ständig installierbare Erweiterungen
Welche Technologien sind für Programmierung der Oberflächen von Erweiterungen nutzbar	Neben EJava keine weiteren Mechanismen vorgesehen, es lassen sich allerdings durchaus z.B. JavaScript-Bibliotheken nutzen	Velocity, Spring MVC, YAML, Javascript-Bibiotheken hierbei nutzbar	Fast alle UI-Technologien für Portlets nutzbar: JSF 2, JavaScript Bibliotheken, wie jQuery, JavaScript-basierte Widgetbibliotheken, Vaadin, Alloy
Zugriff über Services	Einige wenige Funktionalitäten über Services erschlossen	Nicht durchgängig, für bestimmte Integrationszwecke sind aber APIs vorhanden, z.B. für Integration in Portale	Vollständige Liferay-Funktionalität sowohl über REST- als auch SOAP-basierte Webservices erschlossen und nutzbar

Tabelle 5.2: Übersicht vorhandener Funktionalitäten zur Anwendungsprogrammierung in CMS

Wie man Tabelle 5.2 entnehmen kann, bieten WebGenesis und pirobase durchaus gute Möglichkeiten zur Programmierung eigener Webanwendungen oder zur Erweiterung der eingebauten Funktionalitäten, wobei hier WebGenesis von seiner Struktur her wesentlich einfacher für den Programmierer zu verstehen, aber nicht ganz so leistungsfähig ist. Verschiedene Lücken können Programmierer dabei oft durch Hinzufügen weiterer Java-basierter Toolkits oder von JavaScript-Bibliotheken füllen.

Das größte Manko beider Systeme ist, dass sie standardmäßig über kein eingebautes, modernes Toolkit für das Schreiben von Weboberflächen unter Nutzung von AJAX verfügen, das die Benutzung bereits vorhandener Widgetbibliotheken und das Schreiben eigener Widgets erlaubt. Hier lassen sich mit geringem Aufwand aber durchaus JavaScript-basierte Technologien für moderne Oberflächen in beide Systeme integrieren. Die Integration komplexerer Technologien, wie JSF, würde dagegen deutlich mehr Aufwand bei beiden Systemen erfordern. Ein weiterer Nachteil von beiden Systemen ist, dass sie kein explizites Konzept von Design-Templates kennen, über die sehr leicht das Corporate Design der gesamten Webseite auf ein neues Design anpassbar ist. Für die Erstellung eines neuen Corporate Designs muss daher in die System-eigenen EJava- oder Velocity-Templates tiefer eingegriffen werden und die Umstellung eines solchen Systems auf ein neues Design erfordert viel Arbeit. Dies ist aber weniger ein Problem, wenn dedizierte Web-basierte Fachanwendungen entwickelt werden sollen. Hierbei wird man dann sowieso einen großen Teil der Web-UI neu schreiben müssen.

WebGenesis und pirobase besitzen auch keine Konzepte für über die Administrationsoberfläche installierbare eigenständige Softwareerweiterungen. Die Erweiterung von diesen Systemen ist momentan nur auf Systemebene möglich. Diese Schwäche kommt aber nur dann wirklich zum Tragen, wenn im Laufzeitsystem einer Web-basierten Anwendung ständig unabhängig geschriebene Softwarekomponenten nachinstalliert werden müssen. Bei dedizierten Web-Fachanwendungen ist dies aber nicht notwendig, da die Softwareentwicklung für solche Systeme eher in Releasezyklen stattfindet, an deren Ende dann das Komplettsystem aktualisiert wird.

Im Fazit lässt sich noch sagen, dass die Umstellung WebGenesis-basierter Web-Fachanwendungen auf pirobase keinen Sinn macht, da beide Systeme auf der Entwicklungsebene ähnliche Möglichkeiten bieten, aber eine Umstellung auf das deutlich komplexere pirobase den Mehrwert der etwas komfortableren Funktionalitäten nicht wett machen würde.

Beide Systeme bieten andererseits nicht annähernd die Funktionalitäten eines Liferay-Servers, wenn es darum geht, Anwendungen flexibel in ein Gesamtsystem zu integrieren.

5.3 Andere Entwicklungsplattformen im Web-UIS

Neben WebGenesis werden im Web-UIS Umfeld noch eine ganze Reihe anderer Fachanwendungen mit Java-basierter Software betrieben. Hierzu gehören u.a. die Systeme, die mit der Cadenza Software entwickelt werden. Die web-basierte Version von Cadenza „Cadenza Web“ setzt natürlich ebenfalls auf JEE-Technologie auf, so dass hier ebenfalls eine JEE-basierte Grundinfrastruktur für die Programmierung bereits vorausgesetzt werden kann. Auf Basis der Grundfunktionalität stellt „Cadenza Web“ kompliziertere Datenlogik (z.B. das Selektorframework) bereit, um mit strukturierten Umweltfachobjekten und geobasierten Daten umgehen zu können. Aufbauend auf dieser Fachlogik sind dann wiederum andere Anwendungen, wie z.B. Fachanwendungen für den Wasserbereich, geschrieben worden.

Mit Cadenza und „Cadenza Web“ geschriebene Anwendungen sind hochgradig mit dem darunterliegenden Cadenza Framework integriert und wenig modularisiert. Das Herauslösen einzelner funktionaler Teile diese Anwendungen und die getrennte Installation dieser z.B. in einem Portalserver als eigenständige Komponenten ist zurzeit nicht möglich.

Es zeigt sich allerdings ein eindeutiger Trend dazu, dass zukünftig Fachanwendungen verstärkt als Java-basierte Webanwendungen geschrieben werden. Im Zuge dieser Umstellung sollte man darauf achten, dass Funktionalitäten in web-basierten Fachanwendungen verstärkt modularisiert bereitgestellt werden und in JEE-konformen Laufzeitplattformen unabhängig von einer spezifischen Herstellersoftware als wiederverwendbare Softwarekomponenten bereitgestellt werden können.

Ein weiterer Wunsch an die „Cadenza Web“-basierten Systeme ist, dass diese – weil sie im Web-UIS Umfeld wesentliche Systeme zur Lieferung von Grunddaten von Umweltobjekten sind – ihre Daten in Zukunft stärker über lose koppelbare, serviceorientierte Schnittstellen für andere Softwaresysteme im Web-UIS bereitstellen. Der Zugriff auf Cadenza-interne Daten über serviceorientierte Schnittstellen ist momentan nur bedingt und sehr eingeschränkt möglich.

Neben den Java-basierten Anwendungen gibt es im Web-UIS Umfeld noch eine ganze Reihen kleinerer und auch größerer eigenentwickelter Anwendungen, die mit anderen Programmiersprachen geschrieben wurden (.NET Welt, PHP-Scripts, etc.). Solange diese Anwendungen in einem lokalen Kontext verwendet werden, führt das auch zu wenigen Problemen. Stellen solche Anwendungen aber Funktionalitäten bereit, die von anderen Anwendungen entfernt genutzt werden sollen, ist darauf zu achten, dass diese Funktionalitäten einerseits über lose koppelbare Schnittstellen bereitgestellt werden und andererseits die Laufzeit-

infrastrukturen dieser Services entsprechende „Qualtity of Services“, wie Verfügbarkeit und Ausfallsicherheit, Stabilität der Schnittstellen, etc. bereitstellen.

An dieser Stelle ist dann zu überlegen, ob man solche Services nicht eher in größeren Laufzeitinfrastrukturen bündelt, die mit entsprechender Verfügbarkeit und Sorgfalt betrieben werden. Dies kommt auch den Anforderungen aus Betriebssicht entgegen (siehe Kapitel 4.5). Hierzu braucht man aber modularisierte Softwarekomponenten, die sich zur Laufzeit auf den größeren Laufzeitinfrastrukturen installieren lassen. Dabei wird man dann auch entscheiden müssen, wie viel verschiedene Technologieplattformen als Laufzeitumgebung man auf Dauer im Web-UIS zulassen und damit auch warten und pflegen möchte.

5.4 Bisheriger Einsatz von Services und externen Diensten

Obwohl die Vorteile serviceorientierter Softwarearchitekturen (SOA) [77] für größere Organisationseinheiten im Businesskontext lange bekannt sind, beschränkt sich der Einsatz von Services innerhalb des Web-UIS bislang auf einige wenige Stellen. So gibt es im Umfeld des Einsatzes der openGIS-Standards im Cadenza-Umfeld oder bei den eingesetzten Fremdprodukten, wie ArcGIS Server, einige web-basierte standardisierte Serviceschnittstellen (z.B. WMS- oder WFS-Dienste), die genutzt werden können. Im Umfeld dieser Dienste entstanden auch einige weitere Hilfsdienste, mit denen man z.B. zu bestimmten Fachobjekten deren Bounding-Box berechnen kann.

Im Rahmen des SUI-Forschungsprojektes (Semantische Suche nach Umweltinformationen) [86] wurde eine generische Serviceschnittstelle zur Interaktion des Umweltportals mit anderen Fachinformationssystemen entwickelt, die prototypisch z.B. im Themenpark Umwelt für WebGenesis-basierte Webanwendungen umgesetzt wurde.

Obwohl aber im SUI-Projekt auch die Implementierung einer Serviceschnittstelle für „Cadenza Web“ angedacht war, gibt es bis auf kleinere Ansätze bis dato noch keine universell nutzbare serviceorientierte Schnittstelle zum Zugriff auf alle in einem Cadenza-System enthaltenen Fachinformationen, die solche Informationen auch in einem maschinenlesbaren Format zurückliefert.

Bei der derzeit laufenden Analyse des Einsatzes von Internetdienstleistungen für die Bereitstellung von Informationen im Internet hat sich aber bereits jetzt herausgestellt, dass gute Serviceinfrastrukturen, wie sie z.B. von Google über die Google Business Dienste und GME bereitgestellt werden, die Bereitstellung von interaktiven Datenvisualisierungen und interaktiven Kartendarstellungen erheblich vereinfachen und die Nutzererfahrung u.a. durch eine gute Performanz und eine gute Nutzerergonomie stark verbessern. Es ist daher für die Zukunft zu erwarten, dass sowohl die Nutzung externer Dienste als auch der Wunsch nach eigenen serviceorientierten Datenplattformen im Intranet zunehmen wird.

Hierzu bedarf es in Zukunft aber eines Gesamtkonzeptes, das die benötigten Funktionalitäten über die verschiedenen Fachsysteme im Gesamtkontext betrachtet und im Sinne einer serviceorientierten Architektur übergreifend definiert.

5.5 Zusammenfassung

Die Analyse der bestehenden Web-UIS Infrastruktur in Bezug auf die zukünftigen Anforderungen, wie sie in Kapitel 4 definiert sind, und dem „State of the art“ aus Kapitel 3 hat gezeigt, dass die bestehende Infrastruktur nicht mehr auf dem neuesten Stand ist und bereits jetzt den zukünftigen Anforderungen nur noch bedingt gewachsen ist. Eine Modernisierung ist daher zwingend erforderlich.

Hierbei reicht es nicht, nur bestehende Systeme (wie die WebGenesis-basierten Homepages) einfach durch neue Systeme zu ersetzen, sondern über die zukünftige Ausrichtung der Softwarelandschaft im Web-UIS Umfeld muss grundsätzlich nachgedacht werden.

Die zunehmende Serviceorientierung des Internets, der immer stärker aufkommende Dienstleistungscharakter des Internet mit Cloud-basierten Software-Dienstleistungen für jedermann und die stetig voranschreitende Optimierung der Internettechnologien hin zu sehr einfachen Möglichkeiten, Webbrowser-basierte und mobile Clients zu entwickeln, die Desktopanwendungen in Ergonomie kaum noch nachstehen aber enorme Vorteile beim serverseitigen Datenmanagement bieten, wird in Zukunft dazu führen, dass immer mehr Software als Webanwendung geschrieben wird und die Benutzer von diesen einen ähnlich hohen Bedienkomfort wie bei Desktopanwendungen erwarten können. Dies kann bei Eigenentwicklungen nur gelingen, wenn man auf moderne innovative Entwicklungsframeworks und stärkere Serviceorientierung setzt, wobei in Zukunft hier auch verstärkt externe Dienste einzubeziehen sind.

Die momentan vorhandenen Mechanismen zur Entwicklung von web-basierten Anwendungen im Web-UIS haben diesem Innovationsdruck allerdings nur wenig entgegenzusetzen. Das WebGenesis-Framework hat als Java-basiertes Entwicklungswerkzeug durchaus Potential, muss aber an einigen Stellen modernisiert werden. Dies betrifft eine Verbesserung hin zu einer mehr serviceorientierten Grundinfrastruktur, einer Verbesserung der Entwicklungsfunktionalitäten für ergonomische, AJAX-basierte Oberflächen und auf Dauer die Integration eines Konzeptes in sich gekapselter, wiederverwendbarer Erweiterungskomponenten. Mit diesen Änderungen lässt sich WebGenesis durchaus für die Fortentwicklung bereits existierender oder weiterer web-basierter Fachanwendungen einsetzen. Man sollte daher auch nicht ohne triftige Gründe bereits existierende gut laufende WebGenesis-Systeme durch andere Systeme ablösen, sondern eher an eine evolutionäre Weiterentwicklung denken.

Für bestimmte Systeme allerdings, wie die Homepage-Systeme und Portalserver, kann auch über den Einsatz anderer Systeme nachgedacht werden, da hier zurzeit vor allem Funktionalitäten benötigt werden, die in WebGenesis in näherer Zukunft nicht, aber in anderen technischen Systemen bereits vorhanden sind. Ein Wechsel von WebGenesis auf pirobase wird sich dabei aber in der Regel nicht lohnen, da die Enterprise Funktionalitäten, mit denen pirobase besticht, doch eher nur im Bereich von Systemen mit sehr großem Durchsatz an Inhalten und vielen Nutzern den Aufwand des Betriebs von pirobase lohnen. Für Standard-Webseiten sind hier PHP-basierte CMS, wie TYPO3, aber auch ein Liferay-Server, geeigneter.

6 Weiterentwicklung von WebGenesis

6.1 Entwicklungsstand

Wie in den Anforderungen in Kapitel 4 und der Analyse (Kapitel 5) ausgeführt, werden für das Web-UIS 3.0 eine Reihe von Funktionen benötigt, die WebGenesis heute nicht oder nur zum Teil bereitstellen kann. Im Kapitel Analyse wurde allerdings vom Funktionsumfang der aktuell im Einsatz befindlichen WebGenesis-Versionen im UIS ausgegangen und damit aktuelle Entwicklungen noch nicht berücksichtigt. Darüber hinaus ist bereits eine Reihe weiterer WebGenesis-Funktionen verfügbar, die in aktuell im UIS laufenden (A) oder neueren Versionen (V) oder bereits in prototypischer Form aus der Entwicklung in Projekten (P) realisiert, aber noch nicht in die Produktlinie eingeflossen sind. Als Beispiele seien hier genannt:

Ergänzungen der JEE-Basisinfrastruktur

- JAX-WS ab Version 7.5, JAX-RS ab Version 8.x

Bereitstellung von Dienst-Schnittstellen als Dienst-Anbieter:
- WebServices (A,V)
- REST (P)
- WebGenesis als SPARQL-Endpoint (V)
- Open Search (P)

*Bereitstellung von Dienst-Schnittstellen als Dienst-Nutze*r:
- WebGenesis als SPARQL-Client (V)
- Geodienste/Geodaten (P)

Bereitstellung von Social-Media-Schnittstellen als Informationsanbieter:
- Twitter-Integration (P), Wiki-Integration (V)

Bereitstellung von Dienst-Schnittstellen als Informationsnutzer:
- Anzeige von Twitter- (einschl. animierter Geo-Location), Facebook- und Google+-Streams (P)

Integration von Bildern und Bildergalerien (P)

Unterstützung für die Integration von Videoplayern (V)

Ontologie-basierte Entwicklung
- Bereitstellung effizienter Speicherungs- und Zugriffsmethoden (P)
- Visualisierung von Ontologien mit grafisch übersichtlicher Darstellung (P)
- Generierung von Eingabeformularen aus der Ontologie inkl. Hilfetexte (P)

In-Place Autorenumgebung (Bearbeitung direkt im Eintrag)
- mittels AJAX (P)

6.2 Geplanter Ausbau

Die WebGenesis-Entwicklung wird in gewohnter Weise im Kontext von Projekten weiter vorangetrieben. Als Beispiele für derzeit in der Entwicklung befindliche Ausbaumaßnahmen, die für die Entwicklung von Web-UIS 3.0 von Bedeutung sind, seien genannt:

Dienste-Schnittstellen:
Ziel des Ausbaus ist die Bereitstellung generischer Aufrufschnittstellen, die im Sinne der in Kapitel 7 definierten Serviceorientierung als wichtig definiert wurden.

Integration von Messwerten:
Integration einer HTML5-Canvas Diagrammbibliothek zur einfachen Visualisierung der Daten angebundener Datenquellen in der Autorenumgebung

Erweiterte Autorenumgebung:
Die Autorenumgebung umfasst künftig eine Kartenkomponente auf Grundlage von OpenLayers, Anzeigen für RSS- und Twitter-Feeds, Slideshows, Youtube-Video-Einbettung, frei sortierbare Datentabellen, eine verbesserte Kalenderkomponente und die Unterstützung von Mashups (Implementierung ggf. über Portlets).

Verschlankung des Grundsystems:
Das Grundsystem wird auf die benötigte Kernfunktionalität beschränkt. Eine Reihe von Funktionen wird ausgelagert und über Schnittstellen zur Nutzung und zum Ausbau bereitgestellt.

Semantische Sensor-Netzwerke:
- Adaption und Erweiterung der Semantic Sensor Network Ontology von W3C / OGC
- Semantic Registries für die Registrierung von Sensoren auf Grundlage von OGC Sensor Web Enablement (SWE)
- Ontologie-Unterstützung für OGC SWE (SensorML, SOS, SPS, WPS)
- Ontologie-Unterstützung für Linked Open Data

Risiko-/Katastrophenmanagement:
- Generische Architektur für Krisenmanagement-Szenarien einschließlich modularer und resilienter Kommunikation über eine Message Orientierte Middleware (MOM)
- OGC WMS/WFS für Visualisierung von geo-bezogenen Daten
- Elektronische Lagedarstellung aus heterogenen Datenquellen.

Ontologie-gestützte Entwicklung von Anwendungen:
- Anbindung von externen Triple Stores (persistente Speicherung von Ontologien)
- Anbindung von Reasoning Werkzeugen über OWLLink

6.3 Zukünftige Positionierung

Die in Kapitel 6.2 beispielhaft dargelegten Ausbaumaßnahmen zeigen an, dass die Weiterentwicklung von WebGenesis vor allem folgende Nutzergruppen anspricht:

Anwendungs-Entwickler:

Das bestehende Formularframework wird weiter ausgebaut. Der ontologiebasierte Entwurf von Anwendungen wird künftig eine wesentlich zügigere Entwicklung (auch komplexer) Anwendungen erlauben, die automatische Generierung von Eingabeformularen aus Ontologien wird verbessert, das JQuery-Framework ist bereits in WebGenesis integriert. Ein wesentlicher Aspekt ist die direkte Nutzung bestehender, semantischer Vokabulare zur direkten Nutzung in der Anwendung. Ein weiterer Schwerpunkt besteht im formular- und workflowgestützten Anforderungsmanagement (Methode SERVUS) und Rapid-Prototyping von komplexen Fachanwendungen.

Im *Kontext des WebUIS 3.0* positioniert sich WebGenesis in erster Linie als Plattform für die Entwicklung von Fachanwendungen. Bestehende, auf WebGenesis basierte Fachanwendungen (z.B. KFÜ/ELD, DAVE, FISGEQUA) können prinzipiell weiter bestehen, die notwendigen Schnittstellen zur Integration in die in im nächsten Kapitel vorgestellte Architektur sind größtenteils bereits vorhanden bzw. werden entwickelt. Es geht jedoch nicht ausschließlich um den Erhalt und die Pflege bereits vorhandener, sondern auch um die Entwicklung neuer Fachanwendungen auf Basis von WebGenesis. Vor allem in den Bereichen Semantische Integration, intelligente Sensor-Netzwerke, Risiko- und Katastrophenmanagement und Anforderungsmanagement besitzt WebGenesis Alleinstellungsmerkmale, die von konventionellen CMS nicht angeboten werden.

Mit WebGenesis entwickelte Fachanwendungen können als Backend-Lösung eingesetzt werden, die auch mit anderen Frontends (z.B. Liferay oder TYPO3-Systeme) für die Präsentation von Inhalten zusammen genutzt werden können. Für die Übertragung und Darstellung dieser Inhalte auf andere Plattformen bietet WebGenesis bereits heute eine Vielzahl von Schnittstellen an, im Kontext der Web-UIS 3.0 Standardisierung werden weitere dazukommen, die Umsetzung wird sich stringent an die Vorgaben der in Kapitel 7 vorgestellten Architektur halten.

Autoren:

Die Autorenumgebung wird künftig wesentlich komfortabler ausgestattet und modernisiert.

Designer für Web-Auftritte:

Die flexiblen Möglichkeiten zur Gestaltung von Layouts werden weiterentwickelt, die Einbindung medialer Inhalte wird verbessert. Die Unterstützung von Portlets wird geprüft. Allerdings werden nicht bedenkenlos beliebige Funktionalitäten verschiedener Portal- und CMS-Systeme wie Liferay oder TYPO3 in WebGenesis nachgebildet.

7. Lösungsvorschlag für Web-UIS 3.0

Wie bereits in der Einleitung beschrieben, wurden für das UIS-Baden-Württemberg eine Reihe von Web-basierten Informationssystemen entwickelt, die Umweltinformationen, welche von den Umweltbehörden unter Nutzung von internen Fachanwendungen erfasst werden, über das Internet für den Bürger und externe Fachleute bereitstellen.

In den Anfängen des Web-UIS entstanden dabei nicht nur dedizierte Websysteme für die Öffentlichkeit, wie die Homepages der einzelnen Fachbehörden und einige spezielle Web-basierte Informationssysteme für Fachleute, wie FADO, oder der Themenpark Umwelt als Informationssystem für die allgemeine Öffentlichkeit, sondern im Rahmen einer Reihe von Projekten auch ganz spezielle Web-Fachanwendungen für die interne Nutzung, wie z.B. das Sachdatensystem.

Dies führte zu einer stark fragmentierten Weblandschaft innerhalb des UIS, die in der Konsequenz eher zu einer Verschlechterung der Auffindbarkeit von Informationen für den Nutzer führte, da dieser die Informationen in vielen verschiedenen Systemen suchen musste. Aus diesem Grunde wurden für das UIS zentrale Webportale entwickelt, die die Informationen nachgeschalteter Informationssysteme bündeln und über eine Nutzeroberfläche und zentrale Suche an einem Platz verfügbar machen. Zur Anbindung der nachgeschalteten Informationssysteme wurden dabei im SUI-Projekt erste Ansätze entwickelt, diese über serviceorientierte Schnittstellen besser mit den zentralen Portalen zu integrieren.

Dies sind erste wichtige Schritte auf dem Weg zu einer langfristig zukunftsträchtigen IT-Infrastruktur für das Web-UIS. Hierzu muss allerdings eine konsequente Weiterentwicklung des Web-UIS hin zu einer lose gekoppelten serviceorientierten, Web-basierten Dienstleistungsinfrastruktur mit dazu gehörigen Web- und mobilen Anwendungen erfolgen, bei denen nicht nur rein informationelle Angebote für die Öffentlichkeit über Services und Webanwendungen bereitgestellt werden, sondern zentrale Fachanwendungslogik im Sinne einer serviceorientierte Softwareinfrastruktur als Dienstleistung universell verfügbar gemacht wird, die dann in den verschiedenen Web-basierten und mobilen Anwendungen genutzt werden können. Dies reduziert nicht nur die Redundanz solcher Fachlogik in den verschiedenen bis dato entwickelten Anwendungen und damit auf Dauer auch die Entwicklungskosten, sondern ermöglicht auch die einfache Wiederverwendung solcher Anwendungslogik in ganz unterschiedlichen Frontendanwendungen, wie mobilen Apps oder Webanwendungen in unterschiedlichen Websystemen, wie CMS, Portalen oder Fachanwendungen. Die konsequente Nutzung externer Dienste an Stellen, wo es Sinn macht, reduziert dabei ebenfalls wesentlich Entwicklungskosten, spart Entwicklungszeit und bringt andere Vorteile, wie eine permanente Verfügbarkeit von Datendiensten auf dem Internet, ohne dass die eigene Rechnerinfrastruktur ständig zur Verfügung stehen muss.

Der Vorschlag für das Web-UIS 3.0 in diesem Konzeptpapier setzt daher auf eine auf Dauer durchgehende serviceorientierte Infrastruktur für das Web-UIS 3.0.

7.1 Grundlegende Architektur

An Stelle einer Vielzahl von nicht miteinander verknüpften Webanwendungen sollte für das Web-UIS 3.0 auf Dauer eine serviceorientierte Dienstleistungsinfrastruktur von lose koppelbaren Grunddiensten und internen web-basierten Fachanwendungen mit Dienst-orientierten Schnittstellen sowie von Frontend-Webanwendungen, wie CMS und Portalen, die integrierte Zugänge zu Informationen und fachlichen Anwendungen für bestimmte Zielgruppen bereitstellen, aufgebaut werden (siehe Abbildung 7.1).

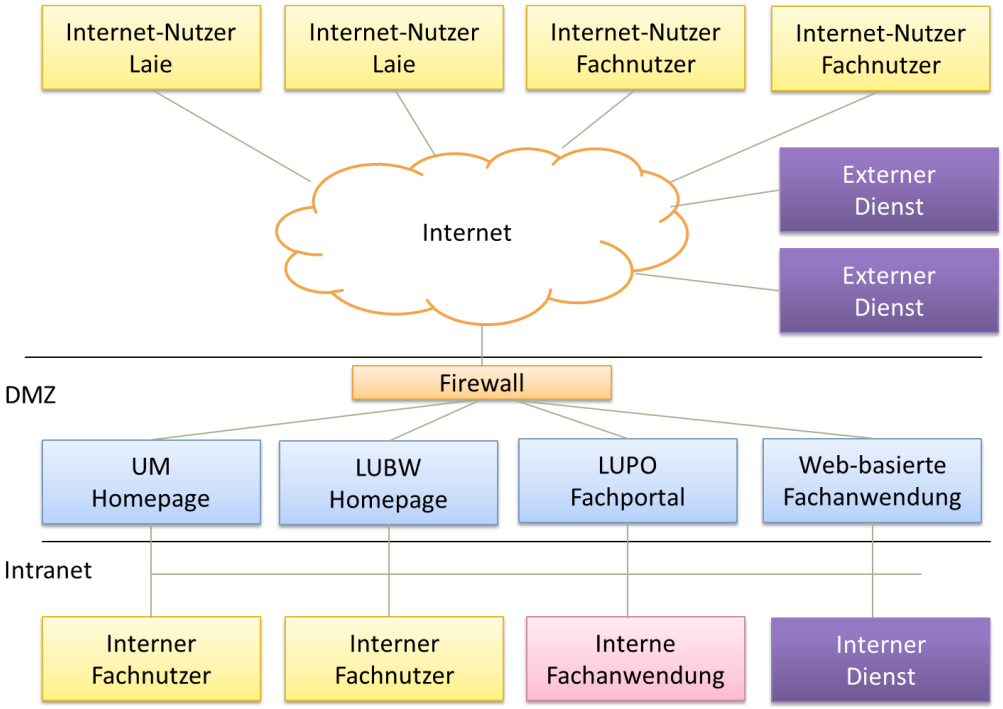


Abb. 7.1: Zusammenspiel einzelner Systeme im zukünftigen Web-UIS

Hierbei ist die gesamte so entstehende Softwareinfrastruktur durchaus *als logische Einheit zu sehen, für die ein Gesamtkonzept aufeinander abgestimmter Dienste und diese nutzender Anwendungen entworfen werden muss, bei der Redundanz weitestgehend vermieden wird*.

Die als serviceorientierte Basisinfrastruktur bereitgestellten Behörden-internen Grunddienste und / oder zugehörigen Web-Fachanwendungen könnten dabei Fachanwendern innerhalb der Behörden als zentrale Autorenwerkzeuge dienen, mit denen zu den Diensten gehörende Daten erfasst, geändert und verwaltet werden. Die Daten und zugehörige Fachlogik werden dann anderen Fachanwendungen wiederum über Dienstschnittstellen zur Verfügung gestellt. Typische Vertreter solcher Anwendungen sind interne Geodatendienste zur Erfassung von Daten mit geographischen Bezug, Sachdatensysteme zur strukturierten Speicherung von Sachinformationen zu Umweltobjekten, aber auch eine interne Bilddatenbankanwendung zum Management aller Bilder oder ein entsprechendes Dokumentenmanagementsystem zum Management von Dokumenten. Auch Anwendungen zur Speicherung und Bereitstellung von Messdaten mit zugehörigen Informationen über Messstellen, Mess-Equipment und Messgrößen sowie Diensten für eine Aggregierung und Interpretation der Messwerte wären gute Kandidaten für solche serviceorientierten Systeme. Über serviceorientierte Schnittstellen können diese internen Datenerfassungsanwendungen und Dienste anderen Anwendun-

gen im UIS Daten zu Umweltobjekten bereitstellen, dazu geographische Informationen über deren Lage oder Dienstleistungen wie die Bestimmung des Mittelpunktes oder die Berechnung eines zugehörigen Höhenprofils. Andere Dienste können Bilder passend zum Umweltobjekt zurückgeben.

Bereits jetzt existieren im UIS eine Reihe solcher Web-basierten Fachanwendungen, z.B. das Sachdatensystem auf Basis von Cadenza Web, INSPIRE konforme OGC-Dienste (z.B. WMS) oder weitere serviceorientierte Dienstleistungsanwendungen (z.T. prototypisch implementiert und „handgestrickt“), wobei ein Gesamtkonzept hierfür allerdings noch fehlt und die gesamte benötigte Funktionalität nur zu einem kleinen Teil serviceorientiert erschlossen ist. Hier ist auf Dauer eine durchgehende Gesamtkonzeption notwendig und sinnvoll.

Die internen Services lassen sich dann an Stellen, wo dies sinnvoll ist, durch externe über das Internet bereitgestellte Dienstleistungsangebote oder durch kommerzielle hinzukaufbare Produkte, die im Intranet bereitgestellt werden können, ergänzen (wie z.B. die GSA). Dies ist sinnvoll, wenn im Internet ein Service für eine gewisse Funktionalität oder ein kommerzielles Produkt bereits existiert, das solche Services bereitstellt, deren Nachbau im Intranet zu aufwändig wäre oder wenn der Betrieb eines solchen Services innerhalb der eigenen Rechnerinfrastruktur mit gewünschter Quality of Service nicht möglich ist. Beim Einsatz solcher externen Dienste oder Third Party Produkte sind allerdings die Geschäftsbedingungen und eventuelle Nutzungskosten genauestens zu prüfen.

Einige wenige oberhalb der Grunddienste und Fachanwendungen angesiedelte Webanwendungen für den Nutzerzugang können dann die bereitgestellten Grunddienste nutzen, um z.B. Daten, Bilder oder Dokumente aus den grundlegenden Fachinformationssystemen zu extrahieren und in einem bestimmten Informationskontext für einen bestimmten Benutzer oder eine bestimmte Benutzergruppe personalisiert zur Verfügung zu stellen. Zur aggregierten Bereitstellung von Informationen und kleineren interaktiven Fachanwendungen eignen sich dabei besonders Portalserver oder Content-Management-Systeme. Diese können dabei Informationen und Dienstleistungen sowohl im Intranet für Mitarbeiter als auch im Internet für bestimmte Fachnutzergruppen (Mitarbeiter oder externe Fachleute) oder auch der allgemeinen Öffentlichkeit bereitstellen. Diese Systeme müssen gegebenenfalls durch spezialisierte Fachanwendungssysteme, wie das KFÜ, ergänzt werden.

Portalserver können dabei alle wesentlichen Daten und Funktionalitäten, die Nutzer benötigen, auch wiederum selbst als Services bereitstellen. Diese Services können dann externe Anwendungen und mobile Applikationen nutzen, um ihrerseits diese Daten und Funktionalitäten Anwendern bereitzustellen. Auf diese Weise können von Anwendern nicht nur Webbrowser, sondern auch native Anwendungen zum Zugang zu Daten verwendet werden.

Das Zusammenspiel der einzelnen Systeme wurde in Abbildung 7.1 bereits verdeutlicht. Nutzer nutzen dabei über einige wenige dedizierte Frontend-Webanwendungen (in blau dargestellt) Informationen entweder über den Webbrowser oder indirekt unter Nutzung von dedizierten Anwendungen (z.B. Apps) mit Hilfe von bereitgestellten Serviceschnittstellen. Spezielle Systeme, wie die Homepage der UM oder der LUBW, stellen Öffentlichkeits-nahe Informationen, wie Presseartikel, Pressmedienberichte, Flyer oder auch Terminankündigungen bereit, während sich andere auf das Auffinden und die Rückgabe von Fachinformationen konzentrieren (UIS-Portal). Die Fachportale sind dabei auch für externe Anwendungen zu-

gängliche Systeme, die Fachanwendungslogik und Daten über Services für externe Anwendungen bereitstellen.

Beide Kategorien von Frontendsystemen können sowohl extern auf dem Internet als auch intern im Intranet der Umweltbehörden befindliche Servicesysteme oder Fachsysteme mit Serviceschnittstellen als zusätzliche Informations- und Datenlieferanten nutzen.

Um die Bereitstellung eigener Daten für die Internet-basierten Nutzer möglichst von der internen Datenbereitstellung für Fachnutzer zu entkoppeln, können Internet-öffentliche Daten bei Bedarf nach einem definierten Verfahren von internen Datensystemen auf öffentliche Services exportiert werden. Solche externen Datendienste garantieren eine hohe technische Verfügbarkeit und einen performanten Zugriff auf diese Daten unabhängig von der eigenen Rechnerinfrastruktur. Änderungen an den internen Daten können dabei gemacht werden, ohne dass sie gleich im Internet reflektiert werden. Durch Qualitätskontrollmechanismen kann dabei auch sichergestellt werden, dass die ins Internet exportierten Daten eine gewisse vorgegebene Qualität besitzen. Gibt man diese Internet-öffentlichen Daten direkt in den externen Services als Daten frei, sind sie als „Open Data“ direkt über die Serviceschnittstellen des externen Dienstes für jedermann nutzbar.

7.2 Systeme für den Nutzerzugang

Die Informationen, die über die Portale und CMS als Systeme für den allgemeinen Nutzerzugang des UIS bereitgestellt werden, sind von unterschiedlichster Natur. So müssen auf der Homepage des UMs sehr viele aktuelle Informationen mit politik-nahem Inhalt zeitnah veröffentlicht werden. Im Zeitalter des modernen Internets sind diese Informationen zudem hochmedial. Neben reinen Presseveröffentlichungen sind auch Interviews mit dem Minister als Videos oder Podcasts als Audiointerview bereitzustellen (Social Media, siehe Abbildung 7.2 auf der nächsten Seite). Dazu wünschen sich die ÖA-Abteilungen Möglichkeiten, Informationen über soziale Netzwerke, wie Twitter zu verbreiten, und diese dann auch in ihre eigene Homepage einzubinden (Anbindung an soziale Netzwerke). Für diese Art der Informationsbereitstellung eignen sich vor allem Content-Management-Systeme, die die oben beschriebenen Anforderungen bereits durch vorhandene funktionale Bausteine abdecken. Diese Voraussetzung erfüllen insbesondere PHP-basierte CMS, wie TYPO3, Joomla oder Drupal.

Dies ist einer der Gründe, warum das neue Landesportal Baden-Württemberg auf das Content-Management-System TYPO3 setzt, da hier die gewünschten Funktionalitäten bereits vorhanden sind. Die bisher verwendeten WebGenesis-Systeme im Web-UIS, die schwerpunktmäßig eher auf Entwicklung komplexer Anwendungen denn auf Bereitstellung von Standard-CMS-Funktionalitäten ausgerichtet sind, erfüllen diese Anforderungen zur Zeit nur zum Teil, da einige der Standardbausteine gängiger CMS hier noch fehlen. Es finden aber auch Entwicklungsarbeiten bei WebGenesis statt, die einen Teil dieser Funktionalitäten in Zukunft bereitstellen könnten (siehe Kapitel 6).

Neben den Öffentlichkeitsarbeits-nahen Informationen müssen über Umweltbehörden aber auch Fachinformationen, die teilweise direkt aus Fachinformationssystemen stammen, z.B. Messdaten-basierte Informationen, wie aktuelle Pegelstände von Flüssen, die Ozonwerte im Sommer oder Informationen über Naturschutzgebiete oder Verordnungsrichtlinien für die Öffentlichkeit bereitgestellt werden (Umweltinformationsgesetz). Die Einbindung solcher In-

formationen in ein CMS lässt sich noch einigermaßen einfach bewerkstelligen, sofern eine serviceorientierte Webanwendung oder ein Portal im Hintergrund für dieses Integrationsszenario einen entsprechenden Datenservice und ein zugehöriges Web-Widget bereitstellt, das z.B. über ein <iframe>-Tag in eine Webseite eines CMS integriert werden kann (siehe auch 4.2.3).

Abb. 7.2: Seite des neuen Landesportals auf Basis von TYPO3 mit integriertem YouTube-Videoplayer (eine Standardkomponente von TYPO3)

Schwieriger wird es allerdings, wenn komplexere interaktive Fachanwendungen oder sogar personalisierbare Web-Desktops mit verschiedenen Fachanwendungen für unterschiedliche Nutzergruppen bereitgestellt werden sollen. Hierzu benötigt man Webumgebungen, die einerseits als standardisierte Entwicklungsumgebungen für eigene Webanwendungen und andererseits als integrative Laufzeitplattform für solche Anwendungen dienen können. Diese Rolle erfüllt das momentan eingesetzte WebGenesis System bereits zu einem gewissen Teil aber darüber hinaus ist dies vor allem das Anwendungsgebiet von (Enterprise) Portalservern, wie Liferay (vgl. Abb. 7.3). Enterprise Portalserver erlauben es, größere Webanwendungen als Bausteine (Portlets) zu implementieren, die modular in ein Portal integriert und dann personalisiert für unterschiedliche Nutzergruppen bereitgestellt werden können (siehe auch 3.4). WebGenesis bietet zurzeit nicht das Hot Deployment von größeren Anwendungs-

komponenten, eignet sich aber als Entwicklungsumgebung durchaus gut zur Integration größerer Fachfunktionalitäten in Web-basierte Fachanwendungen.

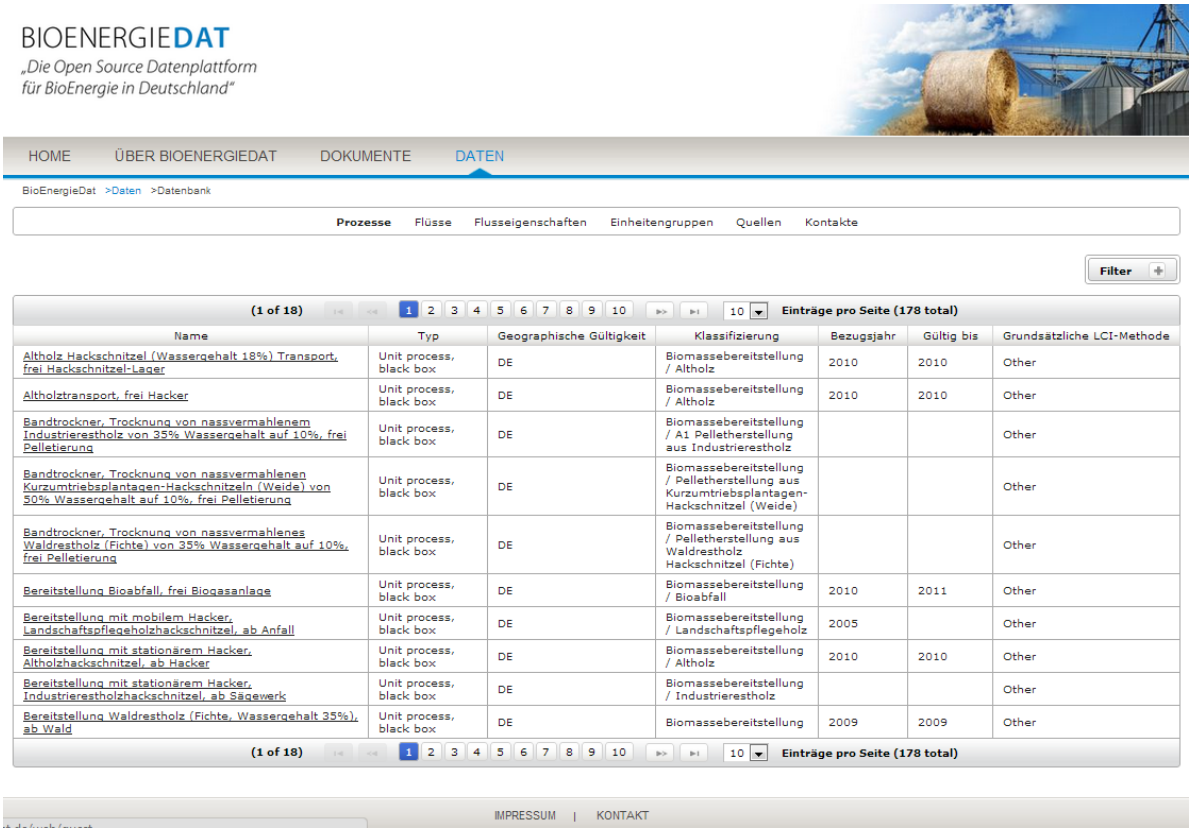


Abb. 7.3: Liferay-Server des BioEnergieDat-Portals mit Portlet zur Anzeige von Ökobilanzierungsdatensätzen. Die angezeigte Tabelle ist eine vollständig interaktive, komplexe Anwendung, die ein Blättern, Sortieren und Filtern der Datensätze dynamisch im Browser ohne Neuladen der Webseite ermöglicht.

Da seit langem im UIS-Umfeld Java als Standardentwicklungssprache für Fachlogik im Einsatz ist, ermöglichen Java-basierte Portalserver oder Entwicklungsplattformen, wie WebGenesis, große Teile bereits vorhandener Implementierungen wiederzuverwenden. Schon aus Gründen der Rückwärtskompatibilität ist daher ihr Einsatz in Bereichen, wo Fachlogik eine große Rolle spielt, sinnvoll.

Eine Migration bereits existierender Fachanwendungslogik, die z.B. für WebGenesis oder andere Java-basierte Fachanwendungen bereits implementiert wurde, in CMS, wie TYPO3, ist ebenfalls weder vom Aufwand noch aus technischer Sicht sinnvoll.

Daher ist es die Empfehlung dieser Studie, die Systemlandschaft der Websysteme für den allgemeinen Nutzerzugang im Umfeld des UIS für die Zukunft so flexibel zu gestalten, dass eine sinnvolle Aufteilung der Funktionalitäten zwischen verschiedenen Systemen vorgenommen werden kann. Dabei kann für die Öffentlichkeits-nahen Homepages, wie die des UM, hierfür z.B. durchaus ein spezialisiertes CMS, wie das landesweit-einsetzbare TYPO3-System für die Verbreitung von Presseveröffentlichungen und Öffentlichkeits-nahen Social Mediainformationen, eingesetzt werden, während für Fachinformations-nahe Portale und Homepages, wie die Homepage der LUBW und das Umweltportal Baden-Württemberg weiter das WebGenesis-Framework oder alternativ auch ein Java-basierter Portalserver, wie

Liferay, als technische Plattform eingesetzt werden kann. Hierdurch können einerseits existierende Investitionen in bereits realisierte Java-basierte Fach-Anwendungsentwicklungen für diese Portale auch in Zukunft weitergenutzt werden. Andererseits lässt sich durch die Integration neuer Systeme, wo nötig, die Modernisierung der Web-UIS Infrastruktur beschleunigen.

Dabei sollte man nicht ohne Not bereits existierende und gut funktionierende fachliche Informationssysteme auf völlig neue Infrastrukturen umstellen wollen, sondern eher auf eine evolutionäre Entwicklung setzen, bei der gut funktionierende, z.B. mit WebGenesis erstellte fach-nahe Webanwendungen weiter mit WebGenesis als Plattform betrieben und evolutionär weiterentwickelt werden können, während man ausgewählte Systeme, wie die Homepage-Systeme, bei denen zur Zeit essentielle CMS-Funktionalitäten bei WebGenesis vermisst werden, auf andere Plattformen migriert.

7.3 Empfehlungen zu den Entwicklungsplattformen für Webanwendungen und Dienste

Eine lose koppelbare, serviceorientierte Infrastruktur von Webanwendungen und Webdiensten ermöglicht es prinzipiell, dass die Entwicklung von Diensten und Anwendungen mit einer Vielzahl an Programmiertechnologien in fast jeder beliebigen Programmiersprache erfolgen kann. Insbesondere die Verwendung REST-basierter Kommunikationsschnittstellen [57] reduziert die Voraussetzungen zur Interaktion mit anderen Diensten auf das Vorhandensein der BasisInternettechnologien, wie HTTP, XML und/oder JSON. Diese Voraussetzungen sind u.a. für Java-basierte als auch .NET-basierte Entwicklungsumgebungen, aber auch für andere Programmiersprachen verfügbar.

Trotzdem wird man sich aus Betriebssicht und zum besseren Know-how Transfer auf einige wenige Entwicklungsplattformen und damit auch Laufzeitplattformen beschränken wollen. In Bezug auf den Betrieb sind dabei unter anderem die Wünsche und Anforderungen aus Sicht des Betriebes zu berücksichtigen (siehe Kapitel 4.5). Diese fordern, dass die Laufzeitumgebungen der zu betreibenden Dienste und Webanwendungen auf standardisierten virtualisierten Rechnerumgebungen mit Standardsoftware, wie Oracle oder MySQL-Datenbank, lauffähig und ihrerseits ausfallsicher, sicher und skalierbar sein sollten. Außerdem sollte man sich auf einige wenige Laufzeitplattformen beschränken, da sonst der Lernaufwand des Betriebspersonals zu groß wäre. Dies schließt eine allzu heterogene Entwicklungslandschaft aus.

Für die Entwicklung dedizierter Web-Fachanwendungen oder komplexer Dienste sollte weiter Java als Standardimplementierungssprache verwendet werden, da Java nach wie vor die am meisten genutzte Entwicklungssprache für komplexe serverseitige Anwendungen ist und hiermit wie bereits beschrieben, vor allem auch Investitionen der Vergangenheit auch in Zukunft genutzt werden können. Außerdem sind Java-basierte Infrastrukturen für Webanwendungen bereits im Einsatz und haben sich auch im Betrieb bewährt.

Dabei sollte sich die Java-basierte Entwicklungsplattform möglichst dicht an den JEE-Standard [66] halten, um Unsicherheiten bzgl. der zukünftigen Weiterentwicklung der Grundtechnologien zu vermeiden.

Die ideale Laufzeitplattform ist dabei für Web-basierte Dienste und –Anwendungen ein Web-Applikationsserver, der das Web-Profil des Java EE 6 oder in Zukunft EE 7 Standards ergänzt durch die Webservice-Standards JAX-RS und JAX-WS implementiert. Hierzu kann ein Apache Tomcat-Server mit entsprechenden Erweiterungen oder ein anderer JEE-konformer Web-Applikationsserver, wie Glassfish, genutzt werden.

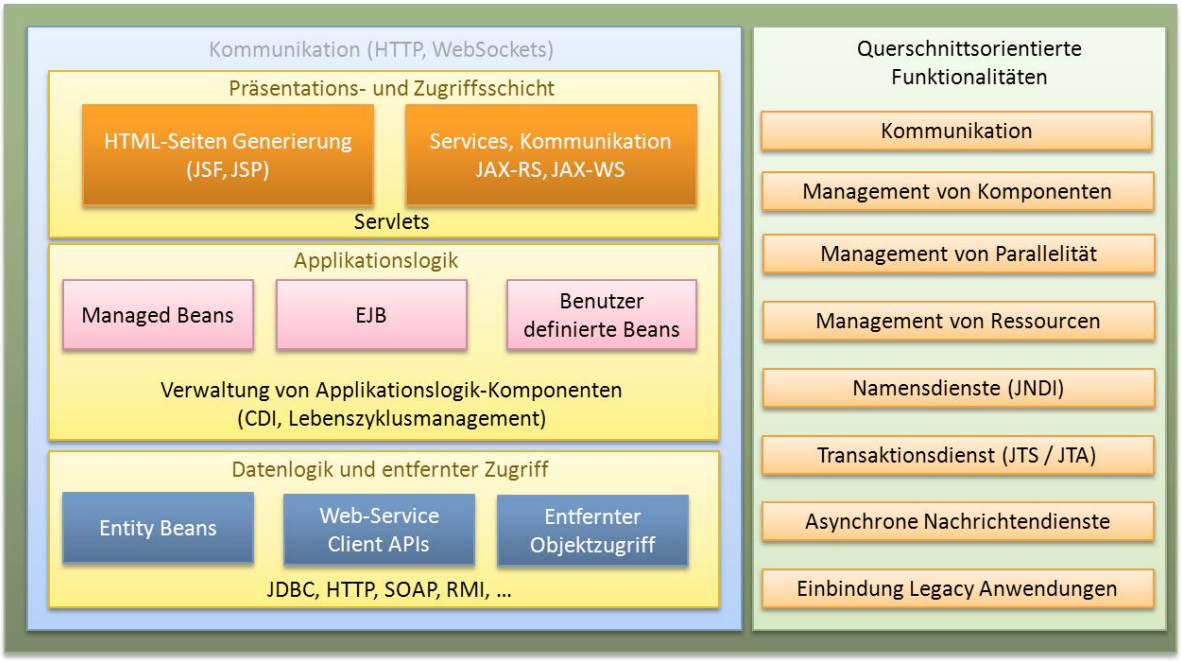


Abb. 7.4: Die Funktionalitäten einer JEE-Entwicklungsumgebung in der Übersicht

Abb. 7.4. stellt die Funktionalitäten einer JEE-Entwicklungstabelle schematisch dar. Die folgende Tabelle 7.1. listet im Einzelnen auf, welche JEE-Technologien Grundbestandteil der auf dem JEE-Standard basierenden Entwicklungsumgebung und Laufzeitplattform sein sollten. Man beachte, dass dabei nicht alle Elemente der vollen JEE-Umgebung genannt werden, sondern eher auf die Technologien gesetzt wird, die zum Web-Profil des JEE-Standards gehören [87]. Es fehlen deshalb im Wesentlichen die Elemente, die mit asynchronen Nachrichtendiensten und Einbindung von Legacy-Anwendungen zu tun haben. Außerdem enthält das Java Web-Profil von der Enterprise Java Beans (EJB)-Spezifikation nur die EJB-Lite Variante, die EJBs mit lokalen, nicht aber entfernten Schnittstellen unterstützt. Die untenstehende Tabelle ergänzt das Web-Profil aber um die für Dienste notwendigen Webservice-Technologien und um JavaScript- und / oder JSF-basierte UI-Komponentenbibliotheken zur komfortableren Programmierung ergonomischer Benutzeroberflächen.

Technologie	**Kurze Beschreibung**
JEE 6 bzw. zukünftig JEE 7 konforme Grundinfrastruktur	Zumindest Web-Container mit Unterstützung aktueller Standards für Servlets, JSP, EL, CDI, Unterstützung von WebSockets und asynchronen HTTP
EJB-Lite	Container-verwaltete Businesskomponenten mit lokalen Interfaces, keine Unterstützung für asynchronen Nachrichtenverkehr

JAX-RS	REST-basierte Services; z.B. durch Integration der Referenz-implementierung Jersey
JAX-WS	SOAP-basierte Services; z.B. durch Integration der Referenz-implementierung Metro-Framework
JPA (Java Persistence API)	Objekt-relationales Mapping zum Zugriff auf Datenbanken; Referenzimplementierung EclipseLink
JSF 2.x	Framework zur Erstellung von Weboberflächen; Referenzimplementierung Mojarra-Framework
Ergänzende Widget und Javascript-Bibliotheken	z.B. primefaces JSF-Komponenten, jquery, jquery-UI, primefaces UI

Tabelle 7.1: Zusammenstellung wünschenswerter JEE-Funktionalitäten für die Entwicklung

Die oben beschriebene Laufzeitumgebung kann sowohl als Grundplattform für WebGenesis wie auch für Java-basierte Enterprise Portalserver, wie Liferay, verwendet werden. Auch Cadenza Web lässt sich als anderes wichtiges Beispiel für eine Java-basierte Entwicklungsumgebung auf einer solchen Laufzeitplattform betreiben.

Die beschriebenen Systeme können dann oberhalb der Grundinfrastruktur weitergehende Funktionalitäten zur Dienste- und Anwendungsprogrammierung bieten. Aus Sicht des Betriebs und der Administration der Laufzeitumgebungen wäre es dabei vorteilhaft, wenn die über der Grundinfrastruktur betriebenen Anwendungsumgebungen auf Dauer eine flexible Erweiterbarkeit zur Laufzeit ermöglichen würden.

7.4 Flexible Erweiterbarkeit durch Komponentenkonzept

Ein Konzept für Anwendungskomponenten setzt bzgl. der GUI-Bestandteile, Services und Businesslogik auf die durch die Basisplattform gegebenen Bestandteile (Widgets, Services, etc.) auf und sollte eine schnelle Entwicklung dedizierter Anwendungskomponenten ermöglichen. Die Bestandteile einer Komponente können in Systemen, die ein solches Komponentenkonzept unterstützen, in spezielle Archive verpackt werden, die dann über eine Administrationsoberfläche in ein bestehendes System geladen und automatisch installiert werden. Das System lässt sich damit über solche Komponenten erweitern, ohne dass das System neu gestartet werden muss. Dies erleichtert Betrieb und Wartung der eingesetzten Laufzeitumgebungen.

Solche Konzepte für Anwendungskomponenten gibt es bereits in vielen Systemen, wie bereits im Kapitel „State of the Art“ (3.) gezeigt wurde. So lassen sich sowohl PHP-basierte Content-Management-Systeme, wie TYPO3, Drupal oder Joomla, als auch Java-basierte Portalserver oder soziale Plattformen über solche Komponenten erweitern.

Der Java JEE-Standard unterstützt die Programmierung solcher Erweiterungen explizit in Form der automatischen Installation von eigenständigen JEE-Businesskomponenten und/oder eigenständigen Webapplikationen über das EJB-Komponentenformat (ear-Datei) oder ein Webanwendungsarchiv (war-Datei). Eine Sonderform einer solchen installierbaren

Webanwendung sind im Fall von Java-basierten Portalservern die Portlets. Portlets [75] sind Webanwendungen, die sich über die Administrationsoberfläche eines Portalservers in einen Portalserver installieren lassen. Sie bilden damit die wiederverwendbaren Komponenten von Portalen, die sich per Drag und Drop auf beliebigen Webseiten des Servers platzieren lassen.

Im Rahmen dieser Konzeptstudie für das Web-UIS 3.0 wird empfohlen, bei der Java-Webanwendungsentwicklung zukünftig verstärkt auf eine durch die JEE Standards genormte Komponenten-orientierte Entwicklung zu setzen, bei der sich komplette Anwendungen und funktionale Erweiterungen modular in den Laufzeitumgebungen installieren lassen. Dies bringt viele Vorteile: Komponenten können unter allen Laufzeitplattformen im laufenden Betrieb ausgetauscht werden und bilden in sich geschlossene, wiederverwendbare Softwarebausteine. Größere Anwendungen lassen sich aus ihnen modular zusammensetzen. Komponenten können dabei Webanwendungen darstellen, reine funktionale Erweiterungen liefern, die nur über Serviceschnittstellen abrufbar sind, oder beides miteinander kombinieren.

7.5 Grundinfrastruktur von Diensten

Wesentliche Bestandteile einer serviceorientierten Web-UIS Infrastruktur sind natürlich die grundlegenden Services. In diesem Kapitel sollen einige mögliche sinnvolle Services vorgestellt werden. Es liegt aber gerade im Wesen einer lose gekoppelten serviceorientierten Softwarelandschaft, dass diese Dienstlandschaft sehr dynamisch ist und von der Erweiterung, Modifizierung, Modularisierung und auch Deaktivierung veralteter Services lebt.

Das wichtigste Element einer serviceorientierten Web-UIS Infrastruktur ist die Kapselung von Funktionalität in abgeschlossenen, eigenständig benutzbaren und installierbaren Softwarebausteinen (Servicekomponenten), die über Serviceschnittstellen erreichbar und von anderer Software nutzbar sind. Dabei sollten die Kommunikationsschnittstellen der Servicekomponenten Hersteller-, Betriebssystem- und sprach-neutral sein, so dass sie über Technologiegrenzen hinweg nutzbar sind. Hierzu eignen sich vor allem REST- bzw. SOAP-basierte Webservices. Im Web-UIS Umfeld ist dabei REST zu bevorzugen, weil REST-Schnittstellen bessere Unterstützung für JavaScript-basierte Webtechnologien und mobile Clients bieten [57].

Die Entwicklung dedizierter Serviceschnittstellen für spezielle Fachanwendungen kann sehr aufwändig sein. Sie lohnt sich nur da, wo Fachanwendungen Informationen in strukturierter Form so verwalten, dass sich die Fachinformationen auf einfache Art und Weise auf Serviceformate abbilden lassen. Im Rahmen des SUI-Projektes wurde hier eine 80-20 % Regel definiert, so dass sich der Aufwand zur Entwicklung dedizierter Schnittstellen nur für einen kleinen Teil von Systemen lohnt, für den Rest der Systeme dagegen mit Standardmechanismen und Schnittstellen gearbeitet werden sollte. Eine solche Standardschnittstelle wurde im Projekt SUI entwickelt und kann an vielen Stellen im Web-UIS universell genutzt werden.

7.5.1 Dienstübergreifende, generische Schnittstellen

Die im Rahmen des SUI-Projektes entwickelte universelle Serviceschnittstelle zum Abruf von Informationen basiert auf Feed-Formaten (Atom-Feed), deren Erweiterbarkeit und dem

OpenSearch-Standard [91] bzgl. der Beschreibung von Such- und Filterkriterien. Diese Schnittstelle, die ursprünglich als universelle Schnittstelle zwischen dem LUPO-Portal und anderen Fachanwendungen im UIS geplant war, eignet sich generell dazu, die ersten zwei Abfrageebenen zur Abfrage von Informationen über Informationsobjekte in Systemen generisch zu bedienen.

Hierbei macht man sich die Tatsache zu Nutze, dass für nahezu jede beliebige Informationsressource (sei es Naturschutzgebiet, Messstelle, Fluss, Naturdenkmal) zunächst einmal Abfragen der Gestalt „Gebe mir eine Liste aller Objekte der vorgegebenen Art, die gewissen Filterkriterien genügen“ (Ebene 1) sowie „Gebe mir eine Übersicht des Objektes einer vorgegebenen Art, dessen Objekt-ID ich dir übergebe“ (Ebene 2) von einem Service realisiert werden müssen.

Die Darstellung eines Objektes in der Listendarstellung bzw. in der Übersicht kann sich dabei im Web zunächst auf einige wesentliche Beschreibungsattribute, wie Name/Titel, Kurzbeschreibung, evtl. Autor, Link auf weitergehende Beschreibung, reduzieren. Dies ist aber gerade mit dem Atom-Feed-Format als Rückgabeformat möglich. Das Atom-Feed-Format erlaubt aber auch durch Einbinden fremder XML-Namespaces die zurückgegebenen Metadaten um beliebige weitere Informationen zu erweitern. So lassen sich gegebenenfalls die zurückgegebenen Metadaten eines Objektes um Medienelemente (Bild, Audio, Video, also durch MediaRSS-Elemente) medial oder durch Hinzufügen von Ortsinformationen (GeoRSS) mit einem Raumbezug ergänzen.

Der OpenSearch-Standard wird bei der generischen SUI-Serviceschnittstelle für die Beschreibung von Filterkriterien auf der Aufrufzeile des Services bzw. innerhalb des Rückgabeformates zur Beschreibung von auf die Suche bezogener Metadaten, wie Anzahl der Treffer oder Informationen zum Blättern innerhalb der möglichen Trefferliste, verwendet. Im Rahmen von SUI wurde eine Reihe von Standardfiltern semantisch vordefiniert, die das Filtern von Ergebnislisten nach Volltextsuchbegriffen, räumlichen und zeitlichen Kriterien und nach Themenbezug vordefinieren. Der Themenbezug bezieht sich dabei auf eine durch die SUI-Ontologie festgelegte Kategorisierung von Umweltobjekten.

Das Rückgabeformat der generischen Serviceschnittstelle wurde sehr flexibel gehalten. Eine anfragende Software kann die Rückgabe als Atom-Feed, RSS-Feed oder aber in einem anderen Anwendungsformat verlangen, je nachdem, was der angefragte Service im Stande ist zu liefern. Für raumbezogene Daten bieten sich dabei auch Formate, wie GeoRSS, KML, etc. und auch Formate wie WMS, für Image-basierte Kartendienste, an. Kalender und Terminereignisse lassen sich z.B. über ICAL-ähnliche Formate als eigene Kalenderfeeds liefern oder in andere Feeds einbetten.

Daher lassen sich viele Anwendungsfälle (die 80% gemäß o.g. Regel) tatsächlich mit solchen Standard-Formaten als Rückgabe abdecken. Andere Systeme können eine Kombination aus Atom-Feed und dedizierteren Formaten (z.B. auf JSON-Basis) implementieren, die im Sinne von REST-Services die abgefragten Objekte vollständig erschließen.

Im dieser Studie wird daher empfohlen, am Web-UIS 3.0 beteiligte Systeme auf die Dauer durchgehend mit serviceorientierten Schnittstellen auszustatten, die die generische im Rahmen von SUI definierte Serviceschnittstelle auf Basis von OpenSearch und Feeds als ein-

fachste Version enthalten, bei Bedarf aber erweitern und so auch detailliertere Informationsstrukturen von Umweltobjekten über die Serviceschnittstelle abrufbar machen.

Auf diese Weise lassen sich Informationen angeschlossener Fachsysteme sowohl mit gängigen vorhandenen Softwarekomponenten, wie Feed-Readern standardmäßig erschließen als auch über spezielle Lesekomponenten detaillierter nutzen. Des Weiteren werden alle Fachsysteme, die diese Schnittstellen benutzen, automatisch für das Umweltportal LUPO und dessen mobile Variante LUPO-mobil nutzbar.

Im Rahmen des SUI-Projektes wurde für den Themenpark Umwelt und die WebGenesis-basierten Systeme bereits eine prototypische Implementierung der generischen Suchschnittstelle implementiert, so dass sich prinzipiell solche Systeme bereits jetzt als serviceorientierte Systeme in das Web-UIS 3.0 integrieren lassen. Diese Schnittstellen lassen sich ebenfalls recht einfach in Portalservern, wie Liferay implementieren, wobei Liferay bereits von Grund auf durchgehend serviceorientiert implementiert ist, so dass sich in jedem Liferay-Server bereits jetzt fast alle Inhaltselemente und Funktionen über detaillierte Services erschließen lassen.

Eine weitere sehr einfache, standardisierte Nutzung der Standardserviceschnittstellen ist durch den Austausch von Neuigkeiten (News) zwischen Websystemen gegeben.

7.5.2 Austausch aktueller Nachrichten

Eine Standardform der Servicenutzung zwischen Öffentlichkeits-nahen Websystemen, wie CMS und Blogs, ist der Austausch aktueller Nachrichten sowie von Kurzinformationen zu neuen Artikeln über sogenannte News-Feeds. Diese Technologie basiert auf den bereits vorgestellten RSS- oder Atom-Feed-Formaten und umfasst damit Spezialfälle der generischen Serviceschnittstellen, wie sie im Rahmen des SUI-Projektes vorgeschlagen wurden.

Da RSS- oder Atom-Feeds direkt von CMS, wie TYPO3 oder WebGenesis-Systemen, aber auch Portalen gelesen oder bereitgestellt werden können, eignen sie sich sehr gut für einen einfachen grundlegenden Informationsaustausch zwischen Fachanwendungen, Portalen und den Öffentlichkeits-nahen CMS im Web-UIS 3.0. Auch Suchergebnisse aus der GSA lassen sich so aufbereitet an Portale liefern.

7.5.3 Kalenderdienste

Kalenderinformationen und Termine lassen sich bei den CMS-Systemen in der Regel direkt mit Kalenderkomponenten des jeweiligen Systems pflegen, wobei die Bereitstellung der Kalenderinformationen dann für andere Systeme nicht mit jedem eingebauten Kalendersystem möglich ist.

Falls es Möglichkeiten zum serviceorientierten Austausch von Kalenderdaten gibt, erfolgt dieser meistens unter Verwendung von Kalender-Feeds. Ein Kalender-Feed ist eine REST-basierte Serviceschnittstelle, über die Kalenderinformationen abgerufen werden können. Das Rückgabeformat ist dabei typischer Weise das ICAL-Format [92], wie es von Apple für den iCalendar-Service verwendet wird.

Kalenderinformationen im ICAL-Format lassen sich auch von Microsoft-Tools, wie Outlook konsumieren oder über Exchange-Server mit Zusatztools erzeugen. In einem Intranet, in dem Microsoft-Produkte verwendet werden, ist es in der Regel allerdings nicht notwendig, solche Termine noch über Webdienste bereitzustellen. Ganz anderes sieht es dagegen mit Terminen aus, die für die Öffentlichkeit bereitgestellt werden sollen, z.B. bei einem Veranstaltungskalender. Für solche Lösungen verwendet man dann in der Regel einen in ein CMS integrierten Kalender oder evtl. sogar externe Kalenderdienste, wie Google Calendar. Im ersteren Fall sollte der verwendete Kalender Kalenderdaten über das ICAL-Format serviceorientiert exportieren können.

Dienstleister, wie Google, bieten für die Nutzung von Kalendern auf der Google Cloud durchgehend serviceorientierte Lösungen. Im Rahmen einer Google Apps Domain, wie sie die LUBW versuchsweise für das UIS bei Google eingerichtet hat, lassen sich beliebig viele Kalender verwalten, die serviceorientiert bereitgestellt werden können. Autorisierte Autoren können Termine in solche Kalender mit nahezu beliebigen Kalenderapplikationen eintragen und der Abruf der Termine ist über Standardkalenderschnittstellen auf einfache Art möglich. Google-Kalender lassen sich auch auf einfachste Art und Weise in andere Websysteme, wie Content-Management-Systeme, einbetten.

Die Calendar-APIs der Version 3 von Google ermöglichen einen Zugriff auf alle Kalenderinformationen über eine REST-API, wobei das Rückgabeformat ein Google eigenes JSON-Kalenderformat ist. Google Kalender unterstützen nicht nur die eigene REST-API, sondern auch die Bereitstellung der Kalenderinformationen über ICAL sowie die Einbettung eines Google-Kalenders per <iframe> in eine beliebige Webseite. Daher werden Google-Kalender auch häufig zur Integration von Kalendern in Websysteme, wie CMS, benutzt. Der Vorteil dabei ist, dass der Kalender nicht nur in die eigene Webseite eingebettet werden kann, sondern aufgrund der API gibt es auch viele weitere Softwareanwendungen, z.B. für mobile Clients wie IPhone, IPad oder Android-Systeme, über die man direkt mit dem Kalender arbeiten kann. Des Weiteren gibt es Werkzeuge zur Synchronisation eines Google-Kalenders mit anderen Kalendersystemen, wie z.B. Exchange Server. Der Einsatz eines Google-Kalenders ist prinzipiell dann sinnvoll, wenn die Termine sowieso öffentlich zugänglich sein sollen. In diesem Fall ermöglicht das Google-Service-Ökosystem eine optimale Nutzung der Kalenderinformationen in verschiedenen Kontexten.

7.5.4 Social Media Dienste

Für die Nutzung von Medienelementen (Audio und Video) bieten viele CMS-Systeme oder Portale Möglichkeiten, Mediendateien in das System hochzuladen, zu verwalten und dann über eingebaute Web-basierte Playerkomponenten im Browser abzuspielen. Die Playerkomponenten basieren dabei auf JavaScript-Technologie, die entweder die durch HTML5 gegebenen Möglichkeiten zum Abspielen von Mediendateien nutzt oder aber z.B. das Flash-Browserplugin nutzen. Die hierbei eingesetzten JavaScript-Bibliotheken (Jwplayer) [89] sind häufig nicht gänzlich kostenfrei, sondern erzeugen in den freien Versionen ein Wasserzeichen im Bild, das auf die JavaScript-Bibliothek hinweist. Möchte man das Wasserzeichen entfernen, muss man die kommerzielle Version des Players erwerben. In Zukunft bleibt zu hoffen, dass die HTML5-Standardisierung bewirkt, dass Funktionalitäten zum Abspielen von Medien in Browsern wesentlich einfacher und kostenfrei verfügbar sind.

Diese Datei-basierten Technologien zum Abspielen von Medien eignen sich so aber nur für kleinere Mediendateien, da ein CMS oder Portal in der Regel nicht über Möglichkeiten zur Kontrolle des Streaming größerer Mediendateien verfügt. Außerdem haben sie den Nachteil, dass die JavaScript-basierten Player nur bestimmte Formate unterstützen, so dass evtl. eine vorherige Konvertierung der Dateien notwendig ist.

Daher ist eine andere elegante Möglichkeit des Einsatzes von Medien, vor allem bei Videos, die Nutzung von YouTube [90]. Im Kontext einer Google Business Domain lässt sich von einer Organisation auch YouTube im Kontext der Business Domain nutzen. Hierzu kann man ein oder mehrere YouTube-Channel (Videokanäle) der eigenen Business Domain zuordnen und den Channels unterschiedliche Autoren zuordnen, die dann Videos in den Channel hochladen können. Die so auf YouTube zur Verfügung gestellten Videos können dann nicht nur YouTube-Nutzern zur Verfügung gestellt werden, sondern lassen sich auch auf einfachste Art und Weise in CMS oder Portale einbinden. Hierzu kann man im einfachsten Fall den im Channel generierten „Einbettungslink“ über einen <iframe> in eine Webseite einbetten. Viele CMS oder Portale verfügen aber auch über spezielle YouTube-Abspielkomponenten, die man als Inhaltskomponente in eine Seite einbetten kann. Damit lassen sich Videos von YouTube direkt im CMS oder Portal anzeigen.

Da YouTube ebenfalls eine Service-API besitzt, lassen sich eigene Komponenten schreiben, die z.B. über eine Komponente in einem CMS oder Portal hochgeladene Videos an einen YouTube-Channel weiterleiten und das entsprechende Video dann in das CMS oder Portal integrieren.Daher verwendet auch das Land mittlerweile einen YouTube-Channel für die Bereitstellung von Videos in ihren Content-Management-Systemen.

Neben der Bereitstellung von „Konservenvideos“ kann YouTube aber mittlerweile auch für Lifeübertragungen genutzt werden. Im Rahmen einer eigenen Google-Business Domain kann eine Organisation daher sogenannte „Hangouts“ oder „Hangouts on air“ dazu nutzen, um geschlossene Videokonferenzen mit kleineren Arbeitsgruppen durchzuführen, wenn die Teilnehmer nicht am selben Ort zusammengeholt werden können, oder aber, um Life-Veranstaltungen für ein größeres Publikum auf dem Internet zu broadcasten. Dies ist z.B. interessant, um Expertengespräche einem größeren Publikum zugänglich zu machen.

Im Kontext einer eigenen Google Business Domain kann die Nutzung eines eigenen YouTube-Channels daher eine gute Alternative zur Bereitstellung von „Social Media“-Beiträgen im Web-UIS 3.0 sein.

7.5.5 Karten- und Sachdatendienste innerhalb der eigenen Infrastruktur

Eine wesentliche Art von internen Dienstsystemen bzw. Fachanwendungen sind die Anwendungen zum Management der Sachdaten und geobezogenen Informationen von Fachobjekten, wie Geotopen, Mooren, Naturschutzgebieten oder auch von solchen Fachinformationen, wie diejenigen der Lärmkartierung. Der geobezogene Teil dieser Informationen wird teilweise innerhalb der Sachdatensysteme selbst als räumliche, vektorielle Information oder als geobezogene Informationen direkt in Kartenservern, wie z.B. ArcGIS Server [88], gehalten. Während die räumliche Information in diverser Form bereits über standardisierte Schnittstellen der Kartensysteme serviceorientiert abgerufen werden können (z.B. als WMS- oder WFS-

Layer), fehlen auf Seiten der Sachdaten hier nach wie vor serviceorientierte Schnittstellen, über die Sachinformationen zu Umweltobjekten einfach von anderen Anwendungen abgerufen werden können.

Die oben beschriebenen Datenmanagementsysteme für Karten- und Sachdaten sind zentrale Elemente der Arbeit von Umweltbehörden. Ein kontinuierlicher Ausbau der Serviceorientierung dieser Systeme ist daher von grundlegender Bedeutung für eine zukünftige Web-UIS 3.0 Infrastruktur.

7.5.6 Bereitstellung von Karten- und Sachdaten über externe Services

Bei Daten und Karteninformationen, die der Internet-Öffentlichkeit zur Verfügung gestellt werden sollen, bietet die Nutzung externer Dienste hierzu einige Vorteile, da diese externen Dienste nicht nur eine hohe Verfügbarkeit (365 Tage x 24 Stunden), sondern auch sehr gute Performanz aufweisen. Beide Eigenschaften lassen sich mit eigenen Servern nur schwer erreichen. Ein weiterer Vorteil der Nutzung externer Dienste ist, dass man dadurch die in Systemen für die Internet-Öffentlichkeit bereitgestellten Daten und Informationen weitgehend von ihren Autorenumgebungen im Intranet entkoppelt. Durch Definition von expliziten Abläufen zur Qualitätskontrolle benötigter Daten und deren Export auf die externen Dienstserver über Software, die die Service-API der externen Dienste nutzt, kann eine genauere Kontrolle erfolgen, welche Daten wann und wie im Internet bereitgestellt werden, und diese Datenbereitstellung wird durch eine interne Bearbeitung der gleichen Daten in den Autorenumgebungen im Intranet nicht gestört. Unbeabsichtigte Änderungen können dabei auch nicht automatisch in die Öffentlichkeit gelangen.

Externe Dienste, wie die Google Business- [78] und Google Maps Engine Dienste [79], ermöglichen die schnelle Erstellung komplexerer, interaktiver Karten und Datenvisualisierungen sowie die Einbettung dieser Anwendungen in eigene Webseiten. Auf Basis der zugehörigen Service-APIs lassen sich insbesondere eigene Softwarekomponenten schreiben, die eine tiefe Integration zwischen einem CMS oder Portalserver und Daten und Anwendungen innerhalb der Google Business Dienste herstellen. Dabei können Daten, wie bereits im Kapitel 3 „State oft he Art“ beschrieben, in Fusion Tables abgelegt werden, und dann entweder als Kartenlayer auf Karten oder aber z.B. als Grafiken angezeigt werden (siehe Abb. 7.5 auf der nächsten Seite für ein komplexeres Beispiel).

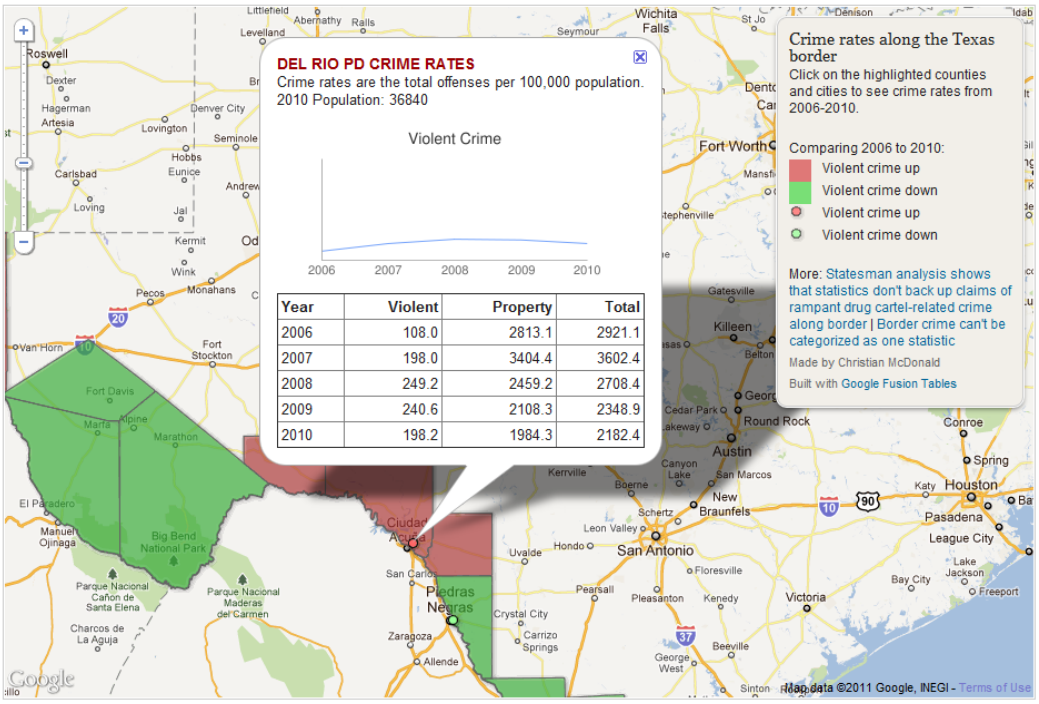


Abb. 7.5: Mit Fusion Tables erzeugte Anwendung zur Anzeige einer Verbrechensstatistik sowohl auf einer Google Maps Karte, an einem ausgewählten Punkt bzgl. der zeitlichen Entwicklung als Tabelle und Kurve (Quelle: Technical Bent, http://www.technicalbent.com/2011/11/adding-google-chart-image-to-fusion.html)

Erste Tests der Google Business Dienste zur Nutzung für die Bereitstellung von Daten und Karteninformationen verliefen sehr erfolgsversprechend. Die Nutzung solcher externen Dienste für zukünftige Internetangebote für Daten, die der allgemeinen Öffentlichkeit zugänglich gemacht werden sollen, könnte daher Bestandteil einer zukünftigen Web-UIS 3.0 Infrastruktur sein.

7.5.7 Dokumenten-orientierte Dienste

Im Kontext der verschiedenen Projekte im Umfeld des Web-UIS wird auch immer wieder über die Notwendigkeit der Einführung eines (zentralen) Dokumenten Management Systems (DMS) [50] gesprochen. Ein Dokumenten-Management-System dient als „Ablage“ für Dokumente unterschiedlicher Arten (Word- oder PDF-Dokumente, Textdokumente, etc.). Dabei ist nicht nur die Ablage und Archivierung wichtig, sondern vor allem auch das Auffinden von Dokumenten soll über ein solches System verbessert werden. So ist eine Kernfunktionalität von DMS-Systemen, Mechanismen zur Verknüpfung von Metadaten mit den zu speichernden Dokumenten bereitzustellen sowie über diese Metadaten eine komfortablere Suche nach Dokumenten zu erlauben.

Ein Web-basiertes DMS-System, wie Alfresco [93], erlaubt dabei häufig auch Webinhalte selbst (z.B. Inhalte von Webseiten) als Assets im System zu verwalten. Man spricht dann auch von Web Content Management (WCM). Darüber hinaus bieten manche dieser Systeme, wie auch Alfresco, einen komfortablen Zugang zum System über serviceorientierte Schnittstellen. Die bereits als Beispiel erwähnte Alfresco-Software lässt sich z.B. über solche Services mit einem Liferay-Portalserver integrieren und kann innerhalb einer oder mehrerer Liferay-Server z.B. als DMS (Speichersystem für Dokumente und andere digitale Inhalte),

aber auch als WCM-System dienen. Sie ersetzt bzw. ergänzt dabei die in Liferay eingebauten Management-Funktionalitäten für solche Inhalte. Verschiedene durch Alfresco bereitgestellte Portlets können dann für den Zugang in Liferay zu den Inhalten benutzt werden. Systeme, wie Alfresco, arbeiten hierbei nicht nur mit Liferay sondern auch mit anderen Enterprise Portalservern und sogar CMS, wie z.B. Drupal oder TYPO3, zusammen. Durch die Integration in verschiedene Systeme können dabei gleiche digitale Inhalte von diesen verschiedenen Systemen aus gemeinsam genutzt werden. Über Schnittstellen für externe Suchmaschine, wie Solr oder GSA, lässt sich der komplette Dokumentenbestand dann auch für eine externe Suche komfortabel erschließen.

Alfresco als typisches serviceorientiertes DMS lässt sich aber nicht nur mit anderen Webanwendungen integrieren, sondern ermöglicht es auch über seine Serviceschnittstellen, dass das DMS direkt von mobilen Anwendungen und mobilen Webclients wie ein Cloud-Dateiservice à la Dropbox oder Google Drive genutzt werden kann. Des Weiteren bietet es auch Integration mit solchen Dateidiensten, wie Google Drive. Dies bietet dann einen von überall mit nahezu jedem Gerät verfügbaren, gesicherten Zugang zu Dokumenten im DMS.

Weitere Integrationstechnologien verbinden ein solches DMS mit der Microsoft-Officewelt im Intranet. Ein direkter Officeconnector fügt dabei z.B. Menüpunkte zum Öffnen und Speichern von Dokumenten direkt aus oder in das DMS in die Officewerkzeuge, wie Word oder Powerpoint, ein. Alfresco kann noch komfortabler von Windowssystemen und Officeprogrammen direkt als kompatibler Sharepoint-Server genutzt werden. Eine solche Lösung ist im Enterprise-Umfeld zum Management von Dokumenten und anderen Inhalten oftmals sinnvoll und wird in vielen Institutionen so eingesetzt.

DMS-Systeme, wie Alfresco, gibt es dabei sowohl als Open Source Community-Lösungen als auch mit Enterprise-Lizenzen, bei denen dann erweiterte Funktionalitäten und verbesserter Support über die Enterprise-Lizenz gekauft werden müssen. Alfresco gibt es dabei entweder als Software, die man auf eigenen Servern im Intranet der Firma installieren kann oder aber einfach als Cloudservice, auf dem man sich für Projekte oder seine Organisation einfach ein DMS mieten kann. Alfresco unterstützt auch die Synchronisierung von Dokumenten zwischen intern im Intranet installiertem System und dem Cloudservice auf der anderen Seite, z.B. für getrennte Bereitstellung bestimmter Dokumente für die Öffentlichkeit.

Ob ein solches System auf Dauer für die Web-UIS Infrastruktur sinnvoll wäre, soll hier nicht weiter diskutiert werden, sondern müsste intern in den Behörden entschieden werden. Falls ein solches System allerdings eingeführt werden sollte, wäre es wegen der Integrierbarkeit ratsam, hier auf Systeme mit lose koppelbaren serviceorientierten Schnittstellen zu setzen, wie sie z.B. Alfresco bereitstellt, um so den Mehrwert solcher Systeme für die eigene serviceorientierte Softwarelandschaft auf alle anderen Systeme über lose koppelbare Schnittstellen übertragen zu können.

7.5.8 Bilddatenbank

Ein Spezialfall der im vorherigen Unterkapitel beschriebenen Dokument Management Systeme sind die Digital Asset Management (DAM) Systeme [94]. Ein Digital Assset Management System beschäftigt sich üblicher Weise weniger mit der Speicherung von Dokumenten als digitale Assets, sondern eher mit der Speicherung multimedialer digitaler Dateien (Bilder, Videos, Audiodateien).

Natürlich können DMS-Systeme als übergeordnete Systeme häufig auch Funktionalitäten zur optimierten Speicherung solcher Digitaler Assets enthalten, so dass sie dann die Funktion eines DAM mit übernehmen. So gibt es für das Alfresco System Plugins, die eine spezialisierte Speicherung von multimedialen Assets (als DAM) ermöglichen. Andere DAM-Systeme sind Spezialisten für multimediale Dateien, was z.B. die automatische Extraktion von Metadaten aus den digitalen Dateien heraus angeht (z.B. Exif-Informationen aus Bildern).

Für die Einsetzbarkeit von DAMs gelten ähnliche Aussagen wie für den Einsatz von DMS. Deshalb sollen die Vorteile solcher DAMs, die durchgehend serviceorientiert mit Standardschnittstellen arbeiten, hier nicht weiter diskutiert werden. Ob man eher ein DMS für beides oder ein spezialisiertes DAM für Multimedia einsetzt, hängt davon ab, wie gut die Managementeigenschaften auf der Medienseite sein sollen (z.B. automatische Transformation von Medien in verschiedene Formate, automatische Skalierung, Einbau von Wasserzeichen, etc.).

7.6 Authentifizierungs- und Autorisierungskonzepte

Innerhalb einer serviceorientierten Softwarelandschaft, wie sie in diesem Konzept vorgestellt wurde, kommt vernetzten Authentifizierungsdiensten und diese nutzenden Autorisierungskonzepten in den beteiligten Systemen eine zentrale Rolle zu. Daher stellen CMS- oder Portalserver als ein wichtiges Konzept Funktionalitäten zum Single Sign On (SSO) bereit. Beim Single Sign On loggt sich ein Benutzer in einem zentralen CMS oder Portal ein und kann dann durch Verknüpfung der Authentifizierungs- und Autorisierungsmechanismen verschiedener beteiligter Systeme mit weiteren Systemen arbeiten, ohne dabei jedes Mal die Accountdaten neu eingeben zu müssen. Eine schwächere Form von SSO ist, dass ein Benutzer zumindest die gleichen Accounts für verschiedene Systeme benutzen kann.

SSO oder die Nutzung ein und desselben Accounts erreicht man im Intranetkontext z.B. durch Verwendung zentraler Benutzermanagementdienste, z.B. auf Basis eines LDAP- oder Active-Directory-Servers, die dann durch SSO-Mechanismen, z.B. einem Central Authentication Service (CAS) [95], wie er z.B. vom Open Source Server JA-SIG (Jasig CAS) bereitgestellt wird, ergänzt werden. An eine solche Lösung lassen sich dann eine Vielzahl von Systemen, wie CMS oder Portalserver, Datenbankserver, RADIUS-Server zum Zugang von Personen über VPN, etc. andocken.

Der Aufbau einer solchen Authentifizierungsinfrastruktur erfordert natürlich einen Aufwand administrativer Art beim Aufsetzen und Konfigurieren der benötigten zentralen Dienste. Zentrale Authentifizierungsdienste sollten auch eine besonders hohe „Quality of Service“ aufwei-

sen, da eine nicht-funktionierende Authentifizierung natürlich dem Nutzer verteilter Dienstleistungen im Intranet wenig Vergnügen bereitet.

Eine ganz andere Situation ergibt sich, wenn sich Personen in CMS, Portale oder Fachanwendungen über das Internet einloggen sollen und als Nutzer in den Intranet-eigenen Nutzerverwaltungssystemen, z.B. Active Directory, nicht registriert sind. In diesem Fall kann man entweder auf eine lokale Nutzerverwaltung solcher Nutzer im CMS oder Portal direkt setzen, oder aber den Nutzern die Benutzung von Accounts anderer Internetdienste (z.B. Google, Facebook oder Amazon) erlauben.

Bei der letzten Variante lassen sich die OpenID [96] und/oder Oauth (Oauth2)-Standards [97] einsetzen. Der Benutzer kann sich hierbei mit einem OpenID-Account, den er bei irgendeinem Internetdienst, z.B. Google, Yahoo, Blogger, flickr, Wordpress, myOpenID, etc., besitzt, in ein Fremdsystem einloggen, das hierbei die Überprüfung der Authentifizierungsdaten an den OpenID-Provider delegiert, der dem Fremdsystem dann in der Regel auch Zugang zu dem zugehörigen Profil (evtl. mit Einschränkungen, die der Benutzer selbst einstellen kann, also dass z.B. seine Telefonnummer sichtbar ist) gestattet. Das Fremdsystem erlaubt dann die Benutzung eigener Funktionalitäten, wobei dem Benutzer über den eingebauten Autorisierungsmechanismus des Systems bezogen auf seinen OpenID-Account Rechte zugewiesen werden können. Das Oauth-Protokoll kommt ins Spiel, wenn nun ein solcher OpenID-Account im Sinne von Single Sign On weiter als Authentifizierungsinformation für den Zugriff auf entfernte Services genutzt werden soll. Unterstützen solche entfernten Dienstschnittstellen Authentifizierung über das Oauth-Protokoll beim Servicezugriff, so lässt sich gegebenenfalls der OpenID-Account ebenfalls für die Authentifizierung bei diesen Diensten nutzen.

Eine ähnliche, aber nicht-standardisierte Lösung bietet das Social Network Facebook mit seinem „Facebook Connect“. Hierbei authentifiziert sich ein Nutzer bei einem Fremdsystem mit seinem Facebook-Account und gibt dabei indirekt (oftmals bemerkt der Nutzer dies nicht einmal) seinen Facebook-Account und sein Facebook-Profil für die Nutzung des Fremdsystems frei. Dieses kann dann über die Facebook-APIs auf Facebook als Dienst zugreifen und Facebook-Funktionalitäten Nutzer-bezogen verwenden. Dies ermöglicht z.B. die Integration von sozialen Funktionalitäten, wie z.B. Kommentierung von Artikeln auf der eigenen Facebookseite durch Nutzer, und deren Integration in eigene Webanwendungen. Das Google+-Netzwerk besitzt einen ähnlichen, aber Oauth 2-konformen Mechanismus, „Google+ Sign-In“ genannt, der eine entsprechende Integration des Google+-Netzwerks mit eigenen Webanwendungen ermöglicht, wobei der Webnutzer hierzu seinen Google (Google+) Account benützt. Auf diese Weise lassen sich dann eigene Webanwendungen mit dem Google+-Netzwerk integrieren.

Ob man solche Verfahren tatsächlich nutzen möchte, sei dahingestellt. Der Trend bei vielen Webdiensten geht allerdings in die Richtung, dass viele kleinere Systeme mittlerweile die Nutzung von solchen Fremdaccounts erlauben. Die Nutzung von Google-Accounts ist dabei auf mobilen Geräten für viele Nutzer nichts Außergewöhnliches mehr. Im Behördenkontext wäre hier natürlich ideal, wenn es einen vom Staat z.B. bundesweit betriebenen OpenID-Provider gäbe, bei dem jeder Bürger einen eindeutigen OpenID-Account besitzt, den er für behördliche Dinge nutzen könnte. Leider sind solche behördlichen Authentifizierungsinfrastrukturen mit wirklich sicheren Accounts auf absehbare Zeit nicht in Sicht. In der Praxis wird es eher darauf hinauslaufen, dass in Zukunft jeder Bürger bei einem der großen Webdienst-

leister (Google, Amazon, Facebook, Apple oder Microsoft) einen OpenID-konformen Account besitzen wird.

Autoren und Administratoren von Webanwendungen sollten sich in der Regel mit ihrem normalen Behördenaccount oder einem dediziert im CMS erstellten Account in die Websysteme einloggen können. Das Einloggen über Fremdsysteme ist demgegenüber eher für den externe Nutzer interessant. Ob Fremdauthentifizierung in diesem Sinne im Rahmen des Web-UIS tatsächlich eingesetzt werden soll oder doch eine überbehördliche Lösung in Frage kommt, müsste noch gesondert diskutiert werden.

8 Fazit und Ausblick

Im Rahmen dieser Konzeptstudie zum Web-UIS 3.0 wurden Empfehlungen für eine zukünftige Ausrichtung der Web-basierten Informationssysteme des UIS Baden-Württemberg erarbeitet. Ausgehend von einer Übersicht der wichtigsten bis dato vorhandenen Web-basierten Systeme in Kapitel 2, wird in Kapitel 3 zunächst der aktuelle „State of the Art“ der Internet- und Webtechnologien beschrieben. In Kapitel 4 werden dann aus Sicht des UM und der LUBW generelle Anforderungen an die zukünftigen Webanwendungen formuliert. Dabei zeigte sich bereits, dass die bestehende Infrastruktur bei weitem nicht mehr alle Anforderungen bedient, die sich Nutzer aus dem UM- und LUBW-Umfeld wünschen.

Auch die Ergonomie der vorhandenen Funktionalitäten zum Content-Management entspricht an manchen Stellen nicht mehr den Benutzerwünschen, was darauf zurückzuführen ist, dass bestimmte zur Entwicklung genutzte Funktionalitäten mittlerweile nicht mehr auf dem Stand der Technik sind. Ein nicht unerheblicher Teil der formulierten Anforderungen zeigt aber auch, dass ein weiteres Defizit der momentanen Webinfrastruktur die nicht vorhandenen Möglichkeiten zur Integrierbarkeit externer Services bzw. zur serviceorientierten Integration interner Webanwendungen in Portale und Homepages ist.

Dass die momentan verwendeten technischen Lösungen im Webbereich nicht in allen Punkten mehr auf dem Stand der Technik sind, wird bereits im Kapitel 3 „State of the Art“ deutlich. Wie in Kapitel 5 „Analyse“ beschrieben, zeigt sich, dass die Weiterentwicklung der WebGenesis-CMS Funktionalitäten auf Grund der recht kleinen Entwicklungsmannschaft und den vielen Anwendungs- und Forschungsprojekten, in denen die WebGenesis-Funktionalitäten als Anwendungsentwicklungsumgebung für Forschungsprojekte optimiert wird, mit der Entwicklung spezieller funktionaler Komponenten für CMS-Systeme in der Vergangenheit nicht Schritt halten konnte, weil der Schwerpunkt bei der WebGenesis-Nutzung auf der Entwicklung technisch-wissenschaftlicher Fachanwendungen liegt. So gibt es für CMS, wie TYPO3, Hunderte von Zusatzkomponenten für fast jede Aufgabenstellung eines CMS, während in diesem Anwendungsbereich essentielle Funktionalitäten bei WebGenesis fehlen. Wie in Kapitel 6 dargelegt wird, arbeitet IOSB aber an Lösungen für die wichtigsten Probleme. Diese Entwicklungen bei WebGenesis konzentrieren sich aber eher auf grundsätzliche Funktionalitäten und bessere Möglichkeiten der Anwendungsprogrammierung, weniger auf Kern-CMS- oder Portalfunktionalitäten. Daher wurde im Rahmen dieser Studie offen diskutiert, dass im Bereich von reinen CMS-Anwendungen und vor allem auch für Portale der Einsatz anderer technischer Systeme denkbar ist. Eine Migration auf das von anderen Landesbehörden verwendete pirobase-System lohnt sich dabei, wie bereits in Kapitel 5 beschrieben, nicht, da hierfür ein sehr hoher Aufwand notwendig wäre und diese Systeme eher für Enterprise Content-Management und weniger als Standard-CMS geeignet sind. Gute Kandidaten wären dagegen das von der Landesregierung eingesetzte TYPO3-System bzw. der Java-basierte Open Source Portalserver Liferay.

Eine andere Situation ergibt sich bei Systemen, deren Bedarf weniger im Vorhandensein gängiger funktionaler Komponenten für CMS liegt, sondern die gute Voraussetzungen zur Programmierung eigener Fachanwendungslogik bieten sollen. Hier kommt es mehr darauf an, dass das verwendete technische System gute Möglichkeiten zur Programmierung der

benötigten Fachfunktionalitäten bietet. Hier bestehen bei dem augenblicklichen WebGenesis-System zwar auch Defizite, diese lassen sich aber an kritischen Stellen durchaus zeitnah entschärfen. Ansonsten bietet die Nutzung von Java als bevorzugte Programmiersprache für web-basierte Fachanwendungen nach wie vor die besten Möglichkeiten. Nicht umsonst ist Java schon seit Jahren auf Platz 1 der am meisten benutzten Programmiersprachen für Softwareentwicklung (gefolgt von C/C++) und wird vorwiegend für die Programmierung serverseitiger Software eingesetzt. Weiter ist Java die bevorzugte Sprache für Web-basierte Anwendungen im Enterprise Bereich. Dies liegt u.a. daran, dass die Java Enterprise Edition, die als Programmierplattform für Enterprise Anwendungen im Java-Bereich als Grundlage eingesetzt wird, alle modernen Konzepte des Webs beinhaltet bzw. zeitnah in neuen Versionen bedient. Daher wird in dieser Konzeptstudie empfohlen, Java auch in Zukunft als Standardentwicklungsplattform für größere Web-basierte Projekte und Fachanwendungen einzusetzen.

Ein wesentliches Ergebnis der Analyse in Kapitel 5 ist aber auch, dass bei aktuellen Entwicklungen des bisherigen Web-UIS trotz vielfacher Diskussion in verschiedenen Forschungsprojekten die konsequente Ausrichtung auf eine serviceorientierte Architektur des Web-UIS nicht stattgefunden hat. So sind in Forschungsprojekten, wie SUI, zwar die konzeptuellen Grundlagen für mehr Serviceorientierung gelegt worden und es wurden hierzu auch Referenzimplementierungen für generische Serviceschnittstellen entwickelt, ein Gesamtkonzept auf breiter Ebene fehlt aber nach wie vor. Dabei zeigt sich insbesondere an verschiedenen Anforderungen aus Kapitel 4 sowie aktuellen Beispielen zur Nutzung externer Cloud-basierter Dienste, wie wichtig eine durchgehende Serviceorientierung für die Zukunft des Web-UIS sein wird.

Daher basieren die in Kapitel 7 gegebenen Empfehlungen für eine Neuausrichtung der web-basierten Umweltinformationssysteme zunächst auf der Hauptempfehlung, in Zukunft *verstärkt auf eine modulare, lose durch Services koppelbare weitgehend web-basierte Softwareumgebung zu setzen*. Innerhalb dieser Softwareinfrastruktur sollten Umweltfachinformationen in web-basierten Fachanwendungen im Intranet der Behörden über Webschnittstellen editierbar und pflegbar sein und gängige Arbeitsvorgänge über das Web von Sachbearbeitern erledigt werden können. Die Fachanwendungen sollten weiter serviceorientierte Schnittstellen bieten, so dass andere Systeme des Web-UIS auf alle essentiellen Informationen über serviceorientierte Schnittstellen zugreifen können. Diese Grundarchitektur des Web-UIS 3.0 ermöglicht einen weitgehend automatisierbaren Austausch zwischen den eigentlichen Fachanwendungen und z.B. Portalen oder externen Dienstplattformen.

Eine geringe Anzahl von Webportalen und CMS-Systemen als Webanwendungen für den allgemeinen Nutzerzugang zu Informationen sollte dann als zentrale Recherche- und Informationsplattform für den größten Teil der Benutzer dienen, so dass diese sich nicht auf verschiedenen Fachanwendungen einloggen müssen, um Informationen aus unterschiedlichen Systemen manuell zusammenzustellen. Diese „Frontend-Systeme“ sollten für unterschiedliche Nutzerklassen spezifische Zugänge bieten, die für bestimmte Nutzergruppen, wie Fachanwender, sogar personalisierbar sein sollten. Für den Zugang über Internet zu Öffentlichkeitsarbeits-nahen Informationen, wie Pressemitteilungen oder Social Media, eignen sich hierfür spezialisierte CMS-Anwendungen, während für den Aufbau größerer Portale für Öffentlichkeit und Fachanwender mit personalisierbaren Inhalten und für das Hosting verschiedener Domainen Portalserver gut geeignet sind, die auch die Trennung zwischen Life-

Inhalten und internen Autorenumgebungen (Staging, Autoren-Entwicklungsserver im Intranet) sowie die Integration von Fachanwendungen unterstützen.

Hier wird in dieser Konzeptstudie empfohlen, durchaus auf eine gewisse Heterogenität der technischen Plattformen, und vor allem auch auf offene Systeme zu setzen, und z.B. für CMS-Anwendungen eher ein Open Source CMS „von der Stange“, wie TYPO3, oder einen Open Source Portalserver zu verwenden, während andererseits nicht ohne Not bereits bestehende Web-Fachanwendungen auf Basis von WebGenesis unnötiger Weise und mit hohem Aufwand auf neue technische Plattformen portiert werden sollten, wenn diese bereits jetzt ihre Aufgabe gut erfüllen und mit vertretbaren Aufwand an aktuelle Anforderungen angepasst werden können.

Eine Serviceorientierung ist dabei in so einer heterogenen Umgebung essentiell. Die Schnittstellen zu den hinter den Frontend-Systemen liegenden Fachanwendungen müssen so gestaltet sein, dass sie möglichst technologieneutral und damit von den verschiedenartigen Frontend-Systemen gleichermaßen nutzbar sind. Dies funktioniert am besten mit Schnittstellen, die universell verfügbar sind. Diesbezüglich zeigen REST-basierte Services deutliche Vorteile, da die Grundinternettechnologien, wie HTTP, heutzutage auf fast allen Geräten Programmiersprachen-unabhängig universell verfügbar sind. Dies gilt z.B. für SOAP-basierte Webservice nur bedingt, da insbesondere Mobilgeräte wesentlich besser mit REST- als mit SOAP-basierten Services umgehen können. Die gleiche Aussage gilt auch für JavaScript-basierte Clients, d.h. für Webanwendungen, die z.B. AJAX zum Zugriff auf Services verwenden. Auch hier werden heutzutage bevorzugt REST-basierte Services eingesetzt. REST-basierte Services bieten weiter gute Skalierungseigenschaften, da sie HTTP-basierte Skalierungstechnologien, wie Cacheserver, Proxy- und Verteildienste unmittelbar transparent nutzen können.

Daher ist die Empfehlung dieser Studie, beim zukünftigen Einsatz von Services vorwiegend auf REST-basierte Services zu setzen, und nur an Stellen, wo SOAP-basierte Services Funktionalitäten implementieren, die mit REST-Services nicht sinnvoll abgedeckt werden können (z.B. Nachrichten-basierte asynchrone Kommunikation) auf SOAP-basierte Services zurückzugreifen. Die Serviceorientierung sollte dabei im Sinne einer ROA (Ressource Oriented Architecture) [98] erfolgen. Bei einer Ressource-orientierten Architektur fokussiert man sich weniger auf die Definition objektorientierter Anwendungsprogrammierschnittstellen (bei diesen liegt die Hauptbetrachtungsweise eher auf der Definition von Aktionen in Form von Objektmethoden) sondern auf die Definition von Ressourcen, deren Repräsentationen und einer verlinkten Navigation zwischen Ressourcen. Man spricht dann im Oberbegriff auch von einer datenorientierten Architektur. In einer ROA ist die Spezifikation der Aktionen nicht so wichtig, da diese sowieso genormt über die Aktionsverben des HTTP-Protokolls erfolgen. Dies vereinfacht aber die Spezifikation der zugehörigen Schnittstellen enorm und erleichtert damit das Verständnis und Schreiben von Server- und Clientschnittstellen.

Die vorgelegte Konzeptstudie ist als interne Diskussionsgrundlage für das UM und die LUBW zur zukünftigen Ausrichtung des Web-UIS gedacht. Sie kann dabei wichtige Impulse für die Überarbeitung der entsprechenden Web-UIS Rahmenkonzeption liefern und als Entscheidungsgrundlage für die Vergabe zukünftiger forschungsorientierter Projekte oder die Ausrichtung größerer neuer Softwareprojekte im UIS-Umfeld, insbesondere wenn es sich um die Entwicklung Web-basierter Anwendungen handelt, dienen.

Folgt die zukünftige Strategie im Web-UIS Bereich Kernaussagen dieser Studie, ist es naheliegend, einige der vorgestellten Konzepte im Rahmen konkreter Entwicklungsmaßnahmen, z.B. bei der Umstellung von Portalen oder Homepages, genauer am Beispiel dieser Systeme auszuarbeiten. Dies betrifft z.B. die Umstellung der Websysteme auf die kommenden neuen Designvorgaben, bei der dann moderne HTML5-basierte Vorlagen für die zukünftigen Systeme erarbeitet werden sollten, die verstärkte Servicenutzung, z.B. durch Integration mit internen und externen Diensten, die Spezifikation grundlegender Serviceschnittstellen in einer ROA oder auch die evolutionäre Weiterentwicklung bestehender Fachanwendungen im Sinne der vorgestellten Architektur. Dies kann in konkreten Entwicklungsprojekten oder zum Ausprobieren forschungsnah auch in forschungsorientierten Projekten oder innerhalb von Projektarbeiten von Studenten geschehen. Die bereits existierende Forschungskooperation MAF-UIS bietet hierfür hervorragende Bedingungen.

Anhang 1: Literaturverzeichnis

1. ixpro.de: *Was ist ein CMS - Definition. Existenzgründung und Selbständigkeit | erfolgreich selbständig machen.* [Online] [Zitat vom: 06. Februar 2013.] http://www.ixpro.de/it-wissen/cms/was-ist-ein-cms-definition.html.

2. Schewe, Gerhard: *Workflow Management. Gabler Wirtschaftslexikon*. [Online] Springer Gabler. [Zitat vom: 06. Februar 2013.] http://wirtschaftslexikon.gabler.de/Definition/workflow-management.html.

3. Universität Bielefeld: *Aufbau CMS. Content Management und Web Application*. [Online] [Zitat vom: 06. Februar 2013.] http://www.techfak.uni-bielefeld.de/ags/pi/lehre/DokV01/lectures/lecture11/aufbau-cms.gif.

4. DATACOM Buchverlag GmbH: *WCMS (Web Content Management System)*. ITWissen. [Online] . [Zitat vom: 06. Februar 2013.] http://www.itwissen.info/definition/lexikon/web-content-management-system-WCMS.html.

5. SAPERION AG: *Content Repository. SAPERION Blog*. [Online] Oktober 2009. [Zitat vom: 06. Februar 2013.] http://www.saperionblog.com/wp-content/uploads/2011/10/WissenHeute_ContentRepos.pdf.

6. Kirchhof, Anja, et al.: *»Was ist ein Portal?« Definition und Einsatz von Unternehmensportalen*. Fraunhofer IAO, 2004.

7. Vlachakis, Joannis; Kirchhof, Anja und Gurzki, Thorsten*: Marktübersicht Portalsoftware 2005*. Stuttgart : Fraunhofer IAO, 2005.

8. Bosch, Andy: *Portale und Portlets (1) – Grundlagen. jaxenter, Portal für Java, Enterprise Architekturen, SOA*. [Online] Dezember 2008. [Zitat vom: 06. Februar 2013.] http://it-republik.de/jaxenter/artikel/Portale-und-Portlets-%281%29-%96-Grundlagen-2075.html.

9. Fraunhofer IOSB: *WebGenesis®.* [Online] 2011. [Zitat vom: 06. Februar 2013.] http://www.iosb.fraunhofer.de/servlet/is/18052/.

10. LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg: *Benutzerhandbuch für WebGenesis-Autoren in der LUBW*. Karlsruhe : 2012.

11. Cron IT GmbH: *TYPO3 - eine Lösung der Enterprise-Klasse! cron IT GmbH - TYPO3-CMS und TYPO3 Hosting.* [Online] [Zitat vom: 06. Februar 2013.] http://www.typo3-anbieter.de/de/typo3-cms/typo3-technik.html.

12. Zentraler Informatikdienst ZID, Universität für Bodenkultur Wien: *TYPO3, Einführung für Web-Autor/innen.* [Online] Mai 2012. [Zitat vom: 06. Februar 2013.] http://www.boku.ac.at/typo3einf/Typo3Einf-45.pdf.

13. The PHP Group. PHP: *Hypertext Preprocessor. PHP*. [Online] [Zitat vom: 06. Februar 2013.] http://www.php.net/.

14. xdot GmbH: *Liferay Workshop-Unterlagen für die LUBW*. Münster : 2013.

15. Simpler Media Group, Inc.: *Gartner's Magic Quadrant For Horizontal Portals: Oracle, IBM, Microsoft, SAP, Liferay Top Dogs*. Covering Customer Experience, Social Business & Information Management. [Online] 11. Oktober 2012. [Zitat vom: 06. Februar 2013.] http://www.cmswire.com/cms/customer-experience/gartners-magic-quadrant-for-horizontal-portals-oracle-ibm-microsoft-sap-liferay-top-dogs-017717.php.

16. Gartner Inc: *Magic Quadrant for Horizontal Portals. Technology Research | Gartner Inc. [Online] 20. September 2012*. [Zitat vom: 06. Februar 2013.] http://www.gartner.com/technology/reprints.do?id=1-1DC3B99&ct=121220&st=sg.

17. Koordinierungsstelle PortalU im Niedersächsischen Ministerium für Umwelt, Energie und Klimaschutz: *Umweltportal Deutschland*. [Online] [Zitat vom: 06. Februar 2013.] http://www.portalu.de/.

18. LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg*: Leitfaden für neue Mitarbeiterinnen und Mitarbeiter*. Karlsruhe : 2013.

19. LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg: *Homepage der Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg*. [Online] [Zitat vom: 06. Februar 2013.] http://www.lubw.baden-wuerttemberg.de.

20. Ministerium für Umwelt, Klima und Energiewirtschaft Baden-Württemberg (UM): *Homepage des Ministerium für Umwelt, Klima und Energiewirtschaft Baden-Württembergs*. [Online] [Zitat vom: 06. Februar 2013.] http://www.lubw.baden-wuerttemberg.de.

21. Ministerium für Umwelt, Klima und Energiewirtschaft Baden-Württemberg: *Umweltportal Baden-Württemberg*. [Online] [Zitat vom: 06. Februar 2013.] http://www.umwelt.baden-wuerttemberg.de.

22. LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg: *Fachdokumente Online*. [Online] [Zitat vom: 06. Februar 2013.] http://www.fachdokumente.lubw.baden-wuerttemberg.de.

23. Ministerium für Umwelt, Klima und Energiewirtschaft Baden-Württemberg. *Willkommen beim Themenpark Umwelt*. [Online] [Zitat vom: 06. Februar 2013.] http://themenpark-umwelt.baden-wuerttemberg.de.

24. LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg: *REACH@Baden-Württemberg*. [Online] [Zitat vom: 06. Februar 2013.] http://www.lubw.baden-wuerttemberg.de/servlet/is/22787/.

25. Umweltbundesamt: *Umweltbundesamt Homepage*. [Online] [Zitat vom: 06. Februar 2013.] http://www.umweltbundesamt.de/.

26. LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg: *PLENUM Baden-Württemberg*. [Online] [Zitat vom: 06. Februar 2013.] http://www.lubw.baden-wuerttemberg.de/servlet/is/47045/.

27. Moxiecode Systems AB; *TinyMCE*. [Online] [Zitat vom: 06. Februar 2013.] http://www.tinymce.com/.

28. Fraunhofer IOSB: *IOSB Sartutor*. [Online] [Zitat vom: 06. Februar 2013.] http://www.xiah.de/img/IOSB-sartutor.gif.

29. TYPO3 Association: *Logo Typo3*. Wikimedia. [Online] 11. Oktober 2012. [Zitat vom: 06. Februar 2013.] http://commons.wikimedia.org/wiki/File:Logo_Typo3.svg.

30. LIGHTWERK Internetagentur: *Liferay Portal - Die Enterprise Lösung. LIGHTWERK Internetagentur in Stuttgart*. [Online] [Zitat vom: 06. Februar 2013.] http://www.lightwerk.de/produkte/liferay-portal.html.

31. Gull, Clemens; Münz, Stefan*: HTML5 Handbuch*. Franzis Verlag; Auflage: 2 (28. November 2011)

32. Stiegert, Heiko: *Modernes Webdesign mit CSS: Schritt für Schritt zur perfekten Website - aktuell zu CSS3*. Galileo Design; Auflage: 1 (28. Juli 2011)

33. Flanagan, David: *JavaScript – Das umfassende Referenzwerk*. O'Reilly; Auflage: 6 (1. April 2012)

34. Zillgens, Christoph: *Responsive Webdesign: Reaktionsfähige Websites gestalten und umsetzen*. Carl Hanser Verlag GmbH & Co. KG (4. Oktober 2012)

35. Maurice, Florence: *Mobile Webseiten: Strategien, Techniken, Dos und Don'ts für Webentwickler. Von Responsive Webdesign über jQuery Mobile bis zu separaten mobilen Seiten*. Carl Hanser Verlag GmbH & Co. KG (4. Oktober 2012)

36. Bongers, Frank: *jQuery: Das Praxisbuch (Galileo Computing)*. Galileo Computing; Auflage: 2 (28. August 2011)

37. Google Inc.: *Google Search Applicance*. [Online], [besucht am 15.03.2013], http://www.google.de/enterprise/search/gsa.html.

38. Mayer-Föll, Roland; Kaufhold, Gerhard (Hrsg.): *Rahmenkonzeption 2006 (RK UIS 2006)*. Universitätsverlag Ulm GmbH, 2006

39. LUBW Landesanstalt für Umwelt, Messungen und Naturschutz Baden-Württemberg: *UDO (Umwelt-Daten und -Karten Online)* [Online] [Zitat vom: 06. Februar 2013.] http://brsweb.lubw.baden-wuerttemberg.de/brs-web/index.xhtml.

40. disy Informationssysteme GmbH: *Cadenza Web*. [Online] [besucht am 13.03.2013] http://www.disy.net/produkte/cadenza/cadenza-web.html.

41. Imperia AG: *pirobase CMS*. [Online] [besucht am 13.03.2013] http://www.pirobase.de/Home-pirobase-CMS.

42. Microsoft Corporation: *Microsoft SharePoint 2010*. [Online] [besucht am 13.03.2013] http://sharepoint.microsoft.com/de-de/Seiten/default.aspx.

43. Bizer, Christian; Heath, Tom; Berners-Lee, Tim: *Linked Data. The Story so Far*. International Journal on Semantic Web and Information Systems, 2009. http://tomheath.com/papers/bizer-heath-berners-leeijswis-linked-data.pdf. [03.08.2011]

44. Campbell, Lorna M.; MacNeill, Sheila: *The Semantic Web, Linked and Open Data. A Briefing Paper*. 2010. http://wiki.cetis.ac.uk/images/1/1a/The_Semantic_Web.pdf. [03.08.2011]

45. Liferay Inc.: *Liferay Homepage*, [Online] http://www.liferay.com.

46. Bundesministerium des Innern (BMI): *Open Government Data Deutschland*. [Online verfügbar] http://www.bmi.bund.de/SharedDocs/Downloads/DE/Themen/OED_Verwaltung/ModerneVerwaltung/opengovernment.pdf?__blob=publicationFile.

47. Robinson, D.; Coar, K.: *The Common Gateway Interface (CGI) Version 1.1*. Request for Comment 3875 (RFC 3875), The Internet Society (2004)

48. Wikipedia: *Artikel zu Servlets*. [Online] http://de.wikipedia.org/wiki/Servlet.

49. Microsoft Corporation: *Internet Information Server (IIS) Homepage*. [Online] http://www.iis.net/.

50. Stefan Schwarz: *Übersicht über Web-basierte Dokumentenmanagementsysteme*. [Online] http://www.stefanux.de/wiki/doku.php/software/webbasiertes-dokumentenmanagement.

51. W3C: *Document Object Model (DOM)*. [Online] http://www.w3.org/DOM/.

52. W3C: *HTML 4.0.1 Specification*. [Online] http://www.w3.org/TR/REC-html40/.

53. Mozilla Developer Network: *Javascript Reference*. [Online] https://developer.mozilla.org/en-US/docs/JavaScript/Reference.

54. Wikipedia: *Artikel zu AJAX*. [Online] https://de.wikipedia.org/wiki/Ajax_(Programmierung).

55. Horn, T.: *Web-Services mit SOAP, WSDL und UDDI*. [Online] http://www.torsten-horn.de/techdocs/soap.htm#WebServices.

56. Heuser, O.; Holubek, A.: *Java Web Services in der Praxis: Realisierung einer SOA mit WSIT, Metro und Policies*. dpunkt Verlag; Auflage: 1., Aufl. (14. Dezember 2009)

57. Tilkov, S.: *REST und HTTP: Einsatz der Architektur des Web für Integrationsszenarien*. Dpunkt Verlag; Auflage: 2. akt. und erw. Auflage (30. Juni 2011)

58. W3C: *Web Design and Applications*. [Online] http://www.w3.org/standards/webdesign/.

59. WHATWG Group*: Homepage der WHATWG Group*. [Online] http://www.whatwg.org/.

60. WHATWG Group: *HTML Living Standard*. [Online] http://www.whatwg.org/specs/web-apps/current-work/multipage/.

61. W3C: *Scalable Vector Graphics (SVG)*. [Online] http://www.w3.org/Graphics/SVG/.

62. Google Inc.: *Google Maps API*. [Online] https://developers.google.com/maps/documentation/javascript/tutorial.

63. Roth, Gregor: *HTML5 Server-Push Technologies, Part 1*. [Online Artikel in Java.net] http://today.java.net/article/2010/03/31/html5-server-push-technologies-part-1.

64. Roth., Gregor: *Asynchronous HTTP and Comet architectures*. [Online Artikel in JavaWorld Magazine] http://www.javaworld.com/javaworld/jw-03-2008/jw-03-asynchhttp.html.

65. Weßendorf, Matthias: *WebSocket: Annäherung an Echtzeit im Web*. [Online Artikel bei Heise Developer] http://www.heise.de/developer/artikel/WebSocket-Annaeherung-an-Echtzeit-im-Web-1260189.html.

66. Oracle Corporation: *JSR 316: JavaTM Platform, Enterprise Edition 6 (Java EE 6) Specification*. [Online] http://jcp.org/en/jsr/detail?id=316.

67. The jQuery Foundation: *jQuery Homepage*. [Online] http://jquery.com/.

68. The jQuery Foundation: *jQuery UI Homepage*. [Online] http://jqueryui.com/.

69. Google Developers: *Google Web Toolkit (GWT)*. [Online] https://developers.google.com/web-toolkit/.

70. Vaadin Ltd.: *vaadin Homepage*. [Online] https://vaadin.com.

71. Burns, E.; Griffin, N.: *JavaServer Faces 2.0. The Complete Reference*. Mcgraw-Hill Professional; Auflage: 1 (1. Februar 2010)

72. Primfaces: *Primefaces JSF Bibliothek*. [Online] http://www.primefaces.org/.

73. JBoss!: *Richfaces JSF Bibliothek*. [Online] http://www.jboss.org/richfaces.

74. Open Source Matters, Inc.: *Joomla CMS Homepage*. [Online] http://www.joomla.org/.

75. Sarin, A.: *Portlets in Action*. Manning Publications; Pap/Psc edition (September 27, 2011)

76. OpenSocial Consortium: *OpenSocial Homepage*. [Online] http://opensocial.org/.

77. Josuttis, N.: *SOA in der Praxis: System-Design für verteilte Geschäftsprozesse*. Dpunkt Verlag; Auflage: 1. (1. Januar 2008)

78. Google Inc.: *Google Apps for Business*. [Online] http://www.google.com/intl/de/enterprise/apps/business/.

79. Google Inc.: *Google Maps Engine*. [Online] http://www.google.com/enterprise/mapsearth/products/mapsengine.html.

80. Zietz, Christian: *Da tut sich etwas innerhalb eines CMS... - Workflows (Teil 1)*. [Online] http://www.contentmanager.de/magazin/da_tut_sich_etwas_innerhalb_eines_cms_workflows_teil_1.html.

81. Google Inc.: *Einführung in Suchmaschinenoptimierung*. [Online PDF] http://static.googleusercontent.com/external_content/untrusted_dlcp/www.google.de/de/de/webmasters/docs/einfuehrung-in-suchmaschinenoptimierung.pdf.

82. Wikipedia: *Artikel zu Lightbox Konzept*. [Online] http://en.wikipedia.org/wiki/Lightbox_(JavaScript).

83. Wikipedia: *Artikel zu RSS*. [Online] http://de.wikipedia.org/wiki/RSS.

84, WiredMinds AG: *WiredMinds.Homepage* [Online] http://www.wiredminds.de/.

85. Sonntag, Michael: *Untersuchungen zur Personalisierung von Webseiten*. [Online] http://www.fim.uni-linz.ac.at/Publications/Aussendung10.98/Personalisierung.htm.

86. Bügel, Ulrich et al. (2011): *SUI für Umweltportale – Entwurf und prototypische Implementierung einer Architektur für die semantische Suche im Portal Umwelt-BW*. In: Mayer-Föll, Roland., Ebel, Renate, Geiger, Werner; Hrsg.: F+E Vorhaben KEWA – Kooperative Entwicklung wirtschaftlicher Anwendungen für Umwelt, Verkehr und benachbarte Bereiche in neuen Verwaltungsstrukturen, Phase VI 2010/11, Karlsruher Institut für Technologie, KIT Scientific Reports 7586, S. 21-32

87. Röwekamp, L.; Weßendorf, Matthias: *Java EE 6 Web Profile*. [Online] http://it-republik.de/jaxenter/artikel/Java-EE-6-Web-Profile-3529.html.

88. Esri*: ArcGIS for Server*. [Online] http://www.esri.com/software/arcgis/arcgisserver.

89. LongTail Ad Solutions. *JW Player Homepage*. [Online] http://www.longtailvideo.com/jw-player/.

90. YouTube Inc. *YouTube*. [Online] http://www.youtube.com/.

91. A9.com Inc.: *OpenSearch Standard Homepage*. http://www.opensearch.org/.

92. Dawson, F.; Stenerson, D.: *Internet Calendaring and Scheduling Core Object Specification (iCalendar)* (RFC 2445). [Online] http://www.rfc-editor.org/rfc/rfc2445.txt.

93. Alfresco Software, Inc.: *Alfresco Homepage*. [Online] http://www.alfresco.com/.

94. Raunig, Michael: *Digital Asset Management – Überblick über die Thematik und Überlegungen zum Einsatz eines entsprechenden Systems an der Universität Graz*. [Online] http://gams.uni-graz.at/fedora/get/o:gams-doku-082-18/bdef:PDF/get/.

95. Jasig: *Central Authentication System (CAS) Homepage*. [Online] http://www.jasig.org/cas.

96. OpenId Foundation: *Homepage von OpenID*. [Online] http://openid.net/.

97. Oauth Community: *Homepage von Oauth*. [Online] http://oauth.net/.

98. Sletten, Brian: *Resource-Oriented Architecture: The Rest of REST*. [Online] http://www.infoq.com/articles/roa-rest-of-rest.