in: Tagungsband der GI-Fachtagung 'Datenbanksysteme in Büro, Technik und Wissenschaft' (BTW'99), A. Buchmann (Hrsg.), Informatik aktuell, Freiburg, März 1999, Springer-Verlag, S. 20-37

iWebDB - Eine integrierte Web-Datenbank auf Basis objekt-relationaler DB-Technologie

Henrik Loeser
AG DBIS, FB Informatik
Universität Kaiserslautern
Postfach 3049, 67653 Kaiserslautern
E-Mail: loeser@informatik.uni-kl.de

Zusammenfassung: Das *World Wide Web* (WWW oder Web) ist auf dem Weg zum Standardkommunikationsmedium mit dem Web-Browser als Benutzerschnittstelle. Datenbanksysteme (DBS) dienen heute schon im großen Maße zur Datenbereitstellung für Anwendungen, die Daten in dynamisch erstellte HTML-Seiten einbinden. DBS werden bislang jedoch nur selten für die Verwaltung der gesamten Dokumente auf einem Web-Server verwendet, was insbesondere an der mangelnden Unterstützung der für die Verwaltung von Web-Inhalten (*Web Content Management*, WCM) relevanten Anforderungen liegt. Da aber das WCM und die Lösung der damit verbundenen Probleme ein Kernpunkt zukünftiger DBS-Aufgaben sein werden, sind entsprechende Lösungen gefordert. In diesem Beitrag stellen wir die Grundkonzepte von iWebDB vor, einem auf objekt-relationaler DB-Technologie basierenden WCM-System (WCMS). Aufgrund der Erweiterungsfähigkeit objekt-relationaler Datenbankverwaltungssysteme (ORDBVS) ist es nun möglich, die für das WCM benötigte Funktionalität in Form von DB-Erweiterungsmodulen in das DBS zu integrieren. iWebDB nutzt dies und versucht im Gegensatz zu vielen auf Einzelaspekte des WCM abzielenden Lösungen mit Hilfe eines integrativen Ansatzes einen einheitlichen Framework anzubieten.

Stichworte: Web Content Management, WWW, objekt-relationale DBVS

1 Einleitung

Seit den Anfangstagen des *World Wide Web* (WWW oder Web, [Cal95]) Anfang der 90er Jahre sind die Zahl der Benutzer und Informationsanbieter und damit auch die Menge der im Internet verfügbaren Daten in einem nicht für möglich gehaltenen Maße gewachsen. Bei der Größe des Web, inbesondere in Bezug auf die Menge der zugreifbaren Daten, rechnet man jedoch erst noch mit der eigentlichen "Revolution" [DM97], d. h. mit einer deutlichen Zunahme gegenüber heutigen Maßstäben. Die im Web verfügbaren Daten werden auf WWW-Servern bereitgestellt und von den Clients, den Web-Browsern, mittels HTTP (*HyperText Transfer Protocol*) angefordert. Zwar sind die HTML-Seiten (*HyperText Markup Language*), die im Web verwendeten und über sogenannte Hyperlinks verbundenen Dokumente, eigentlich rein textbasiert, doch können über spezielle Markierungen (*Tags*) auch Bilder, Videos, 3D-Grafiken, Programme und andere Komponenten eingebunden sowie über externe Applikationen dem Anwender zur Ansicht gebracht bzw. zur Nutzung bereitgestellt werden. Durch die Integration auch anderer Internet-Dienste in den als einheitliche Benutzeroberfläche dienenden Web-Browser kann zudem noch auf andere Datenquellen, wie z. B. FTP-Server (*File Transfer Protocol*), zugegriffen werden.

Durch das CGI (*Common Gateway Interface*, [CGI95]) und andere, später entwickelte Mechanismen können HTML-Dokumente dynamisch erzeugt und aus Programmaufrufen resultierende Daten eingebunden werden. Auf diese Weise können auch die in DBS gehaltenen Daten den Anwendern über HTML-Seiten zugänglich gemacht werden [Loe97, Loe98b]. Datenbanken (DB) sind heute ebenfalls ein fester Bestandteil von sogenannten *Web-Sites*, einem unter einer Adresse kontaktierbarem Web-Server mit seinen Dokumenten. DB werden zur Realisierung von interaktiven Web-Anwendungen eingesetzt, wobei das Einsatzgebiet von Anwendungen des *Electronic Commerce* über *Online Banking* bis hin zu Kommunikationsanwendungen (Chat, Diskussionsforum, Kundenbefragung etc.) reicht. Eine Aufgabe, für die sich DBS zwar aufgrund ihrer Eigenschaften (Mehrbenutzerbetrieb, Konsistenzsicherung etc.) prinzipiell gut eignen, mit der sie aber z. Zt. eher selten betraut sind, ist die Verwaltung der auf einer Web-Site bereitgestellten Inhalte (*Web Content Management*, WCM). Dies resultiert aus der mangelnden Unterstützung der auf einer Web-Site vorhandenen Datentypen. Ein Ansatz ist daher die Bereitstellung spezieller für diese Aufgabe entwickelter Systeme, die in [AMM98] auch *Web-Base Management Systeme* (WBMS) genannt werden. Wir wollen dagegen ein bestehendes ORDBVS um die benötigte WCM-Funktionalität erweitern.

Das Spektrum der im WWW existierenden und zu verwaltenden Web-Sites reicht von kleinen und mittleren privaten Internet-Seiten[1] bis hin zu sehr großen, aus tausenden von aufeinander abgestimmten HTML-Seiten bestehenden professionellen Unternehmenspräsentationen. Neben HTML-Seiten stehen dem Anwender oft auch andere Daten zum Abruf zur Verfügung (siehe Abb. 1). Hierzu gehören neben den direkt in die Seiten eingebundenen Bildern Postscript-, PDF- (*Portable Document Format*) und andere Dokumente, Java-Applets [GJS96] sowie evtl. auch Programm- oder Datenarchive (ZIP-Dateien). Aufgrund der Vielzahl der zu verwaltenden, meist multimedialen Datentypen (BLOB, *Binary Large OBject*) sowie der Anforderungen an solch ein WCMS war der Einsatz von DBS zur Verwaltung dieser Daten wegen des eingeschränkten Typangebots relationaler und eingeschränkter deskriptiver Anfragemöglichkeiten objektorientierter DBVS eher auf Ausnahmen beschränkt. Mit der Einführung objekt-relationaler DBVS (ORDBVS, [Kim96, Sto96]) stehen nun sowohl relationale wie auch objektorientierte Eigenschaften in einem DBVS zur Verfügung. Besonderes Merkmal von ORDBVS ist ihre *Erweiterungsfähigkeit*. Unter diesem Begriff fassen wir die Konzepte der benutzerdefinierten Datentypen (*user defined type*, UDT), der benutzerdefinierten Funktionen (*user defined functions*, UDF), der benutzerdefinierten Indexstrukturen sowie die Möglichkeit zur Einbindung externer Daten zusammen.

ORDBVS scheinen damit die geeignete Plattform für das WCM zu bilden. Wir wollen sie daher für die Entwicklung einer integrierten Web-Datenbank, iWebDB, nutzen. Dazu motivieren wir im folgenden Kapitel zunächst anhand der bei der Verwaltung von Web-Servern und ihrer Inhalte auftretenden Probleme die Anforderungen an ein

1. „Was früher der Fuchsschwanz am Manta war, ist heute die eigene Homepage *bei AOL*.“ Zitat in [CH97].

WCMS. In Kapitel 3 stellen wir nach einem Überblick über unser System iWebDB die einzelnen Komponenten auf konzeptioneller Ebene vor. Kapitel 4 gibt einen kurzen Überblick über die Implementierung von iWebDB und die dabei entstandenen Probleme. In Kapitel 5 grenzen wir unseren Ansatz zu anderen verwandten Arbeiten ab. Der Beitrag schließt mit einer Zusammenfassung sowie einem Ausblick auf weitere Forschungsarbeiten.

2 Anforderungen beim *Web Content Management*

An WCM-Systeme werden besondere Anforderungen gestellt, die wir in diesem Abschnitt erarbeiten wollen. Dazu betrachten wir zunächst die aktuelle Situation auf großen Web-Sites und die bei der Verwaltung von Web-Inhalten auftretenden Probleme. Davon ausgehend erstellen wir dann den Anforderungskatalog.

2.1 Aktuelle Situation und Probleme

Auf großen Web-Sites werden die HTML-Seiten und sonstigen Daten in der Regel nicht nur von einer Person, sondern von mehreren Personen und Interessensgruppen bereitgestellt und gewartet (siehe Abb. 1). Zudem existieren CGI-Programme für die diversen Einsatzgebiete, u. a. auch zum DB-Zugriff und zur Kommunikation mit einer evtl. vorhandenen Suchmaschine. Letztere findet sich auf Web-Sites, um den gezielten Zugriff auf lokal gehaltene Dokumente zu ermöglichen.

Trotz dieses auf den ersten Blick einfachen Szenarios kommt es auf entsprechend großen Web-Sites zu einer Reihe von Problemen. So ist es aufgrund der Zahl der vorhandenen HTML-Dokumente und der in ihre Bereitstellung involvierten Personen und Gruppen bzw. Abteilungen schwierig, alle Informationen und Dokumente aktuell, konsistent und sowohl inhaltlich als auch vom Layout her aufeinander abgestimmt zu halten. So werden in der Regel die HTML-Seiten auf einem Web-Server von unterschiedlichen Anbietern zur Verfügung gestellt (siehe Abb. 1). Hierdurch bedingt kommt es immer wieder vor, daß Hyperlinks auf andere lokale Seiten inkonsistent sind, d. h. auf nicht mehr vorhandene Dokumente verweisen. Diese “toten” Verweise

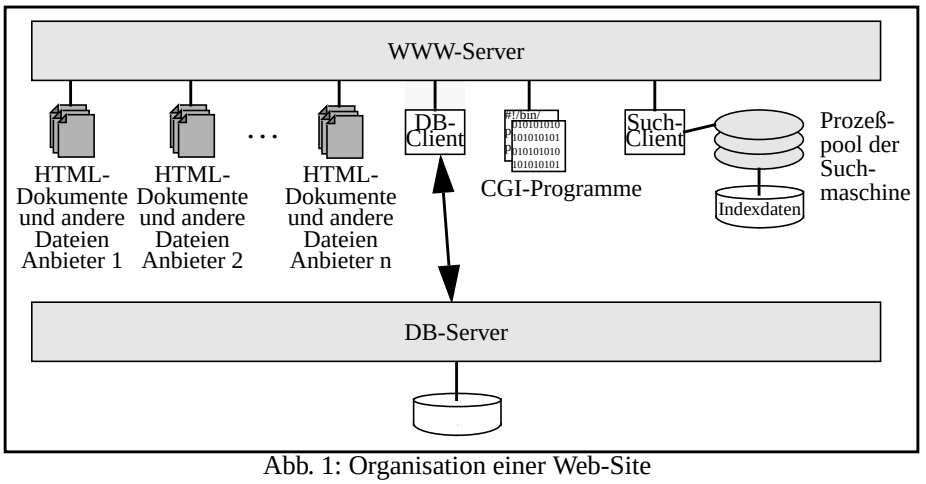

Abb. 1: Organisation einer Web-Site

können zwar über entsprechende Werkzeuge, sogenannte *Link Checker*, entdeckt werden, jedoch gelingt dies nur für Dokumente, die über einen im Wurzeldokument beginnenden Pfad erreichbar sind. Ausschließlich über Quereinstiege erreichbare Dokumente und Verzeichnisbäume werden in der Regel nicht überprüft. Ein weiteres Problem vor allem großer Web-Sites ist die Suche nach gewünschten Dokumenten. Zwar werden hierzu Inhaltsverzeichnisse zusammengestellt, die eine gezielte Navigation erlauben, und Suchmaschinen installiert, um die Suche anhand von Stichworten zu ermöglichen, jedoch werden die Indexdaten der Suchmaschine oftmals nur periodisch, z. B. wöchentlich, und nicht nach Dokumentänderungen aktualisiert. Zudem werden wie beim Link Checker wiederum nur über Quereinstiege erreichbare Dokumente nicht erfaßt. Durch den veralteten und nicht vollständigen Suchindex bedingt lassen sich nicht immer die gewünschten und auch vorhandenen Dokumente auffinden.

Ein weiteres Problem ist die einheitliche WWW-Präsentation. Zwar werden sogenannte *Style Guides* und auch Dokumentvorlagen für die Gestaltung von HTML-Seiten geschaffen, deren Einhaltung läßt sich jedoch nur schwerlich überwachen. Weitere Probleme, die ebenfalls aus der Vielzahl der Informationsanbieter resultieren, entstehen bei der Verwaltung der HTML-Dokumente bzw. den ihnen zugeordneten Namensräumen. So werden die Rechte zum Anlegen und Verändern von Web-Seiten in der Regel über Schreibrechte auf der Verzeichnisebene eines Dateisystems unter Nutzung von Benutzer- und Gruppen-IDs geregelt. Hierbei kann es jedoch durch Zugehörigkeit mehrerer Benutzer zu einer Gruppe und mangelnder Absprache zum Verlust von Änderungen (*Lost Updates*) kommen. Zudem ist die problemlose Zuteilung eines Verzeichnisses für die Benutzung durch unterschiedliche Gruppen nicht möglich, so daß eigentlich inhaltlich zusammengehörende Dokumente, z. B. die Kurzvorstellung aller Arbeitsgruppen eines Fachbereichs, auf unterschiedliche Verzeichnisse aufgetrennt werden müssen.

2.2 Anforderungen

Basierend auf den gerade beschriebenen Problemen bei der Verwaltung bzw. Bereitstellung einer Web-Site lassen sich folgende Anforderungen an ein WCMS formulieren:

- ***Konsistenz***: Lokale Hyperlinks sollten beim Einfügen eines neuen, dem Ändern oder Löschen eines vorhandenen Dokumentes überprüft und gegebenenfalls entsprechende Schritte zur Sicherung der Konsistenz unternommen werden. Konsistenz bedeutet in diesem Fall, daß keine Links auf nicht (mehr) vorhandene lokale Dokumente existieren.
- ***Einheitlichkeit***: Da auf der Web-Site eines Unternehmens oder eines Anbieters in der Regel ein einheitliches Layout angestrebt wird (“Corporate Identity”), sollte es möglich sein, beim Einfügen oder Verändern eines HTML-Dokumentes automatisch die Konformität mit vorhandenen Regeln überprüfen zu lassen.
- ***Einbindung von Nicht-HTML-Dokumenten***: Da neben HTML-Dokumenten auch andere Datei- bzw. Dokumenttypen auf einem Web-Server bereitgestellt werden, sollten auch diese verwaltet werden können. Hierzu gehören neben Dateien von Textverarbeitungsystemen auch Dokumente in Austauschformaten wie *Postscript*

oder *Portable Document Format* (PDF) sowie Daten- und Programmarchive (ZIP-Dateien) und Java-Programme.

- ***Einbindung "normaler" Datenbanken***: Neben den reinen HTML-Seiten mit ihren Komponenten sind aber auch die für die Realisierung von Web-Anwendungen benötigten Geschäftsdaten wichtig, so daß der Einbindung solcher Datenbanken ebenfalls eine besondere Bedeutung zukommt.
- ***Suche auf dem aktuellen Dokumenten-/Datenbestand***: Für das einfache Auffinden eines verwalteten Dokumentes sollte eine Suchmöglichkeit angeboten werden. Anders als bei externen Suchmaschinen sollte die Suche auf einem aktuellen, den Datenbestand wiedergebenden Suchindex möglich sein.
- ***Einfache Verwaltung***: Die Nutzung einer DB-basierten Lösung für die Verwaltung von Web-Inhalten sollte nicht schwieriger sein als eine Lösung auf Basis des Dateisystems. Dies betrifft zum einen das Erstellen und die Wartung von Dokumenten, zum anderen aber auch die Administration der Benutzer, Gruppen sowie der eigentlichen Web-Anwender (Zugriffrechte und Autorisierung).

Somit existiert eine Reihe von Anforderungen, die es beim Entwurf eines WCMS zu berücksichtigen gilt.

3 iWebDB

Im folgenden wollen wir iWebDB, unser integriertes WCMS, kurz vorstellen und anschließend dann genauer auf die einzelnen Komponenten eingehen.

3.1 Überblick

iWebDB basiert auf der Nutzung objekt-relationaler Datenbankverwaltungssysteme (ORDBVS, [Kim96, Sto96]) und setzt deren Erweiterungsfähigkeit zur Erfüllung seiner Aufgaben ein [Loe98a]. Wie schon angesprochen, beinhaltet "Erweiterungsfähigkeit" die Konzepte der UDTs, UDFs und benutzerdefinierten Indexstrukturen sowie die Möglichkeit zur Einbindung externer Daten. iWebDB nutzt alle Formen der Erweiterungsfähigkeit zur Bereitstellung der geforderten Funktionalität und faßt diese in entsprechenden Erweiterungsmodulen zusammen. Je nach Hersteller heißen diese Module *DataBlade* (Informix), *Extender* (IBM) oder *Cartridge* (Oracle).

Das System iWebDB besteht aus vier unterschiedlichen Komponenten, die aufeinander aufbauen (siehe Abb. 2):

- ***iWebDB/HTML***: Dieses Modul stellt die Grundfunktionalität zur DB-internen Dokumentenverwaltung zur Verfügung. Hierzu werden neben entprechenden als

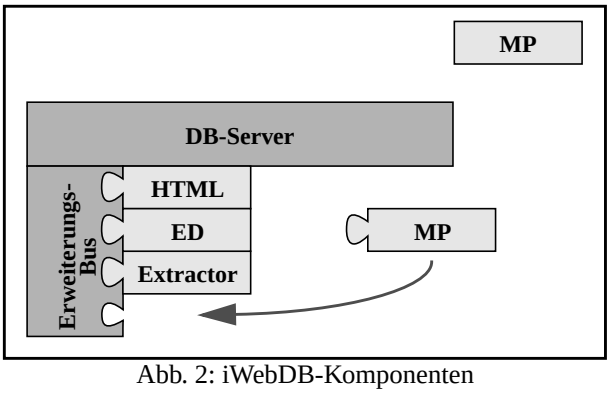

Abb. 2: iWebDB-Komponenten

UDT realisierten Datentypen zur Aufnahme der unterschiedlichen Dokumente einfache Analysefunktionen in Form von UDFs bereitgestellt. Diese Funktionen sind die Basis für die Realisierung der geforderten Konsistenzsicherung. Über Erweiterungsschnittstellen kann auch die Überwachung eines einheitlichen Layouts eingebunden werden. iWebDB/HTML kann unabhängig von anderen iWebDB-Komponenten verwendet werden.

- ***iWebDB/ED*** (*external data*): Über das Modul iWebDB/ED wird der Zugriff auf unterschiedliche externe Datenquellen über die SQL-Schnittstelle verfügbar gemacht. Mit der hierdurch ermöglichten Einbindung von DB-extern gespeicherten Nicht-HTML-Dokumenten und das Kopieren von Inhalten zwischen Dateisystem und DBS schafft iWebDB/ED die Grundlage für die einfache Administration der gespeicherten Web-Inhalte. Die externen Datenquellen werden über die Angabe eines URI (*Uniform Resource Identifier*) spezifiziert, so daß die vom Web vertraute Notation auch in Suchanfragen verwendet werden kann. Neben dem Zugriff auf das lokale Dateisystem (file:) werden auch FTP- (*File Transfer Protocol*, ftp:) und Web-Server (http:) unterstützt. Zusätzlich zur Datenbereitstellung werden dazu einfache Verarbeitungsfunktionen, wie z. B. die Klassifikation von Dateien aufgrund ihrer Suffixe, angeboten. Die von iWebDB/ED bereitgestellte Funktionalität setzt auf den von iWebDB/HTML realisierten Datentypen auf und stellt die Daten dem Anwender bzw. DB-Server im selben Format zur Verfügung.

- ***iWebDB/Extractor***: Über dieses Modul werden iWebDB/HTML und iWebDB/ED um Analysefunktionalität in Form von UDFs ergänzt. Zudem wird über einen entsprechenden Suchindex die Speicherung sowie der spätere, schnellere Zugriff auf die zuvor bei der Dokumentanalyse extrahierten Informationen ermöglicht. Für die Bereitstellung des Suchindex wird das Konzept der benutzerdefinierten Indexstrukturen genutzt. Durch die Funktionalität von iWebDB/Extractor wird die schnelle und komfortable Suche auf dem aktuellen Dokumentenbestand ermöglicht.

- ***iWebDB/MP*** (*macro processor*): Im Gegensatz zu den anderen Modulen ist iWebDB/MP kein direktes Server-Erweiterungsmodul, kann aber in solchen zum Einsatz kommen. Der von iWebDB/MP bereitgestellte Makroprozessor, der zur Web-Anbindung der “iWebDB” dient und spezielle, in HTML-Dokumente eingebettete Markierungen (*Tags*) verarbeitet, ist als Klassenbibliothek realisiert und kann daher sowohl in CGI-Programmen bzw. Web-Server-Modulen als auch in DB-Server-Erweiterungsmodulen eingesetzt werden. Mit Hilfe des Makroprozessors ist es möglich, auf DB zuzugreifen und DB-Kommandos auszuführen, Ergebnisse von (Java-) Funktionsaufrufen in die HTML-Seite einzubinden sowie Programmflußsteuerung und Variablenverarbeitung durchzuführen.

Nach der kurzen Vorstellung der einzelnen Komponenten von iWebDB wollen wir sie nun jeweils genauer diskutieren.

3.2 iWebDB/HTML

Als zentrale Komponente für die DB-interne Dokumentverwaltung dient das Modul iWebDB/HTML. Es stellt neben geeigneten Datentypen auch Verarbeitungs- und einfache Analysefunktionen zur Verfügung.

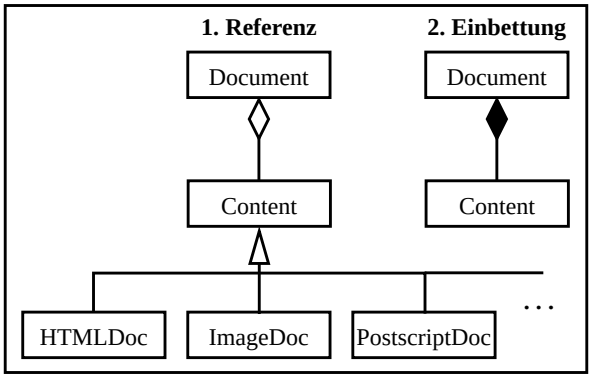

Abb. 3: Typhierarchie

Datentypen

iWebDB/HTML stellt, wie in Abb. 3 zu sehen, mehrere Datentypen bereit. Wichtigster unter diesen ist *Content*, der für die Aufnahme eines Dokumentes zuständig ist. Neben dem reinen Dokumenteninhalt wird auch der Dokumenttyp (HTML, Postscript, Bild etc.) vermerkt, um später gezielter die jeweils passenden Funktionen anwenden zu können. Als Subtypen von Content werden zudem HTMLDoc, ImageDoc, PostscriptDoc und weitere Typen bereitgestellt, die auf den jeweiligen Dokumenttyp zugeschnittene Attribute besitzen. Während z. B. für HTML-Seiten Titel, Überschriften und Hyperlinks charakteristisch sind, sind bei einem Bild u. a. dessen Dimension und Format (GIF, JPEG, TIFF, BMP, etc.) von Interesse.

Neben der Typhierarchie mit *Content* als Supertyp stellt iWebDB/HTML auch den Typ *Document* bereit. In ihm ist in Abhängigkeit von der DBVS-spezifischen Modellierung *Content* entweder über eine Referenz oder als direktes Attribut eingebettet. Document stellt alle zur Simulation einer Datei benötigten Attribute zur Verfügung. Dies sind u. a. der Name, der Pfad, der Besitzer und das Änderungsdatum des Dokumentes bzw. der Datei. Ein besonderes Attribut des Typs Document ist *indexable*, welches angibt, ob das entsprechende Dokument von der integrierten Suchmaschine indexiert werden darf. Dies ist insbesondere von Vorteil, wenn nicht für die gesamte Öffentlichkeit bestimmte Informationen auf einer HTML-Seite stehen oder wenn z. B. sogenannte *Header* oder *Footer*, Standardelemente eines jeden HTML-Dokuments einer Web-Site, verwaltet werden.

Die bereitgestellten Attribute können, wie in Abb. 4 gezeigt, bei der Administration der Dokumente zur einfachen Suche genutzt werden. Während bei Anfrage 1 auf den Inhalt[2] der HTML-Seite mit dem gesamten Dateipfad "/AG-Haerder/LoeserD.html" zugegriffen wird, liefert die zweite Anfrage[3] den Pfad- und Dateinamen aller innerhalb der letzten 10 Tage angelegten und veränderten Dokumente zurück, deren Besitzer der Benutzer "loeser" ist.

---

2. Diese Anfrage verwendet die direkte Einbettung von *Content* in *Document*. Die Tabelle doctable sei vom Typ Document, htmlCont vom Typ HTMLDoc.
3. Die Syntax "CURRENT YEAR TO DAY" entspricht der des Informix Dynamic Servers [Inf97a].

Verarbeitungsfunktionalität

Neben den generell für eigene Datentypen benötigten Hilfsfunktionen existieren noch weitere Verarbeitungs- und Analysefunktionen. So können aus HTML-Seiten der Titel (<TITLE>) sowie die im Dokument enthaltenen Überschriften (<H1> bis <H6>), Hyperlinks (<A HREF>) und Bildverweise (<IMG>) extrahiert werden. Für die Konsistenzkontrolle werden die beiden letzteren verwendet. Bei der Kontrolle werden die lokalen Hyperlinks und Bildverweise extrahiert und dann das Vorhandensein der referenzierten Dokumente überprüft.

Es sind verschiedene Formen der Überwachung möglich. So ist zum einen die Kontrolle der Verweise am Ende einer Änderungstransaktion, zum anderen aber auch bei der Bereitstellung (s. u., "iWebDB-Administration") der HTML-Dokumente sinnvoll. Bei einer Kombination beider Ansätze erhält der Benutzer beim Einfügen und Ändern von Dokumenten eine direkte Rückmeldung über potentielle Fehler, die Bereitstellung fehlerhafter HTML-Seiten wird vermieden.

```
1. SELECT page
   FROM doctable
   WHERE path='/AG-Haerder'
   AND fname='LoeserD.html';

2. SELECT path || fname
   FROM doctable
   WHERE fdate > CURRENT YEAR TO DAY
   - INTERVAL(10) DAY TO DAY
   AND owner='loeser';

3. SELECT d.path || d.fname
   FROM htmlCont h, doctable d
   WHERE h.id=d.id AND
   'Publikationen' IN h.headings;
```

Abb. 4: Beispielanfragen: iWebDB/HTML

Durch die Ablage der bei der Dokumentenanalyse extrahierten Daten in Attributen des Typs *HTMLDoc* wird ein schnellerer Zugriff erreicht, da keine Berechnung zum Anfragezeitpunkt erfolgen muß. Um veraltete Suchdaten zu vermeiden, werden die Attribute über Trigger auf dem aktuellen Stand gehalten. Eine Nutzung der Attribute innerhalb von SQL-Anfragen ist in Abb. 4 dargestellt. So wird in der dritten Anfrage der vollständige Pfad von allen HTML-Dokumenten zurückgeliefert, die eine Überschrift "Publikationen" besitzen.

Da sich die Layoutkontrolle nicht auf die reine Existenzprüfung von besonderen Strukturelementen innerhalb eines HTML-Dokumentes beschränken lassen kann, sind besondere Analysefunktionen notwendig, um die Kontrolle möglichst weit zu automatisieren. Allerdings ist damit nur die Kontrolle von Stilelementen, die erfolgte Einbindung von Stildateien sowie eine einfache Strukturanalyse realisierbar. Letztlich kann keine vollständig automatische Layoutkontrolle erfolgen, die Analysefunktionen können nur Hinweise auf potentielle Abweichungen geben.

3.3 iWebDB/ED

Durch das Modul iWebDB/ED (*external data*) wird von iWebDB der Zugriff auf externe Informationen realisiert. Auf diese Weise soll zum einen die nahtlose Integration bestehender Dokumenten-Sammlungen (Dateisystem des Web-Servers) erreicht und eine Anfragemöglichkeit via SQL auch auf diesen angeboten werden, zum ande-

ren die Arbeit mit dateibasierten Werkzeugen weiterhin ermöglicht werden. Zudem ist die Bereitstellung des Dateisystems in den auch DB-intern verwendeten Datenstrukturen eine geeignete Basis zur Administration (s. u.) der DB-intern verwalteten Dokumente.

Technik

Grundlage bei der Realisierung von iWebDB/ED ist die Fähigkeit von ORDBVS zur Einbindung externer Daten. Diese wird nun genutzt, um Dokumente aus einem Dateisystem, von einem FTP- oder Web-Server in Form einer Tabelle (siehe Abb. 5, Anfrage 1 und 2, Tabelle *website*) dem Anwender zur Verfügung zu stellen. Bei der Definition der Tabelle oder beim Zugriff auf sie (Attribut *path*, siehe zweite Anfrage) kann dabei über eine URI-Notation die Art des externen Datenzugriffs spezifiziert werden. Je nach Protokolltyp (file:, ftp: oder http:) wird dann von iWebDB über das jeweilige Medium auf das entsprechende Verzeichnis zugegriffen und anhand der Qualifikationsbedingungen werden die gewünschten Dokumente aufgesucht und mit ihren Attributen in Form einer Tabelle dem DB-Server als Zwischenergebnis zur Weiterverarbeitung bereitgestellt.

```
1. SELECT path || fname
   FROM website
   WHERE type='html'
   AND 'Loeser' IN headings(page)
   AND path LIKE '/AG-Haerder%';

2. SELECT ... FROM website ...
   AND path LIKE
   'http://www.uni-kl.de/AG-Haerder%';

3. INSERT INTO docs
   SELECT * FROM extDocs
   WHERE fdate > CURRENT YEAR TO DAY
   - INTERVAL(5) DAY TO DAY;

4. INSERT INTO copyDocs
   SELECT * FROM extWeb
   WHERE type='gif' AND path LIKE
   'http://www.uni-kl.de/AG-Haerder%';

5. INSERT INTO extDocs
   SELECT * FROM docs
   WHERE type='html'
   AND owner='loeser';
```

Abb. 5: Beispielanfragen: iWebDB/ED

iWebDB/ED verwendet zur Datenbereitstellung die von iWebDB/HTML verfügbar gemachten Datentypen *Document* und *Content*. Anders als bei der Dokumentenspeicherung in der DB stehen einige Attribute, wie z. B. der Dokumenttyp sowie beim Zugriff mittels HTTP auch der Besitzer und das Erstellungsdatum nicht (direkt) zur Verfügung, sondern müssen in diesem Fall anhand der Dateiendung vermutet oder auf den Wert NULL gesetzt werden.

iWebDB-Administration

Neben dem offensichtlichen Zweck, der Einbindung externer Datenquellen, kann durch die Nutzung von iWebDB/ED eine einfache Administration sowohl der extern als auch der intern gehaltenen Dokumente und ihrer Bestandteile erreicht werden. Wir stellen im folgenden kurz einige der durch iWebDB/ED gewonnenen Möglichkeiten vor. Durch das Bereitstellen interner und externer Daten in Tabellenform mit derselben Struktur können die gewünschten Dokumente mit Hilfe von SQL-Anweisungen auf einfache Weise zwischen dem Dateisystem und dem DB-Server ausgetauscht werden.

Mit der in Anfrage 3 (siehe Abb. 5) dargestellten SQL-Anweisung werden alle in den letzten fünf Tagen veränderten Dokumente in die DB übertragen. Anfrage 4 präsentiert einen einfachen Weg, um alle GIF-Bilder vom Web-Server "www.uni-kl.de" aus dem Verzeichnis "AG-Haerder" samt Unterverzeichnissen in die DB zu übertragen. Das in der Anfrage 5 dargestellte Kommando kopiert schließlich alle dem Benutzer "loeser" gehörenden HTML-Dokumente aus der "iWebDB" in das Dateisystem.

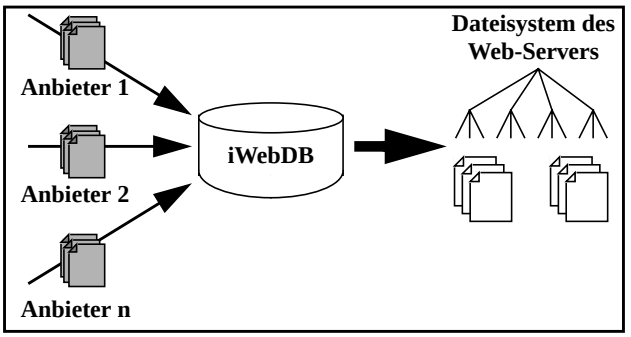

Abb. 6: Administration der HTML-Seiten

```
1. INSERT INTO docs VALUES
   ('Teilnehmer.html','/BTW99/'...);

2. CREATE VIEW btwDocs
   OF TYPE document AS SELECT *
   FROM docs
   WHERE path LIKE '/BTW99/%'
   WITH CHECK OPTION;

3. CREATE VIEW dbisDocs
   OF TYPE document AS SELECT *
   FROM docs
   WHERE path LIKE '/AG-Haerder/%'
   OR path='/'
   AND fname LIKE 'dbis%';
```
Abb. 7: Administration: Beispielanweisungen

Mittels des gerade vorgestellten Datenaustausches zwischen iWebDB und Dateisystem kann iWebDB als zentrales, intelligentes *Repository* für die Verwaltung von Web-Inhalten dienen (siehe Abb. 6). Die einzelnen Anbieter fügen ihre Dokumente unter Angabe des Namens und des Pfads mit Hilfe entsprechender Werkzeuge in die DB ein. So wird mittels der Anfrage 1 (Abb. 7), die von einem solchen Werkzeug auf der iWebDB ausgeführt wird, ein Dokument "Teilnehmerliste.html" im Verzeichnis "/BTW99" eingefügt. Die HTML-Seiten können zu festgelegten Zeitpunkten oder auf Anforderung, z. B. nach einer Aktualisierung des Dokumentenbestandes, im Dateisystem des Web-Servers abgelegt werden. Diese Materialisierung des intendierten Dokumentenbaumes ist insbesondere im Hinblick auf die Zugriffsgeschwindigkeit von großem Vorteil. Es werden zwar die Eigenschaften eines DB-basierten WCMS genutzt, jedoch muß dabei nicht auf die dateibasierten Geschwindigkeitsvorteile verzichtet werden. Durch Erweiterungen von iWebDB/MP ist es zudem möglich, beim Auslesen der HTML-Seiten in das Dateisystem des Web-Servers die Dokumente aus Teilkomponenten zusammenzusetzen und z. B. um Standard-Header und -Footer sowie um eine Navigationsleiste zu ergänzen. Auf diese Weise werden *Server Side Includes* (SSIs) des Web-Servers nicht mehr benötigt, was die Zugriffsgeschwindigkeit erhöht und die Belastung des Web-Servers verringert.

Während im Dateisystem der Zugriff gruppen- und benutzerweise über Schreibrechte auf Verzeichnisebene[4] geregelt wird, werden bei iWebDB Rechte auf Sichten an die einzelnen Informationsanbieter vergeben (siehe Abb. 7). Während, wie in der zweiten Anweisung gezeigt, der Administrator der BTW-Seiten Schreibrechte auf der Sicht btwDocs erhalten könnte, die ihm das Anlegen von HTML-Dokumenten im Verzeichnis "/BTW99" und den Unterverzeichnissen erlauben würde, können für andere Tätigkeiten noch feingranularere Rechte vergeben werden. So wird durch die Anweisung 3 eine Sicht erzeugt, mit der Rechte an der Verzeichnishierarchie "/AG-Haerder/" sowie an mit dem Präfix "dbis" beginnenden Dateien im Hauptverzeichnis assoziiert werden können. Über die SQL-Anweisungen GRANT/REVOKE können diese Rechte jetzt an die entsprechenden Web-Administratoren vergeben bzw. diesen wieder entzogen werden.

3.4 iWebDB/Extractor

Für die Realisierung einer integrierten Suchmaschine und die damit verbundene Bereitstellung aktueller Suchdaten ist das Modul iWebDB/Extractor zuständig. Es ergänzt die von den bereits diskutierten Komponenten angebotene rudimentäre Analysefunktionalität um weitere Funktionen zur Extraktion von Informationen und stellt einen geeigneten Suchindex zur Aufnahme dieser Daten zur Verfügung, so daß ein schnelles Auffinden von Dokumenten ermöglicht wird.

Analysefunktionen

Neben dem zentralen Dokumenttyp HTML werden von iWebDB/Extractor auch andere Dokumentarten unterstützt, wie z. B. Postscript- oder auch PDF-Dateien. Zusätzlich zu den schon bekannten Möglichkeiten zur Extraktion von Titel, Überschriften und Querverweisen (Hyperlinks und Bilder) aus HTML-Seiten werden Funktionen zur Auswertung weiterer HTML-Markierungen angeboten. So können u. a. Anfragen auf die Meta-Informationen (<META>) und auch die logischen und physischen Texthervorhebungen (<EM>, <STRONG>, ..., <B>, <I>, ...) gestellt werden. Für die Analyse von Postscript-, PDF- und anderen Dokumenten werden, bedingt durch ihre Dokumentstruktur und Formatierung, Funktionen zur Volltextsuche angeboten. Die Implementierung soll sich an bestehenden Analysefunktionen, wie z. B. der von Harvest [Har98], orientieren und in Form von UDFs bereitgestellt werden.

Suchdatenbereitstellung

Zur Realisierung einer Suchfunktion auf den von iWebDB verwalteten Daten ist es notwendig, die für den angebotenen Umfang der Suche notwendigen Informationen zu extrahieren und zum Zwecke des schnellen Zugriffs dauerhaft in entsprechenden Attributen vorzuhalten. Diese können entweder Attribute von Subtypen der bereitgestellten Grundtypen oder Spalten für diesen Zweck eingerichteter Suchtabellen sein und werden durch die über Trigger aufgerufenen Analysefunktionen bei Lösch- und Änderungsaktionen automatisch aktualisiert. Eine weitere Alternative stellt die Nutzung

4. Über bei einigen Betriebssystemen vorhandene ACLs (*Access Control Lists*) sind evtl. feinere Rechtevergaben möglich.

sogenannter Funktionsindizes vor, bei denen anstelle von einem oder mehreren Tabellenattributen das Funktionsergebnis indiziert wird. Dies bedeutet, daß bei einem Index auf einer Analysefunktion nach Dokumentänderungsoperationen das Ergebnis (neu-) berechnet wird und somit bei einem Zugriff im Rahmen einer SQL-Anfrage schon vorliegt. Der Vorteil der Verwendung von Funktionsindezes liegt in der automatischen Aktualisierung durch den DB-Server und der einheitlichen Verwendung von Funktionsnamen im Rahmen von SQL-Anfragen.

Zum schnelleren Zugriff auf die Textattribute und zur Unterstützung einer späteren Volltextindizierung stellt iWebDB/Extractor besondere Indexstrukturen bereit. Sie werden unter Nutzung des Konzeptes der benutzerdefinierten Indexstrukturen in den DB-Server integriert und stehen somit wie DBVS-interne Indexstrukturen, z. B. B^*-Bäume, zur Verfügung.

Suche

Aufbauend auf den extrahierten und in einer der diskutierten Formen bereitgestellten Suchdaten kann nun mittels SQL nach Dokumenten gesucht werden (siehe Abb. 8). So wird mit Hilfe der ersten Anfrage nach allen HTML-Dokumenten in der Verzeichnishierarchie unter "/AG-Haerder" gesucht, in denen das Wort "Loeser" als Adresse formatiert ist. Mit der zweiten Anfrage wird nach allen verwalteten Postscript-Dateien gesucht, die den Begriff "World Wide Web" enthalten.

```
1. SELECT path || fname
   FROM doctable
   WHERE path LIKE '/AG-Haerder%'
   AND type='html'
   AND 'Loeser' in address(page);

2. SELECT path || fname
   FROM doctable
   WHERE type='ps'
   AND 'World Wide Web' in
   fulltext(papge);
```

Abb. 8: Beispielanfragen: iWebDB/Extractor

3.5 Makroprozessor (iWebDB/MP)

Für die Anbindung einer "iWebDB" an das WWW, insbesondere für die Einbindung von Geschäftsdaten zur dynamischen Erzeugung von HTML-Seiten, ist ein DB-Zugriff mittels eines CGI-Programms, einer Web-Server-Erweiterung oder mittels anderer Mechanismen notwendig [Loe98b]. Die für die Realisierung eines solchen Programms oder einer Erweiterungsfunktion benötigte Grundfunktionalität wird von iWebDB/MP in Form eines Makroprozessors zur Verfügung gestellt. Dieser liegt in einer Funktionsbibliothek vor und kann daher bei der Entwicklung der benötigten Zugriffskomponente als Basis dienen. Im folgenden wollen wir einen kurzen Überblick über die bereitgestellte Funktionalität geben, welche durch Einbettung spezieller Markierungen in ein HTML-Dokument genutzt werden kann. Alle Makros von iWebDB/MP werden durch ein "?" eingeleitet, um sie beim Parsen von normalen HTML-Markierungen besser unterscheiden zu können, und beginnen mit dem Präfix "iWeb" (siehe Abb. 9). Sie können auch als Markierungen im Rahmen von XML (*Extensible Markup Language*, [BPS98]) interpretiert werden. Die von iWebDB/MP zur Verfügung gestellte Funktionalität orientiert sich weitgehend an den in [Loe97, Loe98b] beschriebenen Werkzeugen. Jedoch werden mit der Bereitstellung in Form

einer Funktionsbibliothek sowie durch die Funktionsauswertung und den Mehr-DB-Zugriff (s. u.) interessante neue Eigenschaften ergänzt bzw. in einem Werkzeug vereinigt.

Im in Abb. 9 dargestellten Beispiel werden alle bisher angemeldeten Teilnehmer der BTW'99 in einer HTML-Seite ausgegeben, wobei die Teilnehmerdaten aus der DB "btw" und ein am Seitenende eingebundenes HTML-Dokument aus der DB "webdata" stammen. Zudem wird über einen Funktionsaufruf das aktuelle Datum inklusive der Uhrzeit ausgegeben.

```
<HTML>
<HEAD><TITLE>Anmdeldungen</TITLE></HEAD>
<BODY><H1>BTW-Anmeldungen</H1><HR NOSHADE>
<!--- Spezifikation der Datenbanken --->
   <?iWebDB NAME=data DBNAME=btw
   DRIVER="..." URL="...">
   <?iWebDB NAME=webObjects DBNAME=webdata
   DRIVER="..." URL="...">
<!--- Ausgabetabelle beginnen --->
<table><TR><TH>Name<TH>Vorname<TH>Ort</TR>
   <?iWebSQL STMT="SELECT name,vorname,ort
   FROM teilnehmer;" DB=data>
<!--- Ergebnisausgabe --->
<TR><TD>$V1<TD>$V2<TD>$V3</TR><?/iWebSQL>
</TABLE>
<HR NOSHADE>
<!--- Datum u. Uhrzeit ausgeben--->
Stand: <?iWebFUNC CLASS=printDateTime>
printDateTime()<?/iWebFUNC>
<!--- Footer einbinden --->
   <?iWebSQL STMT="SELECT page FROM docs
   WHERE fname='footer.html' and path='/';"
   DB=webObjects>$V1<?/iWebSQL>
</BODY></HTML>
```

Abb. 9: Beispiel: iWebDB/MP

DB-Zugriff

Bei der Realisierung einer DB-Zugriffsmöglichkeit im Rahmen von iWebDB sind verschiedene Aspekte von Bedeutung. Zum einen muß beachtet werden, daß Geschäftsdaten und weitere Informationen in der Regel in unterschiedlichen DB abgelegt sind und somit für eine Nutzung aller Daten der gleichzeitige Zugriff auf mehrere DB notwendig ist (siehe Abb. 9, "iWebDB"[5] und "iWebSQL"). Zum anderen ist durch die Bereitstellung des Makroprozessors in Form einer Funktionsbibliothek keine Bindung an eine bestimmte Aufrufschnittstelle sinnvoll, so daß alle DB-spezifischen Aufrufe an eine Abbildungsschicht, d. h. eine DBVS-neutrale Pseudo-API (*Application Programming Interface*), geschickt werden müssen, die diese Aufrufe in die konkret verwendete DB-Schnittstelle umsetzt und dabei die jeweiligen Einschränkungen beachtet. So ist z. B. bei Ausführung des Makroprozessors im Rahmen einer server-internen UDF der Zugriff auf andere DB nicht möglich.

Durch die angestrebte Realisierung von iWebDB/MP in Java sowie die Absicht der DBVS-Hersteller, JDBC (*Java DataBase Connectivity*, [WH98]) bzw. SQLJ [SQLJ] Server-seitig zu unterstützen, ist eine einfache Entwicklung von sowohl JDBC-basierten client-seitigen CGI-Programmen als auch server-intern ablaufenden UDFs auf Basis der bereitgestellten Funktionsbibliothek möglich.

5. *DRIVER* und *URL* sind für die Nutzung eines JDBC-Treibers benötigte Parameter.

Funktionsauswertung

Um die Funktionalität von iWebDB/MP erweitern zu können, wird die Einbindung von Funktionsergebnissen von Java-Funktionen in die HTML-Seiten ermöglicht. Hierzu wird über die Funktionsmarkierung (<?iWebFUNC>, siehe Abb. 9), welche die Spezifikation der Class-Datei als Parameter hat, der Funktionsaufruf mit seinen Argumenten gekapselt. Bei der Makroverarbeitung wird die Class-Datei dynamisch geladen, die Funktion mit den angegebenen Parametern nach einer Variablensubstitution aufgerufen und das Ergebnis anstelle der Markierungen in die HTML-Seite eingefügt bzw. der Variablenverarbeitung übergeben. Da Java-Funktionen sowohl von iWebDB/MP als auch im DB-Server ausgeführt werden können, ist eine einfache Verlagerung von Funktionalität und somit auch eine Verteilung der Rechenlast möglich.

Variablenverarbeitung und Programmflußsteuerung

Die Variablenverarbeitung stellt die Arbeitsgrundlage für die anderen Makros dar, da sie die vom Web-Anwender spezifizierten und vom WWW-Server weitergereichten Aufrufparameter in iWebDB/MP für die Weiterverarbeitung verfügbar macht, Funktionsergebnisse speichert sowie unterschiedliche Manipulationsfunktionen anbietet. So können neben der Wertübertragung von HTML-Formular- und Umgebungsvariablen sowie von Funktionsergebnissen in interne Variablen diese ausgegeben sowie String- und andere Manipulationen an diesen vorgenommen werden.

Eng mit den Veränderungsoperationen verknüpft sind auch die Makros zur Steuerung des Programmflusses. Mit Hilfe dieser ist es möglich, basierend auf dem Auswerteergebnis einfacher logischer Ausdrücke, andere iWebDB/MP-Makros auszuführen bzw. HTML-Text auszugeben. Auf diese Weise ist u. a. eine kontextabhängige Gestaltung von HTML-Seiten möglich.

4 Implementierung

Derzeit wird ein erster Prototyp von iWebDB auf dem ORDBVS IDS/UDO[6] [Inf97a] implementiert. Ein Problem bei der Umsetzung der Konzepte von iWebDB ist die z. Zt. nicht vorhandene Verfügbarkeit aller objekt-relationalen Eigenschaften in einem einzigen ORDBVS, d. h., alle kommerziell verfügbaren Systeme unterscheiden sich in der vorhandenen Funktionalität, wobei kein System alle oder für eine sinnvolle Umsetzung benötigten Mindestmerkmale, wie UDTs, UDFs, Einbindung externer Daten sowie Objektreferenzen, aufweist. Daher sind für eine erste Implementierung immer wieder Notlösungen gefragt, wie etwa die umständliche Simulation von Objektreferenzen durch Primär-/Fremdschlüssel-Verbindungen.

Bei der Implementierung von iWebDB/HTML werden die Datentypen als *opaque types* und als *row types* bereitgestellt, d. h. entweder als komplett neuer Datentyp mit den benötigten Hilfsfunktionen dem DBS bekannt gemacht oder aus bestehenden Datentypen zusammengesetzt. Wie bei iWebDB/Extractor auch werden die Verarbeitungs- und Analysefunktionen in C entwickelt, wobei für die Zukunft ein Umstieg auf Java geplant ist. Dann soll, wie bei iWebDB/MP schon jetzt für den Makroprozessor,

6. Informix Dynamic Server with Universal Data Option, ehemals Informix Universal Server.

der Java-Parser-Generator *JavaCC* [JCC] als Grundlage für die Analysefunktionen dienen.

iWebDB/ED nutzt das *Virtual Table Interface* für die Bereitstellung der externen Daten. Hierbei müssen die sich bei einer Anfrage qualifizierenden Daten dem DB-Server von den bereitzustellenden UDFs zeilenweise zur Verfügung gestellt werden. Auch die Änderungsoperationen liegen unter vollständiger Kontrolle der selbstimplementierten Funktionalität.

Da Teile von iWebDB auch in anderen Projekten eingesetzt werden sollen, gehen wir von einer baldigen Fertigstellung eines ersten, komplett lauffähigen Prototyps aus.

5 Verwandte Arbeiten

Teile der bei der Konzeption von iWebDB verwendeten Ideen und Lösungen sind nicht neu, auf allen behandelten Gebieten, d. h. denen von den einzelnen iWebDB-Komponenten angegangenen Problemen, existieren ähnliche Arbeiten. Im folgenden wollen wir einige von ihnen, bedingt durch die Längenvorgabe jedoch nur sehr kurz, charakterisieren und iWebDB zu diesen abgrenzen. Die relevanten Arbeiten entstammen dem Bereich Digitaler Bibliotheken und der Verwaltung (Semi-)Strukturierter Daten, insbesondere von SGML-Dokumenten (*Standardized General Markup Language*), dem Gebiet der Anfragesprachen für das WWW sowie der Suchmaschinen, der Thematik Web-Site-Verwaltung allgemein und auch den zahlreichen Werkzeugen für die Web-Anbindung von Datenbanken. Das Besondere an iWebDB ist dabei der integrative Ansatz, der alle Aufgaben und Gebiete miteinander in einem einheitlichen, in ein ORDBVS integrierten System zu verbinden versucht.

Ein integrativer Ansatz wird ebenfalls von WebOQL [AM98] gewählt. Jedoch beschränkt sich dieser auf die Struktur- und Inhaltsanalyse, die Dokumentenverwaltung im Sinne von iWebDB wird nicht behandelt. Dagegen wird bei ARANEUS [AMM98] und bei STRUDEL [FFLS97] der Schwerpunkt auf die Verwaltung von Dokumenten und Daten sowie das anschließende Generieren von HTML-Seiten gelegt, wobei über Graphen Abhängigkeiten zwischen Dokumenten bzw. Dokumentteilen gewartet werden. Im Vergleich zu beiden Ansätzen ist iWebDB bei der Dokumentenverwaltung weitaus flexibler, hat aber bzgl. der HTML-Generierung (siehe *iWebDB-Administration*) derzeit nur wenig Funktionalität. Dafür werden von iWebDB jedoch auch andere Aspekte, wie z. B. die Suche, behandelt.

Für die Anfrage und Suche im Web sowie in HTML-Dokumenten existiert eine Vielzahl von auf deskriptiven (z. B. WebSQL [AMM97]) oder deklarativen Anfragesprachen (z. B. FLORID [HL98]) beruhenden Ansätzen, die in der Regel nur die gezielte Informationssuche (anhand der Dokumentstruktur) ermöglichen. Anders als bei iWebDB werden Aspekte der Dokumentenverwaltung und die Suche auf diesem Datenbestand nicht betrachtet. Vergleicht man die Web-Suchmaschinen, die auch zur Indizierung einer Web-Site eingesetzt werden können (Harvest [Har98], Excite, Infoseek, etc.), so hat zwar iWebDB (derzeit) keine Volltextsuche, dafür kann zum einen nach Strukturelementen gesucht werden. Zum anderen steht aber anders als bei den

Suchmaschinen prinzipbedingt eine aktuelle Suchdatenbasis zur Verfügung (siehe auch Abschnitt 2.1, *Aktuelle Situation und Probleme*).

Inzwischen bieten die meisten DBVS-Hersteller ein eigenes Werkzeug zur Web-Anbindung von DB an [IBM98, Inf97b], zudem existieren zahlreiche kommerzielle und auch kostenlos verfügbare Produkte, die alle ihre spezifischen Vor- und Nachteile haben [Loe97, Loe98b]. Daher kann iWebDB/MP auch nur als ein weiteres Werkzeug auf diesem Gebiet gelten. Jedoch vereint es durch die Bereitstellung von sowohl im CGI-Programm und Web-Server als auch im DB-Server einsetzbare Funktionsbibliothek, durch die Realisierung in Java sowie die Ermöglichung des Mehrdatenbankzugriffs interessante und kaum anzutreffende Eigenschaften.

iWebDB bietet für viele der Einzelprobleme weniger Funktionalität als die angesprochenen Speziallösungen, versucht dagegen aber alle bei der Verwaltung von Web-Inhalten relevanten Probleme anzugehen und durch einen integrativen Ansatz eine einheitliche Lösungsbasis zu schaffen. Dies hebt iWebDB deutlich von anderen nur auf einen oder wenige Aspekte beschränkte Ansätzen ab. Zudem ist für die Nutzung kein neues System notwendig, da iWebDB ein bestehendes (objekt-relationales) DBVS um die Verwaltungsfunktionen und die dazu benötigten Komponenten erweitert. Dies bedeutet, daß iWebDB kein neues System an sich darstellt, sondern das vorhandene um WCM-Funktionalität ergänzt. Auf diese Weise können die Anwender in ihrem gewohnten Umfeld weiterarbeiten und der Web-Administrator gewinnt neue Möglichkeiten (siehe Abschnitt 3.3, *iWebDB-Administration*) bei einem deutlichen einfacheren und im Ergebnis konsistenteren Arbeiten hinzu.

6 Zusammenfassung und Ausblick

In diesem Beitrag haben wir iWebDB, ein integriertes System für die intelligente und einfache Verwaltung von Web-Inhalten, das *Web Content Management*, vorgestellt. Während heute noch der Großteil der Web-Sites im und über ein normales Dateisystem verwaltet wird und sich dadurch Probleme bzgl. der Konsistenz der Querverweise, der Rechtevergabe an die unterschiedlichen Nutzer(gruppen), der Aktualität der Dokumente und Suchindizes sowie bzgl. des Erstellungs- und Wartungsaufwands ergeben, bietet iWebDB einen DB-zentrierten Ansatz, der diese Probleme löst. Er sieht die Verwendung eines um entsprechende Verwaltungsfunktionalität erweiterten objekt-relationalen DBVS zur zentralen Speicherung der Web-Dokumente vor. Die Funktionalität wird von iWebDB durch seine Komponenten iWebDB/HTML, iWebDB/ED, iWebDB/Extractor und iWebDB/MP in Form von Erweiterungsmodulen (DataBlades, Extender oder Cartridges) bereitgestellt. Sie ermöglichen die Handhabung von Web-Dokumenten wie DBS-eigene Datentypen (*built-in types*), den Zugriff auf externe Datenquellen und den nahtlosen Datenaustausch mit dem Dateisystem (des Web-Servers), die SQL-basierte Suche mit Hilfe von Analysefunktionen auf dem aktuellen Datenbestand sowie die Web-Anbindung der Dokumenten- und anderer DB.

Neben der Ergänzung der Module um noch fehlende Funktionalität für Nicht-HTML-Dokumente sind in der Zukunft noch weitere Konzepte zu überdenken. So ist z. B. die Möglichkeit zur automatischen, periodischen Materialisierung von auf Geschäfts-DB beruhenden HTML-Seiten im Dateisystem eine sinnvolle Erweiterung. Ein Anwen-

dungsbeispiel ist die täglich erfolgende Generierung einer Teilnehmerliste für die BTW'99, d. h., einmal täglich wird eine solche HTML-Seite basierend auf einer HTML-Schablone mit iWebDB/MP-Makros und den Daten der Teilnehmer-DB erzeugt. Auf diese Weise werden zum einen durch den Verzicht einer CGI-basierten dynamischen HTML-Generierung nach einer DB-Anfrage sowohl Web- als auch DB-Server entlastet, zum anderen können auch viele der bisher manuell erledigten Arbeitsschritte automatisiert und gegenüber der Fehleranfälligkeit sicherer gemacht werden. Für diese Erweiterung können z. B. Dokumentvorlagen auf Basis erweiterter iWebDB/MP-Makros eingesetzt werden. Weitere Arbeiten betreffen die Unterstützung von XML anstelle von HTML sowie die daraus resultierenden notwendigen Erweiterungen.

Durch die Zunahme der Datenmengen im WWW und die Notwendigkeit einer DB-basierten Verwaltung der bereitgestellten Inhalte wird das Web Content Management in Zukunft zu einer bedeutungsvollen Aufgabe für DBS. Bis dahin bedarf es noch zahlreicher Forschungs- und Entwicklungsarbeiten, um eine möglichst gute Aufgabenunterstützung von Seiten der DBS zu erreichen.

7 Literatur

[AM98] Arocena, G., Mendelzon, A.: *WebOQL: Restructuring Documents, Databases and Webs*, in: Proc. of ICDE'98, Orlando, Florida, Februar 1998.

[AMM97] Arocena, G., Mendelzon, A., Mihaila, G.: *Applications of a Web Query Language*, in: Proc. of 6th International WWW Conference, Santa Clara, California, April 1997.

[AMM98] Atzeni, P., Mecca G., Merialdo, P.: *Design and maintenance of data-intensive Web sites*, in: VI Intl. Conference on Extending Database Technology (EDBT'98), Valencia, Spanien, März 1998, S. 436-450.

[BPS98] Bray, T., Paoli, J., Sperberg-McQueen, C. M.: *Extensible Markup Language (XML) 1.0, W3C Recommendation 10-February-1998*, W3C, http://www.w3.org/TR/1998/REC-xml-19980210, Februar 1998.

[Cal95] Cailliau, R.: *A Little History of the World Wide Web*, WWW-Konsortium, http://www.w3.org/History.html, 1995.

[CGI95] *The CGI Specification, Version 1.1*, University of Illinois, Urbana-Champaign, http://hoohoo.ncsa.uiuc.edu/cgi/interface.html, 1995.

[CH97] Conrad, S., Hasselbring W.: *BTW'97 - Datenbanken in Büro, Technik und Wissenschaft*, Datenbank-Rundbrief N^{o}. 20, November 1997.

[DM97] Denning, P., Metcalfe, R.: *Beyond Calculation - The next fifty years of computing*, Copernicus Springer-Verlag, 1997.

[FFLS97] Fernandez, M., Florescu, D., Levy, A., Suciu, D.: *A Query Language and Processor for Web-Site Management*, in: Proc. of the Workshop on Semi-structured Data, Tuscon, Arizon, Mai 1997.

[GJS96] Gosling, J., Joy, B., Steele, G.: *The Java Language Specification*, Addison-Wesley, http://java.sun.com:80/doc/language_specification.html, 1996.

[Har98] *Harvest Web Indexing*, http://www.tardis.ed.ac.uk/harvest/, 1998.

[HL98] Himmeröder, R., Ludäscher, B.: *Querying the Web with FLORID*, in: Tagungsband GI-Workshop Grundlagen von Datenbanken 1998, Konstanz, Juni 1998, S. 47-51.

[IBM98] *IBM Net.Data*, IBM Corporation, http://www.software.ibm.com/data/net.data/, 1998.

[Inf97a] *Getting Started with INFORMIX-Universal Server, Version 9.1*, Informix Software Inc., http://www.informix.com/answers/, März 1997.

[Inf97b] *Informix Web DataBlade Module, User's Guide, Version 3.3*, Informix Software Inc., http://www.informix.com/answers/, Mai 1997.

[JCC] *Java Compiler Compiler - The Java Parser Generator*, Sun Microsystems Inc., http://www.suntest.com/JavaCC/,1998.

[Kim96] Kim, W.: *Object-Relational - The unification of object and relational database technology*, UniSQL White Paper, 1996.

[Loe97] Loeser, H.: *Datenbankanbindung an das WWW - Techniken, Tools und Trends*, in: Tagungsband der GI-Fachtagung 'Datenbanksysteme in Büro, Technik und Wissenschaft' (BTW'97), K. R. Dittrich, A. Geppert (Hrsg.), Informatik aktuell, Ulm, Springer, März 1997, S. 83-99.

[Loe98a] Loeser, H.: *Die Nutzung der Erweiterungsmöglichkeiten von ORDBVS für Client/Server-basierte Anwendungssysteme*, in: Tagungsband 10. GI-Workshop Grundlagen von Datenbanken, Konstanz, Juni 1998, Konstanzer Schriften in Mathematik und Informatik, Nr. 63, Mai 1998, S. 77-81.

[Loe98b] Loeser, H.: *Techniken für Web-basierte Datenbankanwendungen - Anforderungen, Ansätze, Architekturen*, in: .Informatik - Forschung und Entwicklung, Band 13, Heft 4, Springer, 1998, pp. 196-216.

[SQLJ] *Database Language SQL - Part 10: SQL/OLB*, dpANS X3.135.10, http://www.sqlj.org, Juni 1998.

[Sto96] Stonebraker, M.: *Object-Relational DBMSs - The Next Great Wave*, Morgan Kaufman, 1996.

[WH98] White, S., Hapner, M.: *JDBC 2.0 API, Version 1.0*, Sun Microsystems Inc., http://www.javasoft.com/products/jdbc/jdbc20.ps, Mai 1998.