Fachbereich 3 Mathematik und Informatik 10. April 2001
Arbeitsgruppe Datenbanksysteme
R. Kollmann, A. Lattner, S. Prelle,
M. Richters, H. Shi, A. Werner, M. Gogolla

Softwarepraktikum 2001
Aufgabe „XDoc“

1 Einleitung

In dieser Aufgabe soll ein Programm entworfen und implementiert werden, mit dem Informationen über Dokumente (Bücher, Artikel, usw.) angezeigt, bearbeitet und ausgewertet werden können. Die zu verarbeitenden Informationen können beispielsweise zur Erstellung von Literaturverzeichnissen in Diplomarbeiten, Seminararbeiten, Projektberichten und ähnlichem verwendet werden. Darüberhinaus sollen Querverbindungen zwischen verschiedenen Dokumenten deutlich gemacht werden, wie z.B. “Wer zitiert ein bestimmtes Buch?”.

Abschnitt 2 gibt genauere Informationen über den Aufbau von Dokumentdaten. Die eigentliche Aufgabenstellung wird in Abschnitt 3 konkretisiert. Allgemeine Hinweise gibt es schließlich in Abschnitt 4.

2 Dokumentdaten

Für die Dokumentdaten stützen wir uns auf Informationen, die Michael Ley unter [2] über das WWW verfügbar macht. Auf der angegebenen Web-Seite können ca. 200.000 Veröffentlichungen aus der Informatik über Angabe eines Autors oder des Titels gesucht werden. Zusätzlich kann man Zeitschriften, Konferenzbände und ähnliches direkt durchstöbern. Allerdings lassen sich die Daten dort nicht unmittelbar ändern.

Wir benutzen dieselbe Struktur für Dokumentinformationen, wie sie auch für [2] verwendet wird. Die Dokumentdaten liegen dabei im XML-Format vor. XML ist ein Standard für die Beschreibung strukturierter Dokumente [5]. Ein Eintrag für ein Buch kann damit z.B. so aussehen:

```
<book key="books/aw/GoslingJS96">
  <author>James Gosling</author>
  <author>William N. Joy</author>
  <author>Guy L. Steele Jr.</author>
  <title>The Java&#153; Language Specification.</title>
  <publisher href="db/publishers/aw.html">Addison-Wesley</publisher>
  <year>1996</year>
  <isbn>0-201-63451-1</isbn>
</book>
```

Jedes Dokument wird durch einen Schlüssel (durch das Attribut `key`) eindeutig identifiziert. Es folgen weitere Elemente, welche die Autoren, den Titel, den Verlag, das Erscheinungsjahr und die ISBN-Nummer enthalten. Querverweise auf andere Dokumente werden durch `cite`-Elemente angegeben. Im nächsten Beispiel hervorgehoben ist ein Verweis auf das obige Buch. In dem Artikel wird somit auf das Buch von Gosling et al. verwiesen. Einträge mit "..." bedeuten, daß das zitierte Werk nicht in der Dokumentdatenbank enthalten ist.

```
<article key="journals/debu/HiggsC98">
  <author>Mike Higgs</author>
  <author>Bruce Cottman</author>
  <title>Solving the Data Inter-operability Problem Using a Universal
    Data Access Broker.</title>
  <pages>34-42</pages>
  <year>1998</year>
  <volume>21</volume>
  <journal>Data Engineering Bulletin</journal>
  <number>3</number>
  <url>db/journals/debu/debu21.html#HiggsC98</url>
  <ee>db/journals/debu/HiggsC98.html</ee>
  <cdrom>DEBU/98SEP-CD.PDF</cdrom>
  <cite label="CBB97">...</cite>
  <cite label="GJS96">books/aw/GoslingJS96</cite>
  <cite label="Gei94">...</cite>
  <cite label="HCF97">...</cite>
  <cite label="Kle95">...</cite>
  <cite label="Lea97">...</cite>
  <cite label="OLE97">...</cite>
  <cite label="OMG92">...</cite>
  <cite label="OTM97">...</cite>
  <cite label="SQL92">...</cite>
</article>
```

Die genaue Struktur der Einträge wird durch eine Dokumenttypdefinition (DTD) festgelegt, die unter [3] erhältlich ist.

3 Aufgabenstellung

Die Aufgabenstellung umfaßt folgende Teile. Die einzelnen Aufgabenteile werden in den weiteren Abschnitten genauer beschrieben.

1. **Ein-/Ausgabe**: Lesen und Schreiben von XML-Dateien, die Dokumentinformationen enthalten (siehe Abschnitt 3.1).
2. **Anzeige**: Fensterbasierte Anzeige der Literaturdaten (GUI), Navigation zwischen einzelnen Datensätzen (Abschnitt 3.2).
3. **Bearbeitung**: Eingabe, Änderung und Löschen von Dokumentinformationen (Abschnitt 3.3).
4. **Teilaufgaben zur Auswahl**: (mindestens zwei davon müssen bearbeitet werden):
   (a) **Anfragen**: Suche bestimmter Einträge nach verschiedenen Kriterien (Abschnitt 3.4).
   (b) **Netz**: Kommunikation mit anderen XDoc-Programmen übers Netz (Abschnitt 3.5).
   (c) **daVinci**: Darstellung der Querverweise zwischen Einträgen ("wer zitiert wen?") mit dem Visualisierungsprogramm *daVinci* (Abschnitt 3.6).
   (d) **WWW-Server**: Das Programm arbeitet als (sehr einfacher) WWW-Server, um Literaturdaten über Web-Browser zur Verfügung zu stellen (Abschnitt 3.7).

3.1 Ein-/Ausgabe mit XML

Ihr Programm soll Informationen über Dokumente aus einer XML-Datei einlesen und auch schreiben können. Die Dokumente sollen der Definition in [3] entsprechen. Für diese Aufgabe soll auf den XML-Parser Xerces [4] zurückgegriffen werden. Einige Beispieldateien werden auf der WWW-Seite des Softwarepraktikums zur Verfügung gestellt.

3.2 Anzeige

Das Programm soll eine fensterbasierte Oberfläche mit folgenden Funktionen anbieten.

- Anzeige eines einzelnen Eintrags (Buch, Artikel, etc.) mit allen Daten (Autoren, Titel, etc.)

- Anzeige aller Dokumente eines Autors
- Navigation durch eine Menge von Dokumenten

3.3 Bearbeitung

Für ein ausgewähltes Dokument sollen alle Daten änderbar sein. Eingaben sollen auf mögliche Fehler überprüft werden. Zum Beispiel ist zu prüfen, ob der Schlüssel eines Dokuments eindeutig ist und, ob Querverweise auf tatsächlich vorhandene Dokumente zeigen.

3.4 Anfragen

Eine flexible und gezielte Suche nach bestimmten Dokumenten soll durch eine Anfragesprache ermöglicht werden. Beispielsweise soll die Anfrage

```
year > 1998 and (title = "Java" or author = "Gosling")
```

alle Dokumente liefern, die im Titel das Wort Java enthalten oder vom Autor Gosling sind und auf jeden Fall nach 1998 veröffentlicht wurden. Der allgemeine Aufbau einer Anfrage ist durch folgende Grammatik gegeben.

```
expr ::= term | "(" expr ")" | expr "and" expr | expr "or" expr
term ::= ELEMENT_NAME ( "=" | "<>" | "<" | ">" )  STRING
```

Ein Anfrageausdruck besteht aus einem oder mehreren Termen, die miteinander durch "and" oder "or" verknüpft werden können. Ausdrücke können auch geklammert werden, um die Auswertungsreihenfolge zu verändern. Ein Term besteht aus einem Elementnamen wie "title", "author", "editor", etc. gefolgt von einem Vergleichsoperator und einer Zeichenkette. Zeichenketten können (bzw. müssen, wenn sie Leerzeichen enthalten) in Anführungszeichen eingeschlossen sein. Auch Zahlenangaben wie 1998 werden als Zeichenketten behandelt. Ein Ausdruck `title = "foo"` bedeutet "der Titel enthält die Zeichenkette foo", der Ausdruck `title <> "foo"` bedeutet "foo kommt im Title nicht vor".

3.5 Netz

Da XML leicht zu lesen und zu schreiben ist, eignet es sich als Format hervorragend für den Datenaustausch zwischen verschiedenen Programmen. In dieser Teilaufgabe soll das Programm erweitert werden, um die Informationen über Dokumente zwischen verschiedenen Rechnern auszutauschen. Die folgenden Funktionen sind erforderlich:

Server: Er ist bereit Anfragen eines Clients (in Form eines Strings) zu bearbeiten und alle Veröffentlichungen, in denen der String vorkommt, an den Client zurückzusenden.

Client: Er sendet Anfragen an einen Server und trägt die Rückgabe in den lokalen Dokumentbestand ein.

3.6 daVinci

Eine graphische Darstellung der Querverweise zwischen verschiedenen Dokumenten soll mit dem Visualisierungsprogramm *daVinci* realisiert werden. Abbildung 1 zeigt beispielsweise die Querverweise zwischen fünf Dokumenten, wobei *Book 1 Book 3* zitiert, *Book 3 Article 2*, *Article 2 Book 2*, usw.

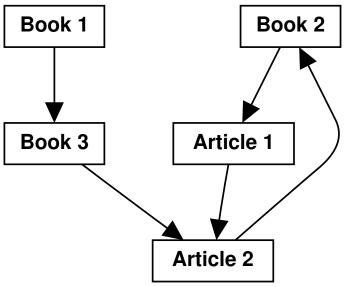

Abbildung 1: Darstellung der Querverweise zwischen Dokumenten

Die Aufgabe besteht hierbei darin, die Daten in geeigneter Form an das Programm *daVinci* zu übergeben. Informationen zu *daVinci* gibt es unter `http://www.informatik.uni-bremen.de/daVinci/`.

3.7 WWW-Server

Das WWW bietet eine einfache Möglichkeit Dokumentinformationen über das Netz zur Verfügung zu stellen. Ihr Programm soll als stark vereinfachter HTTP-Server (siehe [1], Stichwort "GET"-Methode) laufen und folgende zwei Arten von Anfragen von einem Browser bearbeiten können.

- Über den URL `http://localhost:8000/`[1] soll eine einfache Liste aller Dokumente zurückgegeben werden. Für jedes Dokument mit Schlüssel `key` soll ein Hyperlink auf den URL `http://localhost:8000/key` enthalten sein.
- Über den URL `http://localhost:8000/key` sollen Daten des Dokuments mit dem Schlüssel `key` zurückgegeben werden. Beispielsweise soll der URL `http://localhost:8000/books/aw/GoslingJS96` die Informationen zu dem Buch aus Abschnitt 2 liefern.

4 Allgemeine Hinweise

Achten Sie in Ihrem Entwurf auf eine klare Trennung zwischen den verschiedenen Aufgabenbereichen. Das Programm sollte z.B. nicht komplett neuentworfen werden müssen, wenn das Format zum Speichern der Dokumentdaten von XML auf eine SQL-Datenbank oder ähnliches umgestellt oder die graphische Oberfläche durch eine Textschnittstelle ersetzt wird.

Literatur

[1] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1, June 1999. RFC 2616, siehe `ftp://ftp.informatik.uni-bremen.de/doc/rfc/rfc2616.txt`.

[2] Michael Ley. DBLP bibliography collection. `http://dblp.uni-trier.de/`, 2001.

[3] Michael Ley. DBLP document type definition. Kopie erhältlich als `http://www.db.informatik.uni-bremen.de/swp2001/dblp.dtd`, 2001.

[4] Apache XML Project. Xerces XML parser. `http://xml.apache.org/`, 2001.

[5] W3C. Extensible Markup Language (XML) 1.0, W3C Recommendation 10-Feb-98. Technical report, W3C, 1998. `http://www.w3.org/TR/1998/REC-xml-19980210`.

[1] `localhost` muß gegebenenfalls durch den Hostnamen des Rechners, auf dem das Programm läuft, ersetzt werden.