Diplomarbeit

Effiziente Speicherung von XML-Datenströmen

Richard Atterer

27. Januar 2003

Motivation für die Diplomarbeit

Viele Web Services verwalten große Mengen an wertvollen Daten – XL soll für solche Web Services geeignet sein:

- Datenmenge ist größer als der Hauptspeicher
- Daten müssen dauerhaft erhalten bleiben
  (Beenden des XL-Prozesses, Rechnerabsturz. . .)

Beispiel: Kochrezept-Datenbank

```
<?xml version="1.0"?>
<!DOCTYPE recipe SYSTEM "recipe.dtd">
<recipe>
    <title>Kürbisbrot</title>
    <ingredients>
        <ingred name="Mehl" quantity="500–1000g" />
        <ingred name="Kürbis" quantity="1 kleiner" />
        ...
    </ingredients>
    <time>3 Stunden</time>
    <instructions>
        Beschreibung...
    </instructions>
</recipe>
```

Anforderungen

- Unterstützung in XL für persistente Variablen
  (Globale Variablen und Kontextvariablen sind persistent.)

- Abspeichern von XML-Dokumenten auf Festplatte

- Effiziente Modifizierung möglich
  (Einfügen/Löschen im Dokument)

- Modulare Implementierung:
  Möglichkeit für mehrere "Back-ends" zum Speichern der Daten
  (z.B. RDBMS, Berkeley DB, Speziallösungen)

XML-Dokumente als Tokenstrom

Von XL (bzw. XQRL) vorgegeben:
Dokument wird repräsentiert durch eine Folge von Tokens.

Beispiel: <ingred name="Mehl" quantity="500–1000g" />

Tokens:

- Begin element, Name "ingred"
  - Begin attribute, Name "name"
    - Text: "Mehl"
  - End attribute
  - Begin attribute, Name "quantity"
    - Text: "500–1000g"
  - End attribute
- End element

Speicherung des Tokenstroms

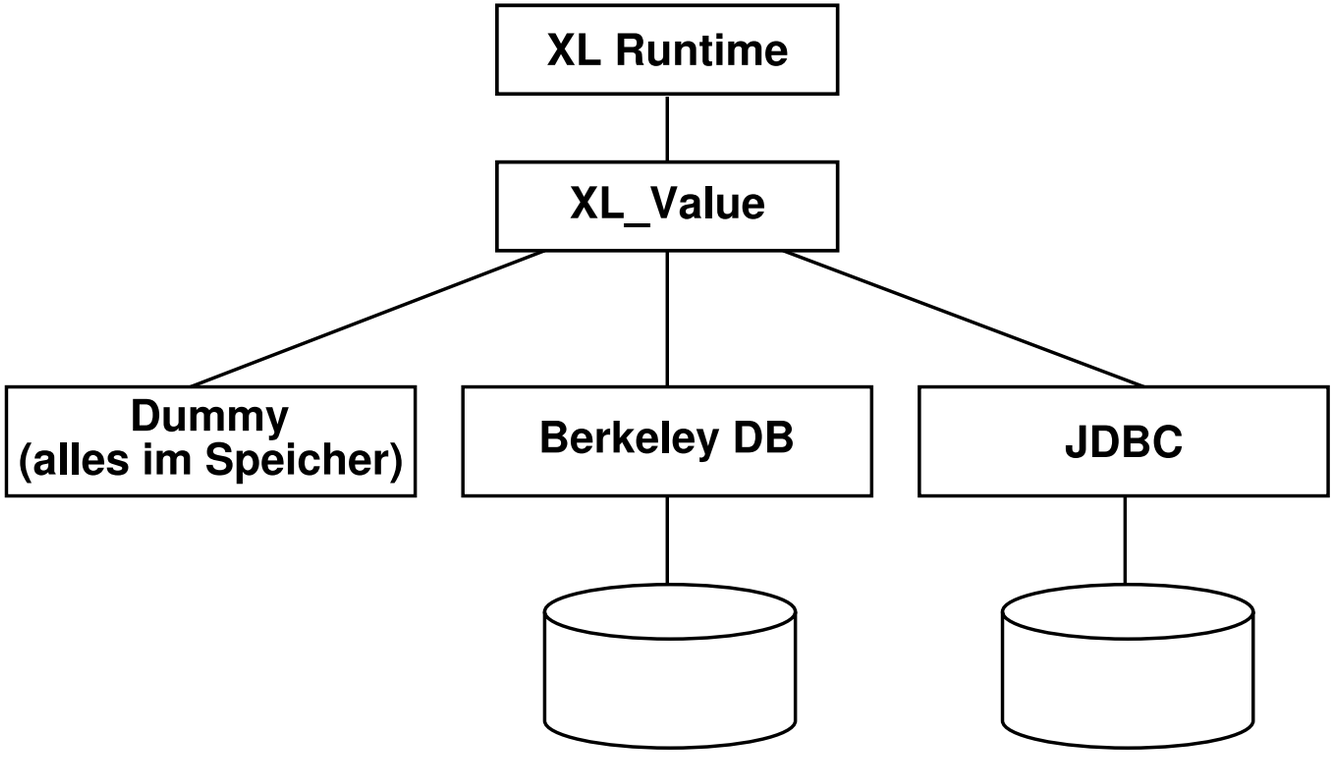
Lösungen zum Speichern der Tokens auf Festplatte:

- **BLOBs** in einem RDBMS
- **Records** (wie oben, ohne SQL-Query-Verarbeitung; Berkeley DB)
- **Tupel** in einem RDBMS (problematisch)
- **Objekte** in einem OODBMS
- Speziallösungen (z.B. "Natix")

Granularität der Speicherung:

- **Einzelne Tokens** in doppelt verketteter Liste: Einfach, Updates billig, Verwaltungsinfos benötigen viel Speicherplatz
- **Chunks von Tokens**: Weniger Verwaltungs-Overhead, wahlfreier Zugriff und Updates aufwändiger, i.A. Speziallösung nötig

Implementierung – Architektur



- XL_Value erlaubt zur Laufzeit (beim Start von XL) Auswahl zwischen Speichermodulen
- Jedes Modul verwaltet eigenen Cache von Token-Objekten

Implementierung – Durchführung

Schritte bei der Implementierung:

- Umstellung der alten XL_Value-Klasse auf neue Architektur
- Implementierung des Dummy-Moduls (keine Persistenz)
- Implementierung des Berkeley DB-Moduls
  (XML-Repräsentation, Einfügen/Löschen, Iteratoren, . . .)
- LRU-Tokencache für Berkeley DB-Modul
- Unterstützung für persistente Variablen in XL
- Anpassen an neue XQRL-Version
- Tests: Stabilität, Testfälle, Performance

Beispiel – "Stack-Server" mit persistenter Variable

Server nimmt Kommandos "push" und "pop" entgegen:

<command name="push"><data/></command>
<command name="pop"></command>

```
service http://localhost:3328/

    let $stack := <stack></stack>;        !! Persistente globale Variable

    operation server
        body

            if ($input/command/@name = "push") then
                insert <entry> $input/command/(text()|*) </entry>
                    into $stack/stack from $stack;
                let $output := $stack;
            endif;

            if ($input/command/@name = "pop") then
                let $output := $stack/stack/entry[position()=last()];
                delete $stack/stack/entry[position()=last()] from $stack;
            endif;

        endbody
    endoperation

endservice
```