Skalierbare Verarbeitung von XML mit Infonyte-DB[1]

Thomas Tesch, Peter Fankhauser, Tim Weitzel

Infonyte GmbH
Julius-Reiber-Str. 15
D-64293 Darmstadt
Tel. +49 (0) 700 INFONYTE
info@infonyte.de

Zusammenfassung: Die zunehmende Durchdringung von IT-Architekturen mit XML führt zu immer größeren XML-Datenvolumen. Diese lassen sich mit den zur Verfügung stehenden XML-Werkzeugen nicht ausreichend skalierbar verarbeiten. Dieser Beitrag stellt den persistenten XML Prozessor Infonyte-DB vor, der auch große XML-Datenvolumen resourcenschonend und effizient verarbeiten kann, und illustriert an Hand von konkreten Anwendungsszenarien seinen Einsatz für XML-basierte Webservices, technische Dokumentation, sowie mobiles Informations-management.

Abstract: The emerging penetration of IT architectures with XML leads to increasing XML data volumes. Available tools often fail in realizing scalable XML processing for large XML data volumes. This paper introduces Infonyte-DB, a persistent XML Processor that economizes on system resources and allows processing large XML data volumes. Based on concrete application scenarios it illustrates, how Infonyte-DB can be deployed for XML based web services, technical documentation, and mobile information management.

1 Einleitung

Die Auszeichnungssprache XML hat sich in den letzten Jahren nicht nur in ihrer angestammten Domäne des Dokumentenmanagements sondern auch zum Nachrichtenaustausch im E-Business-Bereich [RaAS02], zur Konvertierung heterogener Datenbestände [BuLW01] sowie allgemein als plattform- und programmiersprachenunabhängige Basistechnologie zum Datenaustausch wie beispielsweise in Web-Service-Architekturen [BeMW02] durchgesetzt.

Dieser Beitrag illustriert die Rolle von XML bei der Kopplung von IT-Systemen und verdeutlicht die Grenzen von existierenden XML-Werkzeugen bei der Verarbeitung von großen Datenmengen. So stoßen etwa gängige Prozessoren zur Transformation von XML-Daten bei wachsendem Datenvolumen rasch an Leistungsgrenzen. Es stellt sich die Frage nach skalierbaren Werkzeugen zur Speicherung und Verarbeitung von XML-Daten.

[1] Eine frühere Version dieses Beitrags erschien in der Zeitschrift Wirtschaftsinformatik WI 5/2002

Vor diesem Hintegrund wird die native XML-Datenbank der Infonyte GmbH als persistenter XML-Prozessor vorgestellt, der insbesondere die effiziente Verarbeitung großer XML-Datenvolumen ermöglicht und dabei sehr geringe Systemanforderungen stellt. Zentrale technische Grundlagen sind das am Fraunhofer Institut für Integrierte Publikations- und Informationssysteme (IPSI) entwickelte persistente DOM (PDOM, eine persistente Implementierung der W3C-DOM-Schnittstelle) sowie ein webfähiger Anfrageprozessor, der alle gängigen XML-Anfrage- und -Transformationssprachen unterstützt. Die Infonyte GmbH mit Sitz in Darmstadt ist ein Spin-Off des Fraunhofer IPSI.

Die Infonyte-Technologie wird in einer Vielzahl von Produkten wie Web-Portalen, B2B-Nachrichtensystemen, Content Management Systemen und mobilen Informationssystemen erfolgreich eingesetzt. Neben vor allem amerikanischen Luftfahrtunternehmen verwendet eine wachsende Zahl von Kunden aus der IT-Industrie, der Fertigungsindustrie und dem Finanzbereich Infonyte-DB als Komponente in ihren Produkten.

2 Herausforderungen bei der XML-Verarbeitung

Die Herausforderung heutiger IT-Systemarchitekturen ist die kostengünstige und flexible Kopplung von autonomen und heterogenen Systemen. Dabei sind in der Hauptsache Medien- und Prozessbrüche zu überwinden. XML hat sich hier als neutrales Datenaustauschformat durchgesetzt. Mit XML lassen sich auch komplexe Datenstrukturen vergleichsweise einfach darstellen, mit ihren Metadaten kombinieren, und zwischen verschiedenen Plattformen und Programmiersprachen austauschen. Für den praktischen Einsatz gibt es eine breite Basis von freien und kommerziellen Werkzeugen zur Erstellung, Validierung und Transformation von XML. In zwei Bereichen kommen die Stärken von XML besonders zum Tragen:

Medienneutrales Publizieren: Mit der zugrundeliegenden Trennung von Inhalt und Darstellungsinformationen hat XML seine Kernanwendungsdomäne im Bereich von komplexen medienneutralen Publikationsprozessen. Sowohl für Content-Management-Systeme, die Informationen bedarfsgerecht für bestimmte Zielmedien zusammensetzen, als auch Dokumentenmanagementsysteme, die ihren Schwerpunkt in der Verwaltung, Archivierung und Abfrage großer Dokumentenbestände haben, ist XML die ideale Basistechnologie. Dabei müssen häufig verschiedene proprietäre Dokumentenformate vor der Weiterverarbeitung in eine XML-Repräsentation gebracht werden. Mit entsprechenden Transformationssprachen wie der Extensible Stylesheet Language Transformations (XSLT) lassen sich aus den XML-Dokumenten dann beliebige Präsentationsformate (HTML, PDF, eBook-Formate, Druckvorstufe) erzeugen.

Kopplung von Geschäftsprozessen: Bei der innerbetrieblichen Kopplung von Geschäftsprozessen unter dem Schlagwort Enterprise Application Integration (EAI) werden Lösungen zur Überwindung von Unterschieden zwischen Mainframe-Systemen, Legacy-Anwendungen, ERP-Software und Web-Anwendungen entwickelt. Die unternehmensübergreifende Kopplung unter dem Schlagwort Business to Business Integration (B2BI) hat den Umgang mit unterschiedlichen Partnern, Datenformaten,

Technologien, Prozessen und Anforderungen zum Ziel. Mit klassischen Integrationsansätzen müssen entweder die proprietären Datenformate der einzelnen Prozesse direkt ineinander konvertiert werden, oder – bei Nutzung eines traditionellen Datenbanksystems – ein unternehmensweites oder gar unternehmensübergreifendes Datenbankschema entworfen werden. Beide Alternativen führen zu langen Entwicklungszeiten und kaum überschaubaren IT-Architekturen. Durch seine Kombination von Daten mit ihren Metadaten ermöglicht XML einen inkrementellen, bedarfsorientierten Integrationsansatz als goldenen Mittelweg. Daten aus unterschiedlichen Quellen können zunächst in ein von den innerbetrieblichen Prozessen entkoppeltes XML-Warehouse aufgenommen werden. Dort werden sie mit Hilfe etablierter XML-Standards wie XSLT, XPath und XML-Schema validiert, bereinigt, gefiltert, abgeglichen und ähnlich wie beim medienneutralen Publizieren für die verschiedenen Zielsysteme transformiert. Dieser Vorgehensweise minimiert gegenseitige Abhängigkeiten und ermöglicht eine lose Kopplung der Systeme.

Die mit dem Erfolg von XML einhergehende Durchdringung von IT-Architekturen bringt jedoch existierende XML-Werkzeuge schnell an ihre Grenzen beim Umgang mit großen Datenvolumen. Das hat vor allem zwei Gründe:

Verbosität: Der große Vorteil von XML, nämlich die Kombination von Daten mit ihren Metadaten, führt zu großen Dokumenten. In vielen Dokumenten ist die Auszeichnungsinformation größer als die ausgezeichneten Daten. Der direkte Umgang mit textuellem XML belastet daher die Speichersysteme genauso wie die zur Verfügung stehende Netzbandbreite.

Mangelnde Skalierbarkeit: Die existierenden XML-Werkzeuge werden immer wieder wegen ihres Speicherverbrauchs und ihrer Verarbeitungsgeschwindigkeit kritisiert. Dies betrifft insbesondere Realisierungen der Programmierschnittstelle Document Object Model (DOM) des W3C, die XML-Dokumente komplett als Baum im Hauptspeicher darstellen. Je nach XML-Dokument und DOM-Implementierung kann ein textuelles XML-Dokument in einem Hauptspeicher-DOM auf die zwanzigfache Größe anschwellen. Ähnliche Probleme treten bei Implementierungen der Transformationssprache XSLT auf [XSLT99].

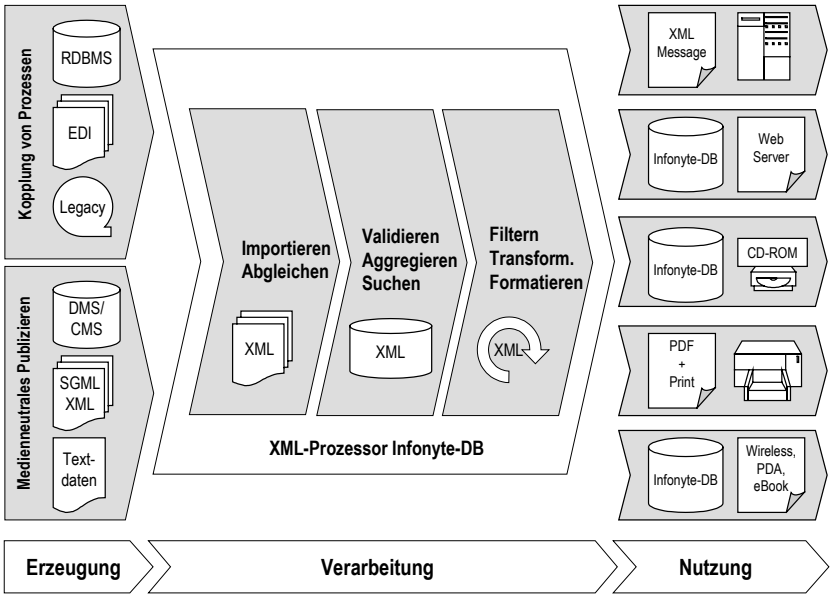

Bild 1: XML-Verarbeitung mit Infonyte-DB

Eine Möglichkeit zum Umgang mit XML-Massendaten sind native XML-Datenbanksysteme oder mit XML-Fähigkeiten angereicherte relationale Datenbanksysteme. Beide stellen die Speicherung von vielen XML-Dokumenten mit entsprechenden Suchmöglichkeiten in den Mittelpunkt. Bei der Nutzung von XML zur Integration zeigt sich jedoch, dass die skalierbare Verarbeitung weitaus stärker im Vordergrund steht als die dauerhafte Ablage. Publikationsumgebungen, die Ur-Domäne der XML-Verarbeitung, erfordern neben leistungsfähigen Autorenwerkzeugen in erster Linie die effiziente Filterung, Transformation und Formatierung zur Erzeugung verschiedener Zielformate. Auch im Bereich der Datenkonvertierung sind weniger die Ablage als vielmehr der effiziente Abgleich mit Quellen beim Import und die anschließende Aggregation und Transformation gefragt. Diese Anforderungen mit reinen XML-Speichersystemen zu realisieren scheitert oft an unzureichend integrierten DOM- und XSLT-Implementierungen. Bild 1 stellt die Nutzung von Infonyte-DB zur XML-Verarbeitung sowohl beim medienneutralen Publizieren als auch zur Kopplung von Geschäftsprozessen dar.

Der im Nachfolgenden beschriebene persistente XML-Prozessor Infonyte-DB hat sich auf den Einsatz von XML für medienneutrales Publizieren und die Integration von Geschäftsprozessen spezialisiert. Das Produkt stellt die XML-Verarbeitung in den Mittelpunkt und betrachtet die Speicherung von XML-Daten lediglich als Voraussetzung, um mit Standard-PC-Hardware auch mit Datenvolumen im Gigabyte-Bereich zu skalieren. Selbst bestehende XML-Anwendungen, die über die Standardschnittstellen DOM oder XSLT arbeiten, können durch den Austausch der

entsprechenden Module mit der Infonyte-Lösung zu skalierbaren XML-Anwendungen umgerüstet werden.

3 Infonyte-DB

Beim Entwurf von Infonyte-DB wurde das Ziel verfolgt, einen modularen und skalierbaren XML-Kern zu bauen, der sich anwendungsspezifisch erweitern lässt. Um ein hohes Maß an Integrierbarkeit zu erreichen, wurde darauf geachtet, die Kommunikation zwischen der Anwendung und dem Infonyte-Produkt soweit wie möglich über standardisierte Schnittstellen zu realisieren. Nutzer erreichen so selbst beim Einsatz einer neuen und innovativen Technologie ein hohes Maß an Investionssicherheit, da Investitionen in proprietäre Schnittstellen vermieden werden.

3.1 Überblick

Bild 2 zeigt die Architektur von Infonyte-DB. Die Architektur ist modular aufgebaut, wodurch die einzelnen Komponenten bis auf wenige Ausnahmen auch unabhängig voneinander eingesetzt werden können. Den Kern bildet ein persistentes DOM, auf dem in der XML-Welt gängige Anfrage- und Transformationsprozessoren realisiert sind. Diese nutzen für Optimierungszwecke zwar spezielle Eigenschaften des persistenten DOMs aus, können aber auch auf anderen DOM-Implementierungen aufsetzen. Für die effiziente Verarbeitung von großen XML-Dokumenten (ab ca. 500 MB) zu ermöglichen, können zusätzlich Indizes auf XML-Dokumenten (oder Kollektionen) definiert werden. Der dargestellte Kollektions-Manager bietet erweiterte Funktionen zum Umgang mit Dokumentmengen.

Das Produkt Infonyte-DB ist vollständig in Java implementiert und so auf praktisch allen marktrelevanten Plattformen ohne Probleme lauffähig. Die Entscheidung, auf Java zu setzen, hat sich als richtig erwiesen, da Java sich bei internetbasierten Serveranwendungen durchgesetzt hat und beim gezielten Einsatz und Verzicht auf ineffiziente Sprachkonstrukte in punkto Performance anderen Sprachen kaum noch nachsteht. Die verfügbaren Just-in-time-Übersetzer haben gegenüber statischen Übersetzern ohnehin den Vorteil, dass ihnen zusätzliche Laufzeitinformationen zur Verfügung stehen, die sie für Optimierungszwecke einsetzen. Die Infonyte-DB-Komponenten haben je nach Ausbaustufe eine Code-Größe zwischen 400 KB und 800 KB und sind bereits ab 16 MB RAM lauffähig.

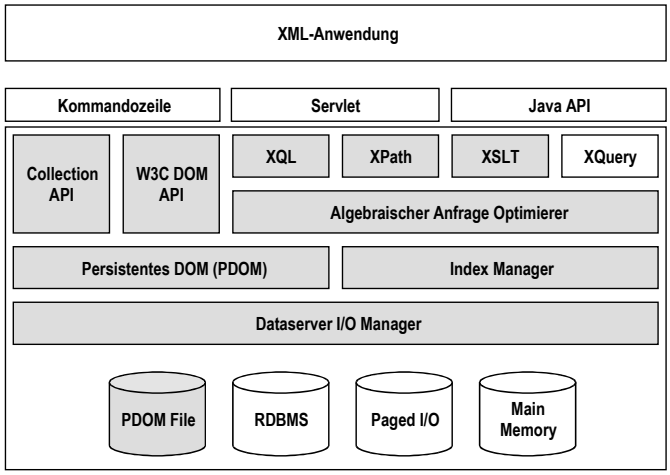

Bild 2: Infonyte-DB Komponentenarchitektur

Das System kann von einer Anwendung direkt über die Kommandozeile, einen Web-Server oder Java-Schnittstellen angesprochen werden. Es verarbeitet direkt wohlgeformtes XML, ohne dass eine DTD oder ein Schema angegeben werden muss. Alle Index- und Speicherungsstrukturen werden aus den Instanzinformationen gewonnen. Somit entfällt die aufwändige Angabe von Abbildungsvorschriften auf physische Datenmodelle, die bei der XML-Speicherung in relationalen Systemen und auch einigen XML-Datenbanken nötig sind und bei jeder Änderung der Dokumentstruktur einen großen Anpassungsaufwand nach sich ziehen.

Im Folgenden werden die einzelnen Komponenten genauer beschrieben.

3.2 Das persistente DOM

Die persistente DOM-Komponente (PDOM) bildet den Kern der Infonyte-Architektur. Das World Wide Web Konsortium (W3C) hat mit dem DOM die Standard-Programmier-Schnittstelle zum Umgang mit XML-Dokumenten geschaffen. Im DOM wird ein XML-Dokument als Baum mit unterschiedlichen Knotentypen dargestellt, welche die speziellen Eigenschaften von Elementen, Attributen etc. modellieren. So kann eine Anwendung über die Dokumentenstruktur navigieren und flexibel Knoten hinzufügen, verändern und löschen.

Es gibt heute eine Vielzahl von DOM-Implementierungen für alle gängigen Programmiersprachen. Sie bauen den Dokumentbaum in der Regel komplett im Hauptspeicher auf, was, je nach Implementierung und Hostsprache, zu erheblichen Speicherproblemen führt. Ein 20 MB XML-Dokument belegt in einem Hauptspeicher-DOM zwischen 200 MB und 400 MB.

Das Infonyte-PDOM geht hier einen anderen Weg. Es bietet eine vollständig W3C-DOM-konforme Schnittstelle, verwaltet XML-Dokumente jedoch in einem hochkompakten, verlustfreien Binärformat, dessen Speicherlayout auf die für XML spezifischen Baumstrukturen ausgelegt ist. Zur Addressierung von Dokumentknoten mit Hilfe einfacher Integer-Arithmetik, sowie zur effizienten Evaluierung von XPath-Ausdrücken, enthält das Format Indizes für Dokumentstruktur und -reihenfolge. Redundante Informationen, wie Elementnamen, werden faktorisiert. Das bewirkt, dass selbst mit den zusätzlichen Indizes, der Speicherbedarf lediglich zwischen 30% und 100% des ursprünglichen XML-Dokuments liegt. Eine Vorläuferversion des verwendeten Formats ist in [HMF99] beschrieben.

Die Kompaktheit des Binärformats sowie das für XML optimierte Speicherlayout führen zu einem hochperformanten IO-Verhalten. Der dazu eingesetzte LRU-Cache operiert auf physischen Speichersegmenten mit einer konstanten Anzahl von DOM-Knoten. Des weiteren stehen Synchronisationsmechanismen für Multithreading-Zugriffe und Funktionen zur Wartung und Defragmentierung der persistenten Dokumente zur Verfügung.

3.3 Data-Server

Das Data-Server-Modul implementiert den eigentlichen IO auf dem Speichermedium. Diese Komponente abstrahiert von den konkreten Eigenschaften eines Speichermediums und bietet wahlfreien Zugriff auf beliebig große Datensegmente. Die Data-Server-Schnittstelle ist vollständig offengelegt, so dass bei Bedarf eigene Implementierungen realisert werden können. Die mit Infonyte-DB mitgelieferte Data-Server-Implementierung verwaltet die Speichersegmente in speziell auf XML und effizienten IO zugeschnittenen Binärdateien. Wie in der Architektur angedeutet, lassen sich Data-Server-Implementierungen auch auf einer relationalen Datenbank aufsetzen oder zur Realisierung einer Hauptspeicherdatenbank verwenden [MBK00]. Genauso könnte ein Data-Server auf seine Daten über Netzdienste zugreifen, um die Skalierung im Mehrbenutzerbetrieb zu erhöhen. Die Offenheit der Backend-Schnittstelle ermöglicht die Erstellung gerätespezifischer Data-Server, so dass der Infonyte-DB leicht auf die Eigenschaften unterschiedlicher Speichersysteme vom Handheld bis zum hochskalierbaren NAT-Server angepaßt werden kann.

3.4 XML-Abfragen und XSL-Transformation mit PXSLT

Infonyte-DB unterstützt die gängigsten XML-Abfragesprachen und Transformationswerkzeuge. Damit können ein oder mehrere Dokumente durchsucht werden, die Suchergebnisse kombiniert werden, sowie logische Sichten realisiert werden. Die Extensible Query Language (XQL) ist ein Vorläufer aus dem Jahre 1999 [XQL99], wurde aber nie vom W3C als Standard verabschiedet. Heute dominieren XPath, das auch in XSLT zur Selektion von Dokumentteilen Verwendung findet, und der zukünftige XQuery-Standard die Szene [XPath99,XQuery02]. Sowohl XPath als auch XQL-Anfragen werden in der Infonyte-Architektur zunächst in einen initialen Ausführungsplan übersetzt, der in einer an XPath angelehnten Algebra dargestellt wird. Dieser Ausführungsplan wird durch die Anwendung von Transformationsregeln

optimiert, um die Ausführungszeit der Anfrage zu minimieren (logische Optimierung). Eine anschließende physische Optimierung unter Einsatz der beschriebenen Indexstrukturen führt dann zu einer optimierten Sequenz von DOM-Operationen, um das Ergebnis der Abfrage zu instanziieren.

Der Infonyte-XSLT-Prozessor („Infonyte-PXSLT“ für Persistent XSLT) bietet erstmalig die Möglichkeit, XSLT-Verarbeitung direkt auf einer persistenten Repräsentation von XML-Dokumenten durchzuführen. Gerade im Bereich der Verarbeitung großer XML-Dokumente mit XSLT gibt es massive Probleme mit den existierenden hauptspeicherbasierten Implementierungen. Dies führt in der Praxis häufig zur Stückelung der XML-Dokumente in mehrere noch handhabbare Einheiten und das Hintereinanderschalten mehrerer XSLT-Skripte. Der PXSLT-Prozessor kann aufgrund seiner Architektur ohne Schwierigkeiten XML-Dokumente im Gigabyte-Bereich mit konstantem Hauptspeicherbedarf verarbeiten. Durch die Verwendung des kompakten PDOM-Formates in Verbindung mit darauf zugeschnittenen Caching-Verfahren geschieht dies praktisch ohne Einbußen bei der Verarbeitungsgeschwindigkeit. Damit ermöglicht der PXSLT-Prozessor auch die Realisierung hochvolumiger XML-Anwendungen.

Der kommende XQuery-Standard ist in der Architektur bereits dargestellt. XQuery nutzt den XPath-Standard zur Auswahl von Dokumentfragmenten und erweitert ihn um SQL-artige Konstrukte für die Kombination und Restrukturierung von Dokumentfragmenten. Zur Darstellung von XML-Anfrageresultaten nutzt XQuery konsequenterweise XML selbst. Darüber hinaus definiert XQuery die notwendigen Funktionen und Operatoren für alle eingebauten Datentypen von XML-Schema. Ein Prototyp, der auf dem DOM-API aufsetzt, ist bereits vorhanden [FGO02].

3.5 Benutzerdefinierte Indizes

Für große XML-Dokumente und Dokumentkollektionen, an die inhaltsorientierte Anfragen gestellt werden, reichen die dargestellten Strukturindizes nicht aus. Um auch die Werte von Textknoten in einem Index zu erfassen, können benutzerdefinierte Indizes angelegt werden. Je nach Typ der zu indizierenden Werte werden Integer-, Double- oder String-Indizes erzeugt. Für textbasierte Suche ist die Erstellung eines Wortindex möglich. Indexeinträge können entweder auf die indizierten Knoten selbst oder wahlweise auch auf andere Knoten (z. B. die Dokumentwurzel) verweisen. So lassen sich beispielsweise auch Wortindizes für große Dokumentmengen aufbauen, die direkt Dokumente zurückgeben. Des Weiteren lassen sich die vorhandenen Indexstrukturen durch den Anwender erweitern, um spezielle Typen wie beispielsweise die Indizierung von Währungsformaten oder proprietären Zahlendarstellungen zu unterstützen.

3.6 Performanz

Experimente mit einer XML-Version der frei verfügbaren CD-Datenbank FreeDB [FreeDB02] sollen einen Anhaltspunkt für die Leistungsfähigkeit des Systems liefern. Bild 3 zeigt die CD-Datenbank im Administrations-GUI des Infonyte-Systems. Die

FreeDB enthält ca. 500.000 CD-Beschreibungen mit einer Datengröße von etwa 500 MB als XML-Rohtext.

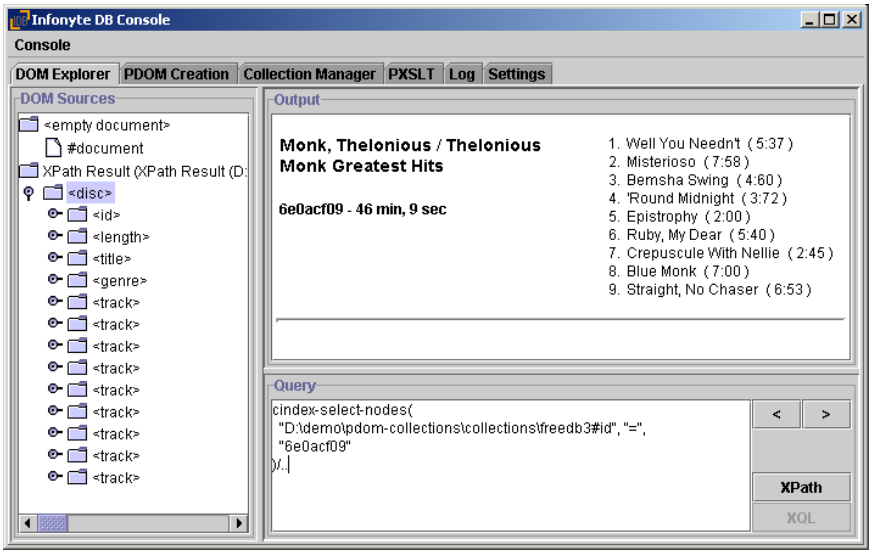

Bild 3: Screenshot der Infonyte-DB Administrationskonsole

Auf einem Standard-PC (1,8 Ghz, 512 MB RAM) dauert das Parsen und die Erzeugung eines PDOM (32 Millionen XML-Knoten, 400 MB) inklusive aller Strukturindizes etwa 4 Minuten (~2MB/Sekunde). Das Anlegen eines benutzerdefinierten Index über die einzelnen CD-Schlüssel führt zur Indizierung von 548.000 Knoten oder 1,7% der gesamten Datenbank und dauert etwa 88 Sekunden. Der Aufbau eines Volltextindex (28 Millionen Knoten, 89% der gesamten Datenbank) nimmt etwa 17 Minuten in Anspruch und resultiert in einer Größe von 90 MB. Je nach Indexdefinition werden zwischen 5 MB und 10 MB XML-Rohtext pro Sekunde indiziert. Bei der XSLT-Verarbeitung, beispielsweise zur Erzeugung von HTML, liegt der Durchsatz bei bis zu 10 MB pro Sekunde. Bei der Suche nach CDs mit bestimmten Titeln oder Tracks über den Volltextindex wird der erste Treffer bereits nach 5-10 Millisekunden geliefert. Jeder weitere Treffer wird in der gleichen Zeit ermittelt. Aufwendiger ist die Verarbeitung bei AND-Verknüpfungen, da hier die Schnittmenge mehrerer Teilergebnisse gebildet werden muss.

In einer eingeschränkten Version, bei der lediglich der Data-Server, das PDOM, sowie das Index- und das Collection-API (zusammen etwa 300 KB) verwendet werden, ist die FreeDB-Anwendung auch auf einem Pocket-PC der neueren Generation unter Realweltbedingungen lauffähig (iPAQ Pocket PC H3800 mit 64 MB Ram, 32 MB Rom, 206 Mhz ARM-Prozessor, 1GB IBM-Microdrive, Personal Java 1.2 Insignia Jeode). Unter Nutzung der Indizes liegen die Antwortzeiten bei boolescher Suche auch auf dieser eingeschränkten Plattform im Bereich von 1-2 Sekunden; die Suche nach nur einem Kriterium erfordert weniger als eine Sekunde.

3 Anwendungsszenarien

Im Folgenden wird die Nutzung von Infonyte-DB in drei Anwendungsszenarien dargestellt. Dabei wird der Einsatz des Systems zur Kopplung von Geschäftsprozessen sowie zum medienneutralen Publizieren im Bereich der technischen Dokumentation gezeigt. Ein weiteres Szenario verdeutlicht, dass Infonyte-DB sich aufgrund der Plattformunabhängigkeit und des sparsamen Umgangs mit Systemressourcen auch zum Aufbau von Informationssystemen für mobile Endgeräte eignet.

4.1 XML-Warehouse

Ein typisches Szenario für den Einsatz von XML zur Kopplung von Geschäftsprozessen ist der Betrieb eines XML-Warehouse, das Daten aus verschiedenen betrieblichen Informationssystemen in eine gemeinsame XML-Repräsentation bringt. Die von den Systemen produzierten XML-Daten werden dann über XSLT oder XQL/XPath-Anfragen zur Weiterverarbeitung wie beispielsweise der Veröffentlichung auf einem Web-Server aufbereitet.

Im vorliegenden Fall eines großen US-Dienstleisters für Finanzinformationen war es aus organisatorischen und technischen Gründen wichtig, das XML-Warehouse von den betriebswirtschaftlichen Anwendungssystemen so weit wie möglich zu entkoppeln, um einen unabhängigen Betrieb zu gewährleisten. Damit kamen Lösungen, die lediglich logische XML-Sichten bereitstellen, nicht in Frage. Auf Basis der Infonyte-Lösung wurde eine Anwendung entwickelt, die individualisierte Nachrichten verschickt und ein Web-Portal steuert, in dem Kunden ihre Handelsinformationen zeitnah abrufen können. Das Infonyte-System muß dabei täglich etwa 10 GB XML-Rohdaten aufnehmen, indizieren und in einem Fenster von zehn Tagen verfügbar halten. Es ließ sich dabei mit geringem Aufwand als skalierbares XML-Backend in den J2EE-konformen IBM WebSphere Application Server integrieren. Die Fähigkeit zur Verarbeitung der geforderten Datenmenge zusammen mit Zugriffen im Millisekundenbereich zur dynamischen Aufbereitung individualisierter Web-Seiten machen Infonyte zu einer idealen Lösung für diesen Problembereich. Mit der Automatiserung eines ehemals auf die Erzeugung und den Ausdruck von individuellen Reports ausgerichteten Geschäftsprozesses (Mainframe-Anwendung) und dem gleichzeitigen Einsatz preiswerter PC-Hardware konnten bedeutende Kosteneinsparungen erreicht werden.

4.2 Interaktive Elektronische Technische Dokumentation (IETD)

Im Bereich der technischen Dokumentation in der Luftfahrtindustrie wurde schon sehr früh SGML-Technologie eingesetzt. Aus Kostengründen werden viele Systeme heute auf XML-Technolgie und Standard-Internet-Browser umgestellt. Die Herausforderungen bestehen dabei in der Realisierung eines verteilten Autorensystems mit zentraler Datenhaltung, eines effizienten Produktionsprozesses zur Zusammenstellung und Formatierung elektronischer Handbücher für verschiedene Zielgruppen, der Bereitstellung leistungsfähiger Lese- und Navigationswerkzeuge sowie in einer möglichst kostengünstigen und sicheren Verteilung.

Aufgrund der leichten Integrierbarkeit von Infonyte und der Fähigkeit, effizient mit großen Einzeldokumenten umzugehen, hat die Sikorsky Aircraft Corporation ein XML-fähiges IETD-System auf Basis von Infonyte entwickelt. Dabei wurde sowohl der Produktionsprozess als auch die Bereitstellung der Dokumente über Web-Server mit der Infonyte-Lösung realisiert. Bei der Produktion steht die Fähigkeit, große XML-Datenmengen bedarfsgerecht zusammenzusetzen, im Vordergrund, wofür sich der Infonyte-XSLT-Prozessor bestens eignet. Bei der anschließenden Nutzung der technischen Dokumente in einer Leseumgebung kann durch die client-seitige Verwendung von Infonyte-DB über die zur Verfügung stehenden XML-Abfragesprachen zielgerichtet in bestimmten Dokumentteilen gesucht werden. Weiterhin wird die Wartung der Dokumente wesentlich erleichtert, da Service-Techniker direkt Änderungsvorschläge oder Reparaturreports über das updatefähige PDOM hinzufügen können.

Gerade im Luftfahrtbereich, wo bisher SGML dominiert hat, werden erhebliche Einsparungen bei der Entwicklung von XML-basierten IETD-Systemen erzielt. Zum einen kann Client-seitig vielfach Standardsoftware (XML/PDF Browser) eingesetzt werden, so dass aufwendige Eigenentwicklungen entfallen, und zum anderen ist der gesamte Entwicklungsprozess durch den Einsatz von XML-Werkzeugen wesentlich günstiger.

Ähnliche Einsatzszenarien gibt es im Bereich formularorientierter Verarbeitungsprozesse und der Aufbereitung von Fachinformationstexten und Nachschlagewerken im medizinischen und juristischen Bereich.

4.2 Mobiles Informationsmanagement

Der geringe Speicherverbrauch von Infonyte-DB, die Plattformunabhängigkeit durch Java sowie die Kompaktheit des PDOM-Formats machen Infonyte zu einer idealen Lösung für XML-basiertes mobiles Informationsmanagement der nächsten Generation. Die Firma Vaultus (http://www.vaultus.com) entwickelt unter anderem auf Basis der Infonyte-Technologie eine mobile Informationsplattform. Neben dem Datenmanagement und Nachrichtenaustausch in XML bietet das System Offline-Fähigkeiten, sichere Transaktionen, Netzwerkunabhängigkeit und Dienste zur Fernwartung und automatischen Aktualisierung mobiler Geräte. Der Infonyte-XML-Prozessor kommt hier aufgrund seines sparsamen Umgangs mit Systemressourcen ausschließlich auf den Java-fähigen Mobilgeräten zum Einsatz. Die XML-Daten werden ähnlich wie bei Nutzung einer relationalen Datenbank direkt von der Anwendung manipuliert, d. h. Updates auf der feingranularen Dokumentenstruktur wie das Infonyte-PDOM sie bietet sind eine Schlüsselfähigkeit zum Einsatz in dieser Anwendungsklasse. Für zukünftige Anwendungen ist auch der Einsatz des kompakten PDOM-Formats zum direkten Austausch von XML-Daten für Synchronisationszwecke denkbar. Das Parsen von Dokumenten könnte dann vollständig entfallen und sämtliche Daten wären sofort über die kodierten Indexstrukturen effizient zugreifbar. Bild 4 zeigt den Screenshot einer mobilen Vertriebsanwendung auf Basis der Vaultus-Plattform.

Bild 4: Web-Demo einer Mobilanwendung mit Infonyte-DB

4 Ausblick

Skalierbare XML-Prozessoren wie Infonyte-DB zeigen einen neuen innovativen Ansatz auf, die Probleme beim Umgang mit großen XML-Daten und damit die Skalierungsprobleme bestehender Werkzeuge zu überwinden. Eigene Experimente mit Open-Source-Implementierungen zeigen, dass viele freie Realisierungen (FOP-Prozessoren, XML-Editoren, WebDAV-Server) durch Austausch des Speicher-Backend mit der Infonyte-Lösung zur skalierbaren XML-Anwendung aufgerüstet werden können. Die Beschränkung auf einen schlanken XML-Kern sowie die performante Realisierung in einer plattformunabhängigen Sprache wie Java machen Infonyte zu einem nahezu universell einsetzbaren Werkzeug für XML-Anwendungen. Der große Bedarf, existierende Architekturen um XML-Fähigkeiten zu ergänzen, zusammen mit der leichten Integrierbarkeit machen das System auch zur kostengünstigen Erweiterung bestehender Anwendungen interessant.

Infonyte-DB kann zur kostenlosen Evaluierung unter der Adresse http://www.infonyte.de heruntergeladen werden.

Literaturverzeichnis

[BeMW02] Beimborn, Daniel; Mintert, Stefan; Weitzel, Tim: Web Services und ebXML. Erscheint in: WIRTSCHAFTSINFORMATIK (2002) 3.

[BuLW01] Buxmann, Peter; Ladner, Frank; Weitzel, Tim: Anwendung der Extensible Markup Language (XML): Konzeption und Implementierung einer WebEDI-Lösung, In: WIRTSCHAFTSINFORMATIK (2001) 3, S. 257-267.

[DOM00] Arnaud Le Hors et al.: Document Object Model (DOM) Level 2 Core Specification. W3C Recommendation. 13. November 2000, http://www.w3.org/TR/DOM-Level-2-Core/, Abruf am 2002-06-06.

[FGO02] Fankhauser, P.; Groh, T.; Overhage, S.: XQuery by the Book: The IPSI XQuery Demonstrator. EDBT 2002: 742-744. http://xml.darmstadt.gmd.de/xquerydemo/ , Abruf am 2002-06-06.

[FreeDB02] http://www.freedb.org, Abruf am 2002-06-06.

[HMF99] Huck, G.; Macherius,I.; Fankhauser, P.: PDOM: Lightweight Persistency Support for the Document Object Model. OOPSLA Workshop "Java and Databases: Persistence Options". November 1999. Denver, Colorado, USA.

[Luo01] Luoma, R.: Using XML to Enable Low-Cost Deployment of Content at Lockheed Martin Aeronautics. XML 2001 Conference and Exposition. December 2001. Orlando, Florida, USA.

[MBK00] Manegold, S.; Boncz, P.A.; Kersten, M.L: Optimizing database architecture for the new bottleneck: memory access. VLDB Journal (2000) 9(3), S. 231-246.

[RaAS02] Rawolle, J.; Ade, J.; Schumann, M: XML als Integrationstechnologie bei Informationsanbietern im Internet - die Fallstudie BertelsmannSpringer, In: WIRTSCHAFTSINFORMATIK (2002) 1, S. 19-28.

[Nwg99] Network Working Group: HTTP Extensions for Distributed Authoring – WEBDAV. RFC 2518. http://www.ietf.org/rfc/rfc2518.txt, Abruf am 2002-06-06.

[XML98] Tim Bray et al.: Extensible Markup Language (XML) 1.0. W3C Recommendation. 10. Februar 1998. http://www.w3.org/TR/1998/REC-xml-19980210.html, Abruf am 2002-06-06.

[XQL99] Robie, J.; Lapp, J.; Schach, D.: XML Query Language: A Proposal. W3C-QL '98 workshop proposal. http://www.w3.org/Style/XSL/Group/1998/09/XQL-proposal.html, Abruf am 2002-06-06.

[XQuery02] Draper, D. et al.: XQuery 1.0 Formal Semantics. W3C Working Draft. 26. März 2002. http://www.w3.org/TR/query-semantics/, Abruf am 2002-06-06.

[XPath99] Clark, James et al.: XML Path Language (XPath) Version 1.0. W3C Recommendation. 16 November 1999. http://www.w3c.org/TR/xpath/, Abruf am 2002-06-06.

[XSD01] Fallside, David C.: XML Schema Part 0: Primer. W3C Recommendation, 2. Mai 2001. http://www.w3.org/TR/xmlschema-0/, Abruf am 2002-06-06.

[XSLT99] Clark, James: XSL Transformations (XSLT) Version 1.0. W3C Recommendation. 16. November 1999. http://www.w3c.org/TR/xslt/, Abruf am 2002-06-06.