1 Einleitung

Die »eXtensible Markup Language« (XML) hat sich in den letzten Jahren als die zentrale Infrastruktur für den elektronischen Datenaustausch und für die Ablage (semi)-strukturierter Datenbestände etabliert.

Der Siegeszug von XML, der im Übrigen noch lange nicht abgeschlossen ist, sondern bei dem permanent weitere Gebiete erobert werden, basiert im Prinzip auf zwei Eigenschaften: Ein XML-Dokument ist zum einen einfach und klar strukturiert (und dadurch von Maschinen leicht verarbeitbar) und bis zu einem bestimmten Grad von Menschen les- und interpretierbar. Zum anderen ist ein XML-Dokument – im Unterschied zu einem ASCII-basierten Textdokument – an ein Schema gebunden und kann gegen dieses Schema validiert werden. Dieses Prinzip, welches als wesentliches Kennzeichen aus dem Bereich der Datenbanken stammt, bildet die Grundlage für jede Art des elektronischen Datenaustauschs, in dem sich die Partner an vereinbarte schematische Definitionen halten.

Die Existenz eines Schemas erlaubt somit den weitreichenden Einsatz von XML, angefangen von der elektronischen Bestellung über das Internet, der Formulierung eines Prozedurfernaufrufs in verteilten Systemen bis hin zur XML-basierten Rekonstruktion bereits in heterogenen Systemen existierender Datenbestände. Insbesondere der letzte Aspekt gewinnt aktuell unter dem Schlagwort der Informationsintegration (»information integration«) immer mehr an Bedeutung, wobei XML als generischer Ansatz zur Beschreibung individueller Datenbestände benutzt wird und dadurch eine XML-basierte Sicht über eine Vielzahl physisch autonom modellierter Datenbestände gelegt werden kann.

Neben Basiskonzepten wie XML selbst, XML Schema, XPath etc. hat sich in den letzten Jahren XQuery als eigenständige Anfragesprache mit dem Anspruch entwickelt, das zentrale Werkzeug zum Zugriff

auf XML-Datenbestände zu sein, die entweder physisch in Form von XML-Dokumenten vorliegen oder nur virtuell existieren und durch eine Middleware-Schicht auf tatsächlich existierende autonome Systeme abgebildet werden. XQuery hat dabei das Ziel, nicht nur einen Zugriff auf XML-basierte Datenbestände zu realisieren, sondern weitreichende Möglichkeiten zur Filterung, Verknüpfung, Transformation und Konstruktion neuer XML-Dokumente zur Verfügung zu stellen. Der Umgang mit XML-Datenbanken und insbesondere die Kenntnis von XQuery wird in Zukunft den gleichen Rang wie SQL im Kontext relationaler Datenbanken einnehmen und damit fundamental sein. XQuery wird Eingang finden sowohl in den Bereich der Anwendungsentwicklung als auch in den Bereich der Lehre und Forschung. Dieser Tatsache versucht das vorliegende Buch Rechnung zu tragen und eine detaillierte Einführung in die zugrunde liegenden Sprachkonzepte von XQuery zu geben.

1.1 Warum ein XQuery-Buch?

Eine Vielzahl von Anwendungen basiert bereits auf XML-Technologien, wobei die strukturierte Ablage, der effiziente Zugriff sowie die Erzeugung abgeleiteter Datenbestände zentrale Anforderungen an XML-Technologien sind. Abbildung 1–1 zeigt die im Umfeld von XML existierenden Konzepte und Basistechnologien, wobei XQuery aus Sicht des Zugriffs auf Datenbestände einschließlich deren Filterung, Verknüpfung und Transformation eine Schlüsselrolle einnimmt.

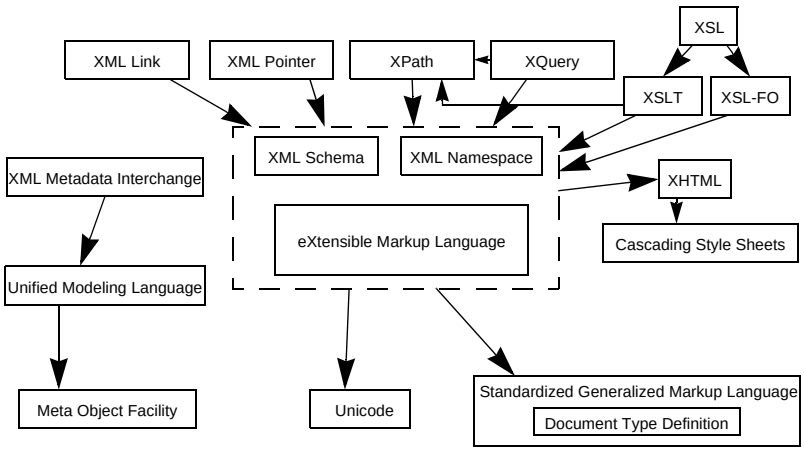

Abb. 1–1 *Übersicht über die XML-Sprachfamilie (nach [Jeck03])*

Konzentration auf XQuery

Insofern ist es an dieser Stelle wichtig, festzuhalten, dass das Buch nicht die gesamte Sprachfamilie abdecken kann, sondern sich auf XQuery konzentriert, wobei zusätzlich eine Auffrischung von XML-Grundkenntnissen wie XML-Strukturen, XML Schema etc. gegeben wird.

Anmerkungen zur Darstellung von XML- und XQuery-Beispielen

Download von Beispielen

Die Darstellung von XML-Fragmenten und XQuery-Anfragen zur Illustration von Sprachkonzepten birgt das Problem der Darstellung, da die Beispiele sehr schnell sehr voluminös geraten. Im Rahmen des Buches wird versucht, nur die zentralen Punkte herauszuarbeiten, so dass auf das Beispielszenario, welches unter `http://www.xquery-buch.de` zum Download zur Verfügung steht, per Referenz verwiesen wird.

Des Weiteren sei an dieser Stelle angemerkt, dass die Ergebnisse der Beispielanfragen oftmals insofern nicht ganz korrekt sind, als dass sie für die Präsentation im Rahmen des Buches formatiert sind, indem zusätzliche Leerzeichen und Zeilenabschlüsse eingefügt worden sind, um die Lesbarkeit zu erhöhen. Der Leser möge diese Unexaktheiten verzeihen.

Notation

Als letzte Anmerkung zur Darstellungsweise sei auf die Funktionsschreibweise hingewiesen. Bewusst wird an den Stellen im Buch, an denen XQuery-Funktionen meist direkt am Beispiel eingeführt werden, die allgemeine Signatur in tabellarischer Form aufgeführt. Dabei werden optionale Parameter mit einem []-Paar gekennzeichnet, so ist zum Beispiel bei der Funktion `fn:compare()` die Angabe des Parameters `collation` optional:

```
fn:compare(
  $comparand1 as xs:string?,
  $comparand2 as xs:string?[,
  $collation as xs:string])
as xs:integer?
```

In analoger Weise werden Signaturen von Funktionen kompakt dargestellt, wenn mehrere Varianten zur Auswahl stehen. Beispielsweise existieren die beiden Funktionen `fn:starts-with()` bzw. `fn:ends-with()`, die überprüfen, ob ein Suchstring in einer Zeichenkette jeweils zu Beginn oder am Ende auftritt. Derartige Varianten in den Signaturen werden verkürzt durch ein {}-Paar mit dem |-Symbol als Trenner repräsentiert:

```
fn:{starts|ends}-with(
  $arg1 as xs:string?,
  $arg2 as xs:string? [,
  $collation as xs:string]) as xs:boolean
```

Das nebenstehende Symbol tritt immer dann auf, wenn eine Eigenschaft von XQuery ungewöhnlich ist oder die Gefahr, einen Fehler zu machen, besonders gegeben ist.

Gliederung des Buches

Die Gliederung des Buches orientiert sich an dem Anspruch, ein Lehrbuch mit Fokus auf die Anfragesprache XQuery und die damit einhergehenden Sprachkonzepte zu sein. Das einleitende Kapitel beschäftigt sich dabei im weiteren Verlauf mit der Historie von XQuery und gibt einen Überblick über den Umfang und den aktuellen Stand der XQuery-Standardisierung. Eine Beschreibung des Beispielszenarios, welches im weiteren Verlauf des Buches zur Illustration der XQuery-Sprachkonzepte herangezogen wird, schließt die einführenden Betrachtungen ab.

Basiskonzepte

Das zweite Kapitel widmet sich den XML-Basiskonzepten. Ziel des Kapitels ist es, die Grundkonzepte von XML so weit aufzuarbeiten, dass die XQuery-Mechanismen verstanden werden können. Beginnend mit dem generellen Aufbau von XML-Dokumenten werden dann ausführlich die Konzepte der Namensräume, der XML-Schemabeschreibung und der Realisierung von Verweisen in XML-Dokumenten erläutert. Insbesondere die Schemabeschreibung mit XML Schema repräsentiert eine wesentliche Grundlage, die für das Verständnis von XQuery fundamental ist. Das vorbereitende Kapitel schließt mit einem Ausblick auf XQuery und illustriert exemplarisch die Mächtigkeit der Anfragesprache.

Datenmodell von XQuery

Das der Anfragesprache XQuery zugrunde liegende Datenmodell wird in Kapitel 3 im Detail eingeführt. Begonnen wird dabei mit dem Konzept der Sequenz, welches die fundamentale Datenstruktur in Kontext von XQuery repräsentiert. Daran schließt sich die Beschreibung von atomaren Werten, des Vorgangs der Atomisierung und der Einführung spezieller Typen für XQuery am Beispiel der XQuery-Typhierarchie an. Die beiden darauf folgenden Abschnitte widmen sich der Aufarbeitung des Knotenkonzeptes, wobei zum einen die unterschiedlichen Knotenarten wie Element, Attribut etc. illustriert werden und zum anderen der Konstruktionsmechanismus einschließlich der unterschiedlichen Eigenschaften beschrieben wird. Das Kapitel über das XQuery-Datenmodell wird inhaltlich abgeschlossen durch die Darstellung von Typausdrücken, die eine Prüfung, Zuweisung, Zusicherung und eine Typabfrage ermöglichen.

XQuery-Sprachkonzepte

Die drei folgenden Kapitel 4, 5 und 6 behandeln den Kern der XQuery-Sprachkonzepte. Dabei konzentriert sich Kapitel 4 auf den

Mechanismus der Pfadausdrücke und erläutert im Detail den Mechanismus der Lokalisierungsschritte in Verbindung mit Navigationsachsen, Knotentests und zusätzlich existierenden Prädikaten. Kapitel 5 fokussiert sich auf die FLWOR[1]-Ausdrücke, wobei in einem ersten Teil die Semantik der unterschiedlichen Klauseln erklärt wird, während ein zweiter Teil sich auf die Formulierung von Verbund und Aggregationsanfragen konzentriert. Kapitel 6 arbeitet schließlich erweiterte Konzepte von XQuery auf. Dabei werden beispielsweise logische, konditionale und quantifizierende Ausdrücke beschrieben und am Beispiel illustriert.

Funktionsbibliothek

Kapitel 7 stellt wesentliche Bestandteile der XQuery-Funktionsbibliothek vor, wobei mit Funktionen auf numerische und boolesche Werte begonnen wird und über Funktionen auf Zeichenketten zu der Verarbeitung von Zeit- und Datumsangaben übergegangen wird. Kapitel 7 wird abgeschlossen durch die Darstellung der Möglichkeit, benutzerdefinierte Funktionen in XQuery zu formulieren.

XQuery-Prolog

Der letzte Teil der Vorstellung von XQuery-Sprachkonstrukten beherbergt in Kapitel 8 eine Beschreibung des Modulkonzeptes, des XQuery-Prologs und des Verarbeitungskonzeptes mit statischem und dynamischem Kontext.

Ausblick

Kapitel 9 gibt einen Ausblick auf zukünftige bzw. bereits stattfindende Entwicklungen, die jedoch (noch) nicht Gegenstand des aktuellen Standardisierungsprozesses sind. Hier werden Themen wie Änderungsoperationen an XML-Dokumenten, erweiterte Lesezugriffe (beispielsweise Sichtenkonzept) und XQueryX als XML-basierte Repräsentation einer XQuery-Anfrage in aller Kürze behandelt. Das Buch schließt mit einer Zusammenfassung (Kapitel 10) und einem umfangreichen Anhang, welcher zum Nachschlagen von Details dient.

1.2 XQuery-Standardisierung

Eine ausführliche Schilderung der teilweise recht komplexen Vorgänge der Standardisierung beim W3C würde den Rahmen dieses Buches sprengen. Ein kurzer Abriss der Geschichte von XQuery und einige Bemerkungen zum Fortschritt der Standardisierung sollen aber dennoch gegeben werden.

1. Das Akronym FLWOR steht für die Folge von `For`-, `Let`-, `Where`, `Order-By`- und `Return`-Klauseln.

1.2.1 Verlauf der Standardisierung

Schon kurz nach der endgültigen Verabschiedung von XML 1.0 durch das *World Wide Web Consortium* (W3C; `http://www.w3.org/`) im Februar 1998 war abzusehen, dass XML in vielen Bereichen des Web, aber auch in ganz lokalen Anwendungen eine entscheidende Rolle spielen würde, und zwar nicht nur als Format für den Datenaustausch, sondern auch als ein Format für die dauerhafte Ablage. Als Konsequenz entstand die Frage nach einer Anfragesprache für XML. Im Dezember 1998 veranstaltete das W3C einen Workshop zum Thema »Query-Sprache für XML« (`http://www.w3.org/TandS/QL/QL98/`), in dem viele Sprachvorschläge, aber auch grundsätzliche Betrachtungen zu den Anforderungen an eine XML-Anfragesprache (beispielsweise [Maie98]) diskutiert wurden. In der Folge wurde die *XML Query Working Group* beim W3C (`http://www.w3.org/XML/Query`) gegründet, deren Ergebnis eine konsolidierte Liste von Anforderungen an eine XML-Anfragesprache war [W3C-10]. Dabei wurde auch untersucht, ob überhaupt die Notwendigkeit einer eigenen Anfragesprache für XML besteht oder ob nicht eine etablierte Sprache wie etwa SQL diesen Zweck ebenso erfüllen könnte. Wegen der zu großen Unterschiede zwischen XML und dem relationalen Modell wurde jedoch die Definition einer eigenen Sprache angestrebt [Katz04].

Anforderungen an die Sprache

Koordination der Aktivitäten

Es war von Anfang an unstrittig, dass die Aktivitäten der XML Query Working Group mit existierenden und entstehenden W3C-Standards koordiniert werden mussten, insbesondere mit XML 1.0 [W3C-3], XML Schema [W3C-14], XML Namespaces [W3C-4], XML Information Set [W3C-5] und XSLT [W3C-21]. Wegen der weiten Verbreitung von XPath [W3C-7], z. B. in W3C-Standards wie XSLT und XPointer [W3C-9], aber auch – in Ermangelung einer Alternative – als Anfragesprache für XML in kommerziellen Datenbanksystemen [Schö03], wurde beschlossen, XPath zu einem Teil von XQuery zu machen. Da gleichzeitig die XSL Working Group (`http://www.w3.org/Style/XSL/`) die Anforderungen für neue XPath-Funktionalität zusammentrug, und diese sich erheblich mit den Anforderungen an die neu zu definierende XML-Anfragesprache überschnitten, wurde beschlossen, XQuery gemeinsam in beiden Arbeitsgruppen zu entwickeln. Tatsächlich bilden XPath 2.0 und XQuery nun eine Familie von Spezifikationen und die Spezifikation für XPath 2.0 und die für XQuery werden aus einer gemeinsamen Textquelle erzeugt. XQuery ist allerdings nicht voll kompatibel zu XPath, XPath 2.0 hingegen schon. Auf derartige syntaktische Details wird im Rahmen dieses Buches aber nicht weiter eingegangen.

Seit 1999 widmet sich die Working Group der Definition von XQuery als der standardisierten Anfragesprache für XML. Zur Ausgangsbasis dafür wurde die Sprache Quilt [ChRF00], die wiederum Bausteine aus verschiedenen anderen Sprachen (XPath, XQL [Robi99], XML-QL [DFF+98], SQL [ISO9075], OQL [CaAt96]) entnommen hatte – der Name weist deutlich darauf hin[2].

1.2.2 Fortschritt der Standardisierung

Obwohl die XML Query Working Group nun schon seit 1999 an der Standardisierung von XQuery arbeitet, liegt die Spezifikation nur als *Last Call Working Draft* vor (Stand Januar 2004). Das bedeutet, dass noch einmal Kommentare gesammelt werden, bevor die Arbeitsgruppe eine *Candidate Recommendation* erzeugt (falls die Kommentare nicht grundsätzliche Probleme aufwerfen). Die Herausgabe einer Candidate Recommendation ist eine Aufforderung, erste Implementierungen der Spezifikation zu erstellen, damit festgestellt werden kann, welche Teile der Spezifikation vielleicht noch der Überarbeitung bedürfen, weil sie schwierig zu implementieren oder aus anderen Gründen nicht praktikabel sind. Wenn es zu jedem Aspekt der Spezifikation Implementierungen gibt, kann die Arbeitsgruppe die Spezifikation zur *Proposed Recommendation* weitertreiben, die dann den Mitgliedern des W3C nochmals zur Begutachtung vorgelegt wird. Ist diese erfolgreich, wird die Spezifikation normalerweise zur *Recommendation*, erreicht also den endgültigen Stand der Standardisierung. In jeder vorherigen Stufe kann die Spezifikation wieder zum Working Draft werden, wenn schwerwiegende Mängel entdeckt werden.

Implementierungen

Es ist also für XQuery noch ein langer Weg zur endgültigen Standardisierung. Allerdings gibt es schon sehr viele Implementierungen von XQuery (`http://www.w3.org/XML/Query`), die jedoch nicht alle den letzten Stand des Working Draft, sondern teilweise ältere Stände realisieren, und diese auch nicht immer vollständig.

1.2.3 Teile der Spezifikation

Hatte die erste XML-Spezifikation (XML 1.0) noch aus einem einzigen Dokument mit wenigen Seiten bestanden, so zeigte sich schon mit XML Schema die Tendenz zu umfangreicheren Spezifikationen. Die XQuery-Arbeitsgruppe hat diesen Trend in den vier Jahren ihres Beste-

2. Quilt bezeichnet eine Steppdeckenart, wie sie in Nordamerika aus verschiedenen Stoffstücken hergestellt wird (Patchwork).

hens fortgesetzt. Die Spezifikation zu XQuery und XPath 2.0 setzt sich (in ihrer ersten Version) aus zehn Einzeldokumenten zusammen:

- Die *XML Query Requirements* [W3C-10] fassen die Anforderungen an die zu standardisierende Anfragesprache zusammen. Sie werden allerdings immer noch fortentwickelt.
- *XPath Requirements Version 2.0* [W3C-15] ist das entsprechende Dokument mit Anforderungen an XPath 2.0.
- In den *XQuery Use Cases* [W3C-12] werden die Anwendungsfälle zusammengestellt, die von XQuery abgedeckt werden sollen. Die »Requirements« und die »Use Cases« waren die ersten beiden Dokumente aus der XQuery-Familie.
- Das *XQuery 1.0 and XPath 2.0 Data Model* [W3C-16] beschreibt das gemeinsame Datenmodell von XPath 2.0 und XQuery. Es übernimmt die sieben Knotenarten des XPath-1.0-Datenmodells. Als grundlegenden Datentyp von XQuery wird zudem noch die Sequenz eingeführt. Dieses Datenmodell wird in Kapitel 3 näher behandelt.
- In *XSLT 2.0 and XQuery 1.0 Serialization* [W3C-23] wird beschrieben, wie aus dem Ergebnis einer XQuery-Anfrage ein XML-Dokument (oder eine andere Ausgabeform) entsteht. Hierauf gehen wir in Abschnitt 8.3.3 näher ein.
- *XQuery 1.0 and XPath 2.0 Formal Semantics* [W3C-17] definiert eine Untermenge von XQuery, auf die alle XQuery-Anfragen abgebildet werden können. Die Semantik dieser Untermenge wird dort ebenfalls spezifiziert.
- *XQuery 1.0: An XML Query Language* [W3C-19] ist die eigentliche XQuery-Spezifikation. Hier werden die Syntax der Anfragesprache sowie ihre Semantik und ihr Verarbeitungsmodell definiert.
- *XML Syntax for XQuery 1.0 (XQueryX)* [W3C-20] definiert eine XML-basierte Syntax für XQuery. Dies war eine der Anforderungen an eine XML-Anfragesprache, die die XML Query Working Group schon sehr früh formuliert hatte. Ob diese XML-basierte Syntax Bestand haben wird, bleibt (aus unserer Sicht) abzuwarten.
- Die in XQuery eingebauten Funktionen sind in *XQuery 1.0 and XPath 2.0 Functions and Operators* [W3C-18] definiert. Wir behandeln sie in Kapitel 7. Außerdem werden in diesem Teil der Spezifikation so genannte Operatoren eingeführt, die nicht direkt aufgerufen werden können, sondern dazu dienen, die Semantik der XQuery-Operatoren (wie z. B. `+` oder `gt`) genau festzulegen.
- *XML Path Language (XPath) 2.0* [W3C-8] definiert XPath 2.0 und ist aus derselben Textquelle erzeugt wie [W3C-19].

Die meisten dieser Spezifikationen liegen im Augenblick (Juni 2004) im Stand vom 12. November 2003 als *Last Call Working Draft* vor. Es gibt schon zwei weitere Spezifikationen (Abschnitt 9.4), die jedoch für die erste Version von XQuery keine Rolle spielen werden.

Stand 12. November 2003

1.2.4 Stand der XQuery-Implementierungen

Die Anzahl von Versuchen, XQuery im Rahmen einer Implementierung umzusetzen, ist mittlerweile recht hoch, wobei prinzipiell zwischen frei verfügbaren und kommerziellen Systemen differenziert werden muss. Alle verfügbaren Implementierungen unterscheiden sich hinsichtlich der Umsetzung des Sprachstandards sowohl untereinander als auch zeitlich gesehen mit Blick auf die ständige Weiterentwicklung. Eine Auflistung der entsprechenden Implementierungen würde den Rahmen des Buches sprengen, so dass an dieser Stelle auf das von W3C gepflegte Verzeichnis (`http://www.w3.org/XML/Query/`) verwiesen wird.

1.3 Beispielszenario

Durch das gesamte Buch hindurch zieht sich das Beispielszenario der »Hochwaldklinik«, welches als Grundlage für die zahlreichen Erklärungen dient. Als Diskursbereich wird (radikal vereinfacht) die Verwaltung eines Krankenhauses mit all seinen Einrichtungen, Mitarbeitern und Patienten betrachtet.

Die Abbildung auf das XML-Datenmodell resultiert in zwei Klassen von Datenbeständen – Stammdaten und Bewegungsdaten. Stammdaten beschreiben das Krankenhaus zusammen mit den dazugehörigen Einrichtungen und Mitarbeitern. Den größten Teil der Bewegungsdaten nehmen die Patientendaten ein, wobei zusätzlich die Zeiterfassung und Verbrauchsartikel aufgezeichnet werden.

Das Beispielszenario ist elektronisch verfügbar unter `http://www.xquery-buch.de`.

Stammdaten

Die Stammdaten sind in der XML-Datei `Hochwaldklinik.xml` abgelegt. Eine Klinik ist in dem Beispielszenario durch eine Bezeichnung der Einrichtung, eine Adresse (Datentyp `Adresse_T`) und durch drei weitere Bereiche beschrieben (Abbildung 1–2):

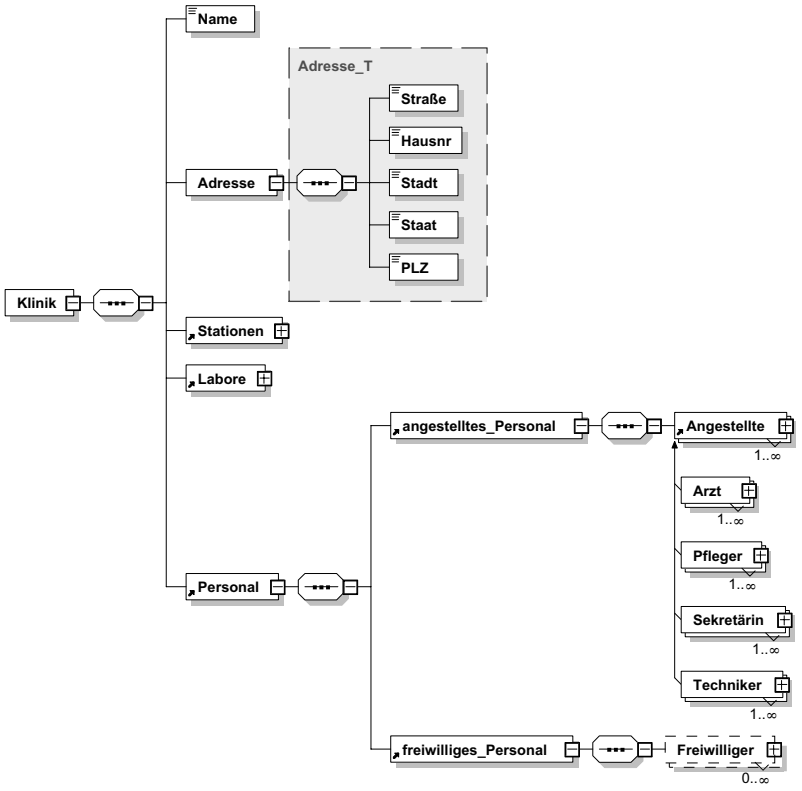

Abb. 1–2 *Aufbau des Krankenhausszenarios*

- *Stationen*
  Ein Krankenhaus kann eine Vielzahl unterschiedlicher Stationen umfassen, wobei jede Station durch einen Namen, einen Standort und eine a priori unbegrenzte Anzahl von Betten beschrieben ist. Darüber hinaus ist jeder Station ein Pfleger in Form einer Referenz auf einen Mitarbeiter des Pflegepersonals zugeordnet (Abbildung 1–3).
- *Labore*
  In Analogie zu den Stationen existieren Laboreinrichtungen, die ebenfalls durch eine Bezeichnung und einen Standort näher charakterisiert werden. Während es jedoch mindestens eine Station in einem Krankenhaus geben muss, kann auf externe Laboratorien zurückgegriffen werden, so dass nicht notwendigerweise ein Krankenhaus auch ein Labor aufweisen muss.
  Abbildung 1–3 zeigt den Aufbau der Stationen und der dem Krankenhaus zugeordneten Laboreinrichtungen.

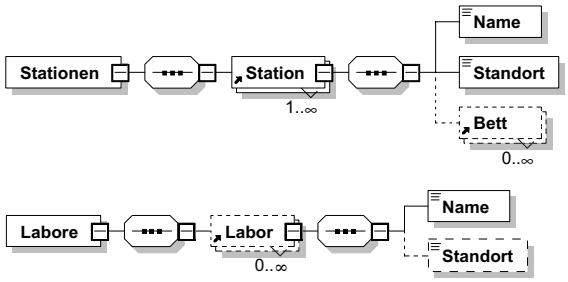

Abb. 1–3 *Strukturierung der Stationen und der Laboreinrichtungen*

- *Personal*
  Das Personal des Krankenhauses gliedert sich in den Teil der Freiwilligen und des fest angestellten Personals, welches wiederum in vier unterschiedliche Kategorien unterteilt wird (Abbildung 1–2).
  Freiwillige Mitarbeiter weisen neben den Angaben zu Name, Adresse, Geburtsdatum und Telefon weiterhin eine innerhalb des Krankenhauses eindeutige Nummer, eine besondere Fähigkeit und Angaben bezüglich der Zugehörigkeit zu einer Berufsklasse auf. Abbildung 1–4 zeigt die Struktur innerhalb des entsprechenden XML-Dokumentes. Der Datentyp `Person_T` für persönliche Angaben wie Name, Adresse etc. wird dabei von allen Entitäten, die Personalinformationen (Ärzte, Patienten etc.) repräsentieren, verwendet.
  Die weiteren Berufsgruppen des angestellten Personals umfassen Ärzte, Pfleger, Techniker und Sekretärinnen, die jeweils neben den Angaben zur Person mit entsprechend eindeutiger Nummer weitere Attribute für E-Mail und Einstellungsdatum und jeweils berufsgruppenspezifische Eigenschaften zur detaillierten Beschreibung besitzen.
  Abbildung 1–5 zeigt die Struktur des bezahlten Personals mit den zusätzlichen Eigenschaften; so besitzen Ärzte neben einer bestimmten Fähigkeit ein Spezialgebiet und eine Pagernummer, unter welcher der Arzt innerhalb der Klinik erreichbar ist. Pfleger können maximal fünf Zertifikate aufweisen; Techniker weisen sich durch ihre besondere Fähigkeit (Labortechniker, Haustechniker, ...) aus. Wichtig ist an dieser Stelle, dass alle Eigenschaften von dem Obertyp `Angestellte` mit Angaben zur Person (`Person_T`) und Angaben zum Angestelltenverhältnis geerbt werden.

In die Kategorie der Stammdaten fällt weiterhin das Verzeichnis aller Verbrauchsartikel, welches in der gleichnamigen Datei `Verbrauchsartikel.xml` abgespeichert ist. Abbildung 1–6 zeigt die entsprechende Struktur, wobei ein Attribut `Medikament` des Elementes `Artikel` eine

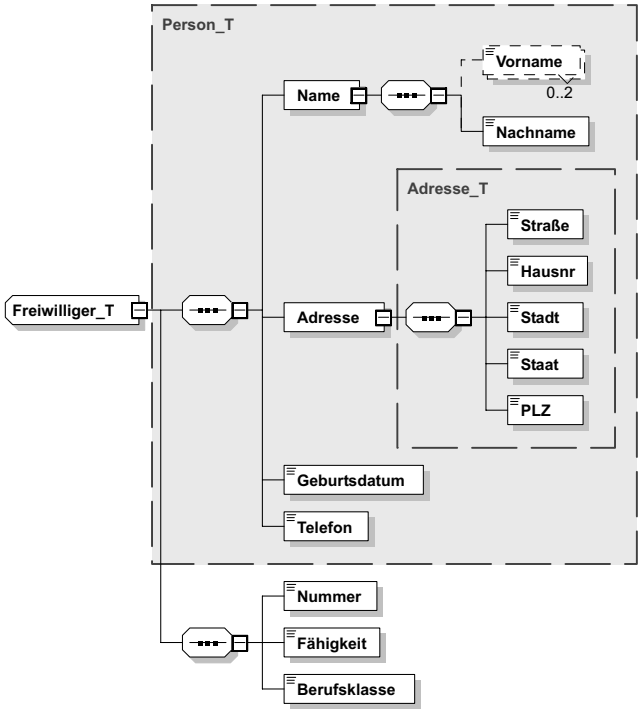

Abb. 1–4 *Strukturierung freiwillig Tätiger*

Unterscheidung in »Allgemeines Verbrauchsmaterial« und »Medikamente« erlaubt.

Patientendaten

Der zweite große Bereich des Beispielszenarios umfasst die Ablage von Patienteninformationen (Datentyp `Patient_T`). Auf oberster Ebene werden die beiden Klassen von stationären und ambulanten Patienten unterschieden (Abbildung 1–7). Jeder Patient hat selbstverständlich Angaben zur Person für Name, Adresse, Geburtsdatum und Telefon und ist damit eine Spezialisierung des Datentyps `Person_T`. Darüber hinaus weisen Patienten ein Kontaktdatum und die Anamnese sowie die durchgeführten Untersuchungen und die erhaltenen Leistungen auf. Eine Untersuchung umfasst ein Datum, eine Angabe über Art und Menge der verbrauchten Artikel, mehrere Befunde und einen Verweis in Form einer XLink-Angabe auf den behandelnden Arzt.

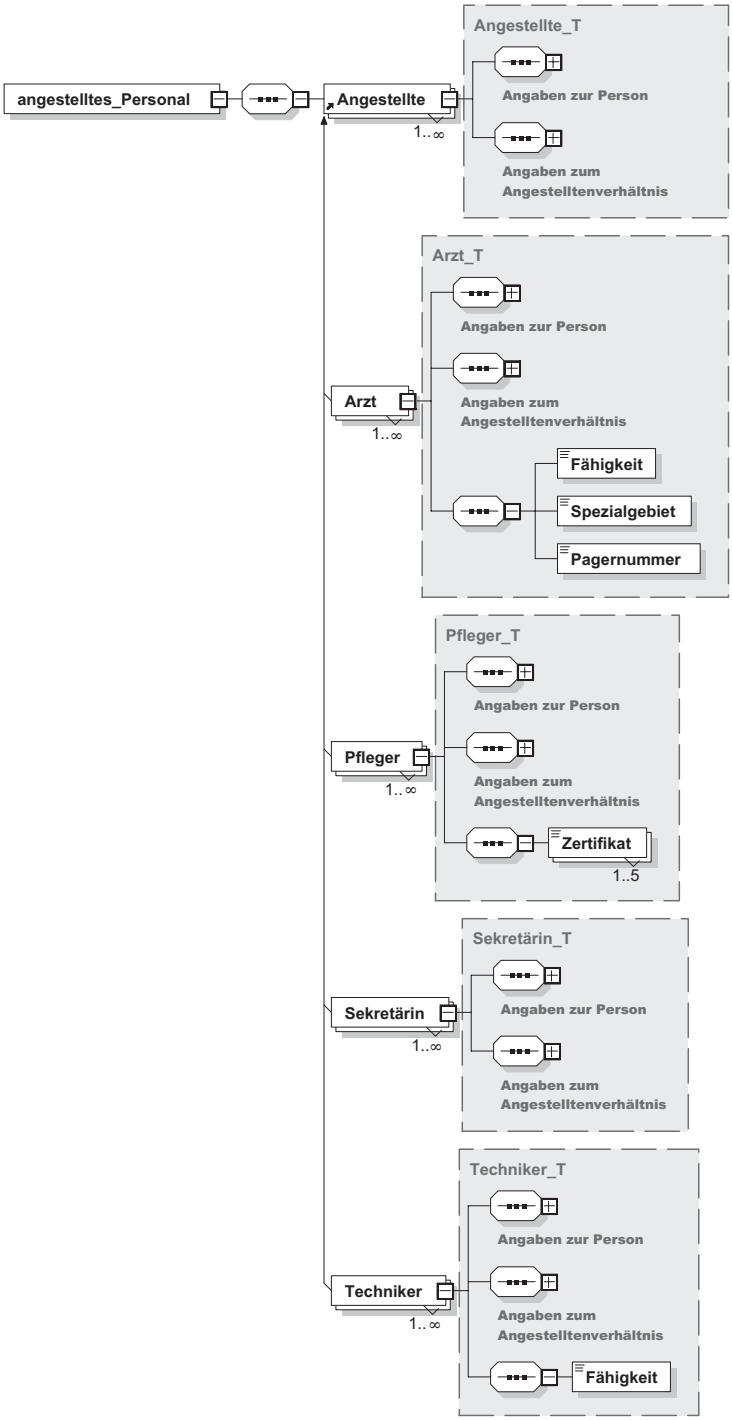

Abb. 1–5 *Strukturierung des angestellten Personals*

Abb. 1–6 *Strukturierung der Verbrauchsartikel*

Ein stationärer Patient enthält zusätzlich Informationen über seine Einlieferung einschließlich einer Referenz auf den einweisenden Arzt. Eine Einlieferung setzt sich wiederum zusammen aus einem Datum, einem Grund für die Einweisung, dem Ort der Unterbringung in der Klinik (Station und Bett) und einer Referenz auf den betreuenden Arzt.

Leistungen

Die zentrale Entität im Datenbestand der Patienten reflektiert die einzelnen Leistungen. Abbildung 1–5 zeigt schematisch die Struktur mit den folgenden näher skizzierten Leistungsarten:

- *Therapie*
  Eine Therapie findet unter Leitung eines Arztes auf einer Station in Form von einer Vielzahl von Sitzungen statt. Jede Sitzung besteht dabei wiederum aus einem Beginn- und Endezeitpunkt sowie optional einer Menge verbrauchter Artikel und einem Ergebnisbericht.
- *Behandlung*
  Eine Behandlung ist eine abgeschlossene Leistung, die von einem Arzt an einem bestimmten Datum durchgeführt wird. Die Angaben zur Station, d. h. dem Ort der Behandlung, zu den verbrauchten Artikeln sowie zu dem Ergebnis sind optional.
- *Operation*
  Eine Operation ist wiederum komplex strukturiert und wird im Folgenden detailliert erläutert.
- *Labortest*
  Ein Labortest wird über eine eindeutige Nummer identifiziert und basiert auf einer Reihe von Testgegenständen (vom Typ `xs:anyType`), wobei optional eine Vielzahl von Testwerten – bestehend aus der Bezeichnung der Kennzahl und dem entsprechenden Wert – ermittelt wird. Ein Labortest wird üblicherweise an einem bestimmten Tag durchgeführt und findet grundsätzlich in einem per Referenz identifizierten Labor statt.

Dabei ist generell anzumerken, dass die einzelnen Leistungen eines Patienten eine Ordnung hinsichtlich der zeitlichen Reihenfolge aufweisen.

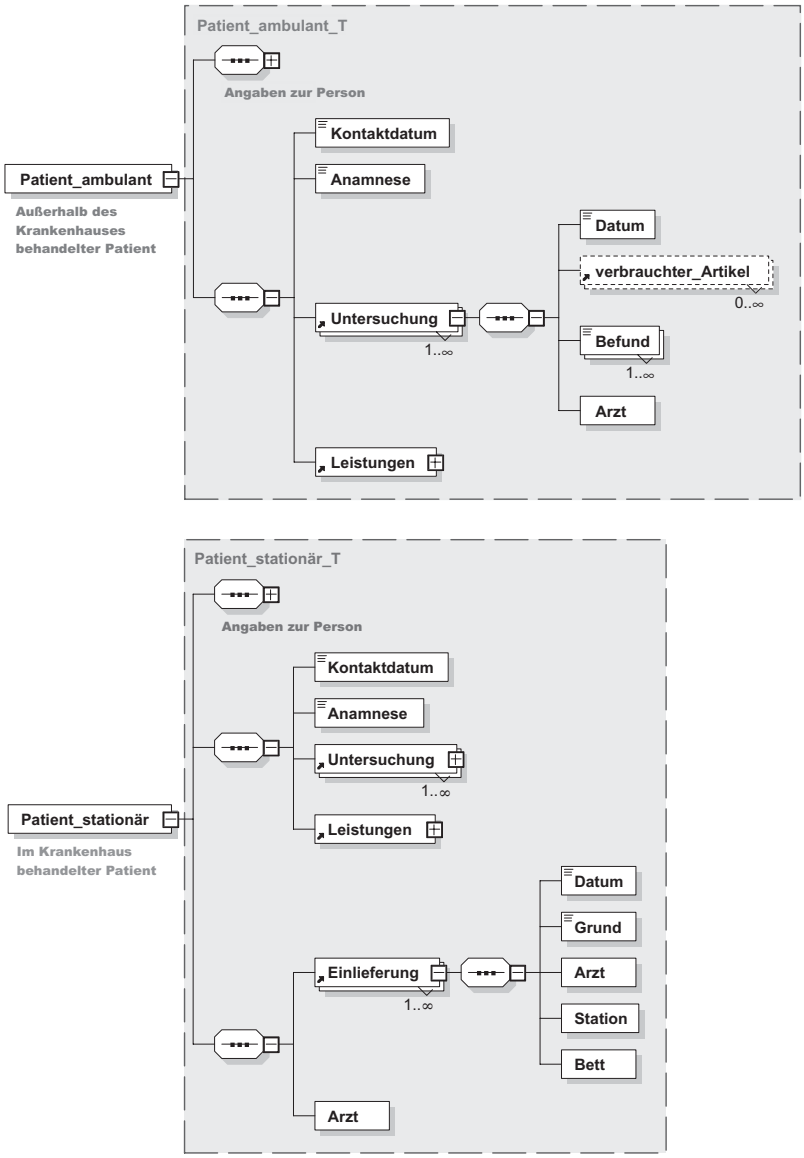

Abb. 1–7 *Strukturierung ambulanter und stationärer Patienten*

Analog gliedert sich die Leistung einer Operation in eine (möglicherweise – im Fall eines sofortigen Abbruchs – leere) Liste von einzelnen Vorgängen. Abbildung 1–9 zeigt die Struktur und insbesondere die unterschiedlichen Vorgänge innerhalb einer Operation. Eine Operation wird von einer Menge von Ärzten durchgeführt, wobei im Attribut `Rolle` die Tätigkeiten bzw. Verantwortung des jeweiligen Arztes im

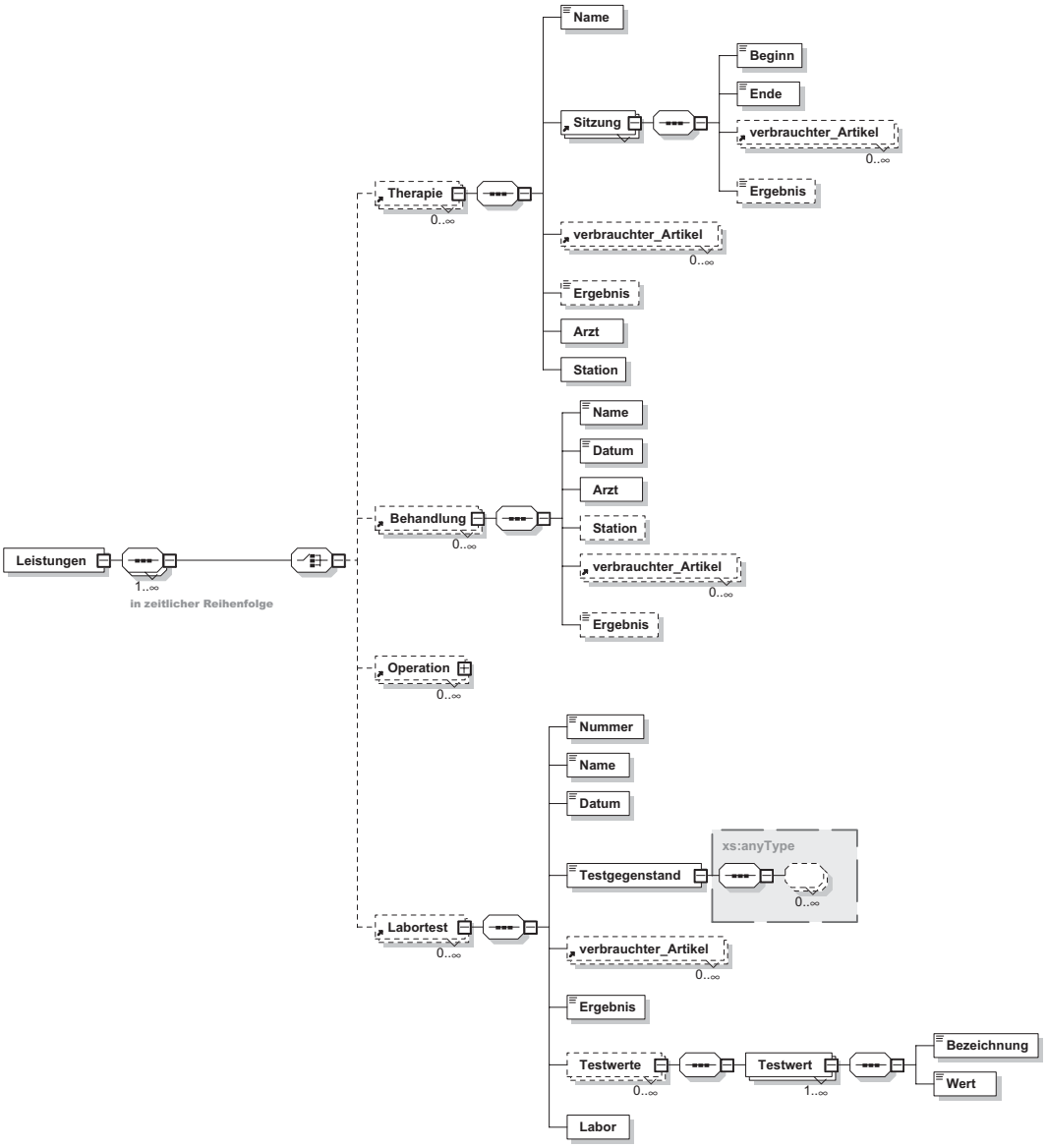

Abb. 1–8 *Strukturierung unterschiedlicher Leistungen*

Rahmen dieser Operation festgehalten wird. Beginn und Ende repräsentieren Werte vom Datentyp `xs:dateTime` und geben den Zeitpunkt bzw. indirekt die Zeitdauer der Operation an. Alle möglichen Vorgänge treten geordnet bzgl. ihres Zeitpunktes in dem Patientendatenbestand auf. Die Vorgänge der Anästhesie, der Injektion und der Infusion beinhalten ausschließlich eine Angabe zum verbrauchten

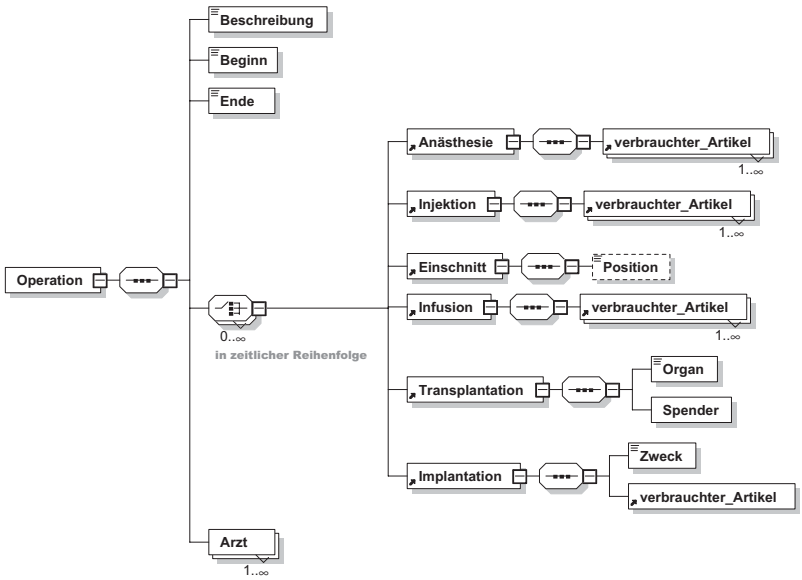

Abb. 1–9 *Strukturierung einer Operation*

Material. Bei einem Einschnitt kann optional die Position aufgezeichnet werden. Eine Transplantation weist ein Organ und – für die Nachvollziehbarkeit – einen Verweis auf den Spender auf. Der Vorgang der Implantation schließlich enthält als Unterelemente einen Zweck und einen Verweis auf das Implantat, welches ebenfalls im Artikelkatalog verwaltet wird.

1.4 Zusammenfassung

XQuery hat sich in den letzten Jahren als Anfragesprache für XML-basierte Datenbestände gebildet und ist – aus Sicht der Standardisierung – zumindest im Bereich des lesenden Zugriffs sehr weit gediehen. Dieses Kapitel gibt zunächst die Motivation für die Anfertigung dieses Buches und erläutert ausführlich den Prozess und aktuellen Stand der Entwicklung und Standardisierung von XQuery. Darüber hinaus wird in Abschnitt 1.3 das im Buch durchgängig verwendete Beispielszenario erläutert, wobei die großen Teile der Stammdaten zur Beschreibung der Klinik einschließlich ihrer Einrichtungen und des Personals von den Patientendaten mit den am Patienten durchgeführten Untersuchungen und Leistungen zu unterscheiden sind.