Dynamische XML-Verarbeitung

Bernhard Eichhorn
MIC Management Consulting GmbH
München

Rainer Krohn
EDV-Beratung Rainer Krohn GmbH
Aßling

Schlüsselworte:
Dynamisch generische Verarbeitung von XML-Daten, Mapping von XML-Komponenten und relationalem Datenmodell, XML-Schema-Generierung, dienst- und anwendungsspezifische Datenvalidierung, Dynamic SQL

Einleitung

XML kommt als Datenaustauschformat in fast allen Bereichen der Datenverarbeitung zum Einsatz. Insbesondere in der Versicherungswelt ist XML die Grundlage für den Datenaustausch z.B. bei Meldungen an die Zentrale Zulagenstelle für Altersvermögen (ZfA) im Falle der Riesterrenten.
Die zugrunde liegenden XML-Strukturen sind dabei mit großen Änderungshäufigkeiten behaftet.
So wird z.B. beim Datenaustausch mit der ZfA nach 5 Jahren Riesterrente aktuell mit der XML-Version 19.b gearbeitet.
Schon heute lässt sich absehen, dass auch in Zukunft immer mehr Zusatzinformationen in die XML-Strukturen integriert werden müssen, z.B. Steuernummer, IBAN, BIC, etc..
Mit der hier vorzustellenden, generischen Realisierung der XML-Verarbeitung wird ein Weg aufgezeigt, wie die mit den Erweiterungen immer wiederkehrenden Anpassungsaufwände minimiert werden können.

Projektbeschreibung und Aufgabenstellung

Vor dem Hintergrund der zu erwartenden umfangreichen Erweiterungen an zentralen XML-Strukturen wurden wir von der Swiss Life AG, Niederlassung Deutschland, beauftragt, eine XML-Verarbeitung zu realisieren, die folgende Anforderungen erfüllt:

- Direkte Verarbeitung von XML-Datensätzen in relationale Datenbankstrukturen
- Die Verarbeitung soll sich invariant bei Strukturänderungen an den Datenbanktabellen und/oder XML-Strukturen verhalten, d.h. die Anpassungsaufwände für die Integration neuer Strukturkomponenten sollen mit minimalem, im Idealfall gar keinem Programmieraufwand möglich sein.
- Erkennung von Strukturänderungen
- Dynamische Einbindung beliebiger Validierungsroutinen:
  - semantische Prüfung, z.B. gültige Wertebereiche, Pflichtfelder, Abhängigkeiten von XML-Komponenten untereinander
  - Datenanreicherung, z.B. Versorgung der Primary-Key-Spalten beim INSERT
  - Rückgabe sämtlicher Fehlermeldungen über kompletten XML

Als fachlicher Hintergrund für die Realsierung diente die Anbindung eines neuen Provisionssystems an das zentrale Partnersystem zur Pflege von Partnerdaten, Adressen und Bankverbindungen, wobei die Kommunikation zwischen der Anwendung und der Datenbank über den Enterprise Service Bus (ESB) erfolgt.

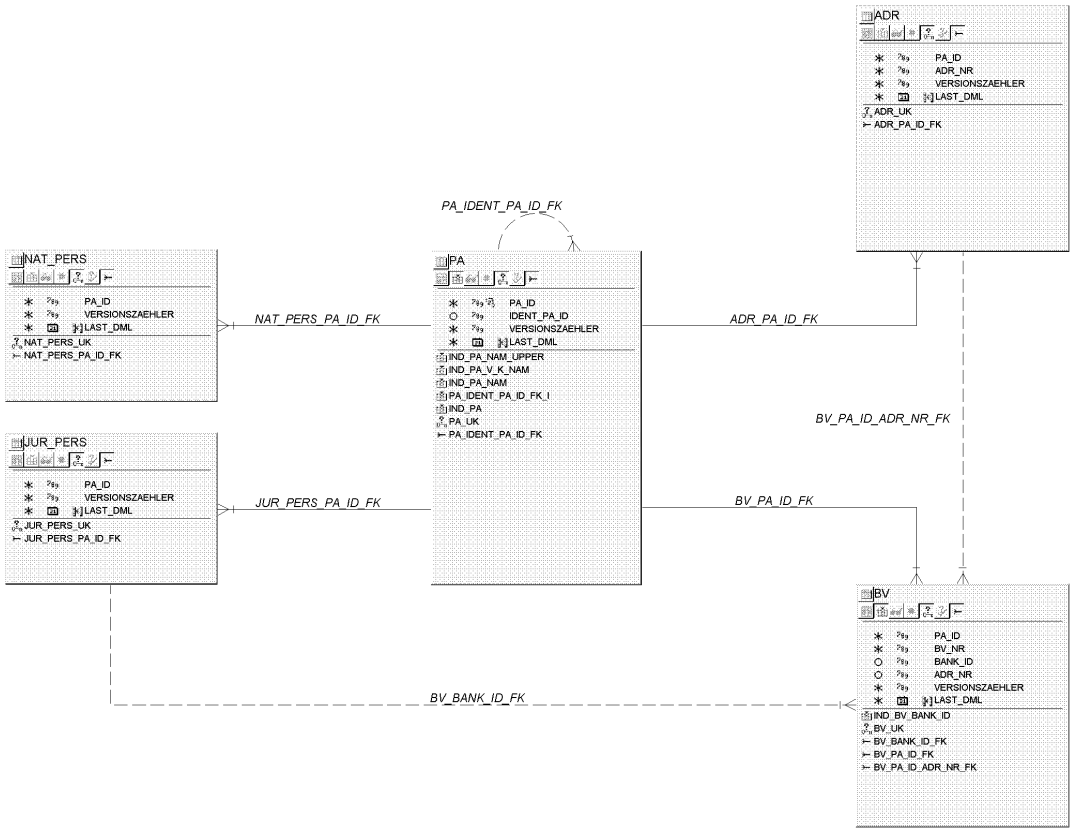

Abb. 1: Auszug Datenmodell PARTNER

Für die einzelnen Bereiche der Pflege von Partner-, Adress- und Bankverbindungsdaten waren eigene XML-Strukturen vorgegeben, die sowohl für Insert-, Update- oder Delete-Operationen verwendet werden sollten. Das Partnersystem sollte für jede XML-Struktur einen eigenen Dienst zur Verfügung stellen, der die XML-Daten validiert und in die entsprechenden Tabellen einarbeitet.

Realisierung

- Entwicklung eines Metadatenmodells
  - Mapping vom XML-Strukturen auf relationale Datenbanktabellen
  - Erkennung von Strukturänderungen
  - XML-Schema-Generierung
  - Zuordnung von dienst- und anwendungsspezifischen Validierungsroutinen
  - Hinterlegung von dienst- und anwendungsspezifischen Regeln
- PL/SQL
  - intensive Nutzung von Dynamic SQL:
    - Einbindung Validierungsroutinen zur Laufzeit
    - Generierung der DML-Statements zur Laufzeit
  - XMLType-Komponenten von Oracle
- keine Oracle XMLDB-Funktionalitäten

Entwicklungsstufe 1
Im ersten Schritt bei der Entwicklung des Metadatenmodells war es das Ziel, ein Mapping zwischen den Komponenten der XML-Struktur und den zugehörigen Tabellenspalten im Zieldatenbankschema zu realisieren. Weiters sollten im Metadatenmodell alle für die Generierung des XML-Schemas eines Dienstes notwendigen Informationen hinterlegt werden können.

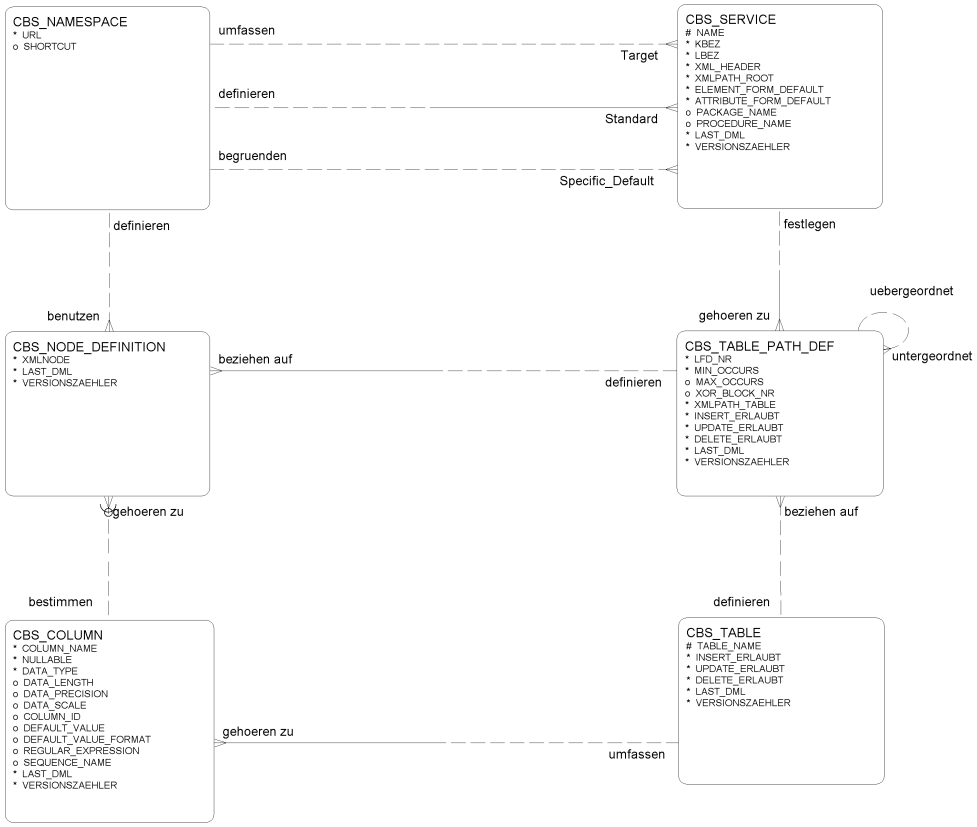

Abb. 2: Auszug Metadatenmodell XML-DB-Mapping

Für die Definition eines Dienstes werden neben seinem Namen in der Tabelle CBS_SERVICE alle Informationen zur Generierung des XML-Headers und der Name des Root-Knotens der XML-Instanz hinterlegt.

Beispiel generierter XML-Header:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsi:schema
   xmlns:xsi="http://www.w3.org/2001/XMLSchema"
   xmlns="http://partner.swisslife.de/services/partner"
   targetNamespace="http://partner.swisslife.de/services/partner"
   elementFormDefault="qualified"
   attributeFormDefault="unqualified"
   version="0.0.1">
```

Mit dem Eintrag einer Tabelle des Zieldatenbankschemas in CBS_TABLE werden alle zugehörigen Spalten aus dem Data-Dictionary in die Tabelle CBS_COLUMN übernommen.

Die Verknüpfung eines Dienstes mit den zu pflegenden Datenbanktabellen erfolgt über die Tabelle CBS_TABLE_PATH_DEF, wobei der hierarchische Aufbau der XML-Struktur in die Tabellenhierarchie des Dienstes übernommen wird.
Jede Tabelle kann innerhalb eines Dienstes mehrfach verwendet werden.
Weiters können Tabellen auf der gleichen Hierarchiestufe mit einer Oder-Verknüpfungen versehen werden, da z.B. die Partner in natürliche und juristische Personen unterschieden werden müssen.
Eine Oder-Verknüpfung findet sich dann im XML-Schema als choice-Gruppe wieder.
Bei der Befüllung der Tabelle CBS_NODE_DEFINITION werden standardmäßig die Spaltennamen als Name des XML-Elements übernommen.

Beispiel generiertes XML-Schema:

```
  <xsi:complexType name="CT_ADR_Partnerdemo">
    <xsi:all>
      <xsi:element name="ADR_NR" nillable="false" minOccurs="1">
        <xsi:simpleType>
          <xsi:restriction base="xsi:integer">
          </xsi:restriction>
        </xsi:simpleType>
      </xsi:element>
      <xsi:element name="ADR_TYP" nillable="true" minOccurs="0">
...
...
...
  </xsi:complexType>

  <xsi:complexType name="CT_PA_Partnerdemo">
    <xsi:all>
      <xsi:element name="ANR" nillable="true" minOccurs="0">
        <xsi:simpleType>
          <xsi:restriction base="xsi:string">
            <xsi:maxLength value="4" />
          </xsi:restriction>
        </xsi:simpleType>
      </xsi:element>
...
...
...
  <xsi:element name="Partnerdemo_in">
    <xsi:complexType>
      <xsi:sequence>
        <xsi:element name="PA" type="CT_PA_Partnerdemo" minOccurs="1"
           maxOccurs="1">
        </xsi:element>
      </xsi:sequence>
    </xsi:complexType>
  </xsi:element>
</xsi:schema>
```

Entwicklungsstufe 2

Das oben generierte XML-Schema enthält als Komponenten alle Spalten der dem Dienst zugeordneten Tabellen. Um jedoch fehlerfreie DML-Statements generieren zu können, werden weitere Informationen benötigt, die zur

- Ausblendung irrelevanter XML-Komponenten pro Operation,
- Versorgung von NOT NULL-Spalten, die von außen nicht geliefert werden können, z.B. Primary-Key-Spalten beim Insert,
- Abbildung von Foreign-Key-Beziehungen,
- Abbildung der Selektionskriterien für Update- und Delete-Statements

im Metadatenmodell hinterlegt werden.

Zusätzlich werden Informationen zur Sicherung der Datenkonsistenz benötigt:

- semantische Prüfung, z.B. gültiger Wertebereich für XML-Komponenten
- Definition von Defaultwerten
- Definition optionaler Spalten der Datenbanktabelle als Pflichtfelder
- Abhängigkeiten von XML-Komponenten untereinander, z.B. muss bei natürlichen Personen ein Geburtsdatum angegeben werden

Diese Regeln und Validierungsvorschriften sollen sowohl auf dienst- als auch anwendungsspezifischer Ebene definiert werden können.

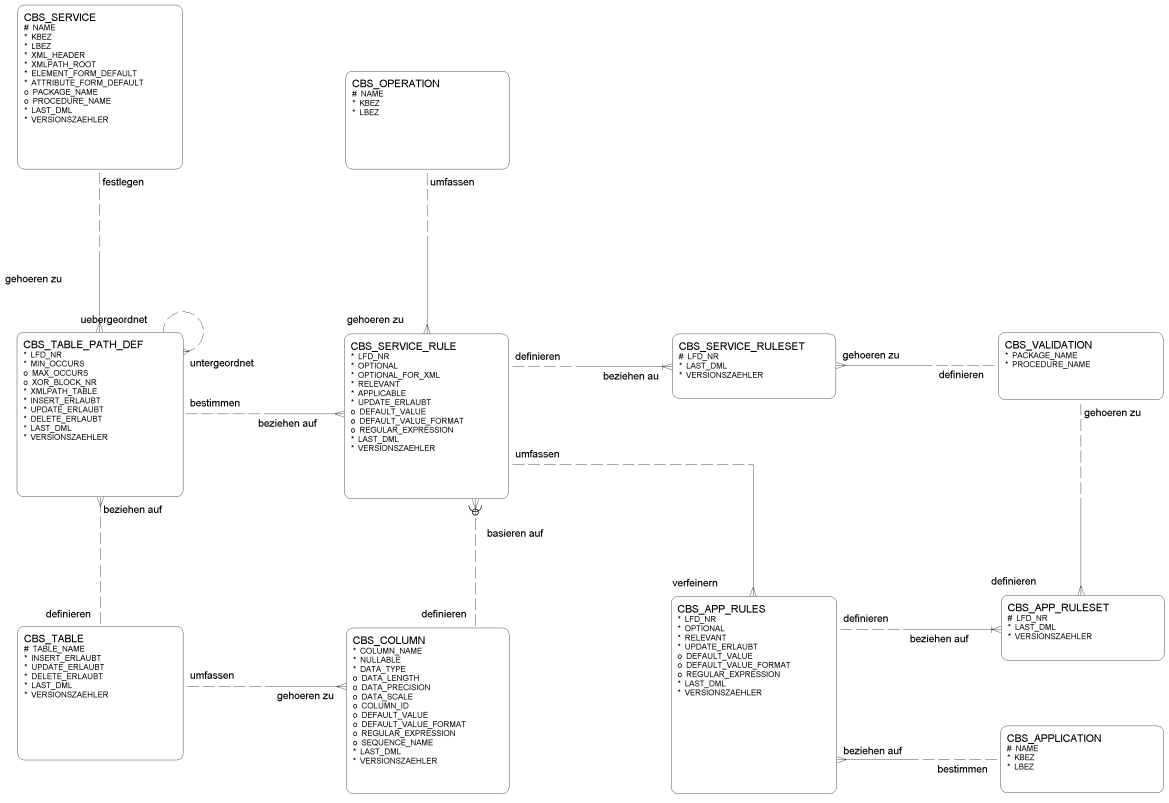

Abb. 3: Auszug Metadatenmodell Regelwerk

Entwicklungsstufe 3
In dieser Entwicklungsstufe war das Ziel, eine Unabhängigkeit zwischen XML- und DB-Struktur zu erreichen.
Beispiel:
Im XML werden Partnerdaten geliefert, wobei das Geburtsgeburtsdatum auf Ebene der natürlichen Personen definiert ist. Im relationalen Datenmodell soll das Geburtsdatum in der Tabelle PA, also auf der Ebene der allgemeinen Partnerdaten gespeichert werden.
Durch Einführung des sog. Virtual-Columns-Konzepts, können Abweichungen zwischen XML- und DB-Struktur überbrückt werden.

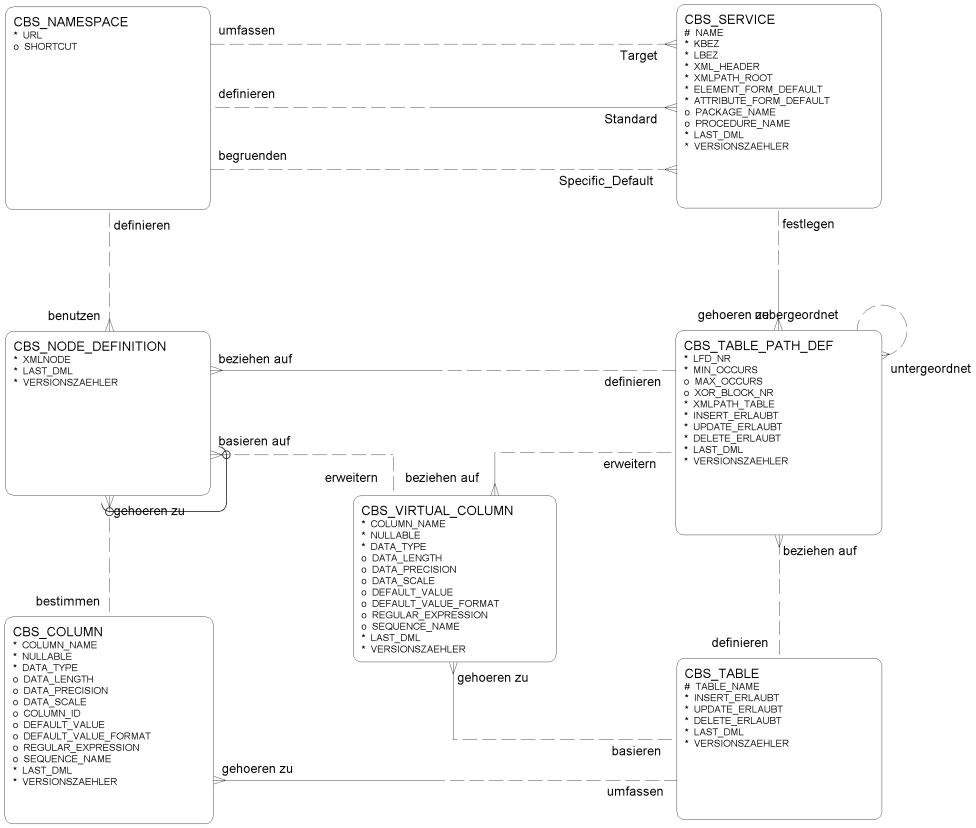

Abb. 2: Auszug Metadatenmodell XML-DB-Mapping Version 2

In der Tabelle CBS_VIRTUAL_COLUMN des Metadatenmodells wird für den Dienst der Tabelle NAT_PERS die virtuelle Spalte GEBURTSDATUM hinzugefügt.
Für diese virtuelle Spalte wird eine Regel in Form einer PL/SQL-Prozedur angegeben, dass der Wert dieses XML-Elements in das XML-Element der Spalte GEB_GR_DAT der Tabelle PA übernommen werden soll.

Umgang mit Strukturänderungen
An die Tabelle PA wird eine neue Spalte DEMO_PID angehängt.

```
ALTER TABLE pa
ADD (DEMO_PID                varchar2 (10));
```

Nach der Strukturänderung liefert der Aufruf eines Dienstes, dem die Tabelle PA zugeordnet ist, folgende Fehlermeldung:

```
interne technische Fehler:
Für folgende Tabellen bestehen strukturelle Inkonsistenzen zwischen dem
Metadaten-Modell und den aktuellen DB-Tabellen:  PA
```

Um den Dienst wieder lauffähig zu machen, muss eine Synchronisation zwischen Data-Dictionary und der Metadatentabelle CBS_COLUMN erfolgen.
Vorschläge für die Synchronisation werden über die PL/SQL-Prozedur

```
set serveroutput on size 1000000
exec cbs_metadaten_manager.metadaten_synchronisation
set serveroutput off
```

generiert:

```
DECLARE
  v_rec_all_tab_columns           all_tab_columns%ROWTYPE;
BEGIN
-------------------------------
-- Die Spalte DEMO_PID der Tabelle PA wurde neu definiert.
-- Bitte führen Sie folgende Kommandos aus:
    SELECT dir.*
    INTO   v_rec_all_tab_columns
    FROM   all_tab_columns    dir
    WHERE  dir.table_name  = 'PA'
    AND    dir.column_name = 'DEMO_PID';
    --
    cbs_metadaten_manager.merge_cbs_columns
         (p_rec_col            => v_rec_all_tab_columns
         ,p_list_index_columns => ', PA_ID, '
         );
-- Zu beachten: evtl. sind noch spezielle Validierungsregeln zu aktivieren!
--
--
  cbs_metadaten_manager.sync_cbs_node_definitions;
  cbs_metadaten_manager.create_metadaten_cursor;
END;
/
```

Projekterfahrungen

- Umstellung einer bestehenden Anwendung zur elektronischen Antragsstellung (Umstellungsaufwand 2 MT ohne Integrationstests)
  - Befüllung des Metadatenmodells
  - Erzeugung der Validierungsroutinen für die XML-Komponenten
- Einbindung Steuernummer als neue XML-Komponente (Umstellungsaufwand 0,5 MT)
  - Tabellenerweiterung im Zieldatenbankschema
  - Aktualisierung des Metadatenmodells
  - Erzeugung der Validierungsroutinen für die neue XML-Komponente
- DDL-Änderungen, z.B. Name einer Tabellenspalte (Umstellungsaufwand < 0,5 MT)
  - DB-Änderung im Zieldatenbankschema
  - Aktualisierung des Metadatenmodells

Kontaktadresse:

Bernhard Eichhorn
MIC Management Consulting GmbH
Kronacher Straße 4
D-81549 München

Telefon: +49 (89) 680 711 61
Fax: +49 (89) 680 711 61
E-Mail bernhard.eichhorn@mic-muenchen.de
Internet: www.mic-muenchen.de