KRP Katalog-Applikation – alles aus einem Guss

Dr. Götz Gleitsmann, ORBIT Gesellschaft für Applikations- und Informationssysteme mbH

Das System „KRP Katalog-Applikation“ dient der Modellierung von Katalogen eines großen IT-Dienstleisters und ihrer Speicherung in einer relationalen Datenbank. Dieser Artikel beschreibt diese wichtige Komponente der Konfigurations- und Rechnerplattform (KRP) dieses Dienstleisters.

Das Projekt ermöglicht die zentrale Speicherung der in XML modellierten Kataloge in einer Datenbank. Hierdurch ergibt sich eine signifikante Beschleunigung der Katalog-Nutzung. Bis auf eine Java-Komponente erfolgte die Implementierung ausschließlich in Oracle. Dies umfasst neben der Datenhaltung auch zahlreiche Schnittstellen zu weiteren Systemen. Der Verfasser ist für die ORBIT Gesellschaft für Applikations- und Informationssysteme mbH in Bonn tätig und hat langjährige Erfahrung in Oracle BI und SQL-Entwicklung.

Hauptziel des Projekts ist der Aufbau der Applikation „KRP Katalog-Applikation“, mit der Kataloge des Kunden im XML-Format modelliert sowie in einer relationalen Datenbank gespeichert und qualitätsgesichert werden. Letzteres erfordert als Basis eine Übersetzung des Common Product Data Model (CPDM-XSD) in ein relationales Datenmodell. Weiterhin müssen Exporte in zwei verschiedene XML-Formate möglich sein. Die Katalog-Arten sind:

- Offering Catalog (Produkt-Katalog)
- Corporate Customer Catalog (besteht aus mehreren Produkt-Katalogen); es gibt genau einen produktiven Produkt-Katalog
- Customer Catalog (spezifische Kunden-Kataloge)

Die Kataloge enthalten Informationen zu Produkten wie Server-Anbindung, Rechenzentrums-Anbindungen, Cloud Services, IT-Infrastruktur und mobilen Netzleistungen. Dabei wird der gesamte Lebenszyklus eines Katalogs berücksichtigt. Für jeden Entwicklungsstand gibt es einen Katalogstatus, im Einzelnen sind dies:

- *LOADED*
  Nach dem Hochladen des XML-Katalogs
- *IMPORTED*
  Nach der relationalen Verteilung des Katalogs in der Datenbank
- *EDIT*
  Katalog ist editierbar
- *TO BE RELEASED*
  Katalog ist zur Qualitätsüberprüfung freigegeben und nicht mehr editierbar
- *IN RELEASE*
  Qualitätsüberprüfung ist erfolgt und Katalog steht bereit zur Freigabe für die Produktion
- *IN PRODUCTION*
  Katalog ist produktiv
- *REJECTED*
  Kataloge, die als gelöscht markiert sind, aber nie produktiv waren
- *RETIRED*
  Als gelöscht markiert, vormals produktiv
- *ARCHIVED*
  Katalog ist archiviert

Sowohl beim Hochladen des Katalogs als auch im Nachhinein bei den relational verteilten Katalogen findet eine Überprüfung statt. Es gibt vier Überprüfungsstatus, nämlich „MINOR“, „MEDIUM“, „MAJOR“ und „FATAL“. Auch bei einem Statuswechsel findet eine automatische Überprüfung statt. Damit ein Katalog für die Produktion freigegeben werden kann, muss sein Prüfstatus „MINOR“ sein. Anderes Beispiel: eine XML-Schemaverletzung ergäbe den Prüfstatus „FATAL“, was keine relationale Verteilung des Katalogs, also Speicherung im Datenbank-Schema zuließe.

Neben der Generierung von Katalogen aus Portfolio-Daten können auch neue Kataloge unter Beachtung von Modellierungsregeln erstellt und anschließend beliebig bearbeitet werden. Dabei werden die typischen IMAC-Geschäftsvorfälle (IMAC = „Install Move Add Change“) unterstützt. Ebenso kann es sich um kostenpflichtige oder kostenfreie Vorfälle handeln. In *Abbildung 1* ist der grobe Ablauf der Bearbeitung eines Katalogs dargestellt. Die ORBIT Gesellschaft für Applikations- und Informationssysteme mbH ist für die Komponente „Katalog-Applikation“ zuständig. Neben der Konzeption inklusive der Schnittstellen wurde auch die Implementierung durchgeführt.

Architektur

Abbildung 2 zeigt die KRP-Gesamt-Architektur. Man sieht, dass die Komponente „KRP Katalog-Applikation“ in folgende Prozesse eingebunden ist:

- Request to Offer/Contract (RtOC)
- Order to Cash (OtC)
- Product-Lifecycle-Management-Prozess (PLM)

Die Komponente „KRP Katalog-Applikation“ hat Schnittstellen zu folgenden Komponenten:

- KRP One Portfolio Navigator zum Auslesen von Portfolio-Daten, die zur Generierung der Rohkataloge verwendet

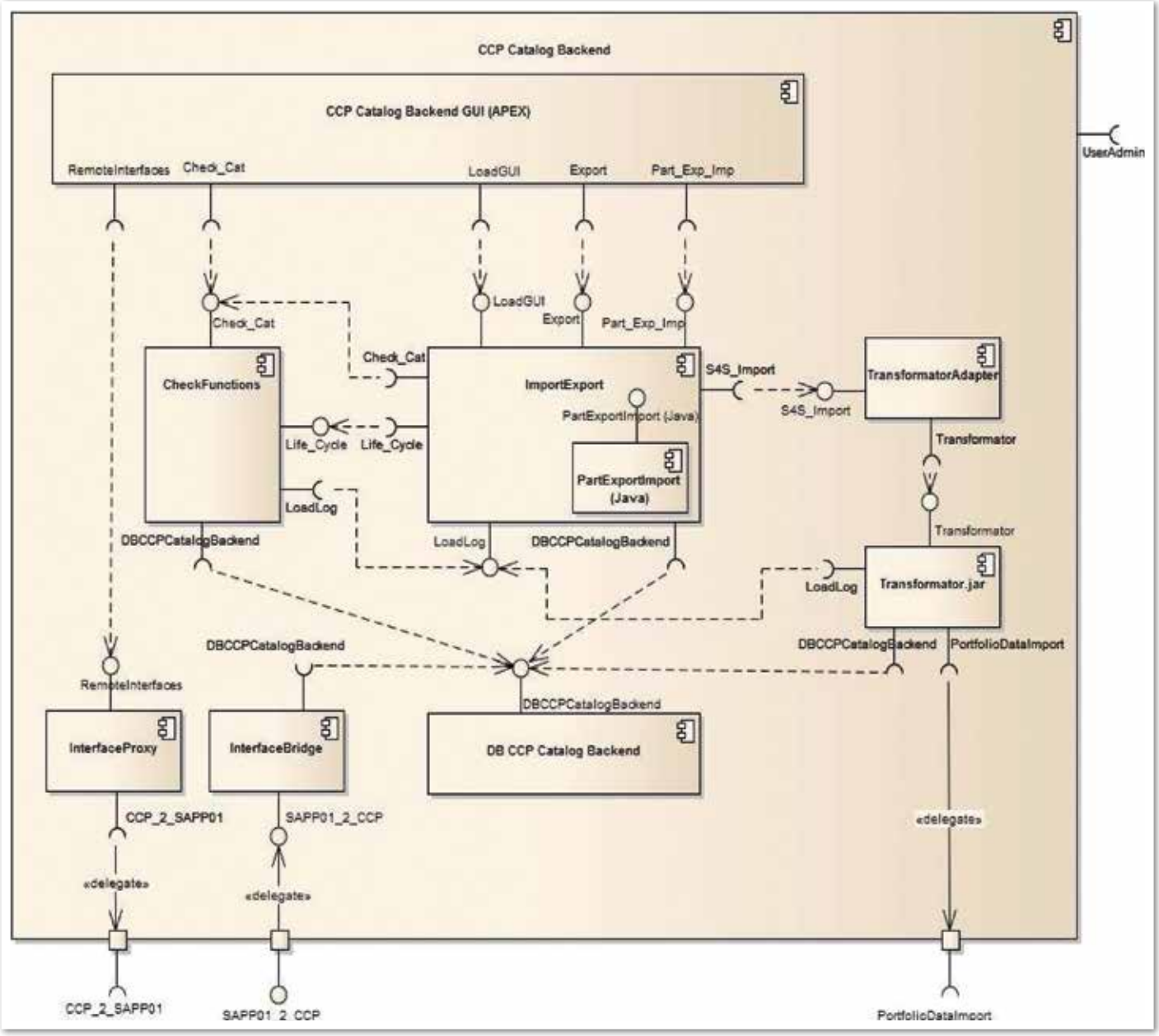

Abbildung 1: Katalog-Modellierungsprozess – prinzipieller Verlauf der Katalog-Bearbeitung

werden. Der Transformator der KRP Katalog-Applikation greift zum Auslesen der Portfolio-Daten auf eine Tabelle von KRP One Portfolio Navigator zu. Er generiert CPDM-XML aus relationalen Daten sowohl der Portfolio- als auch der KRP-Datenbank und kann CPDM-XML in zwei weitere XML-Formate umrechnen.

- SAP P01 (Abnahme) im Product-Lifecycle-Management-Prozess zur Anlage von Materialien für den Verbund- und Fachtest
- SAP P01 (Produktion) im Product-Lifecycle-Management-Prozess zur Anlage von Materialien für die Produktion (produktive Prozessierung der Prozesse „Request to Offer/Contract" (RtOC) und „Order to Cash" (OtC)
- SAP P02 im Product-Lifecycle-Management-Prozess zur Anlage von Materialien für den Verbund- und Fachtest sowie die Produktion (produktive Prozessierung der Prozesse „Request to Offer/Contract" (RtOC) und „Order to Cash" (OtC)
- SALSA im Product-Lifecycle-Management-Prozess zur Anlage von Materialien für den Verbund- und Fachtest sowie die Produktion (produktive Prozessierung der Prozesse „Request to Offer/Contract" (RtOC) und „Order to Cash" (OtC)
- myMDS (Kunden-Kataloge) durch Export von XML-Katalogen in ein anderes Format
- KRP Solution Design durch Export von XML-Katalogen im CPDM-Format

Beschreibung des technischen Ansatzes

Die Implementierung erfolgte nahezu ausschließlich mit Oracle-Komponenten. Einzige Ausnahmen sind die Komponente „Transfor-

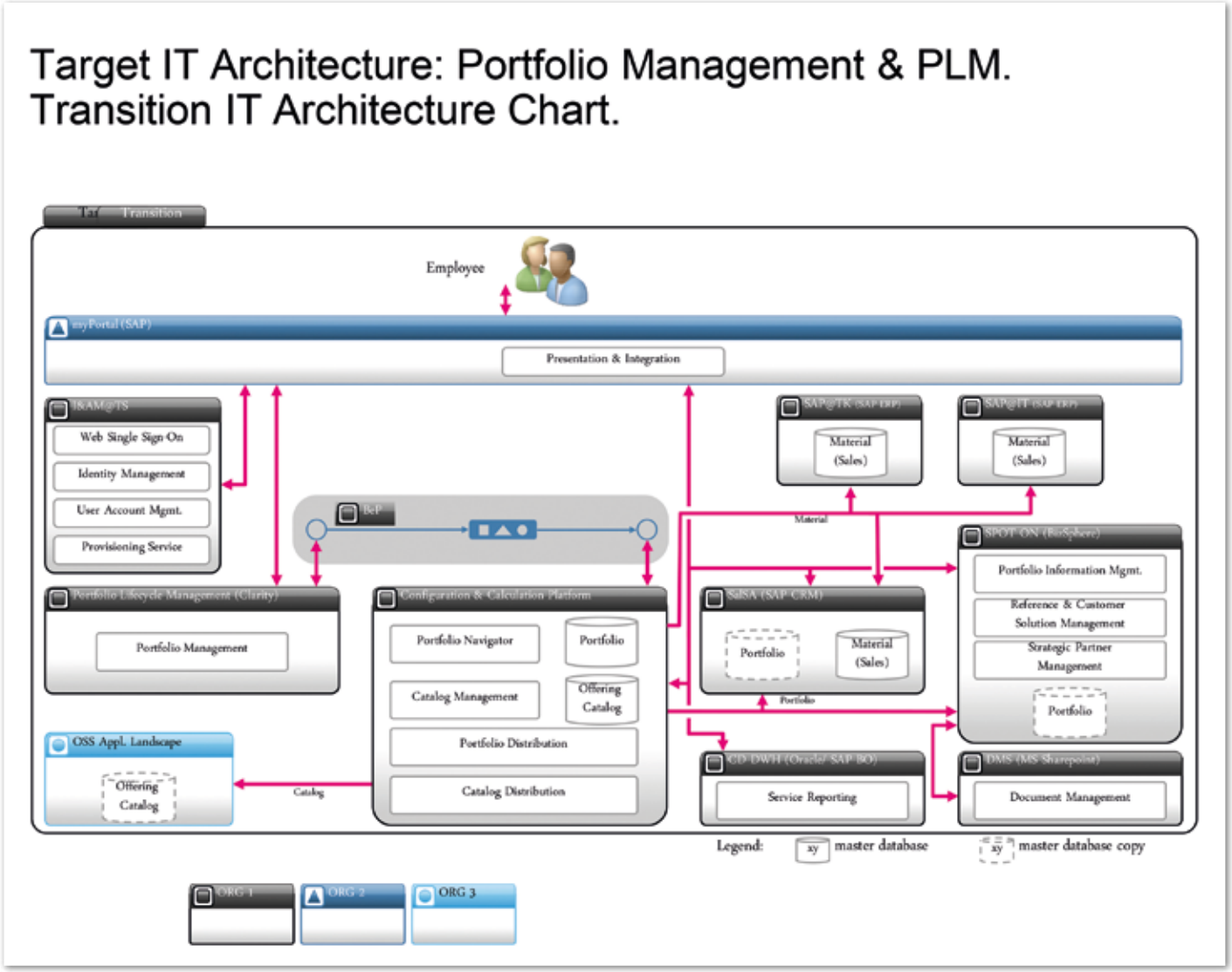

Abbildung 2: Gesamt-Architektur und Anbindung von KRP an Nachbarsysteme

mator.jar", die mit Java programmiert wurde, und die Applikation „Cockpit". Wie aus dem Komponenten-Modell *(siehe Abbildung 3)* zu erkennen ist, enthält der Transformator eine weitere Komponente namens „Transformator Adapter", die im Wesentlichen als Connector zu einem benachbarten, nicht eingezeichneten System dient.

Insgesamt kamen zahlreiche Oracle-Produkte und -Komponenten zum Einsatz. Die Datenbank entspricht der Version 11*g* (Enterprise Edition). Allein das Katalog-Schema enthält insgesamt 261 Tabellen, 69 PL/SQL-Packages, vier integrierte Java-Packages, zwei gespeicherte Funktionen und acht Views. Das Laden der XML-Kataloge erfolgt entweder durch Hochladen einer Datei oder durch Abholen eines Katalogs aus einer Portfolio-Datenbank (OPNav) mithilfe des Transformators.

Zunächst überprüft ein Datenbank-Package das XML auf Wohlgeformtheit und direkt im Anschluss gegen ein XML-Schema (XSD), in diesem Fall CPDM-XML. Im weiteren Verlauf werden rund dreißig konfigurierbare Prüf-Funktionen (Convention Rules, Char Rules, Überprüfung auf veraltete XML-Syntax etc.) ausgeführt.

Für den Import der XML-Daten war zunächst der Oracle Warehouse Builder (OWB 11*g*) vorgesehen. Aufgrund von Präferenzen in der Gesamt-Architektur wurde jedoch schließlich auf manuelles Coding über XPath und SQL-Inserts gesetzt.

Die Komponente SOAP wurde verwendet, um mit zwei SAP-Systemen zu kommunizieren. Die für die verschlüsselte Kommunikation nötigen Sicherheitszertifikate sind in der Komponente Oracle Wallet hinterlegt. Die Kommunikation mit den beiden SAP-Systemen geschieht asynchron über Webdienste in verschlüsselter Form (SSL, HTTPS), wobei im Hinblick auf maximale Datensicherheit die Antwort der beiden SAP-Systeme nicht an das anfragende Datenbank-Schema, sondern an zwei gesonderte, den SAP-Systemen zugeordnete Schemata geht. Erst von dort aus erfolgt eine Weiterleitung. Weitere Operationen finden auf diesem Weg nicht statt.

Die GUI basiert auf Oracle Application Express (Apex 4.2). Die verschiedenen Stadien des Produkt-Lifecycle werden durch spezialisierte Apex- und JSP-Applikationen unterstützt, die in einem gemeinsamen Portal mit SSO-Infrastruktur zusammengefasst sind. Zusätzlich gibt es eine Funktionalität zur dynamischen Definition von „csv"-Dateien, die importiert werden können. Sie können zu Import-Gruppen zusammengefasst

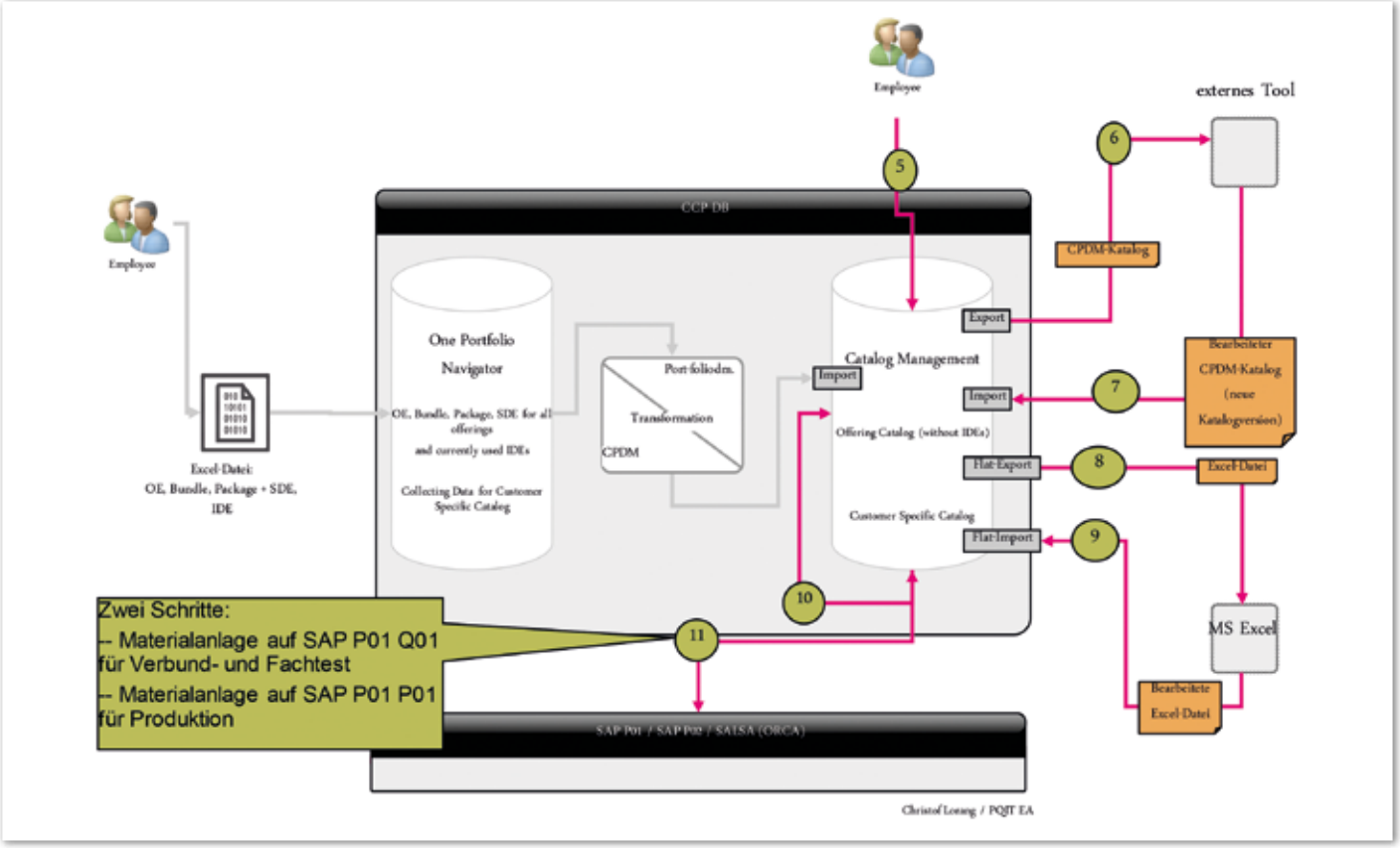

Abbildung 3: Die Komponenten des KRP-Systems. Man beachte die abweichende Ersetzung KRP = CCP

werden, sodass dem Benutzer eine Vorauswahl zusammengehörender Dateien vorgegeben wird. Abschließend sei erwähnt, dass auch ein Excel-Import und -Export auf breiter Basis möglich ist.

Herausforderungen

Im Verlauf der Implementierung ergaben sich zahlreiche Herausforderungen. So stellte sich OWB überraschend als Hürde heraus, weil bei der Überführung der in CPDM-XSD gespeicherten Katalogdaten in die Tabellenstruktur der Datenbank folgende Probleme mit dem XML-Format auftraten:

- Durch zyklische Referenzen im XML-Dokument entstanden Endlosschleifen bei der Analyse und der relationalen Verteilung
- Kein automatisiertes Einlesen der XSD-Datei, weil das Vererben bei komplexen Datentypen nicht möglich ist

Offensichtlich kann OWB nur sehr einfache Strukturen sicher erkennen, was letztlich zu der genannten Ersetzung des OWB durch manuelles Coding führte. Die Komplexität des XML-Schemas führte zu Engpässen im Hauptspeicher der Datenbank. Dies wird hauptsächlich durch den DOM-Parser verursacht, der beim Einlesen großer XML-Dokumente (größer 180 MB) eine sehr große Anzahl von Objekten erzeugt. Implementierungsschwächen des Parsers führten zur Nicht-Freigabe nicht mehr benötigter Speicherbereiche und damit zu betrieblichen Problemen.

Für die Kommunikation mit den SAP-Systemen zur Anlage von Materialnummern und Preisen waren aufgrund der Sicherheitsanforderungen zahlreiche Firewalls und SAP-Web-Dispatcher zu berücksichtigen. Dies erforderte, an vielen Stellen Ports und Zertifikate zu hinterlegen sowie Konfigurationen der Proxys, Router, Firewalls und Dispatcher durchzuführen.

Zur manuellen Bearbeitung der Daten gibt es abgesehen von der GUI die Möglichkeit, die Daten im Excel-Format zu exportieren, manuell zu verarbeiten und wieder zu importieren. Der Transport der verschlüsselten Passwörter zwischen verschiedenen Applikationen erfolgt so, dass sie auch für Systemadministratoren unsichtbar bleiben.

Fazit und Ausblick

Die Bearbeitung großer XML-Dateien ist immer sehr langsam. Durch die hier beschriebene Implementierung konnte selbst mit den umfangreichen automatischen Prozessen eine sehr gute Performance erreicht werden. Das vorliegende System kann an beliebige Systeme angekoppelt werden, da es sowohl relationale als auch im XML-Format gespeicherte Daten anbieten beziehungsweise verarbeiten kann.

Dr. Götz Gleitsmann
goetz.gleitsmann@orbit.de