Groovy Way of Self-Service BI mit Oracle Essbase

Holger Huck
Trivadis GmbH
Stuttgart

Schlüsselworte
Oracle Essbase, Hyperion Essbase, Groovy, Self-Service BI, Prozessautomatisierung, Essbase Java API, Essbase API, Essbase Client

Einleitung

Oracle Essbase ist die führende multidimensionale OLAP Datenbank. Sie verfügt über ein eigenes Datenbankformat und bietet viele Anwendungsmöglichkeiten für Finanz- und Controlling Bereiche. Zum einen ist die Essbase Datenbank in den verschiedenen EPM Produkten von Oracle integriert, kann aber auch als Standalone Server installiert werden. Diese Form ist besonders gut geeignet für den Einsatz in strategischen Planungsprozessen, für Simulationen und Ad-hoc Analysen. Solche Anwendungen haben in der Regel einen kleinen Benutzerkreis und dienen zur Generierung von neuen Daten und Erkenntnissen. Neue Daten werden aus einer Mischung aus bestehenden Daten (z.B. IST-Werte, OP Werte) und Daten aus dem strategischen Zeitraum, z.B. durch Fortschreibungen auf der Basis von Prämissen in der Essbase Datenbank (z.B. Teuerungsraten, Konjunkturdaten) erzeugt. Diese Systeme benötigen keine aufwendigen Userinterfaces. Die Interaktion zwischen dem Benutzer und der Datenbank erfolgt über das SmartView Add-in für Excel. Einzig die Prozesssteuerung für das Laden, Kopieren und Berechnen der Datenbank muss dem Endanwender zugänglich gemacht werden. Out-of-the-Box sind bei einer Essbase Standalone Installation keine Workflow- und Steuerungskomponenten für den Endbenutzer enthalten. Mit der Java API für Essbase, einer relationalen Datenbank und der Skriptsprache Groovy kann man diese Lücke schließen. In einem Framework aus Groovy Skripten lassen sich alle Prozesse, wie z.B. Kalkulationen, Dimensionen und Daten laden starten und kontrollieren. Durch die Verbindung mit einer relationalen Datenbank können Workflows für die Standardprozesse schnell hinterlegt werden. Verbunden mit einem webfähigen Frontend, wie z.B. Oracle APEX wird die Steuerung dem Business Anwender zu Verfügung gestellt. Hiermit werden die Essbase Administratoren entlastet und die Fachabteilung kann unabhängig vom IT-Support arbeiten. Verwendet man dazu noch freie Komponenten wie ORACLE Database EXPRESS Edition und APEX, fallen keine zusätzlichen Lizenzkosten an.

Mit Groovy Scripten kann die Essbase Automatisierung schnell und einfach erfolgen. Viele Essbase Entwickler haben einen betriebswirtschaftlichen Hintergrund. Die Zielsetzung ist deshalb, eine Lösung zu entwickeln, die mit wenig Aufwand und ohne spezielle Programmierkenntnisse gepflegt und weiterentwickelt werden kann. Folgender Artikel stellt einen Ansatz, wie dies durch die Kombination von relationalen Steuerungsdaten, Groovy und der Essbase API realisiert werden kann.

Anforderungen an das Controlling

Controlling Prozesse sind sehr komplex und vielschichtig geworden. Der Trend geht zu immer mehr Informationen und einem stetig wachsendem Detailierungsgrad bei zunehmender Verkürzung der Planungsdauer. Diese Anforderungen können nur erfüllt werden, wenn Controlling Systeme in der Lage sind den Planungsprozess weitgehend zu unterstützen und zu automatisieren.
Das wird erreicht, indem komplexe Berechnungen in das System verlagert werden, Zukunftsdaten auf der Basis von Prämissen erzeugt und Fortschreibungen und TOP-Down Planungen auf der Basis von Erkenntnissen der Vergangenheit berechnet werden. Um auf alle möglichen Ereignisse vorbereitet zu sein, werden verschiedenen Szenarien berechnet, die sich auf unterschiedlichen Prämissen beziehen oder sich aus mehreren Szenarien zusammensetzten.
Mit zunehmendem Jahresfortschritt kommen in der Regel neue Erkenntnisse dazu. So kann eine Planungen mit aktuellen Daten aus den Monatsberichten oder einer Ist-Erwartung ergänzt werden. Diese Vorbelegungen mit IST Werten und die Generierung eines neuen Datenstandes, sowie der Import von Daten aus unterschiedlichen Datenquellen muss vom System unterstützt werden.

Betrachtet man den Prozess einer strategischen oder operativen Planung, so gibt es auch hier Zeitpläne. Diese sind aber nicht mit dem Ablauf einer Regelberichtserstattung, wie z.B. der automatisierten Erstellung eines Monatsberichts zum Monatsanfang zu vergleichen. Datenbankprozesse müssen interaktiv aus dem Controlling gestartet werden können. Die Ergebnisse müssen umgehend zur Analyse und im Reporting zur Verfügung stehen.

Prozessbeispiel

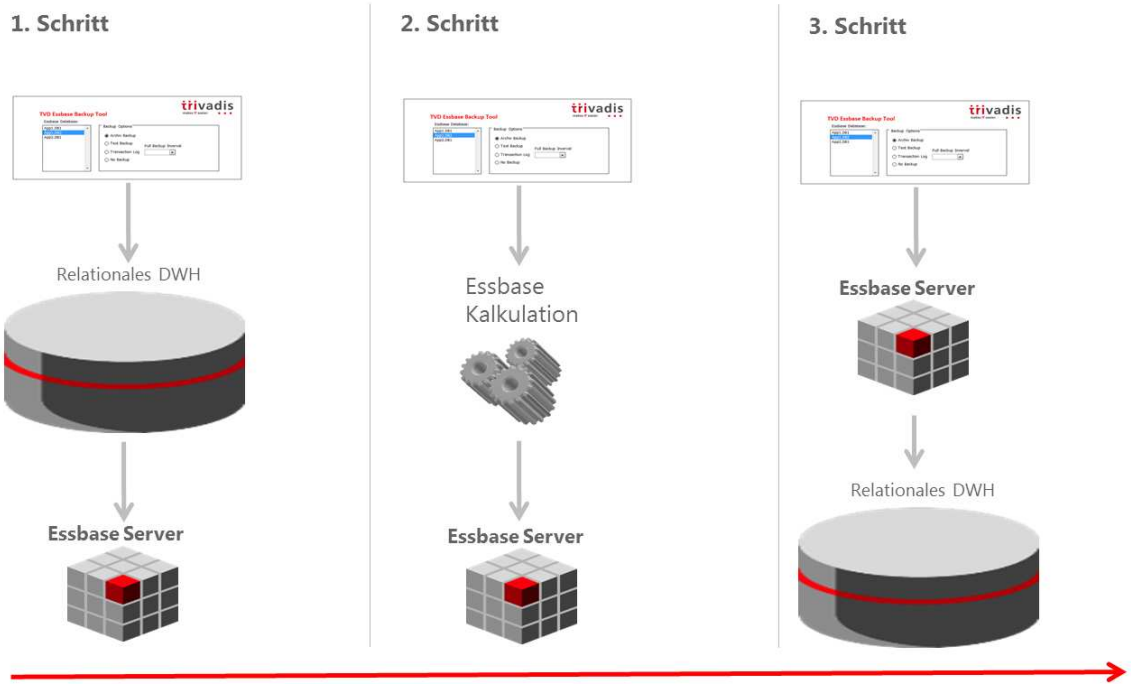


Abb. 1: Prozessbeispiel

Abbildung 1 zeigt eine einfache Prozesskette in drei Schritten.

Schritt 1:
Aus einem relationalen Data Warehouse werden IST Werte für die vergangenen Jahre, sowie Planungsprämissen in den Essbase Würfel geladen. Bevor der Schritt 2 durchgeführt wird, können zusätzlich Daten oder Korrekturen von den Anwendern in den Essbase Würfel geladen werden.

Schritt 2:
Auf dem Essbase Würfel wird ein Datenbank Script gestartet, das die Istwerte anhand der Prämissen in die Zukunft fortschreibt und die Datenbank aggregiert.

Schritt 3:
Der neu erstellte Datenstand wird in das Data Warehouse exportiert.

Folgende Prämissen für die Umsetzung einer Prozesssteuerung sollen berücksichtigt werden:

- Fachanwender kann die Prozesse ohne IT spezifisches Wissen starten
- Steuerung soll über eine Administrationsmaske, möglichst im Web erfolgen
- Planungsart, Szenarien sollen über Auswahlboxen festgelegt werden
- Neue Prozessschritte sollen schnell hinzugefügt werden können

Lösungsansatz und Entscheidungsmatrix

Zur Durchführung der Prozessschritte müssen diese wiederum in Einzelschritte in Essbase und der relationalen Datenbank aufgelöst werden. Somit ist eine Technologie zu wählen, die mit beiden Datenbank Systemen kommunizieren kann.
Die Benutzerinteraktion soll über eine Web Maske erfolgen. Ziel hierbei ist, so wenig Logik wie möglich in der Maske zu implementieren, um das Frontend leicht austauschen zu können.
Um schnell neue Prozessschritte zu erstellen, sollen die Prozessketten in einer relationalen Datenbank gespeichert werden. Diese werden dann ausgelesen und abgearbeitet.

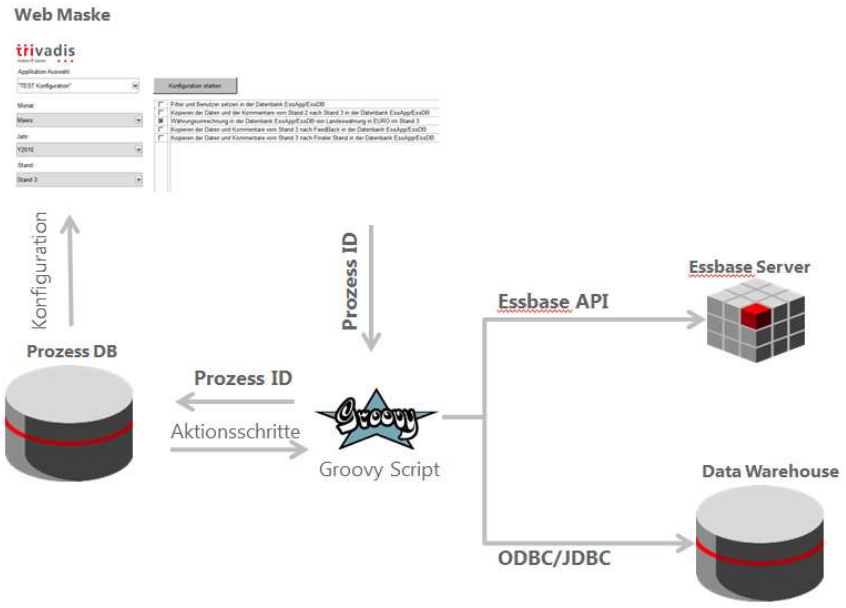


Abb. 2: Lösungsarchitektur

Abbildung 2 zeigt die angestrebte Lösung. Die Prozessketten und Konfigurationsinformationen werden in einer relationalen Datenbank abgelegt. Auf diese Informationen wird aus der Web Maske zugegriffen. Der Fachbereichsanwender wählt aus, welchen Prozess er starten will. Es erfolgt eine Aufruf des Groovy Script mit einer Parameterliste.
Das Groovy Script liest die Prozessschritte aus der Prozessdatenbank und arbeitet diese dann ab. Hierbei erfolgen sowohl Zugriffe auf den Essbase Server, als auch auf die Data Warehouse Datenbank.

Welche der drei Essbase API's verwendet werden soll wurde anhand der Entscheidungsmatrix in Abb. 3 getroffen. Die Java API hat alle Kriterien erfüllt.

	Essbase API's		
Kriterien	**Java**	**Visual Basic**	**C/C++**
Aktuelle Technologie	X	-	X
Leicht zu erlernen	X	X	-
Wird von Oracle auch zukünftig weiterentwickelt	X	-	X
OS und Hardware unabhängig	X	-	-

Abb. 3: Entscheidungsmatrix API Auswahl

	Implementierung	
Kriterien	**Groovy**	**Java Binary**
Programmcode ist schnell einsehbar	X	-
Änderungen können direkt auf dem Server durchgeführt werden	X	-
Keine aufwendige Entwicklungsumgebung notwendig	X	X
Minimaler Installationsaufwand	X	-
Keine speziellen Software Development Kenntnisse notwendig	X	-

Abb. 4: Auswahl Implementierung

Abbildung 4 zeigt die Entscheidungsmatrix für die Implementierung. Ziel war, den Aufwand für die Entwicklung möglichst klein zu halten um schnell Änderungen und Erweiterungen durchführen zu können. Groovy ermöglicht einen schnellen Start mit einfachen Mitteln und war aus diesem Grund die erste Wahl.

Implementierung

Die Prozessketten werden in der relationalen Datenbank hinterlegt. Hierbei ist der kleinste Nenner eine „Aktion". Diese kann entweder auf den Essbase Server oder auf eine relationale Datenbank ausgeführt werden. Folgende Tabelle zeigt ein Auszug aus möglichen Aktionen.

Aktion	**Berschreibung**
Calc	Essbase Kalkulationsskript starten
setVar	Essbase Substitutionsvariable setzen
delVar	Essbase Substitutionsvariable löschen
importwithRuleFile	Essbase Import mit Regeldatei
exportReport	Essbase Report ausführen
importReport	Essbase Report Daten importieren
execStoredProcedure	StoredProcedure auf RDBMS ausführen
execSQL	SQL Befehl auf RDBMS ausführen

Abb. 5: Aktionstabelle

Die nächste Stufe ist eine Aktionsliste. Hier werden alle Aktionen hinterlegt, die eine Aufgabe bilden. Es wird die Reihenfolge festgelegt und auf welcher Datenbank die Aktion ausgeführt wird. Eine Aktionsliste wird einer Aufgabe zugeordnet. Die Aufgabenliste enthält alle Aufgaben mit Informationen zu welcher Applikation und Datenbank sie zugeordnet sind und Einstellungen zur Gültigkeit.
Auf die Aufgabenliste greift die Web Maske zu, um die Auswahlboxen zu füllen.

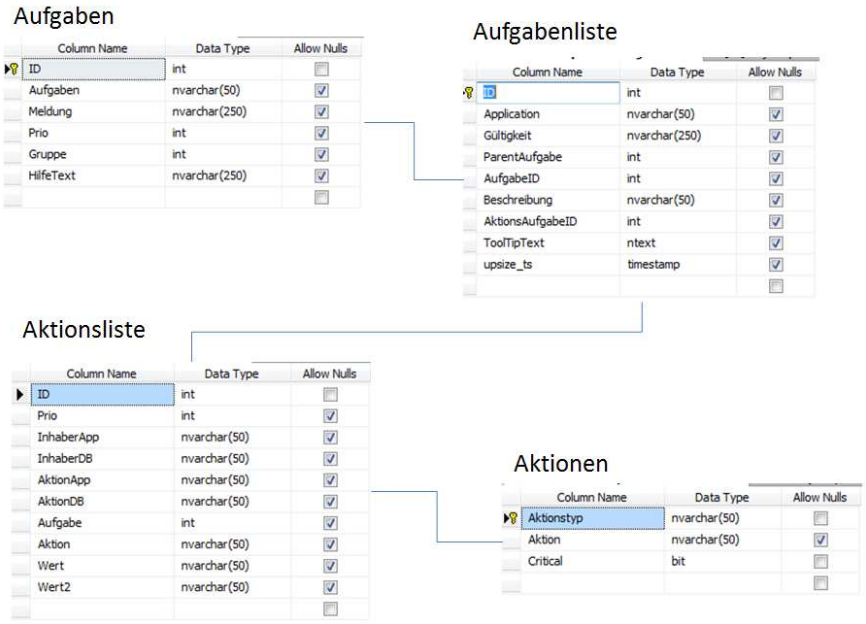
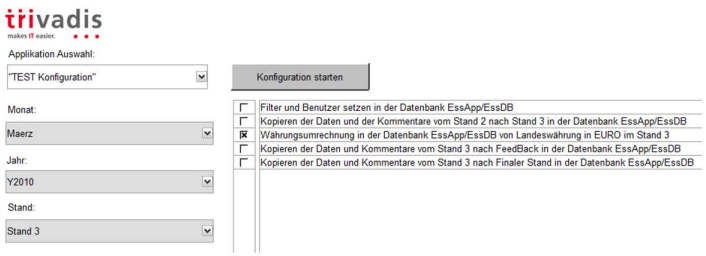


Abb. 6: Datenmodell Prozessketten

Auf Basis der Konfigurationsinformationen wird das Groovy Skript aufgerufen. Die notwendigen Parameter werden als Konfigurationspaar übergeben, d.h. ein Parameter besteht aus zwei Teilen, der Name des Parameters und der Inhalt.



"Januar;mbrMonat" "Y2012;mbrJahr" "Stand 3;mbrAktStd" „essAPP;AppName" „essDB;DBName" "Stand 3 End;mbrStndEnd" "Stand 3 End (LW);mbrStndEndLW" "Stand 6;mbrFinStandOP" "Stand 2;mbrQuell" "Stand 3 Dur;mbrStndDur" "Stand 3 Dur (LW);mbrStndDurLW" "SEDCB016;Server" "admin;EssbaseUser" "password;EssbasePWD" "user;SQLUSER" „password;SQLPWD" „TEST;KONFNAME" "167;TASK"

Abb. 7: Maskenauswahl und resultierende Parameterliste

Das Groovy Script speichert die Parameter und kann diese bei Bedarf abfragen, z.B. um die Variablen für eine „setVar“ Aktion oder abhängige Variablen zu ermitteln.

Für den Einsatz der Essbase API in Groovy muss der Essbase Client mit der Java API auf dem Rechner installiert und der CLASSPATH gesetzt sein.
Die Implementierung in den Groovy Skripten funktioniert analog wie in Java.
Folgendes Beispiel zeigt das Erstellen einer Verbindung zu einem Essbase Server.

```
import com.essbase.api.base.*;
import com.essbase.api.session.*;
import com.essbase.api.datasource.IEssCube;
import com.essbase.api.datasource.IEssOlapFileObject;
import com.essbase.api.datasource.IEssOlapServer;
import com.essbase.api.datasource.IEssOlapUser;
import com.essbase.api.domain.*;


int FAILURE_CODE = 1;
IEssbase ess = null;
try {
    // Create JAPI instance.
    ess = IEssbase.Home.create(IEssbase.JAPI_VERSION);

   // Sign On to the Provider
   IEssDomain dom = ess.signOn(s_userName, s_password, false, null, s_provider);

   IEssOlapServer olapSvr = dom.getOlapServer(s_olapSvrName);
   olapSvr.connect();
   System.out.println("Connected to Analytic server '" +olapSvr.getName();

   String apiVersion = ess.getApiVersion();
   String apiVerDetail = ess.getApiVersionDetail();
   System.out.println("API Version :"+apiVersion);
   System.out.println("API Version Detail :"+apiVerDetail);

   IEssCube cube = olapSvr.getApplication(s_essAPP).getCube(s_essDB);
   cube.beginIncrementalBuildDim();



    } catch (EssException x) {
           System.err.println("Error: " + x.getMessage());
            statusCode = FAILURE_CODE;
        } finally {
            // Sign off.
            try {
                if (ess != null && ess.isSignedOn() == true)
                    ess.signOff();
                    System.out.println("Disconnected from " +olapSvr.getName());
            } catch (EssException x) {
                System.err.println("Error: " + x.getMessage());
            }
        }
```

Das gleiche gilt für Verbindungen zu relationalen Datenbanken. Der entsprechende ODBC/JDBC Treiber muss auf dem Rechner installiert sein und im CLASSPATH hinterlegt werden.

Durch Implementierung mit Groovy lassen sich die Prozessketten zur Freude der Administratoren auch direkte über einen Skriptaufruf starten. Groovy Skripte können schnell kopiert werden, um z.B. schnell auf Ad-hoc Anfragen und Sonderfälle zu reagieren. Der Programmcode ist schnell einsehbar.
Durch die Verbindung mit einer relationalen Datenbank lassen sich Prozessketten effektiv abbilden.

Ein weiterer Vorteil dieses Lösungsansatzes ist, dass Fehlermeldungen und die Interaktionen mit dem Planungssystem in der Prozessdatenbank dokumentiert werden können. Das kann bei der Fehlersuche und bei der Erfüllung von Audit Vorgaben sehr hilfreich sein.

Kontaktadresse:

Holger Huck
Trivadis GmbH
Industriestrasse 4
D-70565Stuttgart

Telefon: +49 (0) 711-90 36 32 30
Fax: +49 (0) 711-90 36 32 90
Mobil: +49 (0) 162 295 96 34
E-Mail: Holger.Huck@trivadis.de
Internet: www.trivadis.com