



Individuelle Datenbanklösungen in einer grafischen Umgebung

Synchronisation mit LIMBAS

Mit dem LIMBAS-Framework lassen sich ähnlich zu Access individuelle Datenbanklösungen in einer grafischen Umgebung umsetzen. Vielfach wird es in den Bereichen CRM, Prüfmittelmanagement, Betriebsmittelmanagement, Access-Alternativen bis hin zu ERP-Systemen eingesetzt. Dabei ist eine häufig gewünschte Funktionalität von Datenbankanwendungen die Verfügbarkeit verschiedener Arten der Synchronisation.

von Alexander Niklaus

Insbesondere im Enterprise-Umfeld darf ein Synchronisationsfeature nicht fehlen. Das LIMBAS-Framework bietet dazu zwei eigenständige Lösungsansätze, die neben den klassischen Datenexport- und -importfunktionen das Publizieren von Projekten sowie eine Datensynchronisation im Master-und-Slave-Verfahren bietet. Beide Funktionalitäten sind unabhängig voneinander, bilden aber zusammen ein wirkungsvolles Werkzeug, um ein Projekt auf mehreren LIMBAS-Instanzen gleichzeitig zu betreiben oder auch strukturierte Entwicklungsumgebungen zu erstellen.

Das LIMBAS Deployment

Das Deployment-Modul ist ein Werkzeug, das es erlaubt, eine in LIMBAS erstellte Lösung, eine Applikation, ein Fachverfahren oder ein Projekt auf eine oder mehrere Instanzen zu klonen bzw. zu veröffentlichen. Nehmen wir als Beispiel eine Mitarbeiterverwaltung, die mit LIMBAS umgesetzt worden ist. Diese Mitarbeiterverwaltung soll nun dezentral an mehrere Filialen weitergegeben werden. Jede Niederlassung betreibt ihre eigene LIMBAS-Instanz. Dazu muss die gesamte Datenbankstruktur wie Tabellen, Trigger oder Abfragen sowie Formulare, Berichte oder Erweiterungen mit dem Zielsystem abgeglichen und synchronisiert werden. Die

vorhandenen Datenbestände dürfen dabei nicht beeinflusst werden. Händisch wäre dieser Vorgang sehr mühsam und würde die Gefahr bergen, leicht Dinge zu übersehen. Das Deployment-Modul ermöglicht dagegen eine sichere und einfache Art der Synchronisation im Master-und-Slave-Verfahren, um eigenständige und unabhängige LIMBAS-Instanzen zu erstellen oder anzupassen. Ebenso ist es damit möglich, die eigentliche Projektentwicklung auf unterschiedlichen Entwicklungsinstanzen bis hin zum Produktivsystem zu verwalten.

Wie funktioniert nun dieses Deployment-Modul? Es wird ein kompletter Export aller Tabellenstrukturen, Systemtabellen sowie Skripterverweiterungen erstellt, in ein Archiv gepackt, zum Slave geschickt, dort ausgepackt und wieder importiert. Während dieses Vorgangs werden vorhandene Tabellenstrukturen abgeglichen und lokale Einstellungen, Nutzerdaten oder individuelle Anpassungen auf die Zielsysteme übertragen. Administrieren kann man die Synchronisation als Administrator im LIMBAS-Toolmenü. Dazu gibt es jeweils einen eigenen Funktionsreiter im Import- sowie im Exportmenü mit der Bezeichnung SYNC-EXPORT und SYNC-IMPORT.

Unter dem Reiter SYNC-EXPORT verbirgt sich die Administrationsoberfläche zum Erstellen und Exportieren eines Archivs. Dieses Archiv kann entweder als Archivdatei gespeichert oder direkt an ein Zielsystem gesendet und dort importiert werden. In diesem Fall beinhaltet die Exportfunktion auch die Importfunktion. Nicht immer möchte man das gesamte System synchronisieren, so bietet sich die Möglichkeit, die Module auszuwählen, die in den Synchronisationsprozess mit einbezogen werden sollen. Dazu gehören unter anderem Tabellen, Formulare, Berichte, aber auch Rechte, Systemtabellen oder Funktionen, die nach dem Import ausgeführt werden. Praktische Funktionen sind beispielsweise das Löschen der Sessions oder das Überprüfen der Indizes nach erfolgtem Import. Die Option „Alle Sessions löschen“ sollte man übrigens immer aktivieren, wenn man möchte, dass allen aktiven Usern die Änderungen sofort und nicht erst nach dem nächsten Anmelden zur Verfügung stehen (Abb. 1). Es ist zu beachten, dass ausgewählte Module in Zusammenhang stehen können. Möglicherweise existieren in Formularen Tabellenelemente, die in der Ursprungstabelle fehlen. In diesem Fall muss auch das Tabellenmodul mit exportiert werden, das dafür sorgt, dass fehlende Tabellenfelder oder gar ganze Tabellen angelegt werden.

Auch ist es möglich, eigene Funktionen oder Erweiterungen (Kasten: „LIMBAS-Erweiterungen“) zu schreiben, die nach dem Import ausgeführt werden. Dazu gibt es auf der Oberfläche ein Eingabefeld mit der Beschriftung RUN. Dort trägt man den Namen seiner Funktion wie zum Beispiel *myExt_UpdateAfterImport* ein, die nach dem Synchronisationsprozess ausgeführt werden soll. Diese Funktion muss dazu in einer eigenen Erweiterung existieren. Ein Beispiel: Wir möchten nach dem Synchronisationsprozess die Tabelle „Kunde“ mit ei-

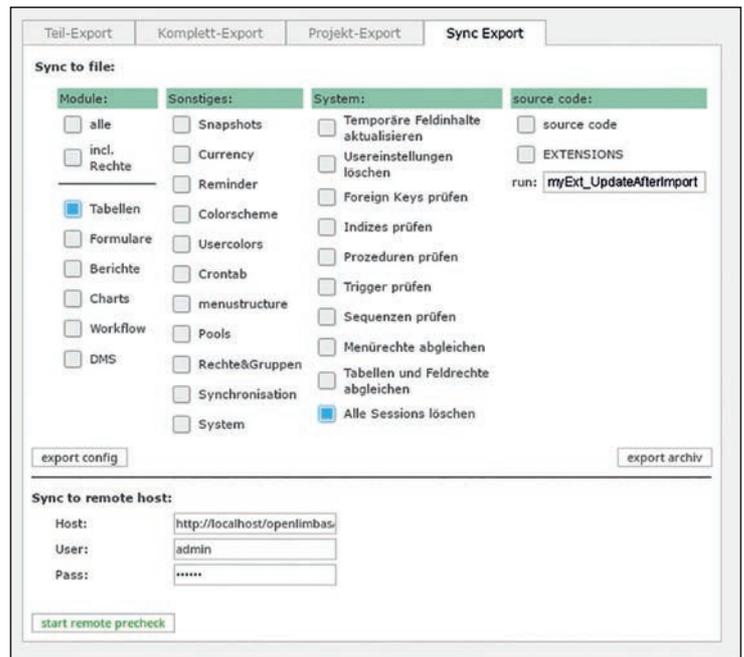


Abb. 1: Sync Export

nem neuen Feld „Kundentyp“ ergänzen. Der Inhalt soll abhängig von verschiedenen Kriterien des Kunden initial gefüllt werden. Obwohl innerhalb des Synchronisationsprozesses das Datenbankfeld „Kundentyp“ schon angelegt wurde, ist es noch leer. Mit Hilfe einer eigenen Erweiterung kann das Feld nun nachträglich gefüllt werden.

Der Knopf EXPORT ARCHIV erlaubt es, die generierten Exportdateien in ein Archiv zu packen und auf dem eigenen Rechner zu speichern, anstatt sie gleich zu einem anderen System zu übertragen. Dieses Archiv kann händisch auf das Zielsystem kopiert und importiert (siehe SYNC-IMPORT) oder einfach nur archiviert werden.

Der Knopf EXPORT CONFIG dagegen erzeugt einzig die Konfigurationsdatei, in der die zu exportierenden Module aufgelistet sind. Sie kann dazu genutzt werden, immer wiederkehrende Konfigurationen nicht jedes Mal neu erstellen zu müssen (Listing 1).

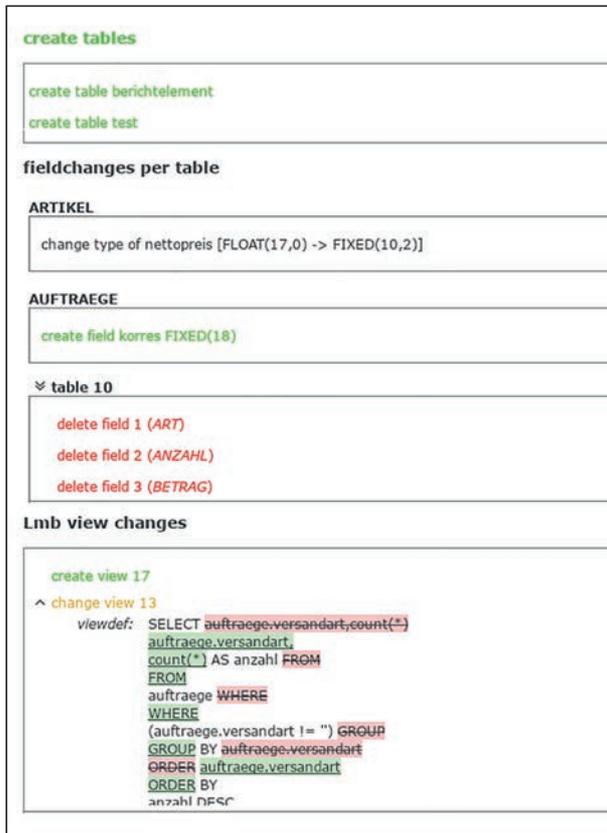
Der Abschnitt „Sync to remote host“ bietet die Option, den Export direkt auf das Zielsystem zu laden und zu importieren. Dazu gibt man den vollständigen URL inklusive Zielpfad des Zielsystems (z. B *https://IP/meinlimbas/dependent/*) und einen Administratorzugang mit

**Listing 1:
Script 1**

```
<SCRIPT 1>
<?php $toSync = array (
    0 => 'tabs',
    1 => 'forms',
    2 => 'rep',
); ?>
```

LIMBAS-Erweiterungen

LIMBAS-Erweiterungen sind ein wichtiger Bestandteil von Limbas. Sie ermöglichen es, an vielen Stellen eigenen Code in das Framework zu integrieren und so den Funktionsumfang zu erweitern. Mehr dazu findet man im Wiki unter <http://www.limbas.org/wiki/Skript-Erweiterungen>.

Abb. 2:
Create
Tables

Username und Passwort an. Ein Klick auf den Knopf START REMOTE PRECHECK exportiert alle nötigen Daten und simuliert einen Import im Zielsystem. Das Ergebnis ist eine Zusammenfassung (Kasten: „System-Jobs“) aller auszuführenden Anpassungen und Änderungen. Diese kann man überprüfen, sich als HTML-Übersicht herunterladen und durch den Knopf TODO bestätigen. Erst jetzt werden die Änderungen im Zielsystem ausgeführt.

Zu guter Letzt bekommt man noch einmal eine Zusammenfassung. Sollte ein Fehler aufgetreten sein, wird neben einer Fehlerbeschreibung das Zielsystem über die Transaktionen in den Ursprung zurückgesetzt. Dabei ist eine Transaktionsfähigkeit der jeweils eingesetzten Datenbank Voraussetzung. Eine Zusammenfassung der Anpassungen zeigt **Abbildung 2**. Das Beispiel zeigt ausschnittsweise, dass zwei neue Tabellen angelegt (grün), ein Feld in seiner Größe geändert (schwarz), ein Feld angelegt (grün) und mehrere Felder gelöscht (rot) werden. Weiterhin wird ein Vergleich einer geänderten SQL-Abfrage gemacht.

System-Jobs

Über das GUI ist eine Vorprüfung mit Zusammenfassung Pflicht. Man kann allerdings über die System-Jobs den Export direkt anstoßen bzw. per Cronjob in regelmäßigen Abständen ausführen lassen, ohne explizit zustimmen zu müssen. Mehr zu System-Jobs findet man unter <http://www.limbass.org/wiki/System-Jobs>.

Nachdem wir weiter oben im Artikel den Funktionsreiter SYNC-EXPORT vorgestellt haben, hier noch eine ausführlichere Erklärung zum Funktionsreiter SYNC-IMPORT.

Die schon angesprochene Funktion *Sync-Import* findet man im Importmenü. Sie benötigt weder einen URL zu einem anderen LIMBAS-System, noch nutzt sie eine Socket-Verbindung. Sie dient zum manuellen Importieren vorheriger exportierter und gespeicherter Archive, wie unter SYNC-EXPORT beschrieben. Analog zum Export wird nach Auswahl und Upload eines gespeicherten Archivs zuerst eine Übersicht aller Anpassungen ausgegeben, die bestätigt werden muss. Der Importprozess ist ab diesem Zeitpunkt analog zum Exportprozess auf einen Remote Host.

Der Revisionsmanager

Eine weitere nützliche Funktion ist im Menü TOOL unter REVISIONS-MANAGER zu finden. Werden Revisionen erstellt, werden sie analog zum Sync-Export in einem Archiv zusammengefasst und gespeichert. Das Interessante dabei ist: Erzeugte Revisionen können miteinander verglichen werden. So ist es möglich, schnell die Unterschiede im Datenmodell der Version 3 und des aktuellen Systems aufzuzeigen. Theoretisch ist es auch machbar, auf alte Versionsstände zurückzumigrieren, was aber einer Erzeugung einer neuen Version mit entsprechenden Änderungen gleichkommt. Mehr zum Thema Revisionsmanager findet man unter <http://www.limbass.org/wiki/Revisions-Manager>.

LIMBAS-Datensynchronisation

Die LIMBAS-Datensynchronisation synchronisiert im Gegensatz zum Deployment-Modul nicht die LIMBAS-Applikation, sondern die echten Dateninhalte der Tabellen. Es können alle in LIMBAS registrierten Tabellen synchronisiert werden, sofern man es will. Dabei werden die Daten in beide Richtungen synchronisiert und nach unterschiedlichen Regeln priorisiert. Die Besonderheit dieses Moduls besteht in seiner Flexibilität, unterschiedlichste Varianten (Templates) der Synchronisation zu erstellen und per Job oder manuell auszuführen.

Die Synchronisation erfolgt über PHP Sockets. Das schafft die Voraussetzung, zusammenhängende Aufgaben auszuführen, ohne den PHP-Prozess zu unterbrechen. Das ermöglicht es wiederum, für den gesamten Prozess Datenbanktransaktionen zu nutzen und damit eine hohe Datenintegrität zu erreichen. Insbesondere in einer Webumgebung ist die größte Gefahr, nach einem Verbindungsverlust oder Datenbankfehler inkonsistente Daten zu generieren. Die Nutzung von Transaktionen gewährleistet auch bei Verlust der Datenverbindung eine Zurücksetzung begonnener Änderungen. Der Ablauf der Synchronisation lässt sich wie folgt beschreiben:

- Herstellung der Socketverbindung zum Slave
- Prüfung auf Änderungen auf dem Slave

