Replication Mechanism of ZEMIS Ref

Tanja Küry

University of Bern

tanja.kuery@students.unibe.ch

04.07.2017
What is ZEMIS?
Zentrales Einwohner Migrations-System

- Manages immigrant data
- Lots of interfaces
What is ZEMIS Ref?
"ZEMIS Referenzdatenverwaltung"

- Administration application for so called 'reference data'
- About 380 tables with relational dependencies
- Several applications use the data
ZEMIS Ref - Overview

ZEMIS

ZEMIS ref

eArb/eAsyl

MIDES

Message Broker

eDossier

Tanja Küry

Replication Mechanism of ZEMIS Ref
Reference Data

What is reference data?
Data in the ZEMIS application landscape which ...

- is shared among applications
- supports multi-lingual text content
- controls application behaviour
- is used to interpret data from ZEMIS (e.g. nation codes)
- changes only occasionally
Problems

- Replication issue: Direct database access to clients
- New applications outside scope of project ZEMIS
- Web framework End-Of-Life since 2013 ⇒ security risks
- Business logic integrated into web layer ⇒ rewrite required
Replication: Current State

Push architecture with direct database access
ZEMIS Ref replicates changes directly into the client’s db

- Direct database access
- Client - data mapping
- Database connection and schema
- ZEMIS Ref release required to add new clients
- Tight coupling - makes changing the schema complex
- Robust, no synchronization issues, failures are detected, no heavy workload
Replication: Variant 1

Push architecture with web service
ZEMIS Ref pushes changes via web service

- Client - data mapping
- No direct database access
- Coupling loosen, schema can be changed
- Dynamic subscriber list
- Clients need to provide web service
- Robust, no synchronization issues, failures are detected, no heavy workload
Replication: Variant 2

Direct pull architecture (on the go)
Clients pull each time they need information

- Always up to date
- Permission and authentication instead of mapping
- No direct access to database
- No need to store the ref data
- Robustness low, heavy workload possible ⇒ performance requirements
Replication: Variant 3

Pull architecture with caching

ZEMIS ref offers web service, clients pull and cache the data

- No direct access to database
- Deltas can be pulled
- Loose coupling, clients can be added easily, schema can be changed
- Clients can have differing data $\Rightarrow$ conflicts
- Robustness medium high
Replication: Variant 4

Push notify to pull architecture with web services
ZEMIS ref pushes notifications about updates, clients pull if needed

- No direct access to database
- Loose coupling: Easy to add new clients, schema can be changed
- No heavy workload, responses can be scheduled
- Clients need to know whether to pull or not
- Robustness good, no differing data
- What if one client doesn’t get a notification?
Replication: Variant 4

Push notify to pull architecture with web services
ZEMIS ref pushes notifications about updates, clients pull if needed

- Publish-subscribe pattern
- Web service for ZEMIS ref and clients → overkill?
- Use real publish subscribe like JMS or RabbitMQ instead
Current Work

- Data is sent via SOAP as XML
- Basic XSD and WSDL are done
- Building prototypes to test the variants
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<x:multipleTablesRefDataRequest xmlns:x="http://ejpd.admin.ch/sem/zemis/refdataservice/types/v1">
  <x:clientId>MIDES</x:clientId>
  <x:startDate>2017-06-13T09:00:00</x:startDate>
  <x:requestedTable>
    <x:tableName>Gemeinde</x:tableName>
    <x:modifiedOrNew>true</x:modifiedOrNew>
  </x:requestedTable>
</x:multipleTablesRefDataRequest>
<xml version="1.0" encoding="UTF-8" standalone="yes"/>
<x:refDataResponse xmlns:x="http://ejpd.admin.ch/sem/zemis/refdataservice/types/v1">
  <x:table name="Adressdaten">
    <x:row index="1" status="new">
      <x:active>true</x:active>
      <x:entry column=""last name"" dataType="alphanumerisch">DJ</x:entry>
      <x:entry column=""first name"" dataType="alphanumerisch">Bobo</x:entry>
      <x:entry column=""adress line 1"" dataType="alphanumerisch">Hauptstrasse 33</x:entry>
      <x:entry column=""adress line 2"" dataType="alphanumerisch">4108 Witterswil</x:entry>
      <x:entry column=""phone number"" dataType="numerisch">0610123456</x:entry>
      <x:entry column=""swissJN"" dataType="boolean">J</x:entry>
      <x:entry column=""birth date"" dataType="datum">08.08.1988</x:entry>
    </x:row>
    <x:row index="2">
      <x:active>true</x:active>
      <x:entry column=""last name"" dataType="alphanumerisch">Doe</x:entry>
      <x:entry column=""first name"" dataType="alphanumerisch">John</x:entry>
      <x:entry column=""adress line 1"" dataType="alphanumerisch">Münstergasse 14</x:entry>
      <x:entry column=""adress line 2"" dataType="alphanumerisch">4053 Basel</x:entry>
      <x:entry column=""phone number"" dataType="numerisch">0790123456</x:entry>
      <x:entry column=""swissJN"" dataType="boolean">N</x:entry>
      <x:entry column=""birth date"" dataType="datum">09.09.1999</x:entry>
    </x:row>
    <x:row index="3">
      <x:active>false</x:active>
      <x:entry column=""last name"" dataType="alphanumerisch">Dupont</x:entry>
      <x:entry column=""first name"" dataType="alphanumerisch">Pierre</x:entry>
      <x:entry column=""adress line 1"" dataType="alphanumerisch">Maieńweg 11</x:entry>
      <x:entry column=""adress line 2"" dataType="alphanumerisch">4055 Basel</x:entry>
      <x:entry column=""phone number"" dataType="numerisch">0760123456</x:entry>
      <x:entry column=""swissJN"" dataType="boolean">J</x:entry>
      <x:entry column=""birth date"" dataType="datum">09.09.1999</x:entry>
    </x:row>
  </x:table>
</x:refDataResponse>
Future Work

- Test the prototypes
- Determine which variant to use
- Define next steps
Challenges

▶ Variants are quite complex
▶ Several stakeholders with different views
▶ Government reference architecture has to be met
Summary

- ZEMIS ref administrates reference data shared among applications
- Decapsulation of ZEMIS ref and its clients
- Developed several approaches to solve the issue