

20/09/2009



Jonathan Clarke
jonathan@phillipoux.net

Introduction

- LDAP directories are commonly used to store identity information
- Provisioning for identity management is easy ...
 - Just put all employee information in a directory!
- Simple, right? ... well, yes, but ...
 - « *HR already has software that only stores identity information in a database* »
 - « *We use Active Directory for our desktops and we need users' identities there too* »
 - « *XYZ software already uses a different directory* »

Introduction

- Several different identity repositories
 - How to make sure the same changes apply?
 - New employees
 - Name changes (marriage), transfers...
 - Employees leaving
- Manual synchronization?
 - Leads to a **mess**, leaving old accounts active ...
- Automatic synchronization?

Introduction

- Automatic synchronization
 - It already exists, and works great
 - Directory- / database-*specific* replication
 - Application-*specific* connectors (AD, SAP, etc)
 - What about the rest?
 - Between different databases, directories, files?
 - Different data models?
 - Using standards: LDAP, SQL, etc...?

About LSC Project

- What is LSC?
 - LDAP Synchronization Connector
 - Open Source project
 - BSD licence
 - Written in Java
 - 4 years in the making
 - 1 year ago *LSC-project.org* created
 - 6 regular contributors

- Website: <http://lsc-project.org>



Goals – functionality

- Read/write to any **database** or **LDAP directory**
 - Standard LDAPv3 operations
 - JDBC connectors for databases
- **Transform** data on-the-fly
 - Adapt to a different data model
 - JavaScript based engine to manipulate data
- Adjustable updates: default, keep, force or merge

Goals – usability

- **Quickly** implement a new synchronization
- Highly **configurable**
 - What *exactly* do we read?
 - Powerful **transformations** (correctness is important)
 - What *exactly* do we write?
- Run **fast** (performance is important)
- Easy to setup

Philosophy

- Make it **possible**, now!
- Make it more **stable** and **safer**
 - Open Source benefits over home-grown scripts
 - More secure and better tested
 - Don't reinvent a buggy wheel!
- Make it **faster** and **simpler**
 - Faster than writing home-grown scripts
 - Provide methods for IAM and directory-specific tasks
- This is not the ultimate solution ...

LSC synchronization principles

- Two levels of information per identity
 1. Existence – equivalent to an *account* (LDAP entry)
 2. Identity specific details – names, phone numbers (LDAP attributes and values)
- A unique ID: the *pivot* attribute(s)
- Synchronization operations
 - Create: Add entries from source to destination
 - Delete: Delete entries from destination not in source
 - Update: Compare and set specific details

LSC synchronization principles

- First step: sync
 - Get a list of all pivots from the source
 - For each pivot
 - Read the source object
 - Search for the destination object with pivot
 - Build up desired destination object by applying transformations to source object
 - If the destination object exists, calculate modifications
 - Apply: create or modify

LSC synchronization principles

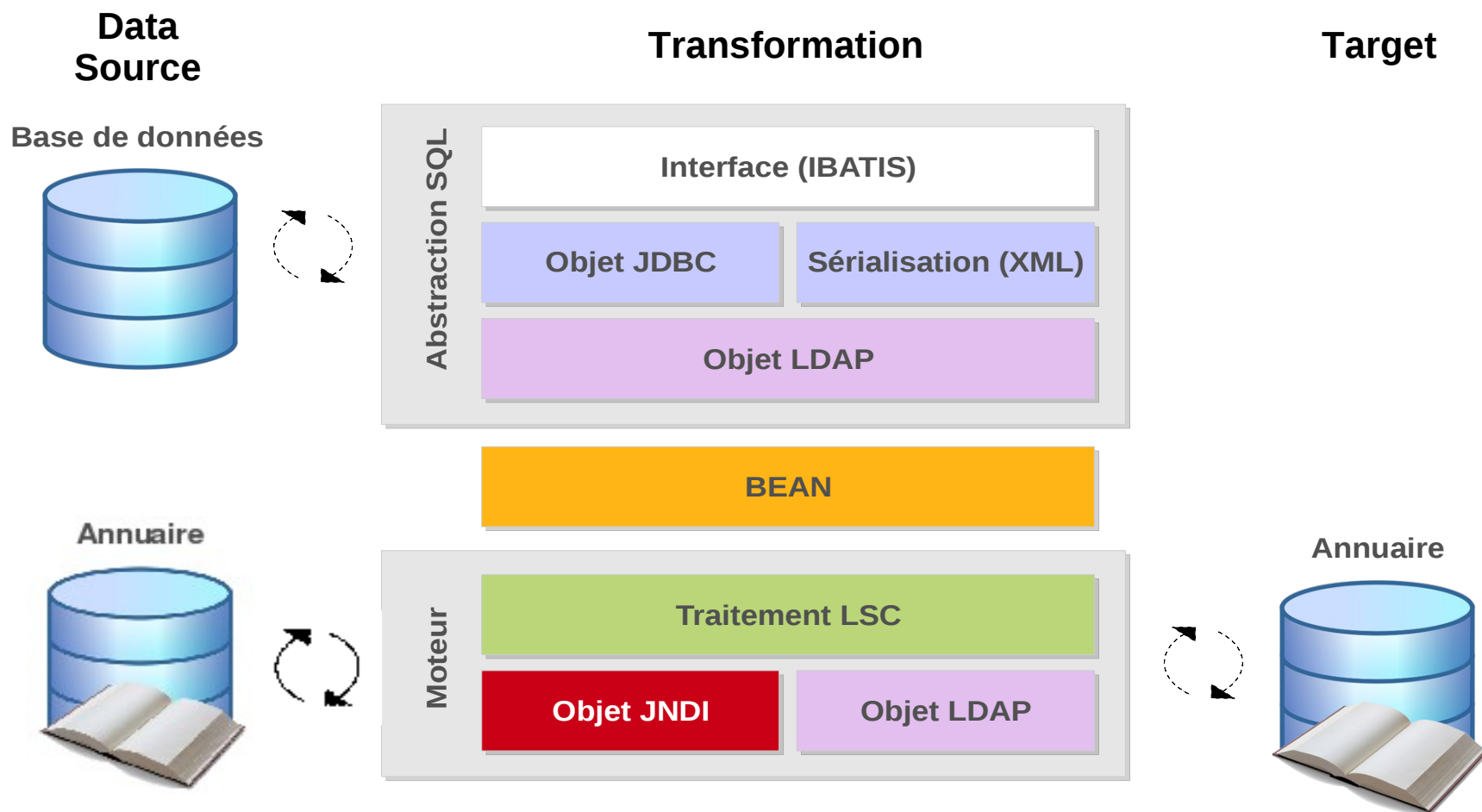
- Second step: clean (optional)
 - Get a list of all pivots from the destination
 - For each pivot
 - Search for the source object with pivot
 - If the source object doesn't exist, delete from destination
 - Apply: delete

Defining a synchronization

- Source type: LDAP / SQL database / CSV file ?
- Population: Which users? Which *pivot*?
- Information: Attributes? Transformations?



Software design



Example: MySQL to OpenLDAP

- MySQL: a simple users table (HR-style)

Field	Type	Values
id	INT	Auto-increment
first_name	VARCHAR	« Jane »
last_name	VARCHAR	« Doe »
marital_status	ENUM	« Single » / « Married » / « Divorced »
salary	INT	42000
start_date	DATE	1 st October 2009

Example: MySQL to OpenLDAP

- Configuring the source database
 - JDBC connector: *com.mysql.jdbc...*
 - URL, username, password
 - Simple SQL request

```
SELECT id AS uid, first_name AS givenName,  
last_name AS sn, start_date AS startDate FROM users
```

Example: MySQL to OpenLDAP

- OpenLDAP: inetOrgPerson entries

Field	Type	Values
givenName	String	first_name (ex: « Jane »)
sn	String	last_name (ex: « Doe »)
cn	String	LAST_NAME, first_name (ex: « DOE, Jane »)
userPassword	Binary string	Defaults to « CHANGEME »
uid	String	Unique id from MySQL table

Example: MySQL to OpenLDAP

- Configuring the destination directory

```
dst.java.naming.provider.url = ldap://localhost/dc=lsc-project,dc=org  
dst.java.naming.security.authentication = simple  
dst.java.naming.security.principal = cn=Manager,dc=lsc-project,dc=org  
dst.java.naming.security.credentials = secret
```

Example: MySQL to OpenLDAP

- Configure the synchronization task
 - Source directory searching

```
lsc.tasks = MyTask
lsc.tasks.MyTask.type = db2ldap
lsc.tasks.MyTask.dstService.baseDn = ou=People
lsc.tasks.MyTask.dstService.pivotAttrs = uid
lsc.tasks.MyTask.dstService.filterAll = (uid=*)
lsc.tasks.MyTask.dstService.attrs = uid sn cn givenName userPassword
lsc.tasks.MyTask.dstService.filterId = (uid={uid})
```

- DN generation

```
lsc.tasks.MyTask.dn = "uid=" + srcBean.getAttributeValueById("uid") \
+ "ou=People"
```

Example: MySQL to OpenLDAP

- Configuration data transformations (syncoptions)

```
lsc.syncoptions.MyTask.default.action = F
```

```
lsc.syncoptions.MyTask.cn.force_value = \  
    srcBean.getAttributeValueById("sn").toUpperCase() + ", " \  
    + srcBean.getAttributeValueById("givenName")
```

```
lsc.syncoptions.MyTask.userPassword.action = K
```

```
lsc.syncoptions.MyTask.userPassword.default_value = \  
    SecurityUtils.hash(SecurityUtils.MD5, "CHANGEME")
```

Features overview

- Syncoptions offer unlimited possibilities
 - Text transformations
 - cn = givenName + SPACE + SN in caps
 - Filter accents: convert « Hélène » to « Helene »
 - Hash passwords (SSHA, MD5, etc)
 - Simple LDAP bind test
 - Active Directory specifics:
 - UserAccountControl: deactivate accounts, force password changes, etc ...
 - UnicodePwd: update passwords in AD-style
 - Anything else you can write in Java!

Features overview

- Operation conditions
 - Perform ADDs / MODIFYs / MODRDNs / DELETES conditionally
- Use-cases:
 - Update-only synchronizations (never create, never delete)
 - Only update the password if it's changed (perform a LDAP bind operation to check on the fly)
 - Delete an account after 60 days of inactivity

Features overview

- Attribute-level priorities for update
 - FORCE: replace the destination value whatever
 - KEEP: leave the destination value as-is
 - DEFAULT: value to use if the destination is empty
 - CREATE: default value for new entries
- Use cases:
 - Provide a default password but don't squash real one
 - Force phone numbers if we're authoritative for them

Features overview

- Detailed and configurable logging
 - LDIF format (fully RFC-compliant)
 - CSV format

- Audit or play back modifications

Perspectives

- Project is currently in stable status
 - Version 1.1.0 released
- Ideas for improvement are everywhere:
 - Implement directory-specific replication systems
 - LDAP sync (RFC 4533) for OpenLDAP, ApacheDS
 - DirSync for Microsoft AD
 - Others?
 - Support other scripting languages
 - Plugins to integrate into enterprise workflows
 - Anything else ...

Try it out! Get involved!

- Main website: <http://lsc-project.org/>
 - Tutorials: quickstart demo, detailed tutorials
 - Reference documentation

The screenshot shows the homepage of the LSC Project website. At the top left is the LSC Project logo. A navigation bar contains links for ABOUT, DOWNLOAD, DOCUMENTATION, COMMUNITY, ROADMAP, and LOGIN. Below the navigation bar, a grey box contains the text: "Ldap Synchronization Connector provides tools to synchronize a LDAP directory from a list of data sources including any database with a JDBC connector, another LDAP directory, flat files..." followed by "Download | Read more...". The main content area features the heading "LDAP SYNCHRONIZATION CONNECTOR (LSC)" and the sub-heading "Database/LDAP/flat files to LDAP configurable engine". There are four main sections with icons: "About LDAP Synchronization Connector (LSC)" with a flag icon, "Documentation" with a lightbulb icon, "Community" with a group of people icon, and "Roadmap" with a gear icon. On the right side, there is a sidebar with a "Download" button, "Latest Release" section listing "LSC version 1.1 coming soon!" and "Nightly builds available to test", "Events" section listing "10/07/2009 - RMLL (Nantes)" and "25/06/2009 - LinuxTag (Berlin)", "Community" section with text "Get help, contribute or find professional services ... Find out more!", a "Search" input field with a "Search" button, and "Ohloh statistics" showing "Ohloh" logo and "\$304.3K Cost".

Try it out! Get involved!

- Getting help (keep in touch!)
 - Mailing lists: <http://lists.lsc-project.org/>
 - IRC: **#lsc-project** on Freenode
- Development tools:
 - Redmine forge: <http://tools.lsc-project.org/>
 - Bugtracker, SVN repository ...
 - Continuous build server
 - Lots of tests based on OpenDS

Success stories

Private:

The logo for LAPEYRE, featuring the word "LAPEYRE" in white capital letters on a red rectangular background.

Database to directory
8 different instances



Active Directory
to OpenLDAP

Public:



Oracle and MySQL to OpenLDAP
250 000 entries



CSV files to OpenLDAP

Thanks for your attention!
Any questions?



Jonathan Clarke
jonathan@phillipoux.net