Open XAL Database Abstraction

Introduction

Database abstraction refers to the abstraction of both the vendor and the schema such that a high level API has all necessary access to the data source without requiring vendor specific or schema specific details.

Vendor Abstraction

Vendor abstraction refers to a high level API that is independent of the database vendor. Vendor abstraction is already supported in the current XAL project, and this current architecture is the proposed support in Open XAL.

JDBC provides most of the database vendor abstraction, but some functions are still vendor specific including making a new connection, creating BLOB data, creating arrays and fetching nontrivial schemas. XAL introduces a DatabaseAdaptor class which provides abstract methods for all of these functions. Concrete subclasses (one for each supported vendor) provide the implementation details.

A database configuration file provides a list of adaptors, servers and accounts with defaults for each. Each server entry specifies an identifier, URL and its associated database adaptor which in turn specifies an identifier and the concrete database adaptor subclass to load. Each account entry specifies an identifier and the user name and password. A connection dictionary is composed of an account and server pair and is used to establish the database connection. Figure 1 shows an example of a database configuration file.

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<dbconfig date="Thu Jun 11 10:46:03 DST 2010" version="1.0.0">
	<adaptors default="oracle">
		<adaptor name="oracle" class="gov.sns.tools.database.OracleDatabaseAdaptor" />
		<adaptor name="mysql" class="gov.sns.tools.database.MysqlDatabaseAdaptor" />
	</adaptors>
	<servers default="production">
		<server name="production" adaptor="oracle" url="jdbc:oracle:thin:@xal.org:1521:prod" />
		<server name="development" adaptor="oracle" url="jdbc:oracle:thin:@xal.org:1521:dev" />
		<server name="internal" adaptor="mysql" url="jdbc:mysql:thin:@xal.org:1521:dev" />
	</servers>
	<accounts default="reports">
		<account name="reports" user="reporter" password="xyz123" />
		<account name="score" user="backup" password="abc456" />
		<account name="personal" user="tom" />
	</accounts>
</dbconfig>
```

Figure 1: Example of a database configuration file.

This database vendor abstraction described above is already implemented in the current version of XAL in use at SNS. However, we should add to the database adaptor error code translation so XAL can define common SQL error code symbols and have them numerically assigned the vendor specific values.

Schema Abstraction

Schema abstraction refers to a high level API that is independent of the details of the data table organization within the database. Schema abstraction will be a new feature introduced in Open XAL.

For each module that requires database access, the database request interface will be encapsulated into one or more abstract classes and/or interfaces. Implementations of these data sources will be provided by individual site projects, but the core will provide placeholder implementations so the core can be built successfully and tested without reference to a real database. The placeholder classes will be excluded from the core jar, but instead archived into *datasource.jar* and will not be distributed to the sites. Site code will provide site specific substitutions for the placeholder classes using identical class names and packages. Note that these implementation packages must be unique from those in the core to allow the core packages to be sealed.

Consider the PV Logger as an example. The following table show a possible package distribution among jar files.

Jar File	Package	Class / Interface	Role
xal-core.jar	xal.tools.pvlogger	PVLoggerDataSource	Data source interface or abstract class
xal-datasource.jar	xal.tools.pvlogger.datasource	PVLoggerDatabaseSource	Placeholder data source implementation not to be distributed. Built alongside core.
xal-*site*.jar	xal.tools.pvlogger.datasource	PVLoggerDatabaseSource	Site specific data source implementation

Document Revision History

This table describes the changes to this Open XAL database abstraction document.

Date	Notes
June 25, 2010	Draft proposal that describes the plan for supporting database abstraction in Open XAL.
October 7, 2010	Clarify the organization of the packages and the jar file names.