Alternative Technologies-Cayenne

Comparison of EOF to Cayenne

Introduction

EOF is the data access portion of WebObjects that provides Object-Relational Mapping (ORM). Apache Cayenne is an alternative ORM that can be used within a WebObjects app as a replacement for EOF or in any sort of Java app, like a Servlet application.

Below is a description of the differences between EOF and Cayenne with the goal of providing information to a WebObjects/EOF developer who is curious about switching to Cayenne.

Key Value Coding

Cayenne doesn't really have a KVC mechanism like EOF does. Properties of entity objects can be accessed by name by using the readProperty and writeProperty methods of CayenneDataObject, but these methods do not invoke custom getters and setters like EOF does, they simple return the value. So they are analogous to storedValueForKey and takeStoredValueForKey (however, readProperty also supports a key path, not just a simple key).

The closest match to KVC is found in a utility class: PropertyUtils.getProperty and PropertyUtils.setProperty. These can be leveraged to provide access to property values that go through custom getters and setters. BUT the getter methods must be prefixed with "get", following the java-beans convention.

It should be noted however, that complete KVC can be added on top of Cayenne by using the JavaFoundation framework in WO, or by using the custom implementation found in the NSFoundation project that is in Project Wonder. This approach would provide a completely compatible KVC implementation on top of Cayenne. An example of this is available in the ERCayenne framework in Project Wonder.

Concurrency

Cayenne has much better concurrency support that EOF. Locking the framework stack is not required as it is in EOF. Cayenne also supports jdbc connection pooling out of the box. I'm not aware of any benchmarks comparing performance, but Cayenne should be much faster at handling concurrent operations.

Raw Rows

In Cayenne raw row fetches are limited to returning columns from a single table. This differs from EOF which allows for any related key path to be fetched. In Cayenne the qualifier may still reference key paths, they just cannot be returned in the select. You could consider rewriting complex EOF Raw Row fetches with SQLTemplate in Cayenne.

Inheritance

Cayenne currently supports only single table inheritance (completely) and vertical inheritance (partially). However, horizontal inheritance is slated for implementation in the next release of Cayenne (3.2?).

Tooling

CayenneModeler is a standalone application that can be used on any platform. There is also an Eclipse plugin under development that should be viable soon. The Eclipse plugin will allow for automatically regenerating entity classes when the model is modified, but the standalone app doesn't this (not directly anyway).

Entity Templates

The default entity templates in Cayenne are pretty sparse, without a lot of convenience methods like the WonderEntity template has for EOF. But this is easily remedied, and the ERCayenneExample in Wonder has and EOF-like template you can use.

Modeler / Model API

In Cayenne each Entity has two parts: 1) an Object mapping and a DB mapping, referred to as ObjEntity and DbEntity. These appear separately in Modeler and are linked together within Modeler. ObjEntity contains the attribute name and java class type; DbEntity contains the column name and database type. This differs from EOF where these two concepts are combined into a single editor in the Modeler and into a single EOEntity class at runtime. Whilst I find the separation in Cayenne to be a bit cumbersome, it allows for for possibility for operations to be performed at a lower level using just the database information and bypassing the object layer. This may be useful in a small number of specific situations.

Deployment

Cayenne is agnostic to the specific deployment environment, however it is almost always deployed to a standard Java Application Server (Servlet container) like JBoss or Glassfish, etc. This is an industry standard approach, which has some advantages. Personally, I quite like that the app is entirely contained in a single .war file.

However, Cayenne can **easily be used in a WO application** as it is just another jar and doesn't have any special deployment requirements.

Feature Matrix

Feature	EOF	Cayenne	Notes
Open Source		X	
Still Developed / Maintained		X	WebObjects/EOF has been given legacy status by Apple.
Development on any Platform		X	WebObjects/EOF may only be developed on Apple hardware.
Eclipse Integration	X	*	Cayenne Eclipse integration is being developed and an alpha version is available.
Good Concurrency Support		X	EOF is single threaded and requires explicit locking, preventing concurrent operations.
Support for all common DBs	X	X	
Primitive Attributes (int, double, long, etc)		X	
Enum Attributes	X*	X	Enum attributes are supported in EOF via the javaEnum prototype in Wonder.
Relationships with Map or Set collection types		X	
Custom Attribute Types	*	X	The ERAttributeExtension framework in Wonder can provide this support for EOF.
Embeddable relationships (related objects stored in same table as the source)	*	X	EOF allows mapping relationships to the same table as the parent and using the same entity class. Cayenne allows embedding objects of a different class as de-normalized relationships in a single column.
Single Table Inheritance	X	X	
Horizontal Inheritance	X	*	Horizontal inheritance is being developed and targeted for the next release of Cayenne (3.1)
Vertical Inheritance	X	*	Cayenne's vertical inheritance support is not yet complete
Flattened attributes/relationships	X	X	
Automatic Bi-directional Relationship Management	X*	X	EOF support is optionally provided by Wonder.
Generics support (Java 5)			Generics support is mostly artificial in both EOF and Cayenne (implemented by casting in entity template).
Support Transient Objects (objects not in editing context)	X	*	Cayenne supports Transient object manipulation for attributes, but not relationships.
Support for Generic Objects (no custom classes)	X	X	
Optimistic Locking	X	X	
Optional Editing Context Synchronization		X	EOF automatically synchronizes editing contexts which can hide Optimistic Locking problems.
Key Value Coding	X		Cayenne has some utility methods to access bean properties, and support for raw data access like storedValueForKey provides for EOF, but no direct analog to KVC.
Customizable entity templates	X	X	
Raw fetching (Raw Rows / Data Rows)	X	X*	Cayenne's raw fetching is limited to fetching attributes, not relationships (see here), but other lower level support is available without resorting to raw SQL.
Direct SQL support	X	X	
Nested Editing Contexts	X	X	
Editing Context Undo	X		

Remote / Distributed Data Access (Client apps) using API directly	X	X	
Android Support		*	Android support is available with a patch to Cayenne. EOF will run on Android, but the license does not allow it.
Pluggable Cache Implementation		X	
Database Migrations	X	*	Migrations are available in Cayenne with a pending project in the sandbox and an open issue (vote it up!). CayenneModeler has the ability to inspect schema and migrate it as an one-time Admin function; this could be extended to generate a migration upgrade classes in additional to the initial migration.
EJBQL Query support (defined by JPA spec)		X	
Raw SQL Templating		X	Cayenne allows SQL to be written using the velocity template language and stored in the model so it can support multiple database types.
Batch Querying		X	Cayenne provides this using QueryChain
Documentation in the model (EO model doc)	X		
Default values defined in the model (for migrations and in entity objects)	X		
Servlet Deployment	X	X	
JavaMonitor / wotaskd Deployment	X	X	
Incredible Community and Conferences	X		

API Comparison

For EOF users wanting to use Cayenne there is very little to learn to get started. The frameworks are conceptually very similar at the user level, so you just have to translate your EOF knowledge to some new method and class names.

EOF	Cayenne
EOEditingContext	**DataContext**
.objectWithFetchSpecification	.performQuery
.saveChanges	.commitChanges
.revert	.rollbackChanges
.lock	Locking is not necessary in Cayenne!
.unlock	Locking is not necessary in Cayenne!
.insertObject	.registerNewObject
.deleteObject	.deleteObject
.undo	No analog in Cayenne
EOGenericRecord	**CayenneDataObject**

.storedValueForKey	.readProperty
.takeStoredValueForKey	.writeProperty
.valueForKey	PropertyUtils.getProperty (sort of, see KVC section above)
.takeValueForKey	PropertyUtils.setProperty (sort of, see KVC section above)
.__globalID	.getObjectId
.entity()	Cayenne.getObjEntity(eo)
.validateForSave	.validateForSave
.validateForInsert	.validateForInsert
.validateForUpdate	.validateForUpdate
.validateForDelete	.validateForDelete
.validateValueForKey	No analog in Cayenne
.<validateSpecificAttributeViaReflection>	No analog in Cayenne
.willUpdate, willInsert, etc	provided by Cayenne lifecycle callbacks
EOQualifier	**Expression**
EOQualifier.qualifierWithQualifierFormat	Expression.fromString
ERXKey	**Property**
ERXKey.eq	Property.eq
ERXKey.ne	Property.ne
EOFetchSpecification	**SelectQuery**
.qualifier	.getQualifier
.setSortOrderings	.addOrderings
.setPrefetchingRelationshipKeyPaths	.addPrefetch
EOEntity	**ObjEntity + DbEntity**

EOModel	**DataMap (entities) + DataNode (connection dictionary)**
EOAttribute	**ObjAttribute + DbAttribute**
EOClassDescription	**DataObjectDescriptor**