



Introduction To Cayenne

Bill Dudney

Virtuas Open Source Solutions





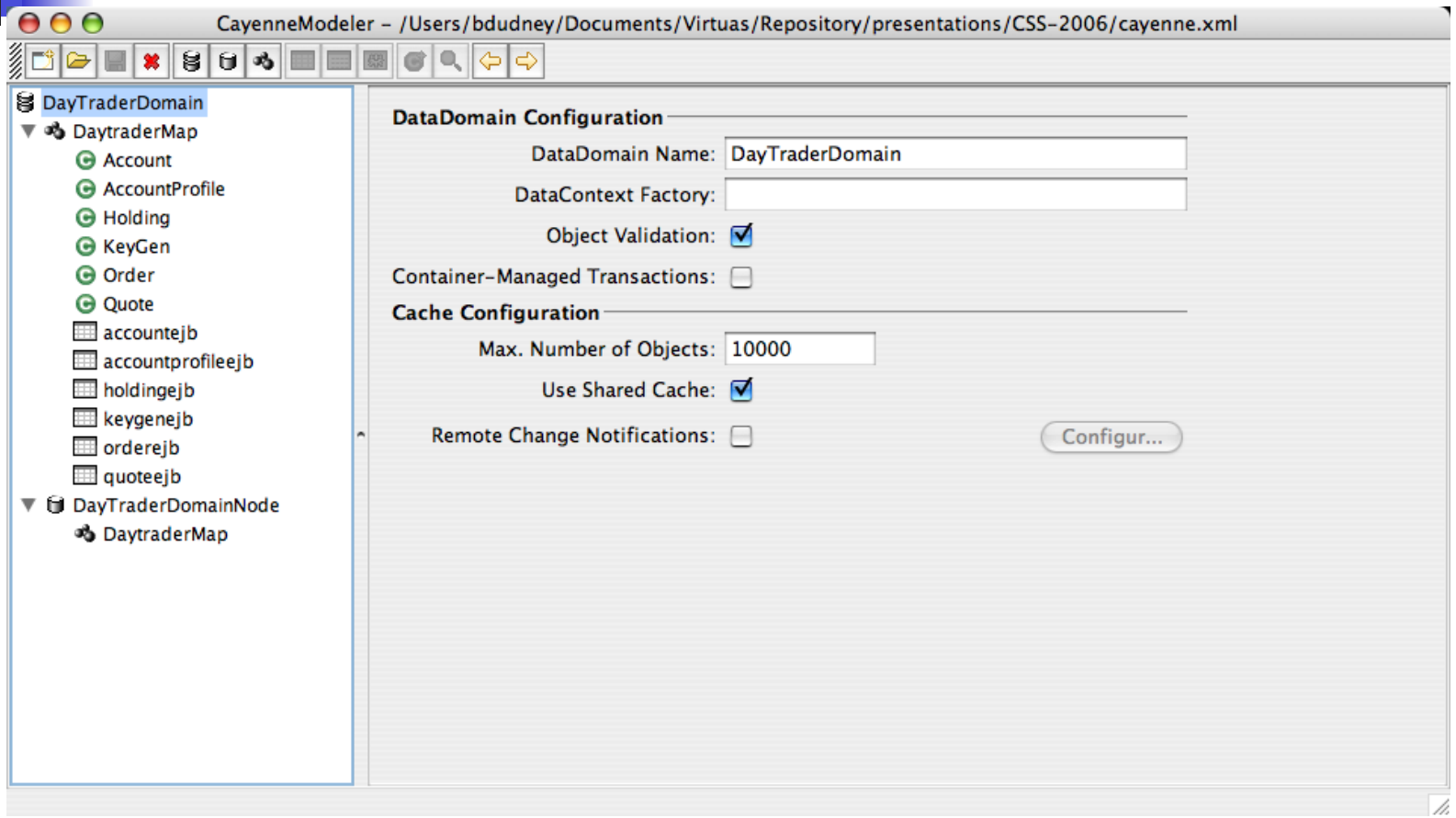
R/O Mapping?

- Mainstream Technology
 - Hibernate
 - TopLink
 - EJB 3
- Long history of success (TopLink started in late 80's)



The Successful MDA

Cayenne Modeling



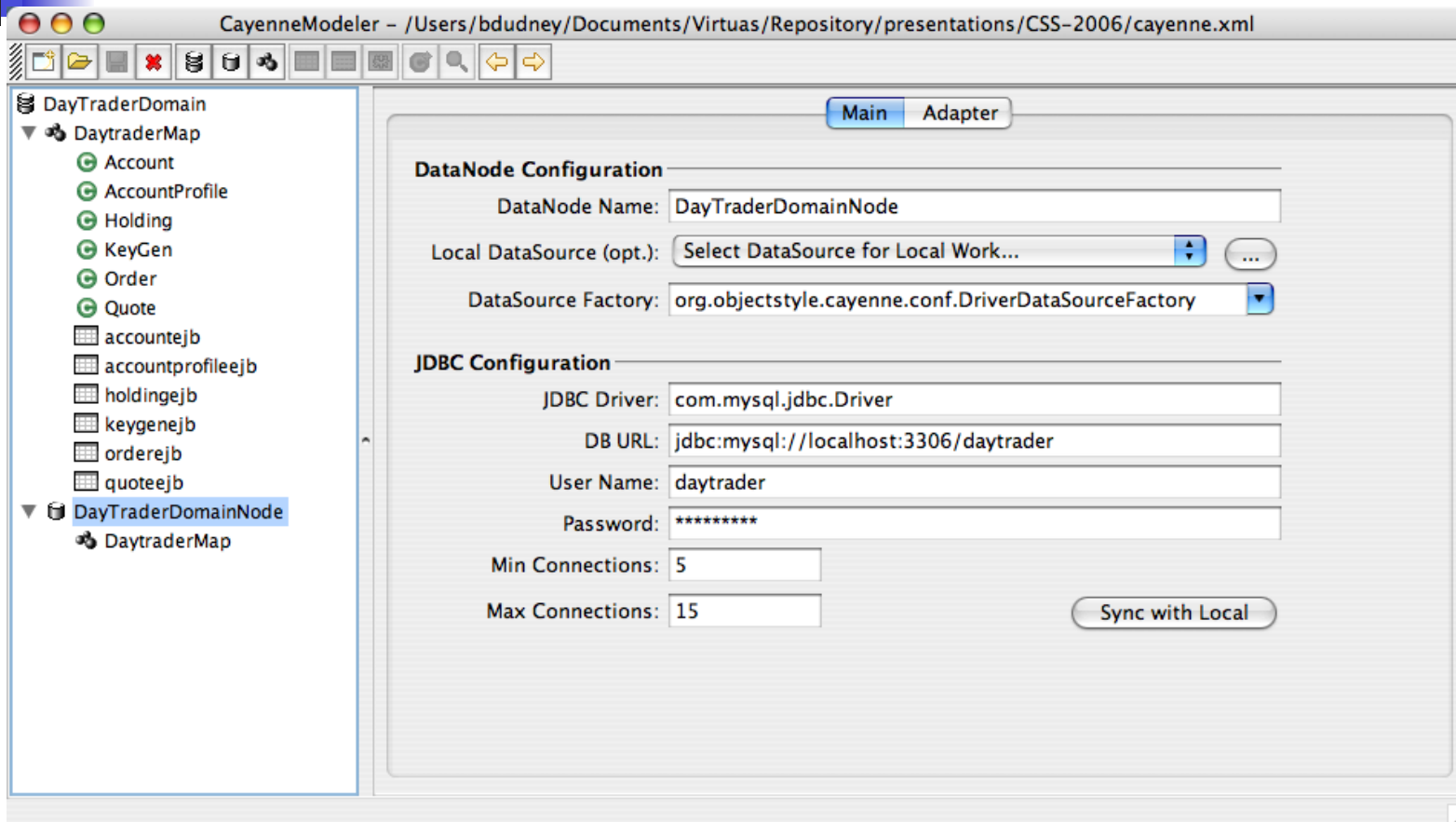
Cayenne Modeling

The screenshot shows the CayenneModeler application window titled "CayenneModeler - /Users/bdudney/Documents/Virtuas/Repository/presentations/CSS-2006/cayenne.xml". The interface is divided into two main sections. On the left is a tree view of the project structure under "DayTraderDomain". The "DayTraderMap" folder is expanded, showing several entities: Account, AccountProfile, Holding, KeyGen, Order, and Quote. Below these are several entity classes with ".ejb" suffixes: accountejb, accountprofileejb, holdingejb, keygenejb, orderejb, and quoteejb. The "DayTraderDomainNode" folder is also expanded, showing the "DayTraderMap" entity class. On the right is the "DataMap Configuration" panel. It contains the following fields and controls:

- DataMap Name:** DaytraderMap
- File:** DaytraderMap.map.xml
- DataNode:** DayTraderDomainNode (selected from a dropdown menu)
- Entity Defaults:**
 - DB Schema:** [text field] [Update...]
 - Java Package:** [text field] [Update...]
 - Custom Superclass:** [text field] [Update...]
 - Optimistic Locking:** [Update...]
- Client Class Defaults:**
 - Allow Client Entities:**
 - Client Java Package:** [text field] [Update...]



Cayenne Modeling



Cayenne Modeling

The screenshot shows the CayenneModeler application window. The title bar reads "CayenneModeler - /Users/bdudney/Documents/Virtuas/Repository/presentations/CSS-2006/cayenne.xml". The interface is divided into a left-hand tree view and a main right-hand pane.

The tree view on the left shows a hierarchy starting with "DayTraderDomain", which contains "DaytraderMap". Under "DaytraderMap", there are several entities: "Account" (selected), "AccountProfile", "Holding", "KeyGen", "Order", and "Quote". Below these are several EJBs: "accountejb", "accountprofileejb", "holdingejb", "keygenejb", "orderejb", and "quoteejb". At the bottom of the tree is "DayTraderDomainNode", which contains "DaytraderMap".

The main pane is titled "Attributes" and contains a table with the following columns: "ObjAttribute", "Java Type", "DbAttribute", "DB Type", and "Used for Lockin".

ObjAttribute	Java Type	DbAttribute	DB Type	Used for Lockin
balance	java.math.BigDecimal	balance	DECIMAL	<input checked="" type="checkbox"/>
creationDate	java.util.Date	creationdate	TIMESTAMP	<input checked="" type="checkbox"/>
lastLogin	java.util.Date	lastlogin	TIMESTAMP	<input checked="" type="checkbox"/>
loginCount	java.lang.Integer	logincount	INTEGER	<input checked="" type="checkbox"/>
logoutCount	java.lang.Integer	logoutcount	INTEGER	<input checked="" type="checkbox"/>
openBalance	java.math.BigDecimal	openbalance	DECIMAL	<input checked="" type="checkbox"/>
profileUserID	java.lang.String	PROFILE_USERID	VARCHAR	<input checked="" type="checkbox"/>



Cayenne Modeling

The screenshot shows the CayenneModeler application window. The title bar reads "CayenneModeler - /Users/bdudney/Documents/Virtuas/Repository/presentations/CSS-2006/cayenne.xml". The interface is divided into a left-hand tree view and a main central pane.

The tree view on the left shows a hierarchy starting with "DayTraderDomain". Underneath it is "DaytraderMap", which contains several entities: "Account", "AccountProfile", "Holding", "KeyGen", "Order", "Quote", "accountejb" (highlighted), "accountprofileejb", "holdingejb", "keygenejb", "orderejb", and "quoteejb". Below these is "DayTraderDomainNode" containing "DaytraderMap".

The main pane has three tabs: "Entity", "Attributes", and "Relationships". The "Attributes" tab is selected. It contains a table with the following columns: "Name", "Type", "PK", "Mandatory", "Max Length", and "Precision".

Name	Type	PK	Mandatory	Max Length	Precision
PROFILE_USERID	VARCHAR	<input type="checkbox"/>	<input type="checkbox"/>	250	
accountid	INTEGER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	11	
balance	DECIMAL	<input type="checkbox"/>	<input type="checkbox"/>	10	2
creationdate	TIMESTAMP	<input type="checkbox"/>	<input type="checkbox"/>	19	
lastlogin	TIMESTAMP	<input type="checkbox"/>	<input type="checkbox"/>	19	
logincount	INTEGER	<input type="checkbox"/>	<input checked="" type="checkbox"/>	11	
logoutcount	INTEGER	<input type="checkbox"/>	<input checked="" type="checkbox"/>	11	
openbalance	DECIMAL	<input type="checkbox"/>	<input type="checkbox"/>	10	2





Demo





Getting At Data

The Context API



Running Queries

```
public java.util.List performQuery(Query query)
```



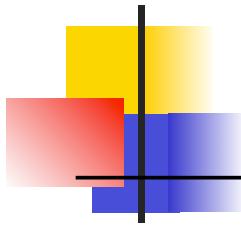
public void commitChanges()



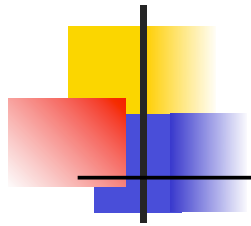
```
public void deleteObject(java.lang.Object object)
```



```
public java.lang.Object newObject(java.lang.Class clazz)  
public void registerNewObject(java.lang.Object object)
```



Code



Remotely Getting At Data





API Stays the Same

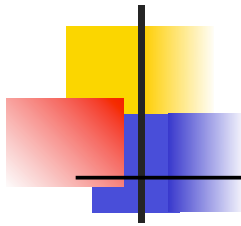
```
public java.util.List performQuery(Query query)
```

```
public void commitChanges()
```

```
public void deleteObject(java.lang.Object object)
```

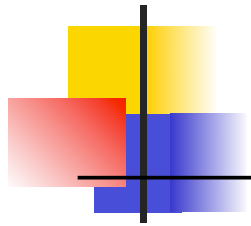
```
public void registerNewObject(java.lang.Object object)
```

```
public java.lang.Object newObject(java.lang.Class clazz)
```



Code





Conclusion & Questions

