Scale, Share and Store your Models with CDO 2.0

Eclipse Live Webinar, January 29, 2009

Eike Stepper
stepper@esc-net.de
http://www.esc-net.de
http://thegordian.blogspot.com
Berlin, Germany

Ed Merks
Ed.Merks@gmail.com
http://www.macromodeling.com
http://ed-merks.blogspot.com
Ballantrae, Ontario, Canada
<table>
<thead>
<tr>
<th>Eike</th>
<th>Ed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typical Application</td>
<td>The Wonder of EMF</td>
</tr>
<tr>
<td>answers questions</td>
<td>EMF Snippet</td>
</tr>
<tr>
<td>Typical CDO Deployment</td>
<td></td>
</tr>
<tr>
<td>Typical CDO Application</td>
<td>answers questions</td>
</tr>
<tr>
<td>Typical CDO Server</td>
<td></td>
</tr>
<tr>
<td>Live Demonstration</td>
<td></td>
</tr>
<tr>
<td>Live Discussion</td>
<td></td>
</tr>
<tr>
<td>(CDO Snippets)</td>
<td></td>
</tr>
<tr>
<td>(Next Steps)</td>
<td></td>
</tr>
</tbody>
</table>
Typical Application

Input

Business Logic

Persistence Logic

Output

Business Objects

.Scale, Share and Store your Models with CDO 2.0
© 2009 by Eike Stepper, Berlin, Germany. Made available under the EPL v1.0
Cross References by EMF
Model View Controller by EMF

- Input
- Output

Business Logic

Persistence Logic

ResourceSet

Business Objects

.xml
.dat
RDB

Scale, Share and Store your Models with CDO 2.0
© 2009 by Eike Stepper, Berlin, Germany. Made available under the EPL v1.0
Business Logic is Yours
Models Generated by EMF
Persistence Handled by EMF

Persistence Logic

.ResourceSet

.xml
.dat
.RDB

Scale, Share and Store your Models with CDO 2.0
© 2009 by Eike Stepper, Berlin, Germany. Made available under the EPL v1.0
Default Serialization by EMF

XMLResource

BinaryResource

RDB

Scale, Share and Store your Models with CDO 2.0
© 2009 by Eike Stepper, Berlin, Germany. Made available under the EPL v1.0
ResourceSet rs = new ResourceSetImpl();
rs.getResourceFactoryRegistry().getExtensionToFactoryMap()
    .put("xml", new XMLResourceFactoryImpl());
rs.getPackageRegistry().put(MODEL.getNsURI(), MODEL);

URI uri = URI.createFileURI("C:/business/company.xml");
Resource resource = rs.getResource(uri, true);
resource.setTrackingModification(true);

Company company = (Company)resource.getContents().get(0);
executeBusinessLogic(company);

if (resource.isModified())
{
    resource.save(null);
}
Typical CDO Deployment

Client 1

Client 2

Client 3

Server

Model Repository

RDB

Scale, Share and Store your Models with CDO 2.0
© 2009 by Eike Stepper, Berlin, Germany. Made available under the EPL v1.0
Multiple Repositories

Client 1

Model Repository

Client 2

Model Repository

Client 3

RDB
Typical CDO Application
Integration with ResourceSet

Diagram showing the integration with XATransaction, ViewSet, ResourceSet, View Audit Transaction, Session, and Model Repository.

Scale, Share and Store your Models with CDO 2.0
© 2009 by Eike Stepper, Berlin, Germany. Made available under the EPL v1.0
Each Repository Once per ViewSet
Multiple Views per Session

Each Repository Once per ViewSet
Scalability through Revisions

View

Object

Session

Cached Revisions

7 v1: 13, 263, 94

7 v2: 13, 263

7 v3: 82, 263
Views are Revision Selectors

View

Object

Transaction

Dirty Object

Local Revision

Session

Cached Revisions

Audit

Historical Object

Scale, Share and Store your Models with CDO 2.0
© 2009 by Eike Stepper, Berlin, Germany. Made available under the EPL v1.0
Object State Machine

TRANSIENT

attach to view

detach from view

NEW

commit

DIRTY

commit

read

remote invalidate

write

remote invalidate

PROXY

read

remote invalidate

CLEAN

CONFLICT

reload
Live Demonstration
public CDOSession openSession()
{
    // Create TCP connector through wiring container
    IConnector connector = (IConnector)IPluginContainer.INSTANCE
        .getElement("org.eclipse.net4j.connectors", "tcp", "dev.foo.com:2036");

    // Create CDO session configuration
    CDOSessionConfiguration config = CDONet4jUtil.createSessionConfiguration();
    config.setConnector(connector);
    config.setRepositoryName("repo1");
    config.setLazyPackageRegistry();

    // Open CDO session
    return config.openSession();
}
Opening Views, Audits and Transactions

```java
CDOSession session = openSession();

// Open multiple views on a session
CDOView view = session.openView();
CDOAudit audit = session.openAudit(new Date("2009-01-19").getTime());
CDOTransaction transaction= session.openTransaction();

// Use own ResourceSet
ResourceSet resourceSet = new ResourceSetImpl();
CDOTransaction transaction2= session.openTransaction(resourceSet);

// Associate transactions with a distributed transaction
CDOXATransaction xa = CDOUtil.createXATransaction();
xa.add(transaction.getViewSet());
xa.add(transaction2.getViewSet());
```
// Open CDO session and view
CDOSession session = openSession();
CDOView view = session.openView();

// Navigate through the resource folder structure
for (CDOResourceNode node : view.queryResources(null, "business", false))
{
    if (node instanceof CDOResourceFolder)
    {
        EList<CDOResourceNode> subNodes = ((CDOResourceFolder)node).getNodes();
    }
    else
    {
        EList<EObject> contents = ((CDOResource)node).getContents();
    }
}

// Close the session when done
session.close();
Explicit Locking

```java
CDOSession session = openSession();
CDOTransaction transaction = session.openTransaction();
transaction.setAutoReleaseLocksEnabled(true);

// Lock a single object for writing
CDOResource resource = transaction.getResource("/my/resource");
resource.cdoWriteLock().lock();

// Modify that object
resource.getContents().add(new Company());
resource.getContents().add(new Company());
resource.getContents().add(new Company());

// Commit atomically and release all locks
transaction.commit();
session.close();
```
Save Points

CDOSession session = openSession();
CDOTransaction transaction = session.openTransaction();

// Create and populate a resource
CDOResource resource = transaction.getOrCreateResource("/my/resource");
resource.getContents().add(new Customer());
resource.getContents().add(new Customer());

// Set save point, modify and rollback
CDOSavepoint savepoint = transaction.setSavepoint();
resource.getContents().add(new Supplier());
resource.getContents().add(new Supplier());
transaction.rollback(savepoint);

// Commit only the first changes (customers)
transaction.commit();
session.close();
Passive Updates

```java
CDOSession session = openSession();
CDOView view = session.openView();
CDOResource resource = view.getResource("/my/resource");

// Decouple from passive updates
session.options().setPassiveUpdateEnabled(false);
for (EObject object : resource.getContents())
{
    Company company = (Company)object;
    // Work with local model…
}

session.refresh();
// Work with refreshed model…

// Stay in sync from here
session.options().setPassiveUpdateEnabled(true);
```
Change Subscriptions

```java
CDOSession session = openSession();
CDOView view = session.openView();

// Subscribe to repository for CDO adapters
view.options().setChangeSubscriptionPolicy(CDOAdapterPolicy.CDO);

// Define your CDO adapter
class MyAdapter extends AdapterImpl implements CDOAdapter
{
    @Override
    public void notifyChanged(Notification msg)
    {
        System.out.println("Modified remotely: "+ msg.getNotifier());
    }
}

// Attach your adapter to any object to trigger a particular subscription
CDOResource resource = view.getResource("/my/resource");
resource.eAdapters().add(new MyAdapter());
```
CDOSession session = openSession();
CDOView view = session.openView();

// Create query
CDOQuery query = view.createQuery("SQL",
   "SELECT cdoid FROM Company WHERE name LIKE ${name}");
query.setParameter("name", "Foo%");
query.setMaxResults(35);

// Send query to server and iterate the result asynchronously
for (Iterator<Company> it = query.getResultAsync(Company.class); it.hasNext())
{
  Company company = it.next();
  System.out.println(company);
}

// Closing the view closes all open queries
session.close();
Offline mode
- Checkout
- Re-sync

Legacy model support
- Ecore
- UML
- 3rd party models

OCL server-side
- As common query language
- For commit validation

Model code server-side
- Custom data types (persistence)
- Operations (e.g. in OCL)

Integration
- GMF integration
- Workspace integration
- Team integration
- CDO Explorer