**Scale, Share and Store your Models with CDO 2.0**

Modellieren und generieren Sie Ihre Anwendungen bereits mit dem *Eclipse Modeling Framework* (EMF)? Großartig! Dann genießen Sie bereits automatisch generierte, modellbasierte Runtimes, strukturierte oder grafische Benutzeroberflächen, XML-Speicherung und so vieles mehr…

Aber was ist, wenn

* Ihre Modelle wachen und wachsen und irgendwann nicht mehr in den Speicher passen?
* Sie eine unternehmensgerechtere Persistenzform als XML-Serialisierung benötigen?
* Ihre Anwendung muss für mehrere Benutzer zur gleichen Zeit verfügbar sein?
* Sie benötigen objektgranulares Locking und Multi-Resource Transaktionen?
* Ihre Benutzer wollen jederzeit auf den neuesten Datenstand des Zentralmodells blicken?
* Sie Ihre Sicht auf das Modell einfach auf historische Stände zurückschrauben wollen?

Das *Connected Data Objects* (CDO) Model Repository und Distributed Shared Model Framework für EMF bietet Lösungen für diese Herausforderungen und mehr. CDO ist seit vielen Jahren eine Komponente des EMF Projekts und wurde seitdem von vielen Unternehmen und Organisationen erfolgreich integriert.

CDO erlaubt Ihnen, Ihre Anwendung gegen Standard EMF Schnittstellen zu entwickeln und erst später zu entscheiden, ob Sie Ihre Modelle in irgendeiner SQL Datenbank, möglicherweise durch Hibernate, in einer objektorientierten Datenbank oder lediglich im Server-Hauptspeicher abzulegen. Ohne Änderungen an Ihrer Anwendung werden Ihre Modelle sowohl auf Client-, wie auf Serverseite wirklich skalierbar. Speichern und benutzen Sie mehrere Gigabyte große Modelle und laden Sie bis 30.000 Objektepro Sekunde in Ihre Clients. Betten Sie Ihr Repository in Ihre Clients ein, wenn Sie weder Verteilung noch Mehrbenutzerbetrieb benötigen. Oder betten Sie es in einen J2E Container Ihrer Wahl ein, wenn Sie andere Verteilungsmechanismen benötigen. Lassen Sie Ihre Benutzer die Kollaboration auf einem wirklichen Distributed Shared Model erfahren.

**Inhalt**

**Scaling Up**

* Transparente Integration mit EMF
* On-Demand Laden enthaltener Objekte
* Unterstützung für Querreferenzen und externe Referenzen
* Automatische Garbage Collection aller unbenutzten Objekte
* Teilweises Laden von Collections
* Speichern von Change Deltas
* Adaptives Pre-Fetching

**Distributing und Sharing**

* Bedienung der CDO Schnittstellen
* Sessions und Fail-Over
* Views und Audits
* Transactions und Save Points
* Verteilte Transaktionen
* Explizites Locking
* Change Subscriptions
* Query Framework

**Repositories, Persistenz und mehr**

* Einen CDO Repository Server aufsetzen
* Den Transport Layer konfigurieren
* Authentifizierung und Autorisierung
* Den richtigen Backend-Typ wählen
* DB, Hibernate, Objectivity, MEM, …
* Cache Management
* Eingebettete Server

**Author**

Eike ist ein unabhängiger Berater mit aktuellem Fokus auf den Gebieten OSGi und Modeling. Insgesamt verfügt er über 25 Jahre Softwareentwicklungserfahrung. Mit seinem Beratungsunternehmen [ES-Computersysteme](http://www.esc-net.de) hat er seit 1991 dutzende erfolgreicher Kundenprojekte durchgeführt oder begleitet. Eike hat bei Eclipse die beiden Projekte [CDO Model Repository](http://wiki.eclipse.org/CDO) und [Net4j Signalling Platform](http://wiki.eclipse.org/Net4j) initiiert und leitet sie seit 2004 aktiv.

Besuchen Sie Eike's [Blog](http://thegordian.blogspot.com/) für mehr Informationen...