CDO 2.0 Preview

Power to the People
Appealing to a Broader Community
Lighter, Faster and Better
2.0 Themes

☞ **Power to the People**

Major improvements in the API and configuration will give clients considerably more control over the behaviour and performance of their repositories and sessions.

☞ **Appealing to a Broader Community**

The CDO project has a proven record of robustness and flexibility. However, in the past best results could only be achieved with EMF models specifically generated for the usage with CDO. The upcoming 2.0 release will be less invasive to the client models and applications by being better aligned with EMF and other standards.
2.0 Themes

§ Lighter, Faster and Better

The CDO project has a proven track record of quality, reliability, and extensibility but all these aspects can be continuously improved.

Existing and new clients often desire more functionality from the framework while long established clients generally desire mostly improvements in performance.

Often these two are at odds since additional extensibility often has a performance cost and any changes, even performance boosting ones, potentially impact reliability.
Every effort will be made to introduce new capabilities so that they have negligible impact on established clients based on the guiding principle: *If you don't use it, you don't pay for it.*

Because service and support has and always will take priority over new development work, development commitments and schedules are subject to arbitrary change, slippage, or even deferral.

Be sure to establish direct communication to ensure that any hard commitments are seen as that by the development team itself!
Power to the People

215688: Create save points
208689: Add resource queries
230832: Make remote invalidation configurable
233273: Develop Query mechanism
233277: Automatic attachment of dangling references
233314: CDOTransactionHandler.rollingbackTransaction()
233490: Change Subscriptions
241463: Configurable timeout in Connector.openChannel()
241464: Configurable timeouts in CDO read-access requests
243279: Add caching properties to CDOSessionConfiguration
244801: Export resources to XMI...
Power to the People

245944: Make MEMStore more configurable (list sizes)
247143: Add detach object handling to CDOTransactionHandler
247428: Prefix for EClass names in DBStore
247709: Override standard Teneo extensions for generating mapping
248017: Provide a CDOSession.setInitialReferenceChunkSize()
248325: Support CDOAudit.setTimeStamp()
249278: Interceptors at the server-side to run validation, filtering, etc.
249296: Provide userID for IChannel through IUserAware
249297: Provide userID for CDOProtocolSession through IUserAware
249536: Public view lock to protect clients against remote invalidation
249847: Store resources in a hierarchical structure
250064: CDOPackageRegistryImpl should offer delegation...
Appealing to a Broader Community

204890: Implement detach
248296: Manage CDOResource .isLoaded
226778: getURIFragment and getEObject
242349: Net4J PostgreSQL DBAdapter
246705: Support containment proxies
246844: Implement Resource.delete()
213402: Support external references
247226: Transparently support legacy models...
Lighter, Faster and Better

213369: Exception handling
216586: Optional pooling of store accessors
243287: Shortcut for getObject(object.cdoID())
243987: Decouple Signal from IChannel
244006: Decouple IConnector from IManagedContainer
245794: Provide memory sensitive revision cache
245947: Provide a two level revision cache
Lighter, Faster and Better

245950: Make TwoLevelRevisionCache the default
246454: Optimize transfer of Package URIs
246465: Optimize
   CDOTransactionImpl.analyzeNewPackages
247174: Remove uniqueResourceContents
247764: Optimize ModelUtil.getCDOFeature()
248062: Refactor CDOView.
   setLoadRevisionCollectionCollectionChunkSize
248997: Protect closed sessions and their views against illegal usage