Refactor Lockengine

Refactor Lockengine

Table of content:

1 Project Description

Student: **Wensheng Dou**
Mentor: **Ralf Joachim**

Since Java 5, Java has introduced the java.util.concurrent API , which supports flexible locking constructs, more concurrent mechanism, and many lock-free and thread-safe atomic data structure. But the Castor project has not supported java.util.concurrent API yet. In the Castor JDO module, the LockEngine implements the read/write lock, and use custom lock mechanism. So the LockEngine is very complicated and error-prone. The “Refactor Lock Engine” project will refactor the Castor to support the java.util.concurrent API, and make LockEngine more readable and maintainable, and repair some concurrency bugs.

2 The ideas

2.1 Refactor the ObjectLock class (CASTOR-3085)

The ObjectLock class uses custom mechanism to implement the read/write locks, and it is too complicated and error-prone. I’ll use the java.util.concurrent.locks.ReentrantReadWriteLock to refactor the ObjectLock class.

1. Refactor some variables’ names to make them more intuitive.
2. Use HashSet to simplify the LinkedTx.
3. Invetigate the meaning of the property _deleted, and then simplify the meaning of the property _deleted (CASTOR-3121)
4. Use ReentrantLock and Condition to replace the old wait() and notify(); (CASTOR-3122)

2.2 Refactor the LockEngine class

1. LockEngine takes too much responsibility, some functions should be separated.
   - Move LockEngine.TypeInfo from an inner to a normal class (CASTOR-3076)
   - Extract 'molder' from TypeInfo, LockEngine, OID, and move it into ClassMolderRegistry (CASTOR-3104)(CASTOR-3107)(CASTOR-3106)
   - Seperate cache handling which are both handled by TypeInfo and LockEngine (CASTOR-3105)
   - Refactor invalidate and exipre (CASTOR-3163)
2. LockEngine is too complicated, and there are some concurrency bugs which are difficult to repair, so it should be refactored.
   - Eliminate the synchronized on basemolder in in LockEngine.load() (CASTOR-3109)(CASTOR-3119)(CASTOR-3120)
     Why: Deadlock causes by sync block in LockEngine.load
     Solutions:
     (1) Change the oid's meaning, and make it unique for one instance (no matter extended or not). Then one instance just has one oid, just one lock. (CASTOR-3119)
     (2) Move registration of entity at TransactionContext to a later point in the process when we already know the correct class? (CASTOR-3120)(CASTOR-3140)(CASTOR-3141)
     (3) Just register the entity into TransactionConetxt once. (CASTOR-3148)
   - Eliminate double loading of extends hierarchy (CASTOR-3074)
     Why: When an extended object is loaded as the base object, the first loading will know what the concrete class is, and then the second loading will load the extended object. It is a performance problem.

Possible Solutions:
(1) When the concrete class is determined, the we return the right objects. (Accepted)
(2) Store the result fields when the first loading finishes, the second loading just molds the concrete instance. (Rejected)

2.3 Remove the EDU.oswego.cs.dl.util.concurrent library (CASTOR-3050)

Since Java 5, Java has introduced the java.util.concurrent API, which supports ReentrantReadWriteLock. But Castor has not used the java.util.concurrent API to improve the performance. The Castor should be compiled on JDK 1.5 (or higher). We should convert the Castor to use java.util.concurrent API, and benefit from the java.util.concurrent package's improvement.

2.4 Replace Hashtable and synchronized HashMap by ConcurrentHashMap to improve cache performance (CASTOR-2940)

In the org.castor.cache.* cache implementations Hashtable and synchronized HashMap will be used as underlying data structures. This issue will use ConcurrentHashMap, ReentrantReadWriteLock and so on to improve the cache performance.

1. Improve synchronization/locking at org.castor.cache.* (CASTOR-3154)(CASTOR-3155)(CASTOR-3156)(CASTOR-3158)(CASTOR-3170)
2. Add some multithreaded tests for org.castor.cache.hashbelt.* (CASTOR-3176)
3. Improve extends hierarchy of caches (CASTOR-3179)
4. Use generics at Cache interface (CASTOR-3180)(CASTOR-3186)

2.5 Some other bugs

1. ArrayIndexOutOfBoundsException occurs when loading an extended object (CASTOR-3065)
2. Delete and the insert with same id in the same transaction fails (CASTOR-1044)
3. Some inconsistent errors between pom.xml and .classpath (CASTOR-3051)
4. A maven building error : castor-xml has dependency on castor-codegen (CASTOR-3048)

---

3 Related JIRA tasks

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated	Due
	CASTOR-3186	Improve generics at Cache and CacheFactory	Wensheng Dou	Ralf Joachim		Resolved	Fixed	Aug 16, 2011	Aug 16, 2011	
	CASTOR-3185	Add full Javadoc to ObjectLock and TypeInfo	Wensheng Dou	Wensheng Dou		Resolved	Fixed	Aug 12, 2011	Aug 16, 2011	
	CASTOR-3180	Use generics at Cache interface	Wensheng Dou	Ralf Joachim		Resolved	Fixed	Aug 04, 2011	Aug 16, 2011	
	CASTOR-3179	Improve extends hierarchy of caches	Wensheng Dou	Ralf Joachim		Resolved	Fixed	Aug 04, 2011	Aug 10, 2011	
	CASTOR-3176	Add some multithreaded tests for org.castor.cache.hashbelt*	Wensheng Dou	Wensheng Dou		Resolved	Fixed	Aug 04, 2011	Aug 10, 2011	
	CASTOR-3175	Various cleanup at org.castor.cache.hashbelt.*	Wensheng Dou	Wensheng Dou		Resolved	Fixed	Aug 04, 2011	Aug 04, 2011	
	CASTOR-3170	Eliminate extra synchronization/locking at hashbelt's containers	Wensheng Dou	Wensheng Dou		Resolved	Fixed	Jul 30, 2011	Aug 04, 2011	
	CASTOR-3163	Refactor invalidate and exipre	Wensheng Dou	Ralf Joachim		Resolved	Won't Fix	Jul 28, 2011	Aug 16, 2011	
	CASTOR-3162	Various cleanup at ObjectLock	Wensheng Dou	Ralf Joachim		Resolved	Fixed	Jul 28, 2011	Jul 28, 2011	
	CASTOR-3159	Eliminate duplicated code for acquiring lock in ObjectLock and TypeInfo	Wensheng Dou	Wensheng Dou		Resolved	Fixed	Jul 27, 2011	Jul 27, 2011	
	CASTOR-3158	Improve synchronization/locking at hashbelt cache	Wensheng Dou	Wensheng Dou		Resolved	Fixed	Jul 26, 2011	Jul 28, 2011	
	CASTOR-3156	Improve synchronization/locking at	Wensheng	Ralf		Reso	Fixed	Jul 22,	Jul 27,	

	CountLimited cache	Dou	Joachim	lved		2011	2011
CASTOR-3155	Improve synchronization/locking at TimeLimited cache	Wensheng Dou	Ralf Joachim	Resolved	Fixed	Jul 22, 2011	Jul 27, 2011
CASTOR-3154	Replace Hashtable by ConcurrentHashMap at Unlimited cache	Wensheng Dou	Ralf Joachim	Resolved	Fixed	Jul 22, 2011	Jul 22, 2011
CASTOR-3150	Clean and correct some coments in ObjectLock	Wensheng Dou	Wensheng Dou	Resolved	Fixed	Jul 19, 2011	Jul 20, 2011
CASTOR-3148	Omit multiple initialization of objectInTx in LockEngine.load	Wensheng Dou	Wensheng Dou	Resolved	Fixed	Jul 17, 2011	Jul 17, 2011
CASTOR-3141	Improve exception handling in LockEngine.load	Wensheng Dou	Ralf Joachim	Resolved	Fixed	Jul 06, 2011	Jul 06, 2011
CASTOR-3140	Omit multiple calls to track/untrack in LockEngine.load	Wensheng Dou	Ralf Joachim	Resolved	Fixed	Jul 06, 2011	Jul 06, 2011
CASTOR-3136	Refactor cpactf/test04 through CPAThreadedTestCase and CPAThreadedTestRunnable	Wensheng Dou	Wensheng Dou	Resolved	Fixed	Jul 01, 2011	Jul 01, 2011
CASTOR-3132	Simplify the acquireUpdateLock method	Wensheng Dou	Wensheng Dou	Resolved	Fixed	Jun 29, 2011	Jul 04, 2011

Showing 20 out of 37 issues