CloudTran

CloudTran V2 – MVCC and Transactional Replication

Previously on Coherence we've supported TopLink Grid and a cache-level API with Pessimistic Locking. For the 2.0 release, we've added two major new features to CloudTran – while continuing to support the previous features.

First, MVCC (Multi-Version Concurrency Control) is added as an alternative to Pessimistic Locking on the cache-level API. It supports:

- consistent reads across multiple distributed cache entries or services. This effectively gives a snapshot, with all the values being consistent with the time the reading transaction started. For example, if you start a transaction at 9:01 and you want to read the price of an item which was updated at 9:00 and 9:02, you will get the 9:00 price – being the correct value as of 9:01.
- multiple simultaneous updates. If you want to calculate a new price and note it as "the current price" regardless of whether other threads are also calculating the price, then you can turn off update checking – so the price can be continuously updated, using a consistent snapshot of input values for each calculation.
- non-conflicting updates. You can specify write conflict checking – i.e. no intervening write can have occurred on any cache entry changed in a transaction. You can also do the same, but for any entry read by the transaction as well; this is the MVCC equivalent of pessimistic locking and prevents a "write skew anomaly".

Second, CloudTran 2.0 introduces transactional replication between data centers. The key here is "transactional" across multiple cache entries (in different partitions); if you don't need this, the Coherence push replicator or GoldenGate will make more sense depending on your system of reference.

By doing transactional replication, you know that the secondary grid will be consistent – updates are done atomically – so if the primary grid goes down, you can switch over to the secondary grid immediately the outage is detected.

One of the big issues in replication is the time to send to the other data center, which is typically 100ms or more. It doesn't make sense in a grid environment to wait that long, so what the CloudTran replicator does is to save the transactions to SSD at the sending data centre, and then relays them to the other grid while the transaction carries on locally.

Writing to SSDs only takes 13µs from our Java code, which is likely to be acceptable in most applications – to get the benefit of knowing the transaction will be replicated eventually (even if the replicator node or data centre goes down). The replicator supports failover and dual-ported SSD drives, avoiding a single point of failure – it's like a message appliance, but built on Coherence.