OptimisticConcurrencyControl

OptimisticConcurrencyControl

The RDB DAS is intended for use in disconnected scenarios. When the DAS returns a graph of SDO DataObjects as the result of a query, there are no locks held on the underlying database and the data is no longer associated with any database connection or transaction. However, although locks are not held, the DAS does employ a mechanism called Optimistic Concurrency Control (OCC) to manage concurrency.

Basically, when OCC is being utilized, the DAS checks whether the state of the database has changed since the original data was read before applying any writes to the same location. For example, suppose that a DAS client reads a set of Customers, modifies one Customer and then attempts to write the change back to the database:

```
Command select = das.createCommand("Select * from CUSTOMER where LASTNAME = 'Williams'");
DataObject root = select.executeQuery();

DataObject customer = (DataObject) root.get("CUSTOMER[1]");
customer.set("LASTNAME", "Pavick");

//Some period of time passes in which "CUSTOMER[1]" could be modified by another process or application

das.applyChanges(root);
```

If OCC is enabled then as part of "applyChanges" processing the DAS will ensure that the table row representing "CUSTOMER[1]" has not been changed since it was read. This is possible because a feature of the original SDO graph is to "remember" its original state along with any changes that have been made. The DAS has access to this original state and uses it to create an "overqualified" update statement. When the DAS executes this update statement against the database, it will fail if the underlying data has been modified (a collision has occurred) and, if so, an exception will be thrown and the transaction rolled back. If there is no collision then the update statement execution succeeds and the change is applied.