Market Requirements Document

Feature Name: Non-Quorum Updates

Version: 2 **Date Submitted:** 11/25/08 **Completed By:** Leon Guzenda
* Removes extraneous Appendix.
Version: 1 **Date Submitted:** 11/25/08 **Completed By:** Leon Guzenda

Description of the Problem

Background

Objectivity/HA implements synchronous database replication with a quorum scheme for allowing or disallowing updates to a replicated database. If a client can obtain N/2+1 votes (where N is the total allocated) it is allowed to update. A vote can be a weighted value assigned to a database replica or a value attributed to a lock server. Objectivity/HA also provides a transaction mode option that allows a client to view potentially stale data if the client is running outside of a quorum.

The basic philosophy is that database replicas may fall behind the synchronized quorum state, but replicas cannot be updated unless they are up to date and a quorum has been obtained. This avoids the need for content aware resynchronization mechanisms. If a logical group of data, e.g. Sales_Orders, has to be updateable by multiple, disconnected users then each of them has to have their own “private” database and a replica weight that gives them a quorum. A central replica has a vote of one, or less than that required for a quorum.

Federations can be logically subdivided into partitions. Each partition has its own lock server, a copy of the system data in the federated database (catalogs and schema), and one or more databases or replicas of a database.

Sometimes there are links in a network that are much slower than others. In this case it is convenient to mark such partitions as offline. They can be resynchronized when the load on the quorum replicas is lighter. There is also a means to force a particular group of replicas to become the quorum, but this must only be used with great caution and there must be an external mechanism, such as a processor heartbeat monitor, to ensure that the other replicas cannot be updated.

Some DBMSs, such as Sybase, DB2 and Oracle, allow uncoordinated updates and then put the onus on the user to resynchronize datasets. The exact mechanisms vary, but most of them depend on a notify and subscribe model.

Problems

1. Although the mechanism involving private databases works well, it forces application writers to deal with logical groups of databases, e.g. queries have to be directed to multiple databases rather than one. It is also still possible for two users to create different objects representing exactly the same item, e.g. a new customer name.
2. If a group of replicas is forcibly made the quorum and the other replicas are actually still being updated there is no automatic way to resynchronize the replicas correctly. It is impossible for Objectivity/DB to determine the policy to be used as it may vary according to the data types and exact nature of the transactions. For Example:
    a. In the case of a change to a bank account PIN the latest update should apply.
    b. A security application would want to record all changes to PINs.
    c. In the case of recording ATM withdrawals all updates should apply.
    d. If an item is deleted all later changes to replicas are void unless some external action needs to be rolled back.

Description of the Requested Feature

Objectivity/HA should provide these additional features:

1. A mechanism for assigning a property to a database that makes it possible to update replicas without requiring a quorum.
2. A mechanism for checking the version and date/time stamps of database replicas, recognizing discrepancies and invoking a user replaceable hook for synchronizing the contents of individual containers across all replicas. Note that this implies access to multiple versions of objects with the same OID, some of which may not be of the same type or usage as the original object. The mechanism may be added to the current resynchronization mechanism, be a separate tool, or both.
3. A prototype user replaceable component that can access each version/date/time variant of a container within the same transaction and reconcile the contents. This may involve creating new data, updating existing data or deleting data, or even the whole container. This component must only be allowed to run in a quorum situation.

An alternative solution would be to implement an operation log for each such database replica and provide the user with a means to selectively reapply the log. However, this would be a radical departure from the way that the current kernel works, so it is only mentioned here for completeness.

Part of an existing feature or does it require another feature, if so, which one?

- This feature should be a standard part of Objectivity/HA.

How is this problem being solved now, and why isn't that acceptable?

1. See “Problems” above.

What languages must support this capability?

- C++, Java, .Net for C# and Python in that order of priority.

Which platforms must be supported?

- Windows and Linux.
- Solaris, but only if there is sufficient demand.

Do any competitors already have this feature?

- Most RDBMSs have a replica synchronization mechanism.
- Birdstep uses the SyncML protocol.
- Db4objects has an object level synchronization mechanism.

Customers who require this feature

- Any application that has mobile or intermittently connected users, e.g. in the DoD or Intelligence Community and field workers.
- Fugro-Jason.

Revenue at risk, or which could be won

- There is no immediate risk, but there is a small probability that a user will accidentally create an unrecoverable situation by misusing the current quorum-forcing feature.
- This is a competitive feature in regard to db4objects.

When is this required?

- Post Release 10, preferably in late 2009.

Additional Notes

1. We will also need:

- Marketing collateral, including promotional material and updates to our web site.
- Amended technical publications and web based training material.
- New QA material to prove that the feature works and is interoperable with other platform and language combinations.

2. Licensing costs are to be determined.