Active/Active DB2 Clusters for HA and Scalability

Ariff Kassam

xkoto, Inc

Tuesday, May 9, 2006 • 2:30 p.m. – 3:40 p.m.

Platform: DB2 for Linux, Unix, Windows
Market Focus

Solution
GRIDIRON™ 1808 Database Load Balancer

Value Proposition
High Performance and Continuous Availability for Corporate Data on Low-Cost Systems

Target Markets
- OLTP
- Enterprise Linux
- Distributed/Grid Computing
- Commodity Platforms
- High Availability

Release History
v1.1 – released Dec. 2005

Operating Environment
- DB2 UDB v8
- SUSE/Red Hat/AIX/Solaris
- Intel x86/SPARC/POWER

Background
xkoto Inc. is a new venture-backed startup spun off from Halcyon Monitoring Solutions, an Infrastructure Management ISV in business since 1994.

IBM Business Partner
“The increase in scalability and performance offered by the combination of DB2 Express-C and xkoto’s GRIDIRON is impressive. Clients … are choosing GRIDIRON with DB2 because it delivers a strong competitive solution and effectively addresses the current and future state of their information infrastructure.”

Bob Picciano, Vice President IBM Data Servers
The Need for GRIDIRON: Data On Demand

Web and Application Tiers: typically have clustered, scaled out commodity systems with a web/app load balancer to distribute workload – this provides good availability & scalability – if performance is too slow, just add another server to the farm.

Data Tier: typically has a single SMP server (or two servers in an active/passive HA cluster), sized big enough to handle peak load.

xkoto’s goal:

Provide a lower cost solution to deliver scalable performance and continuous availability for data.
May 7–11, 2006
Tampa Convention Center
Tampa, Florida, USA

Database Clusters

IDUG® 2006
North America

GoFurther
Shared Nothing / Partitioned Data Clusters

Whenever the data can be partitioned (typically for Data Warehouses), database performance can potentially be improved by distributing mutually exclusive data among multiple servers.

With DB2, this is done using DPF (Data Partitioning Facility). Since data is distributed by single or multi-dimensional (MDC) cluster keys, performance gains result from increased parallelism and reduced resource contention.

Caveats:

1. Solutions like DPF require additional solutions to provide HA & disaster recovery support.
2. Schema changes (e.g. data shuffling) and application changes may be required to optimize performance.
3. OLTP environments typically are unsuitable for partitioning solutions.
Shared Storage / Non-partitioned Clusters

When data cannot be partitioned, high performance may be sought through a cluster of data servers using Shared Storage architecture. By distributing workload across multiple servers that share disk and memory, performance can potentially be increased.

**Oracle RAC** uses a Shared Storage model and a complex distributed memory cache among the servers to try to provide higher performance.

**Caveats:**

1. Requires a SAN and HBAs which are typically expensive.
2. Typically requires a separate solution for Disaster Recovery (e.g. Oracle Data Guard or EMC SRDF).
3. With Shared Storage, there is potential for a single point of failure and I/O performance bottlenecks.
Shared Nothing / Non-partitioned Clusters

When data cannot be partitioned, high performance and availability may also be sought through a cluster of data servers using a Shared Nothing architecture, whereby each database server in the cluster maintains its own complete copy of the database – *no memory or disk is shared between servers*. By distributing workload across multiple servers, performance is potentially increased. By sharing nothing, *data replication is built-in*.

**GRIDIRON** uses a Shared Nothing design to provide horizontal scalability, high availability and disaster recovery.

Caveats:

Shared Nothing architectures require multiple copies of data so storage costs and synchronization must be accounted for.
### Data Clustering Strategies – Summary

<table>
<thead>
<tr>
<th></th>
<th>Shared Nothing / Partition Data</th>
<th>Shared Storage / Non-partitioned</th>
<th>Shared Nothing / Non-partitioned</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Examples</strong></td>
<td>IBM DB2 DPF</td>
<td>Oracle RAC</td>
<td>xkoto GRIDIRON</td>
</tr>
<tr>
<td><strong>Sweet Spot</strong></td>
<td>Data Warehousing</td>
<td>OLTP</td>
<td>OLTP</td>
</tr>
<tr>
<td><strong>Architecture</strong></td>
<td>Data distributed among different servers by cluster key(s).</td>
<td>Shared storage and (sometimes) memory.</td>
<td>Independent copies of all data per server – no shared storage of any kind.</td>
</tr>
<tr>
<td><strong>Load Balancing</strong></td>
<td>Yes.</td>
<td>Yes.</td>
<td>Yes.</td>
</tr>
<tr>
<td><strong>Replication (native)</strong></td>
<td>No.</td>
<td>No (except via SAN architecture).</td>
<td>Yes.</td>
</tr>
<tr>
<td><strong>HA (native)</strong></td>
<td>No.</td>
<td>Yes (for data server only).</td>
<td>Yes.</td>
</tr>
<tr>
<td><strong>DR (native)</strong></td>
<td>No.</td>
<td>No.</td>
<td>Yes.</td>
</tr>
</tbody>
</table>

Most OLTP applications are not suitable for partitioning, so a Non-partitioned architecture must be selected.
Caveat:

The following Data Replication slides only apply to Shared Nothing / Non-partitioned architectures. As discussed, partition clusters (e.g. DB2 DPF) and shared storage clusters (e.g. Oracle RAC) do NOT provide replication out-of-the-box.
**Master/Slave + Synchronous**

<table>
<thead>
<tr>
<th>WHEN will the data be replicated</th>
<th>Synchronous</th>
<th>Asynchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHERE will the data be replicated from</td>
<td>Master/Slave</td>
<td>Multi-Master</td>
</tr>
</tbody>
</table>

**DESIGN** (e.g. IBM HADR, Oracle Data Guard in synchronous mode)

writes (inserts/updates/deletes) go to only **one** designated server (the “master”) and are replicated **synchronously** to the other servers (the “slaves”).

Reads can go to any server (not supported with HADR).

**ADVANTAGES**
Guaranteed replication across all servers. Guaranteed data availability across all servers.

**DISADVANTAGES**
Reduced performance since **all** servers must be committed before the application can continue.
Master/Slave + Asynchronous

<table>
<thead>
<tr>
<th>WHEN will the data be replicated</th>
<th>Synchronous</th>
<th>Asynchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHERE will the data be replicated from</td>
<td>Master/Slave</td>
<td>Multi-Master</td>
</tr>
</tbody>
</table>

**DESIGN** (e.g. IBM HADR in asynchronous mode, WebSphere II)

Writes (inserts/updates/deletes) go to only one server and are replicated asynchronously.

Reads can go to any server (not supported with HADR).

**ADVANTAGES**
Better performance than synchronous replication.

**DISADVANTAGES**
Potential unreliable data availability & read consistency (some writes can be missed if the master fails before the slave commits).
Multi-Master + Synchronous

<table>
<thead>
<tr>
<th>WHEN will the data be replicated</th>
<th>Synchronous</th>
<th>Asynchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHERE will the data be replicated from</td>
<td>Master/Slave</td>
<td>Multi-Master</td>
</tr>
</tbody>
</table>

DESIGN (e.g. Two Phase Commit)
Updates can go to any server and are replicated synchronously.
Reads can go to any server.

ADVANTAGES
Guaranteed read consistency across all servers.
Guaranteed data availability across all servers.

DISADVANTAGES
Complex - increased chance of failure.
Performance issues.
# SQL Replication vs. Log Shipping

<table>
<thead>
<tr>
<th>HOW will the data be replicated</th>
<th>SQL Replication (e.g. xkoto GRIDIRON)</th>
<th>Log Shipping (e.g. IBM DB2 HADR, WebSphere II, Oracle Data Guard)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advantages</td>
<td>Can take advantage of active/active configurations.</td>
<td>Good performance – logs are compact and complete data images.</td>
</tr>
<tr>
<td>Disadvantages</td>
<td>Constraints in a distributed environment – more complexity to maintain consistent data.</td>
<td>Must be used in a master/slave architecture because a transaction must commit before the log can be shipped to a replica server.</td>
</tr>
</tbody>
</table>
## Data Replication Strategies Summary

<table>
<thead>
<tr>
<th>WHEN will the data be replicated</th>
<th>Synchronous</th>
<th>Asynchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHERE will the data be replicated from</td>
<td>Master/Slave</td>
<td>Multi-Master</td>
</tr>
</tbody>
</table>

### Master/Slave

<table>
<thead>
<tr>
<th>Synchronous</th>
<th>Asynchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>(e.g. IBM HADR, Oracle Data Guard)</td>
<td>(e.g. Two Phase Commit)</td>
</tr>
<tr>
<td>Data consistency.</td>
<td>Complex.</td>
</tr>
<tr>
<td>Data availability.</td>
<td>Increased chance of failure.</td>
</tr>
<tr>
<td>Poor performance.</td>
<td>Poor performance.</td>
</tr>
</tbody>
</table>

### Asynchronous

<table>
<thead>
<tr>
<th>Synchronous</th>
<th>Asynchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td>(e.g. IBM HADR, Oracle Data Guard, Avokia ApLive)</td>
<td>(not recommended)</td>
</tr>
<tr>
<td>Unreliable data availability.</td>
<td>Requires manual reconciliation.</td>
</tr>
<tr>
<td>Reads of stale data.</td>
<td></td>
</tr>
</tbody>
</table>

GRIDIRON takes a different approach: not Master/Slave, not Multi-Master …
May 7–11, 2006
Tampa Convention Center
Tampa, Florida, USA

xkoto GRIDIRON

IDUG® 2006
North America

GoFurther
A New Model: xkoto GRIDIRON

Database Scalability & Availability

- Virtualized access for applications (GRIDIRON appears like a single database).
- Cluster of active/active data servers for high performance and availability.
- Guaranteed read consistency across all servers.
- Commodity server & Linux support.

The GRIDIRON approach: transactions are sent to ALL replicas at the same time (since the cluster is active/active, there’s NO “master” and replication is built-in), and the first server to successfully commit responds to the application (so all processing is asynchronous). Queries are load-balanced and only serialized behind transactions when required for consistency.

The result: high performance, high availability, lower cost.
GRIDIRON Features

- Intelligent SQL Load Balancing
- Horizontal Scalability
- Continuous Availability
- 100% Data Consistency
- Shared Nothing
- Asynchronous SQL Replication
- No Distance Limitations
- No Application Code Changes
GRIDIRON Architecture

- Application
  - JDBC driver
- Application
  - JDBC driver
- Application
  - JDBC driver
- Command shell
  - xdsql

GRIDIRON server

DB connector
- DB2
- DB2
- DB2
- DB2

JDBC driver/xdsql

active, identical & consistent databases
Applications only need to be configured to use the xkoto JDBC driver instead of the DB2 JDBC driver.

```
driver=xkoto.gridiron.sql.Driver
url=jdbc:xkoto://10.20.169.75:4444/petstore
username=db2
password=db2
```

The GRIDIRON cluster appears like a single database to an application – access is transparent. No special database privileges are required by GRIDIRON – it accesses data with the user’s credentials.
xdsqI Application Interface

SQL shell interface to the GRIDIRON database cluster

> xdsqI -d PetStore -u db2inst1 -P db2inst1 -h petit

xdsqI> create table out_of_stock (itemname varchar(32), inventory_no integer);

Query successful (0.033 seconds)

xdsqI>
GRIDIRON Server

The GRIDIRON Server is the heart of the Load Balancer.

It tracks the state of every table in each database server in the cluster; client requests are distributed based on type of statement and database state.

It maintains an integrity log which contains all transactions and additional information needed to restore a database in the case of failure.

Recommended Configuration

dual CPU x86 server
2 GB RAM
Gigabit Ethernet
40 GB disk
Linux Enterprise Server

Competitors typically implement their solutions in Java – GRIDIRON performs up to 2x faster because it uses C code.
DB Connector

Installed on every database server in the GRIDIRON cluster.

Receives statements from the GRIDIRON Server and applies them to each database via direct C API calls.

Increases performance and ensures data consistency.

Many competitive solutions take a “lowest common denominator” approach – they use tools like C-JDBC which do not tailor to a database vendor’s specific SQL extensions. The DB Connector embodies vendor-specific engineering.
Security Features

Connection operations can be encrypted for over-the-wire transmission.

GRIDIRON data integrity log is encrypted.

Separate username / password for GRIDIRON management tasks.
A Web GUI is provided for all administration

A command line utility is also provided – this is useful for automating actions
Write statements are broadcast to all database servers and processed asynchronously.

The response from the first database server to process the statement is sent back to the application.
Query Processing (“Reads”)

Read statements are load balanced and sent to the most up-to-date database server.

To maintain read consistency, GRIDIRON executes Virtual Read Statements on other database servers.
“Cluster Safe” Design

GRIDIRON maintains:

1. **Data Consistency** across the cluster
   a) nondeterministic functions
   b) row order
   c) automatically generated values

2. **Distributed Concurrency** across the cluster
   a) table dependencies
   b) internal updates

Many competitive solutions do NOT maintain 100% data consistency in all circumstances – user beware!
High Availability Configuration

An active/passive cluster of two GRIDIRON servers with the same virtual IP address is available as an HA option for GRIDIRON itself.

State-full failover with 1 second heartbeat.

5 – 10 sec. failover time.
GRIDIRON Failure Recovery Scenario

1. Application sends COMMIT.

2. GRIDIRON server fails after the statement is sent to all the database servers.

cont’d …
GRIDIRON Failure Recovery Scenario

3. GRIDIRON reestablishes connection and determines database state.

4. GRIDIRON will automatically replay the transaction on any database that did not process the commit.

...cont’d
Database Failure Recovery Scenario

1. Application sends a READ request.

2. Database server fails after receiving the READ request.

cont’d …
3. GRIDIRON instantaneously re-routes the READ to another database server transparently.

4. When the database server is back on line, GRIDIRON will reintegrate the server transparently.
GRIDIRON enables rolling upgrades for continuously available database services.

Database servers can be taken off-line for maintenance without stopping operations.

GRIDIRON enables rolling upgrades for continuously available database services.
One or more database servers can be placed in a geographically distant data center.
GRIDIRON Performance

TPC-W (e-commerce) benchmark (80% read mix)

89% average speed up
“Ideal” Workload / Customer Profile

• Application:
  • Java application
  • e-commerce web / OLTP application
  • small to medium size database (< 1 TB)

• Customer profile:
  • looking to reduce costs
  • interested in Linux
  • looking for active/active high availability solution
  • considering scale out architectures
  • architecting a new application
## Product Comparison

<table>
<thead>
<tr>
<th>Focus</th>
<th>IBM HADR</th>
<th>IBM DPF</th>
<th>IBM WebSphere II</th>
<th>xkoto GRIDIRON</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HA</td>
<td>Scale</td>
<td>HA</td>
<td>HA &amp; Scale</td>
</tr>
<tr>
<td>Architecture</td>
<td>active/passive</td>
<td>active/active</td>
<td>active/active</td>
<td>active/active</td>
</tr>
<tr>
<td>Data</td>
<td>Non-partitioned</td>
<td>Partitioned</td>
<td>Non-partitioned</td>
<td>Non-partitioned</td>
</tr>
<tr>
<td>Load Balancing</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Replication (native)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Replication Method</td>
<td>Log Shipping (instance level)</td>
<td>N/A</td>
<td>Log Shipping (table level)</td>
<td>SQL Replication (instance level)</td>
</tr>
<tr>
<td>HA (native)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>DR (native)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
GRIDIRON: Better Than Oracle RAC

xkoto GRIDIRON + DB2

✓ runs faster
GRIDIRON + DB2 provides better horizontal scalability then Oracle RAC

✓ costs less
GRIDIRON + DB2 is a fraction of the cost of Oracle RAC

✓ continuously available
GRIDIRON enables continuous availability for DB2
GRIDIRON: Better Than Other Vendor Solutions

xkoto GRIDIRON + DB2

✓ higher performance
GRIDIRON provides up to 89% scalability

✓ cluster safe
GRIDIRON ensures that data is stored and returned consistently across the cluster

✓ simpler license model
users are not taxed on the number of systems, CPUs, or applications
Session ####
Active/Active DB2 Clusters for HA and Scalability

Ariff Kassam
xkoto Inc.
ariff.kassam@xkoto.com