Emmanuel Cecchet

C-JDBC status

c-jdbc@objectweb.org
Outline

- Overview
- Advanced concepts
- Query caching
- Horizontal scalability
- Monitoring
- Perspectives
Motivations

- J2EE performance scalability bounded by database performance
- Database tier must be
  - scalable
  - fault tolerant (high availability + failover)
  - without modifying the client application
  - using open source databases
  - on commodity hardware
How do we want to use JOnAS?

- end-to-end open source solution for large scale J2EE clustered application servers
- cost-effective solution for both small and large configurations
Redundant Array of Inexpensive Databases

**RAIDb controller**
- gives the view of a single database to the client
- balance the load on the database backends

**RAIDb levels**
- RAIDb-0: full partitioning
- RAIDb-1: full mirroring
- RAIDb-2: partial replication
- composition possible
Middleware implementing RAIDb

Two components
- generic JDBC 2.0 driver (C-JDBC driver)
- C-JDBC Controller

C-JDBC Controller provides
- performance scalability
- high availability
- failover
- caching, logging, monitoring, …

Supports heterogeneous databases
Using JOnAS with C-JDBC

- Without C-JDBC
Using JOnAS with C-JDBC

- Update the datasource to use the C-JDBC driver

Diagram:
- JOnAS
- Virtual Database
- Cache
- Pooling
- C-JDBC
- JDBC Driver
- Backend
What C-JDBC offers

- No application modification
- No JOnAS modification
- No database modification

Diagram:
- JOnAS
- C-JDBC
- Virtual Database
- Driver
- Backend recovery
- Cache
- Pooling
- Load balancing
- JDBC Controller
- JDBC Driver

JOnAS developer workshop – http://www.objectweb.org - c-jdbc@objectweb.org
Outline

- Overview
- Advanced concepts
- Query caching
- Horizontal scalability
- Monitoring
- Perspectives
Inside the Controller

XML configuration file

C-JDBC Controller

Virtual database
- Authentication Manager
- Request Manager
  - Scheduler
  - Request Cache
  - Load balancer

Database Backend
- Connection Manager
  - MySQL JDBC driver
  - Oracle JDBC driver

MySQL
Oracle

Administration console

JMX

Java client program (Servlet, EJB, ...)

Sockets

RMI

Recovery
Log

Authentication Manager

Request Cache

Load balancer

Request Manager

MySQL
Oracle

JMX
Fault tolerant recovery log

C-JDBC Controller
- Checkpointing service
- Recovery virtual database
  - Authentication Manager
  - Request Manager
    - Scheduler
    - Load balancer
  - Database Backend
    - MySQL

Application virtual database
- Authentication Manager
- Request Manager
  - Recovery Log
  - C-JDBC Driver
  - MySQL
- Scheduler
- Request Cache
- Load balancer
- MySQL
- MySQL
- MySQL
- MySQL

Client application
(Servlet, EJB, ...)

MySQL

JOnAS developer workshop – http://www.objectweb.org - c-jdbc@objectweb.org

12 - 23/02/2004
Cascading controller
Heterogeneity support

- application already written for a specific [commercial] database
- user defined rules for on-the-fly query rewriting to execute on heterogeneous backends

Diagram:
- C-JDBC Controller (RAIDb-2)
  - Java client program
  - Servlet container: Tomcat, Jetty, ...
  - EJB Container: JOnAS, WebLogic, JBoss, WebSphere, ...
- C-JDBC driver
  - JVM
- MySQL JDBC driver
  - MySQL
  - MySQL
  - MySQL
- Oracle JDBC driver
  - Oracle
- C-JDBC Controller
  - RAIDb-2
Outline

- Overview
- Advanced concepts
- Query caching
- Horizontal scalability
- Monitoring
- Perspectives
Query caching status

- Cache contains a list of SQL->ResultSet
- Policy defined by queryPattern->Policy
- 3 policies
  - EagerCaching: variable granularities for invalidations
  - RelaxedCaching: invalidations based on timeout
  - NoCaching: never cached

<table>
<thead>
<tr>
<th>RUBiS bidding mix with 450 clients</th>
<th>No cache</th>
<th>Coherent cache</th>
<th>Relaxed cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput (rq/min)</td>
<td>3892</td>
<td>4184</td>
<td>4215</td>
</tr>
<tr>
<td>Avg response time</td>
<td>801 ms</td>
<td>284 ms</td>
<td>134 ms</td>
</tr>
<tr>
<td>Database CPU load</td>
<td>100%</td>
<td>85%</td>
<td>20%</td>
</tr>
<tr>
<td>C-JDBC CPU load</td>
<td>-</td>
<td>15%</td>
<td>7%</td>
</tr>
</tbody>
</table>
RUBiS - Tomcat without C-JDBC caching

Throughput: 3900 pages/min

- 1 Database: 100% cpu
- Tomcat: ~50% cpu

Processor usage

Time in seconds
RUBiS - Tomcat with C-JDBC relaxed caching

Throughput: 4200 pages/min

Tomcat ~55% cpu
1 Database ~20% cpu
C-JDBC <10% cpu
## JOnAS and C-JDBC query caching

<table>
<thead>
<tr>
<th>RUBiS</th>
<th>Throughput (req/min)</th>
<th>Response time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servlet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No cache</td>
<td>3890</td>
<td>744</td>
</tr>
<tr>
<td>C JDBC</td>
<td>4140</td>
<td></td>
</tr>
<tr>
<td>JOnAS SF CMP 1.1</td>
<td>2947</td>
<td>2929</td>
</tr>
<tr>
<td>No cache</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C JDBC</td>
<td>3418</td>
<td></td>
</tr>
<tr>
<td>JOnAS CMP 2.0</td>
<td>2178</td>
<td>6372</td>
</tr>
<tr>
<td>No cache</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C JDBC</td>
<td>2450</td>
<td></td>
</tr>
</tbody>
</table>
Outline

- Overview
- Advanced concepts
- Query caching
- Horizontal scalability
- Monitoring
- Perspectives
Horizontal scalability

- JGroups for controller synchronization
- Groups messages for writes only
Horizontal scalability

Centralized write approach issues

- Centralized write approach issues

  ![Diagram of Centralized Write Approach Issues]

Issues with transactions assigned to connections

- Issues with transactions assigned to connections

  ![Diagram of Issues with Transactions Assigned to Connections]
Horizontal scalability

- General case for a write query
  - 3 multicast + 2n unicast

```
DistributedVirtualDatabase@controller1
execWriteRequest(AbstractWriteRequest) -> return result
DistributedRequestManager
Scheduler
Distributed scheduling handler
Send request to all controllers
Wait for schedulers replies
Distributed load balancer handler
Send execute request order to all controllers
Wait for load balancer replies
Notify controllers of completion success or not
Return result

DistributedVirtualDatabase@controller2
DistributedRequestManager
Distributed load balancer handler
Load Balancer
Notify scheduler of completion
Scheduler
Distributed scheduling handler

DistributedVirtualDatabase@controller3
DistributedRequestManager
Distributed load balancer handler
Load Balancer
Notify scheduler of completion
Scheduler
Distributed scheduling handler
```
Horizontal scalability

Solution: No backend sharing
- 1 multicast + n unicast [+ 1 multicast]
Horizontal scalability

Issues with JGroups
- resources needed by a channel
- instability of throughput with UDP
- performance scalability

TCP better than UDP but
- unable to disable reliability on top of TCP
- unable to disable garbage collection
- ordering implementation is sub-optimal

Need for a new group communication layer optimized for cluster
Horizontal scalability

- JGroups performance on UDP/FastEthernet
Outline

- Overview
- Advanced concepts
- Query caching
- Horizontal scalability
- Monitoring
- Perspectives
Design choices

- JMX for both administration and monitoring
  - JMX server is MX4J
- HTTP console for basic features but not suited for monitoring
- Swing console for monitoring
  - uses JMX/RMI connector
- Currently working on a scalable monitoring infrastructure
Online Monitoring

- Cache Hits Ratio
- Cache Entries Count
- Controller Used Memory

JOnAS developer workshop – http://www.objectweb.org - c-jdbc@objectweb.org
Viewing logs remotely
Outline

- Overview
- Advanced concepts
- Query caching
- Horizontal scalability
- Monitoring
- Perspectives
Current status

- C-JDBC 1.0 rc2 release
- Web site: >100,000 hits per month
- Downloads: >1200 per month
- Mailing lists: 134 subscribers

- Ready for production environments
Ongoing work

- **Scalable group communication middleware for clusters**
  - Needed for horizontal scalability
  - Decouple application streams from network channels

- **Monitoring**
  - Continuous monitoring stored in repositories
  - Generic monitoring console for both online monitoring and monitoring history browsing

- **Administration**
  - Fully featured administration console
  - Graphical configuration and deployment of centralized/distributed backends and controllers (offline/online)
  - Dynamic reconfiguration

- **Misc.**
  - Automated Load testing
  - RPM packaging (JPackage)
Questions & Answers

Thanks to all users and contributors ...

http://c-jdbc.objectweb.org
Virtual Database

- gives the view of a single database
- establishes the mapping between the database name used by the application and the backend specific settings
- backends can be added and removed dynamically
- configured using an XML configuration file
TPC-W Performance
(Amazon.com)

Throughput in requests per minute

Number of nodes

- Single DB
- RAIDb-0
- RAIDb-1
- RAIDb-2
Outline

- Overview
- Fault tolerance
- Performance
- New features
- Roadmap
Octopus is an ETL tool
Use Octopus to store a dump of the initial database state

Octopus

dump for initial checkpoint

EJB Container
JOnAS, WebLogic, JBoss, WebSphere, ...

C-JDBC driver

JVM
disabled

c-JDBC Controller

Recovery Log

PostgreSQL JDBC driver

PostgreSQL
disabled
Journaling

- Backend is enabled
- All database updates are logged
  (SQL statement, user, transaction, ...)

Diagram:
- JDBC Recovery Log
- C-JDBC Controller
- Octopus
- PostgreSQL JDBC driver
- PostgreSQL

Remarks:
- Recovery Log enabled
- Dump for initial checkpoint
- EJB Container enabled
- JOnAS, WebLogic, JBoss, WebSphere, ...

JOnAS developer workshop – http://www.objectweb.org - c-jdbc@objectweb.org
Adding backend on the fly

» Add new backends while system online

» Restore dump corresponding to initial checkpoint with Octopus

EJB Container
JOnAS, WebLogic, JBoss, WebSphere, ...

C-JDBC driver
JVM

Enabled

C-JDBC Controller

Recovery Log

Octopus

dump for initial checkpoint

JDBC Recovery Log

PostgreSQL JDBC driver

PostgreSQL

disabled enabled disabled

JOnAS developer workshop – http://www.objectweb.org - c-jdbc@objectweb.org
Synchronizing backends

ษา Replay updates from the log

Octopus

dump for initial checkpoint

JDBC Recovery Log

C-JDBC Controller

Recovery Log

PostgreSQL JDBC driver

PostgreSQL

disabled

enabled

disabled

EJB Container

JOnAS, WebLogic, JBoss, WebSphere, ...

C-JDBC driver

JVM

enabled

PostgreSQL

disabled

enabled

disabled
Expanded Cluster

Enable backends when done

C-JDBC Controller

EJB Container
JOnAS, WebLogic, JBoss, WebSphere, ...

C-JDBC driver
JVM

Recovery Log

PostgreSQL JDBC driver

Octopus

dump for initial checkpoint

PostgreSQL enabled

PostgreSQL enabled

PostgreSQL enabled

JOnAS developer workshop – http://www.objectweb.org - c-jdbc@objectweb.org
Handling a backend failure

A node fails!
Automatically disabled but should be fixed or changed by administrator

Octopus

JDBC Recovery Log

C-JDBC Controller

PostgreSQL JDBC driver

EJB Container
JOnAS, WebLogic, JBoss, WebSphere, ...

C-JDBC driver

JVM

enabled

dump for initial checkpoint

dump for last checkpoint

dump for last checkpoint

dump for initial checkpoint

PostgreSQL
disabled
enabled
enabled

JOnAS developer workshop – http://www.objectweb.org - c-jdbc@objectweb.org
Restore latest dump with Octopus

- C-JDBC Controller
  - JDBC Recovery Log
  - PostgreSQL JDBC driver
  - EJB Container
    - JOnAS, WebLogic, JBoss, WebSphere, ...
  - C-JDBC driver
  - JVM

- Octopus
  - Recovery Log
  - PostgreSQL JDBC driver
    - PostgreSQL
      - enabled
      - disabled
      - enabled
Re-synchronization

→ Replay missing updates from log
Healed Cluster

Re-enable backend when done

C-JDBC Controller

EJB Container
JOnAS, WebLogic, JBoss, WebSphere, ...

JVM

enabled

C-JDBC driver

JDBC Recovery Log

PostgreSQL JDBC driver

Octopus

...dump for initial checkpoint...

dump for last checkpoint

dump for last checkpoint

dump for last checkpoint

dump for initial checkpoint

PostgreSQL enabled
PostgreSQL enabled
PostgreSQL enabled
C-JDBC Management Framework

Shared design
Making new checkpoints

-Disable one backend to have a coherent snapshot
-Mark the new checkpoint entry in the log
-Use Octopus to store the dump
Making new checkpoints

- Replay missing updates from log

C-JDBC Controller

JDBC Recovery Log

EJB Container
JOnAS, WebLogic, JBoss, WebSphere, ...

C-JDBC driver

JVM

enabled

Recovery Log

PostgreSQL JDBC driver

- PostgreSQL enabled
- PostgreSQL enabled
- PostgreSQL enabled

dump for last checkpoint

dump for initial checkpoint

Octopus
Making new checkpoints

- Re-enable backend when done

1. C-JDBC Controller
2. PostgreSQL JDBC driver
3. JVM
4. EJB Container
5. JOnAS, WebLogic, JBoss, WebSphere, ...

- Recovery Log
- Recovery Log
- Recovery Log
- PostgreSQL JDBC driver
- PostgreSQL enabled
- PostgreSQL enabled
- PostgreSQL enabled

dump for initial checkpoint
... dump for last checkpoint
... dump for last checkpoint
... dump for initial checkpoint

Octopus