Using XA with Rdb
(DECdtm XA Gateway)

John Howard
Oracle New England Development Center

Copyright 2001 Oracle Corporation
Agenda

• Background
  – ACID 101 (Distributed Transactions)
  – DECdtm
  – XA Standard

• DECdtm XA Project
  – Summary
  – XA Gateway

• Future Directions
Distributed Transactions

• Transactions ensure database integrity through ACID:
  – Atomic – all changes associated with the transaction are a single unit of work
  – Consistent – The whole transaction succeeds or fails together.
  – Isolated – The “work in progress” doesn’t show up outside the transaction until it is committed
  – Durable – Committed transactions are permanent
Distributed Transactions (cont.)

- **Classic Example is Bank Transaction**
  - Atomic - Money is withdrawn from Account A and put in Account B in a single transaction which is either committed or rolled back as a unit
  - Consistent - If the transaction fails, the money won’t disappear or double
  - Isolated - Queries against the account balances don’t show the withdrawal from Account A until commit
  - Durable - Once committed, the transaction is guaranteed to be reflected in the database even through system failure and recovery.
Distributed Transactions (cont.)

• A DBMS provides transactions for a single database.
• But what if the two accounts are in two databases possibly on different systems on different continents?
• Transaction processing need to be distributed to provide an ACID transaction across all the resources and nodes involved.
Distributed Transactions (cont.)

• Distributed Transactions require an external Transaction Manager (TM) to coordinate the various Resource Managers (RMs), e.g. database managers, participating in the transaction

• The TM uses a Two Phase Commit (2PC) protocol to ensure transaction integrity

• A Transaction Processing Manager (TPM) such as ACMS or BEA Tuxedo™ uses a TM to coordinate transactions
Two Phase Commit (2PC) Protocol

- A 2PC allows the TM to guarantee ACID transactions across multiple RMs
- 2PC Protocols include these ingredients:
  - Start transaction
  - Prepare to commit (this is Phase I of the commit)
  - Commit (this is Phase II of the commit)
  - Rollback
  - Recovery
Two Phase Commit (2PC) Protocol

Application Program (AP) communicates with Transaction Manager (TM) for transaction services.

Start Trans Request

Start Trans

AP communicates with TM for transaction services.
Two Phase Commit (2PC) Protocol

Transaction Manager (TM)

Resource Manager (RM)

Resource

Application Program (AP)

Resource Manager (RM)

Resource

Retrievals & Updates

AP communicates with RMs directly to do the work of the transaction
Two Phase Commit (2PC) Protocol

- **Transaction Manager (TM)**
- **Resource Manager (RM)**
- **Application Program (AP)**
- **Resource**

**Process Flow:**
- **Prepare (Phase I):** Transaction Manager (TM) sends a prepare request to the Resource Manager (RM).
- **Commit Request:** Application Program (AP) requests a commit.
- **Commit:** Resource Manager (RM) commits the transaction and sends a commit request to the Transaction Manager (TM).

**Diagram Elements:**
- Arrows indicate the flow of requests and responses.
- **Prepare (Phase I)** and **Commit Request** are explicitly labeled.
Two Phase Commit (2PC) Protocol

- **Transaction Manager (TM)**
  - **Application Program (AP)**
  - **Commit Request**
  - **Vote to Commit** *

- **Resource Manager (RM)**
  - **Resource Manager (RM)**
  - **Vote to Commit** *

*Any “no” votes result in transaction rollback for both RMs*
Two Phase Commit (2PC) Protocol

- **Transaction Manager (TM)**
- **Application Program (AP)**
- **Resource Manager (RM)**
- **Resource**

Flow:
- **Commit Request** from AP to TM
- **Commit (Phase II)** from TM to RM
- RM to Resource
DECdtm

- DECdtm is a Distributed Transaction Manager (TM) for OpenVMS
- Implements a Two-Phased Commit (2PC) protocol interface
- Used by many OpenVMS native products as the TM:
  - Oracle Rdb (an RM)
  - Oracle CODASYL DBMS (an RM)
  - RMS Journaling (an RM)
  - ACMS (a TPM)
XA Distributed Transaction Processing Standard

- XA is a standard for Distributed Transaction Processing (DTP) from the X/Open Group
  - Specifies interfaces for 2PC services
  - Specifies behavior that must be provided by TMs and RMs
- Many vendors provide XA-compliant TMs and RMs
  - Oracle8/8i/9i
  - BEA Tuxedo and WebLogic 6.0 Application Servers
- XA is commonly used for cross-platform, cross-OS DTP including OpenVMS
- XA is the standard underneath J2EE transactions
DECdtm XA Interface Project

- Customers have wanted to use components compliant with X/Open’s XA 2PC protocol
- Compaq has developed software which allows DECdtm-compliant components to participate with XA-compliant components
  - DECdtm XA Veneer puts an XA RM in a DECdtm-managed transaction
  - DECdtm Wrapper lets an Application Program talk to DECdtm via the XA Transaction Demarcation (TX) I/F
  - DECdtm XA Gateway allows DECdtm to be an XA-compliant RM
DECDtm XA Veneer

ACMS EXC (TPM)

DECDtm (TM)

Oracle9i (RM)

Oracle Rdb (RM)

ACMS Task Shell

Application Code (AP)

Transaction Interface
DECDtm XA Gateway

- **Tuxedo Monitor (TPM)**
- **Tuxedo TM**
- **Application Server (AP)**
- **Oracle9i (RM)**
- **Oracle Rdb (RM)**
- **DECdtm (RM a.k.a. Sub-TM)**
- **XA Gateway**
So What?

• An Oracle Rdb database can now be integrated into an application using XA transactions
  – BEA Tuxedo applications
  – BEA WebLogic applications

• Oracle Rdb can participate in cross-platform, cross-OS XA transactions
DEMONSTRATION

BEA Tuxedo with Oracle Rdb
DECdtm Interface Project Release

• Compaq has a downloadable SDK for the DECdtm Interface components
  – OpenVMS 7.2 or higher

• Compaq is looking for field test participants
  – Called the DECdtm v2.0 Field Test
  – Contact Rick McLaughlin for more information
    email: Rick.McLaughlin@Compaq.com
    phone: (603) 884-0992

• Will be released with OpenVMS 7.3 (Ruby)
  – Called DECdtm v2.0
  – Standard part of OpenVMS, no additional licensing
Considerations

• **Oracle Rdb doesn’t fully support tightly coupled transaction contexts**
  - You can’t insert a row in one AP and see the uncommitted row in another AP even if both are in the same transaction
  - You can’t modify a row in one AP and see the uncommitted modification in another AP even if both are in the same transaction

• **All testing was done with Oracle Rdb7.1**
  - Dispatch layer of Oracle Rdb (where this work is done) is virtually identical between 7.0 and 7.1
  - No code changes to Oracle Rdb were needed

• **Tuxedo insists on IEEE floats**
  - Not available with Oracle Rdb
  - Being added for Module Language and Precompiled SQL and will be available post 7.1.0.1
Possible Future Directions

• Integration into SQL Services
  – Use the DECdtm XA Gateway on Oracle Rdb end of the Oracle DB Link
  – Would allow multiple Oracle Rdb databases in 2PC
• Support of Tightly Coupled Transactions
• Integration with J2EE
• Direct integration with Oracle Application Server
For More Information

- www.openvms.compaq.com/commercial/decdtm/index.html
- John.Howard@oracle.com
- Rick.McLaughlin@compaq.com
- www.oracle.com/rdb
- www.openvms.compaq.com
QUESTIONS & ANSWERS