Introducing Oracle Queuing/Messaging Technology

Anthony D. Noriega
MSCS, MBA, BSSE, OCP-DBA
Objectives

- Emphasize technical concepts and Oracle queuing infrastructure technology.
- Highlight programming techniques, methodologies, and relevant architectures.
- Discuss the relevant of queuing business models in various industries, with emphasis on industry and business requirements.
Objectives

Emphasize the reliability of interprocess communication imposed by AQ technology.

Position Oracle as the de facto leader in the integrated database/advanced message queuing market.
Technical Concepts

- Queue (FIFO data structure)
- Message Queuing
- Header
- Payload
- Channel
- Port
- Propagation
Technical Concepts

- Producer (enqueueing)
- Consumer (dequeuing)
- Recipient
- Enqueue
- Dequeue
Technical Concepts

- Peer-to-Peer Mode
- Publish/Subscribe Mode
  - Broadcasting
  - Multicasting
- Streams AQ
- Model View Controller (Message-Driven Beans)
Business Models Concepts

- Message Queuing
- Data Replication (Logical Change Record)
- Data Protection
- Data Warehouse Loading
- Event Management and Notification
- Workflow
- Serializable Distributed Processing
Business Concepts

- Information Integration
- Automation and Business Event Management
- Message Queuing in SCM, ERP, CRM
- Data Protection and Information Hiding
Queuing Infrastructure

Oracle Applications

Oracle Integration Server Gateway

Queues

PeopleSoft
Siebel

eBusinessSuite

Oracle Net

Oracle Net

MQ Series Applications

Websphere MQ Manager

SOA Framework
Types of Oracle Queues

Based on Producer/Consumer Cardinality
- Peer-to-Peer (P2P) Mode
- Publish/Subscribe Mode

Based on Persistency
- Persistent
- Non-Persistent
Types of Oracle Queues

Based on Enqueue/Dequeue Capabilities

- Normal
- Exception
Types of Oracle Queues

Based on Payload Data Type
- ANYDATA
- RAW
- LOB
- XML

On Based Transaction Type
- Transactional
- Non-Transactional
An application can enqueue messages that represent events into a queue explicitly, or a Streams capture process can capture database events and encapsulate them into messages called LCRs. These captured messages can be the results of DML or DDL changes. Propagations can propagate messages in a stream through multiple queues. Finally, a user application can dequeue messages explicitly, or a Streams apply process can dequeue messages implicitly. An apply process can reenqueue these messages explicitly into the same queue or a different queue if necessary. “
AQ Downstream Model

- Capture
- Propagation
- Staging and Propagation
- Enqueue
- Consumption
- Dequeue
Streams AQ Capabilities

- Automatic Shared Memory Management of the Streams Pool
- Streams Tool in Oracle Enterprise Manager
- Procedures for Starting and Stopping Propagations
- Queue-to-Queue Propagations
- Declarative Rule-Based Transformations
- Commit-Time Queues
- Supplemental Logging Enabled During Preparation for Instantiation
- Configurable Transaction Spill Threshold for Apply Processes
- Conversion of LCRs to and from XML
- Retrying an Error Transaction with a User Procedure
- Enhanced Support for Index-Organized Tables
- Row LCR Execution Enhancements
- Information About Oldest Transaction in V$STREAMS_APPLY_READER
Architectural Considerations

- Oracle integration server
- Agent
- Queue table
- Queuing processes
- Listener configuration
- Database links
- Message-Oriented Middleware (MOM)
Architectural Considerations

- Server Process 1
- Server Process 2
- Background Processes

- PGA

- System Global Area (SGA)
  - Java Pool
  - Shared Pool
  - Streams Pool
  - Large Pool

- Database Buffer Cache
  - Default
  - 2k
  - 4k
  - 16k
  - recycle
  - keep

- Oracle Process
- Server Process 3
Security Framework

- Rule-based Security
  - Object Level
  - Schema
  - Global
- Virtual Private Support
Security Framework

Enhancements:
- Database Volt
- LDAP Support
- XA Support
- Encryption Support via asymmetric authentication (PKI)
Planning the AQ Environment

- Oracle-based only or third-party, e.g., gateway-based or heterogeneous system involvement
- Transactional or non-transactional queue
- Peer-to-peer or Publish-Subscribe Mode (Broadcasting or Multicasting, i.e., custom recipient)
Planning the AQ Environment

- Propagation-type (queue-to-queue or queue-to-database link)
- Payload format
- Sending Mechanism (Producer Application)
- Receiving Mechanism (Consumer Application)
Planning the AQ Environment

- **Model View Controller**
  - Data Source, domain, data model
  - Presentation
  - Controller/ Mediator

- Payload content management for intelligent rule-based filtering or routing
Planning the AQ Environment

- Queue browsing without consumption
- Queue consumption and removal
- Queue consumption without payload removal for auditing, non-repudiation, or logging.
System Requirements

Configuring:

- PATH, LD_LIBRARY_PATH or equivalent platform parameters, as needed
- Streams pool instance and/or aq_tm_processes (9i only) > 0.
System Requirements

Configuring:

- Extproc listener
- Messaging Gateway, if applicable, involves gateway software installation and packages, including messaging home.
Configuring:

- Create AQ user and administrator with appropriate privileges, namely, `AQ_USER_ROLE` and `AQ_ADMINISTRATOR_ROLE`.
- Database links accordingly.
- Heterogeneous Services, if applicable (involved package and instance configuration).
**Software Requirements**

- **Certified OS Platform**
- **Oracle Streams AQ**
- **Oracle Streams AQ Gateway**
  - Procedural Gateway (Websphere MQ/Tibco)
  - Transparent Gateway (SQL Server)
- **Heterogeneous Services Gateway**
- **Configure Gateway homes with API provided**
Initialization Parameters

Oracle9i

- \texttt{Aq_tm_processes=\textit{n}}, \(1 \leq \textit{n} \leq 10\)
- \texttt{Qmnc}, master process
- \texttt{Qxxx}, spawned slave processes
- \texttt{=0}, then no queue monitoring
Initialization Parameters

Oracle10g

- Streams_pool_size, configured with dynamic memory management. Recommended default setting about 10% of shared_pool_size parameter.
Installation and Configuration

Information Integration Components

Install

Copying files for *Oracle Ultra Search Extension for EMD 9.0.4.0.0* 
Setup pending...
Configuration pending...

extracting file 'prev_off.gif' from 'C:\oracle\product\10.1.0\enm\adr\1\syms\man\web...'

You can find a log of this install session at:
C:\Program Files\Oracle\Inventory\logs\InstallActions2005-08-18_03-33-27PM.log

Manage from Anywhere, Anytime
- Web-based management, using Enterprise Manager console
- Mobile device-based management, using EM2Go

Oracle Universal Installer: Install

Stop installation...

Help | Installed Products... | Back | Next | Install | Cancel
Data Dictionary Views

```sql
anthony@adm>SELECT owner, name, queue_table, queue_type, retention, enqueue_enabled, dequeue_enabled, network_name
FROM dba_queues
ORDER BY 1,2,3
/
```

<table>
<thead>
<tr>
<th>OWNER</th>
<th>NAME</th>
<th>QUEUE_TABLE</th>
<th>QUEUE_TYPE</th>
<th>RETENTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS</td>
<td>ALERT_QE</td>
<td>ALERT_QT</td>
<td>NORMAL_QUEUE</td>
<td>0</td>
</tr>
<tr>
<td>SYS</td>
<td>AQ$_ALERT_QT_E</td>
<td>ALERT_QT</td>
<td>EXCEPTION_QUEUE</td>
<td>0</td>
</tr>
<tr>
<td>SYS</td>
<td>AQ$_AQ$_MEM_MC_E</td>
<td>AQ$_MEM_MC</td>
<td>EXCEPTION_QUEUE</td>
<td>0</td>
</tr>
<tr>
<td>SYS</td>
<td>AQ$_AQ_EVENT_TABLE_E</td>
<td>AQ_EVENT_TABLE</td>
<td>EXCEPTION_QUEUE</td>
<td>0</td>
</tr>
<tr>
<td>SYS</td>
<td>AQ$_AQ_SRVNTFN_TABLE_E</td>
<td>AQ_SRVNTFN_TABLE</td>
<td>EXCEPTION_QUEUE</td>
<td>0</td>
</tr>
</tbody>
</table>
Heterogeneous Productivity
### PL/SQL Supplied Packages

<table>
<thead>
<tr>
<th>ORACLE10g</th>
<th>ORACLE9i</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBMS_APPLY_ADM</td>
<td>DBMS_AQ</td>
</tr>
<tr>
<td>DBMS_AQ</td>
<td>DBMS_AQADM</td>
</tr>
<tr>
<td>DBMS_AQADM</td>
<td>DBMS_AQELM</td>
</tr>
<tr>
<td>DBMS_AQELM</td>
<td>DBMS_MGWADM</td>
</tr>
<tr>
<td>DBMS_AQIN</td>
<td>DBMS_MGWMSG</td>
</tr>
<tr>
<td>DBMS_CAPTURE_ADM</td>
<td>DBMS_FLASHBACK</td>
</tr>
<tr>
<td>DBMS_FLASHBACK</td>
<td></td>
</tr>
<tr>
<td>DBMS_MGWADM</td>
<td></td>
</tr>
<tr>
<td>DBMS_MGWMSG</td>
<td></td>
</tr>
<tr>
<td>DBMS_PROPAGATION_ADM</td>
<td></td>
</tr>
<tr>
<td>DBMS_STREAMS</td>
<td></td>
</tr>
<tr>
<td>DBMS_STREAMS_ADM</td>
<td></td>
</tr>
<tr>
<td>DBMS_STREAMS_MESSAGING</td>
<td></td>
</tr>
<tr>
<td>DBMS_TRANSFORM</td>
<td></td>
</tr>
</tbody>
</table>
Java Supplied Packages

javax.jms

Interface MessageConsumer

All Known Subinterfaces:
- AQimsQueueReceiver
- AQimsTopicReceiver
- AQimsTopicSubscriber
- QueueReceiver
- TopicReceiver
- TopicSubscriber

All Known Implementing Classes:
- AQimsConsumer

public interface MessageConsumer

A client uses a MessageConsumer object to receive messages from a destination. A MessageConsumer object is created by passing a Destination object to a message-consumer creation method supplied by a session.

MessageConsumer is the parent interface for all message consumers.

A message consumer can be created with a message selector. A message selector allows the client to restrict the messages delivered to the message consumer to those that match the selector.

A client may either synchronously receive a message consumer's messages or have the consumer asynchronously deliver them as they arrive.

For synchronous receipt, a client can request the next message from a message consumer using one of its receive methods. There are several variations of receive that allow a client to poll or wait for the next message.
Enqueuing Concepts

- Enqueue Options
  - visibility
  - relative_msgid
  - sequence_deviation
  - transformation
  - delivery_mode

- Message Properties
  - priority
  - delay
  - expiration
  - correlation
  - attempts
  - recipient_list
  - exception_queue
  - delivery_mode
  - enqueue_time
  - state
  - sender_id
  - original_msgid
  - transaction_group
  - user_property
Monitoring Staging

- Message delay
- Message expiration
- Retry delay
- Garbage collection for the queue table
- Retention and Message History
- Cleaning Up Message Queues
- Tracking and Event Journals
- Non-repudiation
- Queue Forwarding
Programming AQ

Oracle Developer 10g (C:\adoprojects\aq\pkgogy bailoutcode.sql)

lv_options := sys.shp.properties(sys.shp_property('MQ_openOptions', '2066'));
lv_prop := sys.shp.properties.properties.construct();
lv_prop.max_connections := 1;
lv_prop.queue_name := ipqManager;
lv_prop.hostname := ipHost;
lv_prop.port := ipPort;
lv_prop.channel := ipChannel;
lv_prop.user_name := ipUser;
lv_prop.password := ipPwd;
MQAddSubscriber(ipUser, ipPropType, ipQueue, ipDest);
lv_clob := lvMessage;
lv_payload := UTL_RAW.CAST_TO_RAW(lvMessage);
lv_payload := lvPayload.lvPayload;
message := sys.shp_basic_msg_t.construct;
-- message.text_body := sys.shp_text_value_t(null, lv_clob);
message.text_body := sys.shp_text_value_t(lvPayload, null);
-- N.B. the first param is for varchar(200), the second for clob

DEMS_AQ.ENQUEUE(queue_name := ipQueue,
enqueue_options := enqueue_options,
message_properties := message_properties,
payload := message,
msgid := message_handle);

EXCEPTION
WHEN OTHERS THEN
DEMS_OUTPUT.put_line('From local enqueuing := '||SQLERROR);
END;

PROCEDURE enqueue_message(ipMsgID IN RAW DEFAULT NULL,
ipMsgPayload IN RAW DEFAULT NULL,
ipMsgMetaDataInfo IN VARCHOR2 DEFAULT NULL,
ipQueueName IN VARCHOR2 DEFAULT 'OISSHIPSEQQ',
ipTargetQueueName IN VARCHOR2 DEFAULT 'OISSHIPSEQQ',
ipQueueManager IN VARCHOR2 DEFAULT 'MQ_AIDUG_89')
IS
lv_enq_opt := DEMS_AQ.enqueue_options_t;
DECLARE
enqueue_options   DBMS_AQ.enqueue_options_t;
message_properties DBMS_AQ.message_properties_t;
message_handle RAW(16);
message test.message_typ;
BEGIN
message := test.message_typ(001, 'MESSAGE 1 ', 'First message to adm_queue');
DBMS_AQ.ENQUEUE(
queue_name => 'aqadmin.adm_queue',
enqueue_options => enqueue_options,
message_properties => message_properties,
payload => message,
msgid => message_handle
);
COMMIT;
END;
PROCEDURE MqkAddSubscriber(ipHiUser IN VARCHAR2,
ipProcParType IN BINARY_INTEGER DEFAULT 1,
ipQueue IN VARCHAR2 DEFAULT 'GLSSH1\$REQ2',
IpDest IN VARCHAR2 DEFAULT 'GLSSH1\$REQ2') IS
BEGIN
    dmas_mqadm.add_subscriber(subscriber_id => ipHiUser,
        propagation_type => ipProcParType,
        queue_name => ipQueue,
        destination => ipDest
    );
EXCEPTION
    WHEN OTHERS THEN null;
END;

PROCEDURE setConnectInfo(ipHiUser IN VARCHAR2 DEFAULT 'MQAGENT',
ipPwd IN VARCHAR2 DEFAULT 'MQAGENT',
ipDatabase IN VARCHAR2 DEFAULT 'DEMONG1') IS
BEGIN
    dmas_mqadm.db_connect_info(
        username => ipHiUser,
        password => ipPwd,
        database => ipDatabase);
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.put_line(SqlError);
END;

PROCEDURE create_queue_table(ipQueueTableName IN VARCHAR2,
ipQueueName IN VARCHAR2) IS
BEGIN
    null;
END;
DECLARE
    enqueue_options  DBMS_AQ.enqueue_options_t;
    message_properties  DBMS_AQ.message_properties_t;
    message_handle  RAW(16);
    message  test.message_typ;
BEGIN
    message := test.message_typ(001, ' "MESSAGE 1 " ', 'First message to adm_queue');
    DBMS_AQ.ENQUEUE(
        queue_name  => 'aqadmin.adm_queue',
        enqueue_options  => enqueue_options,
        message_properties  => message_properties,
        payload
        => message,
        msgid  => message_handle
    );
    COMMIT;
END;
DECLARE
    enqueue_options   DBMS_AQ.enqueue_options_t;
    message_properties DBMS_AQ.message_properties_t;
    message_handle    RAW(16);
    message           aqadm.message_typ;
BEGIN
    message := test.message_typ(001, 'APPLE', 'APPLE enqueued first.');
    DBMS_AQ.ENQUEUE(
        queue_name    => 'aqadm.fruit_queue',
        enqueue_options => enqueue_options,
        message_properties => message_properties,
        payload       => message,
        msgid         => message_handle);
    message := test.message_typ(001, 'GRAPE', 'GRAPE enqueued second.');
    DBMS_AQ.ENQUEUE(
        queue_name    => 'aqadm.fruit_queue',
        enqueue_options => enqueue_options,
        message_properties => message_properties,
        payload       => message,
        msgid         => message_handle);
    EXCEPTION
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20999,'At least a message could not be enqueued.'-awesome
    END;
Using AQ with MQ
Using AQ with MQ

Mqe
IBM SMF
IBM JVM
Server Side Application
IBM WAS

Secure transactions
MQe Gateway
MQ Server

Install and maintain software

Client

Server

Operate over secure, optimized, fragile network connections

- IBM SMF: IBM Service Management Framework
- IBM JVM: IBM Java Virtual Machine
- IBM WAS: IBM WebSphere Application Server

Database
DECLARE
  _l_options sys.mgw_properties;
  _l_prop sys.mgw_mqseries_properties;
  _l_type_in VARCHAR2(12) := 'INBOUND';
  _l_type_out VARCHAR2(12) := 'OUTBOUND';
BEGIN
  _l_options := sys.mgw_properties(sys.mgw_property('MQ_SendExit','CDEM_QDEV_SendExit'));
  _l_prop := sys.mgw_mqseries_properties.construct();
  _l_prop.max_connections := 1;
  _l_prop.queue_manager := 'MQ_NYCMGW_A7';
  _l_prop.hostname := 'researchportal.adncorp.com';
  _l_prop.port := 1724;
  _l_prop.channel := 'OLSS.HIPAA1.UPDATE';
  _l_prop.username := 'WEBMQJAT';
  _l_prop.password := NULL;
  _l_prop.inbound_log_queue := funGetQueueName(_l_type_in);
  _l_prop.outbound_log_queue := funGetQueueName(_l_type_out);
  DBMS_MGWADM.CREATE_MGSYSTEM_LINK(
      LINKNAME => 'MQS_CDEMQUAT',
      PROPERTIES => _l_prop,
      OPTIONS => _l_options
    );
  DBMS_MGWADM.REGISTER_FOREIGN_QUEUE(NAME => 'OLSS.HIPAA1.UPDATE.REQUESTQ2',
    LINKNAME => 'MQS_IPC_LINK1',
    PROVIDER_QUEUE => 'OLSS_HIPAA1.UPDATE.REQUESTQ2',
    OPTIONS => MGW_PROPERTIES(MGW_PROPERTY('MQ_openOptions', '1724'),
      COMMENT => 'Websphere MQ Series Test on OLSS.HIPAA1.UPDATE.REQUESTQ2'
    );
  EXCEPTION WHEN OTHERS THEN
  DBMS_OUTPUT.put_line(SQLERRM);
END;
DECLARE
    lv_options sys.mgw_properties;
    lv_prop sys.mgw_mqseries_properties;
BEGIN
    lv_options := sys.mgw_properties(sys.mgw_property('MQ_SendExit','ADNMQDEVSendExit'));
    lv_prop := sys.mgw_mqseries_properties.construct();
    lv_prop.max_connections := 1;
    lv_prop.queue_manager := 'MQ_NYCGW_A10';
    lv_prop.hostname := 'portal.adncorp.com';
    lv_prop.port := 1414;
    lv_prop.channel := 'MQIIH.ADN1.UPDATE';
    lv_prop.username := 'WEBMQADN';
    lv_prop.password := NULL;
    DBMS_MGWADM.CREATE_MSGSYSTEM_LINK(LINKNAME => 'MQS_IPC_LINK1',
        PROPERTIES => lv_prop,
        OPTIONS => lv_options
    );
    DBMS_MGWADM.REGISTER_FOREIGN_QUEUE(NAME => 'MQIIH_ADN1_UPDATE_REQUESTQ2',
        LINKNAME => 'MQS_IPC_LINK1',
        PROVIDER_QUEUE => 'MQIIH.ADN1.UPDATE.REQUESTQ2',
        OPTIONS => MGW_PROPERTIES(MGW_PROPERTY('MQ_openOptions', '1414'),
            COMMENT => 'MQ Series Test on MQIIH.ADN1.UPDATE.REQUESTQ2'
        );
EXCEPTION WHEN OTHERS THEN
    DBMS_OUTPUT.put_line(SQLERRM);
END;
Using AQ with MQ
Using AQ with MQ

```
BEGIN
  lv_payload := MCDEV.PAYLOAD_MSG_STRING('Test message on 17 FEB 2005 at 4.30 PM');
  AQ_CTL.enqueue_message(lv_msg_id,
      lv_payload,
      ''OLSS.HIPAA3.UPDATE.REQUESTQ2'',
      ''OLSS.HIPAA3.UPDATE.REQUESTQ2'',
  );
  DBMS_OUTPUT.put_line('Enqueuing succeeded.');
END;
```
Using Database and Grid Control

Database Instance: adm1 > Streams

Streams

Setup Options
Streams Setup wizard allows you to setup and replicate the whole database, specific schemas or specific tables between 2 databases.
Streams Global, Schema, Table and Subset Replication Wizard
Streams Tablespace replication wizard allows the replication and maintenance of tablespaces between databases.
Streams Tablespace Replication Wizard
Messaging allows creation and setting up of queues.
Messaging

Copyright © 1996, 2005, Oracle. All rights reserved.
About Oracle Enterprise Manager 10g Database Control
Using Database and Grid Control

Database Instance: adm1 > Streams
Streams
Overview Capture Propagation Apply Messaging

Search

By default, the search returns all uppercase matches beginning with the string you entered. To run an exact or case-sensitive match, double quote the search string.

Select | Queue Name | Queue Table Name | Schema | Type | Enqueue Enable
--- | --- | --- | --- | --- | ---
1 | ALERT_QUE | ALERT_QT | SYS | Normal Queue |
2 | AQ$_ALERT_QT_E | ALERT_QT | SYS | Exception Queue |
3 | AQ$_AQ$ _MEM_MC_E | AQ$ _MEM_MC | SYS | Exception Queue |
4 | AQ$_AQ_EVENT_TABLE_E | AQ_EVENT_TABLE | SYS | Exception Queue |
5 | AQ$_AQ_SRVNTFN_TABLE_E | AQ_SRVNTFN_TABLE | SYS | Exception Queue |
6 | AQ$_DEF$_AQCALL_E | DEF$_AQCALL | SYSTEM | Exception |

49 NYOUG September, 2006
Using Database and Grid Control

Oracle Streams enables information sharing. Oracle Streams can share database changes and other information in a stream, which can propagate events within a database or from one database to another. The specified information is routed to specified destinations. The result is a feature that provides greater functionality and flexibility than traditional solutions for capturing and managing information, and sharing the information with other databases and applications.

- A capture process is an Oracle background process that scans the database redo log to capture DML and DDL changes made to database objects. It formats these changes into events called logical change records (LCRs) and enqueues them into a queue.
- Propagations send events from one queue to another, and these queues can be in the same database or in different databases.
- An apply process is an Oracle background process that dequeues events from a queue and applies each event directly to a database object or sends events to apply handlers for custom processing.
- Oracle Streams Messaging, also called as Oracle Streams Advanced Queuing, provides database-integrated message queuing functionality.
Using Database and Grid Control

Transformations
Transformation is a mapping from one Oracle data type to another, represented by a SQL function that takes the source data type as input and returns an object of the target data type. If a transformation is specified with a remote consumer, then the message is transformed before propagating it to the destination queue.

Search
Schema
Transformation
Name
Go

By default, the search returns all uppercase matches beginning with the string you entered. To run an exact or case-sensitive match, double quote the search string. You can use the wildcard symbol (%) in a double quoted string.

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Schema</th>
<th>From Type</th>
<th>To Type</th>
<th>Attribute Name</th>
<th>Transformation Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>HAEN_TXFM_OBJ</td>
<td>SYS</td>
<td>SYS.ALERT_TYPE</td>
<td>SYS.VARCHAR2</td>
<td>ENTIRE</td>
<td>SYS:haen_txfm_text</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(source user_data)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSSERVICE_METRICS_GEN_TS</td>
<td>SYS</td>
<td>SYS.SRLB_TYP</td>
<td>SYS.SRLB_TYP</td>
<td>ENTIRE</td>
<td>source user_data</td>
</tr>
<tr>
<td>SYSSERVICE_METRICS_TS</td>
<td>SYS</td>
<td>SYS.SRLB_TYP</td>
<td>SYS.VARCHAR2</td>
<td>ENTIRE</td>
<td>source user_data.payload</td>
</tr>
</tbody>
</table>
Advanced Strategies

- OCI - and Precompiler-based Custom Implementation
- RAC-Support (best strategy for the large enterprise)
- Message Priority
Advanced Strategies

- Message-Driven Beans and XA Support
  - Synchronous distributed transactions
  - Two-phase commit (2PC) implementation
- JMS-based custom implementation with Connection and Context Factory
- Protocol support (LDAP, and SMTP, SNMP APIs)
Advanced Strategies

Buffered messaging, a new feature in Oracle Streams AQ 10g Release 2 (10.2), combines the rich functionality that this product has always offered with a much faster queuing implementation. Buffered messaging is ideal for applications that do not require the reliability and transaction support of Oracle Streams AQ persistent messaging.
Buffered messaging is faster than persistent messaging, because its messages reside in shared memory. They are usually written to disk only when the total memory consumption of buffered messages approaches the available shared memory limit.
Asymmetric authentication via PKI

- The producer application encrypts the message payload prior to enqueuing.
- The consumer application knows the key and decrypts the message.
Managing Encryption

- The approach is also valid for intermediate or repeating queues under the SOA infrastructure, in conjunction with service requestor and receiver, accordingly.

- Encryption can be congruent with payload transformation.
Industries of Application

- Financial Sector
  - Banking
  - Trading
- E-Business (SCM, e.g., B2B transactions)
- E-Business (CRM, e.g., Order Entry)
- Direct Marketing
Related Technologies

- SOA
- Web Services Security and Transaction
- Oracle Streams
- Oracle Advanced Replication
- RPC
Related Technologies

**AMQP (competitor).**

**In the news:** LONDON - June 20, 2006 - JPMorgan Chase & Co., Cisco Systems, Envoy Technologies, Inc., iMatix Corporation, IONA® Technologies, Red Hat, Inc., TWIST Process Innovations, and 29West, today announced the formation of the AMQP (Advanced Message Queuing Protocol) Working Group and an effort by its members to create a new specification for defining and developing messaging infrastructure that is technology agnostic, standards-based, open and interoperable. The AMQP is a binary level protocol that is divided into two layers and designed with a flexible, plug-in architecture. Both the functional layer and the transport layer can be easily evolved to enable AMQP to respond to changing technology requirements.
Strategic Group Partners

Products working with Oracle Procedural Gateways

- IBM Websphere MQ (formerly MQ Series)
- Microsoft MSMQ
- Tibco Rendez-Vous
Available Literature

Oracle Documentation

- Oracle Streams Replication Administrator's Guide
- Oracle Database PL/SQL Packages and Types
- Oracle Database Heterogeneous Connectivity
- Oracle Streams Advanced Queuing User's Guide and Reference
- Streams Concepts and Administration
- Streams Advanced Queuing Java API Reference
- Streams Advanced Queuing User's Guide and Reference
- Application Developer's Guide - Advanced Queuing (Oracle9i)

White Papers


Metalink

- 238070.1, 198523.1, 212587.1, 188833.1, 198523.1
Envisioning AQ Future

- Who
- Where
- When
- Extent
- How
- How much
Concluding Remarks

- Future of information integration
- Importance of protocol standards
- Information Privacy
- Vendor interoperability
- Business Operational effectiveness.
Demonstration
Discussion

AQ
Q/A

?