Oracle Database 11*g* on Windows: Development and Deployment

An Oracle Technical White Paper
September 2007

ORACLE

Oracle Database 11*g* on Windows: Development and Deployment

Oracle Database 11*g* on Windows: Development and Deployment

EXECUTIVE OVERVIEW

Oracle Database 11*g* provides a comprehensive database solution to the Windows platform to make developing and deploying Oracle on Windows and .NET easier, faster, and more cost-effective.

The Oracle Developer Tools for Visual Studio .NET is a powerful "add-in" for Microsoft Visual Studio 2005 and Visual Studio .NET 2003 that makes it easier and faster for .NET developers to write Oracle applications. It is tightly integrated with Visual Studio and includes powerful features such as designers, automatic .NET code generation and a fully integrated PL/SQL editor and debugger.

.NET developers can access the Oracle database via Oracle Data Provider for .NET (ODP.NET). ODP.NET is a native .NET data provider that offers the best performance and most Oracle database functionality available from a .NET environment. In the most recent releases, ODP.NET supports 64-bit .NET Framework (Windows x64 and Itanium), faster performance for queries and LOB retrieval, and local transactions for System.Transactions.

As part of Oracle Database 11*g*, the Oracle Database Extensions for .NET is a database option for deploying stored procedures and functions written in a .NET-managed language into Oracle Database on Windows. A .NET-stored procedure can be called from within .NET code; from a SQL statement or trigger; from another .NET, PL/SQL or Java stored procedure; or from anywhere else a stored procedure or function call is allowed.

Oracle Database 11*g* on Windows includes new features for Active Directory and Volume Shadow Copy Service. Oracle integrates with Windows native security and Active Directory, allowing Oracle to operate in heterogeneous operating system and directory environments. Oracle database and configuration tools can use a Windows user's login credentials to connect to Active Directory without having to re-enter the login credentials. In addition, setting up Kerberos authentication for Oracle users in Active Directory is now more flexible and easier to set up.

The new Oracle Volume Shadow Copy Service (VSS) writer allows Oracle databases to participate in VSS-initiated backup and recovery on Windows Server 2003 and higher. Administrators can reliably back up and recover Oracle data in conjunction with popular VSS requestor and provider vendors.

Oracle Fail Safe provides high availability solutions for Oracle software on one or more Windows clusters with an easy to use manageability console.

ORACLE ON WINDOWS

By being the first with a relational database on Windows NT in 1993, Oracle had made a commitment to offer the best information management solutions available on the platform. Since that time, Oracle technology has adapted to the latest changes in Windows computing starting from the early client/server solutions to the Internet applications, and now to grid technology. Today, Oracle remains a leader on the Windows platform through its world-class database and its close integration with the operating system's and .NET Framework's underlying technologies. Oracle has over twenty-five years of data management expertise and is the leading vendor with the technology, experience, and services needed to deploy business-critical applications on Windows—whether on a corporate intranet or the Internet.

Oracle Database 11*g* for Windows provides all the features needed for data management, whether it is used for a departmental or enterprise-wide deployment. It allows users to take advantage of the cost-effectiveness and ease of use of Windows, while providing the scalability, reliability, and performance traditionally available from Oracle. This new Oracle database contains additional enhancements for application integration on the Windows platform.

Two of these focus areas are application development and deployment on Windows. Oracle Database 11*g* has made significant strides to improving performance and the ease of use and manageability for both Windows developers and administrators.

One of the great benefits of using Oracle products is its support of multiple programming standards. By supporting Java, .NET, PHP, and C/C++ applications, Oracle ensures all developers can use Oracle's advanced database features, providing true flexibility for development organizations. Each of Oracle's data access drivers is designed to maximize performance and to have access to the latest database features. This white paper focuses specifically on .NET development with the Oracle database and Oracle database server integration with Windows.

.NET APPLICATION DEVELOPMENT

Oracle provides a number of products for .NET programming. Oracle Developer Tools for Visual Studio .NET is a plug-in for use at design time. Oracle Data Provider for .NET enables data access to all of Oracle's key database features. Oracle Database Extensions provides support for .NET stored procedures.

Oracle .NET developers must typically perform various database tasks, such as modifying database table or view designs, updating data, and editing and debugging PL/SQL stored procedures. The Oracle Developer Tools for Visual Studio .NET (ODT) makes these tasks easy. ODT is a tightly integrated "add-in" for Microsoft Visual Studio 2005 and Visual Studio .NET 2003 that includes powerful features, such as designers and wizards, automatic .NET code generation, and a fully integrated PL/SQL editor and debugger.

Oracle offers ADO.NET data access support via ODP.NET. ODP.NET is a native .NET data access provider for Oracle databases. It can be used from any .NET language, including C# .NET, Visual Basic .NET, and ASP.NET, and from .NET application servers or .NET stored procedures. ODP.NET offers the best performance and access to more Oracle database features than any other data provider. ODP.NET was designed specifically to allow .NET developers to maximize Oracle database's capabilities.

ODP.NET 11*g* introduces performance features that use both new functionality available in Oracle Database 11*g* as well as enhance existing functionality in the Oracle database. As such, developers will benefit using the latest ODP.NET version whether they are deploying new database applications or enhancing legacy applications.

Oracle Database Extensions for .NET is a database option that lets you deploy stored procedures and functions written in a .NET-managed language, such as C# or VB.NET, into the database server on Windows. A .NET stored procedure can be called from within .NET code; from a SQL statement or trigger; from another .NET, PL/SQL or Java stored procedure; or from anywhere else a stored procedure or function call is allowed.

Oracle Developer Tools for Visual Studio .NET

The Oracle Developer Tools for Visual Studio .NET makes it easier and faster for .NET developers to write Oracle applications.

.

The Oracle Developer Tools for Visual Studio .NET is a powerful "add-in" for Visual Studio .NET that makes it easier and faster for .NET developers to write Oracle applications. Available now as a free download on the Oracle Technology Network, it is tightly integrated with Visual Studio 2005 and Visual Studio .NET 2003 and includes powerful features such as designers, automatic .NET code generation and a PL/SQL editor/debugger.

All of these features are designed with the goal of making developing for Oracle on Windows as intuitive and as easy as possible. Developers coming from a SQL Server background will find it to be familiar and intuitive, minimizing any learning curve. And for those who are new to Oracle development, this is an easy way to learn.

Major features include:

- Oracle Explorer – Browse and alter the Oracle schema via a tree control
- Designers and Wizards– e.g. Table Designer - makes database tasks easy
- Automatic Code Generation – Drag and drop to create working code.
- PL/SQL Editor and debugger – Edit and debug PL/SQL stored procedures, functions, packages and triggers
- Stored Procedure Testing – Run stored procedures and functions
- Oracle Data Window – View and edit your Oracle data
- SQL Query Window – Execute any ad-hoc SQL statement or script

- Integrated Help System - SQL, PL/SQL and Error Reference Manuals
- .NET Deployment Wizard - Deploy .NET stored Procedures easily

Oracle Explorer

The Oracle Explorer is a tree control that allows you to view the structure of the Oracle schema. All of the various Oracle schema types such as tables or stored procedures are included here. Metadata such as column datatypes or stored procedure parameter types are displayed in the Visual Studio .NET "properties pane" when a schema object is selected. A context menu for each schema object offers additional features: SQL Scripts can be generated for schema objects, filters are provided to limit what is shown, and designers and wizards can be spawned to alter the schema. For example, while viewing the structure of a table, the context menu can be used to spawn the "Oracle TableDesigner" to modify the design of the table.

Designers and Wizards

The Oracle Developer Tools offer a variety of designers and wizards to provide step by step assistance to create or alter database objects such as tables, views, stored procedures, stored functions, PL/SQL packages, sequences, indexes, constraints, triggers, synonyms, and more. For example, the table designer makes looking up datatypes or memorizing SQL syntax a thing of the past. In the case of the table designer, the user simply provides the table name and the names of the columns and chooses the appropriate datatype from a drop down box. Pressing the "save" button would then cause the table to be created or altered. A newly created schema object will then immediately appear in the Oracle explorer.

Any SQL that is generated by a designer to create or modify a schema object is displayed in both a "Preview" window (before a change is committed) as well as in the Oracle Database Output window (after the change is committed).

Automatic .NET Code Generation

Dragging and dropping a schema object from the Oracle Explorer onto an application's form results in .NET code being automatically generated for SELECT, UPDATE, INSERT, and DELETE operations on that schema object.

Dragging and dropping a schema object from the Oracle Explorer onto an application's form results in .NET code being automatically generated for SELECT, UPDATE, INSERT, and DELETE operations on that schema object. The code utilizes the OracleDataAdapter class which is provided by the Oracle Data Provider for .NET, Oracle's robust .NET data provider. The DataAdapter can then be used to connect UI elements (such as a DataGrid) on the application's form to the Oracle database, with a minimum of additional coding required. This use of a DataAdapter is standard and will be very familiar to anyone with programming experience with other databases.

The user is also given the option to generate code for a typed or untyped dataset based on an Oracle table or view. These datasets can then be bound as a data source to UI elements or can be used in other standard ways throughout Visual Studio .NET, such as with the Visual Studio XML Schema designer.

PL/SQL Editor

The PL/SQL editor provides a tightly integrated development environment for Oracle developers while freeing them from having to leave Visual Studio .NET when stored procedure development is required. Standard Visual Studio features include syntax coloring to improve readability and collapsible regions to hide stored procedures or functions that are part of a very large and complex package - this makes it easier for attention to be focused on theprocedure or function that is being worked on. Additionally, as SQL or PL/SQL statements are entered, a drop down list of tables or columns will pop up to provide autocompletion of the statement.

When the PL/SQL is compiled and errors are found, they are listed in the Visual Studio .NET task list. Clicking on them will take the developer to the line of code that failed. Pressing the context sensitive help key will open the error messages manual to the error code in question.

Fully Integrated PL/SQL Debugger

Now with ODT's integrated PL/SQL debugger, you can step through the PL/SQL code, view and modify variable values, look at the call stack, set break points, and more. You can step from your VB.NET or C# application code directly into PL/SQL code and back out again.

A feature introduced in ODT 10.2.0.2 is a PL/SQL debugger that is fully integrated with Visual Studio. You can now debug the PL/SQL code inside of stored procedures, functions, or triggers in exactly the same way that you debug VB.NET or C# code. You can step through the PL/SQL code, view and modify variable values, look at the call stack, set break points, and more.

You can also step from your VB.NET or C# application code directly into PL/SQL code and back out again. For example, if your C# code is calling a PL/SQL procedure or function using the Oracle Data Provider for .NET, you can debug the C# code, and then step into the PL/SQL code to continue debugging. You can examine all of the parameter values passed into the stored procedure or function, including complex data types and arrays from within the PL/SQL debugger. When the PL/SQL code execution has completed, you are then returned to the C# code to continue debugging.

You can also set breakpoints and debug PL/SQL stored procedures, functions, or triggers called by any application located on any platform without requiring any modification or rebuilding of that application. For example, a PHP application running on Linux calls PL/SQL stored procedure FOO and passes in an array of values. After you set an environment variable in the environment of the PHP application, you can set a breakpoint in FOO within Visual Studio and begin debugging when FOO executes.

Stored Procedure Testing

Stored procedures and function may be tested quickly via the "Run" context menu on the Oracle Explorer. Input parameters are requested, and then the procedure is executed. Output parameters are provided in an easily readable format. Complex output types, such as REF CURSOR yield a link that when clicked will open up a grid containing the complex data.

Oracle Data Window

The Oracle Data Window displays table or view data in an easy to read grid and allows the developer to insert, update or delete data without having to leave Visual Studio .NET development environment. For large tables, the user can provide a row number to jump ahead to a particular row.

SQL Query Window

Ad hoc SQL statements, such as those containing SELECT, CREATE, ALTER, etc may be executed from the SQL Query Window. If the output of the SQL statement is a table or view, the user is given a choice of text or grid output. Multiple statements may be highlighted and run as a group. The script operator "@" can be used to run SQL scripts.

Schema objects may also be dragged and dropped from the Oracle Explorer onto the SQL Query Window surface to automatically generate the appropriate SQL.

Integrated Online Help

The Oracle Developer Tools for Visual Studio .NET includes key Oracle documentation that has been converted into Visual Studio help format. The SQL Language Reference Guide, the PL/SQL User and Reference Guide, and the Error Messages and Codes manuals have each been included.

The chapters in these manuals may be read sequentially via the Visual Studio .NET help pane. In addition, this documentation has been integrated with the Visual Studio context sensitive help. For example, while writing a stored procedure, the developer can highlight a SQL or PL/SQL keyword, such as "SELECT", press the context help key, and be taken automatically to the appropriate page in the SQL Language Reference Guide. In addition to these manuals, the Oracle Developer Tools documentation contains helpful getting started and "walkthrough" chapters as well as a reference guide.

Oracle Data Provider for .NET

The Oracle Data Provider for .NET provides fast data access performance to Oracle databases. It supports the latest database functionality, including database change notification, XML DB, native Oracle data types, and Real Application Clusters.

ODP.NET is a native .NET data access driver for the Oracle database. Because it was developed for Oracle-specific data sources in mind, ODP.NET is optimized for fast performance and to access the latest database functionality. As such, ODP.NET makes using Oracle databases more flexible, faster, and more stable than many other data access solutions for Oracle. Beginning with version 10.2.0.2, ODP.NET has been compliant with both ADO.NET 1.x and 2.0. ODP.NET supports all versions of the .NET Framework currently available, including .NET Framework 3.0.

Other .NET data providers for Oracle are designed for maximum portability in accessing different data sources. What is lost when using these providers is the ability to take advantage of the database's inherent performance and data management features. For example, one of ODP.NET's unique features is support for native Oracle data types, such as REF Cursors and LOBs. These data types provide greater flexibility for data retrieval than their .NET data type counterparts.

Developers can choose the data types that best fit their data usage and performance requirements. For example, REF Cursors will defer result set retrieval until the data is actually read, a key performance optimization often used. With Oracle LOB data types, an application can choose to retrieve all LOB data in a result set in one database round trip or defer LOB retrieval until some later time when the LOB data is actually consumed by the client. Then when the LOB is read, the developer can decide how much of that data is retrieved. Perhaps the application may only need the last 100KB of a very large CLOB, rather than the entire CLOB. This is an example of how ODP.NET provides greater control to performance tune and take advantage of native database features.

With the advent of ADO.NET 2.0, even code portability is no longer an issue. Developers can create a single set of code that can access multiple vendors' databases much more easily now with the ADO.NET data provider factory classes.

ODP.NET 11*g* provides a plethora of features for the Oracle database, including data access performance tuning; database change notification; connection pooling for RAC and Data Guard; XML support; native Oracle data type support; ADO.NET 2.0 support; and many other features.

Performance

ODP.NET includes numerous performance tuning features to optimize data retrieval and data changes. Some of these tuning options include using connection pooling; statement caching; using LOB data types; and employing PL/SQL associative arrays. 64-bit ODP.NET is supported on Windows x64 and Windows Itanium.

One of ODP.NET's key differentiators over other providers is its out-of-the-box performance and numerous tuning options. Under the covers, a lot of optimizations have been done to ensure fast .NET access to Oracle data sources. Additionally, ODP.NET has many tunable parameters for specific data retrieval and data update scenarios. Many of these optimizations were developed for retrieving and manipulating Oracle native types, such as LOBs and REF Cursors.

Connection Pooling and Statement Caching

One of the most widely used performance optimizations is connection pooling, which is critical for applications with large numbers of users that connect and disconnect from the database. ODP.NET creates a pool of connections with tunable settings that include connection lifetime and timeout, minimum and maximum pool sizes, and the numbers of connections to increment or decrement from the pool at a time. These parameters give developers greater control over how their application handles large user populations and the changes in those populations over time. This ultimately leads to better application response time and quality of service for end users.

If a particular query or PL/SQL statement is executed multiple times, ODP.NET can use statement caching to speed statement execution. By caching the server cursor created during the initial statement execution, statement caching eliminates the need to re-parse each statement before subsequent executions. Each subsequent statement execution reuses the saved parsed information, and then executes the statement. The result set data itself is not cached, only the parsed statement information. ODP.NET will still retrieve the latest data from the

database server. Statement caching just allows these queries to be executed more quickly.

When employing statement caching, SQL or PL/SQL statements should use parameters rather than literal values. Doing so takes full advantage of statement caching since parsed information from parameterized statements can be reused even if the parameter values change in subsequent executions. If literal values were used instead and those literal values changed, the parsed information could not be reused and the database would need to parse the statement anew.

By default, ODP.NET will cache the last ten executed statements. The number of statements to cache and which statements to cache can be configured at the .NET application level or machine level.

Controlling Data Fetch Size

To tune data retrieval performance, ODP.NET can specify a set amount of data to return for each database round trip. Many times, a developer may not need to retrieve the data queried all at once because the end user is consuming the data over a period of time.

The query's data fetches can be spaced in distinct chunks defined by the developer through two ODP.NET OracleCommand properties: FetchSize and RowSize. FetchSize tells ODP.NET how much data to retrieve per database roundtrip. RowSize indicates how large each row of data is. RowSize is a read-only property that is set after a query is executed. If a developer wishes to fetch ten rows of data per database roundtrip, all that is required is to set FetchSize equal to ten multiplied by RowSize. The wonderful thing about RowSize is that its value is determined at run-time. So if there is a schema change or a query change in the future, there is no need to modify the code to ensure ten rows of data are fetched per round trip.

Optimizing LOB Data

A similar fetch size tuning feature exists with LOB data types. These data types are used to store images and documents, which can sometimes be in the range of gigabytes each. For LOB applications, performance is often a key issue due to LOB's potential data sizes and how LOB data is consumed. Sending gigabytes of data between the server and client can clog a network unless data retrieval is handled intelligently.

With ODP.NET, developers can specify how LOB data should be retrieved. Upon LOB query execution, developers can choose to fetch all the LOB data immediately in a minimal number of database round trips or defer the LOB fetch until the user attempts to read the data. If developers choose to defer the LOB fetch, they can then specify how much data to retrieve for each LOB read call. If the end user only needs to read 10KB of data at a time, developers can retrieve just 10KB of data for each LOB read.

In addition, ODP.NET developers can retrieve any portion of a LOB via random access. Perhaps they may need only the last 100MB of data from a 1GB LOB.

Developers can tune LOB retrieval to only fetch the last 100MB without returning data from the first 900MB to the client. These tuning options provide .NET developers the flexibility to build better performing applications.

Because LOB data can often be large, by default, the LOB data fetch is deferred after a query is executed. When retrieving many sets of large LOBs, this behavior is optimal to prevent overloading the network with LOB data delivered to the client. However, for small LOBs, this behavior can be slow, producing more database round trips than should be necessary.

To allow all the small LOB data to be fetched immediately, ODP.NET has an InitialLOBFetchSize property on the OracleCommand and OracleDataReader classes. If InitialLOBFetchSize is set to a value greater than zero, the initial LOB data from all the LOBs queried is fetched in one round trip up to the number of characters or bytes that is specified in this property. For instance, if InitialLOBFetchSize were set to 10 KB, then the first 10 KB of all the LOBs selected would be retrieved to the client in one database round trip. This can significantly speed up applications using lots of small LOBs.

Array Data

One of ODP.NET's unique features is the ability to pass arrays between the database and .NET Framework. Arrays make sharing large sets of data of the same data type between the database and client much easier. ODP.NET uses PL/SQL associative arrays in the database to pass data into and out of .NET arrays.

64-bit .NET Framework

With 64-bit .NET Framework's introduction, .NET developers have access to more scalable and high-performing hardware systems. They have a choice between AMD64 and Intel EM64T processors for Windows x64 and Itanium processors for Windows Itanium. 64-bit systems have the capability to address larger amounts of memory directly than 32-bit systems can and have optimized hardware components for high performance computing. Beginning with the 10.2.0.3 release, ODP.NET supports both 64-bit .NET Frameworks with a native 64-bit data access driver for each platform. Developers can now deploy their Oracle .NET mid-tier as a 64-bit application to take advantage of the more scalable hardware.

Performance – New for ODP.NET 11*g*

New Oracle Database 11*g* features are available that enhance ODP.NET performance. These features include a client result cache, faster LOB fetching, and faster performance with statement caching.

Oracle Database 11*g* introduces new performance optimizations, many of which .NET application developers can use without any changes to their client code. These new features include a client result cache, faster LOB retrievals, and enhanced statement caching.

Client Result Cache

With Oracle Database 11*g* server and client, ODP.NET applications can use the Oracle client result cache to improve response times of repeatedly executed queries. This feature enables client-side caching of SQL query result sets in memory. The client result cache is completely transparent to ODP.NET applications, and its

cache of result set data is automatically kept consistent with any session or database server side changes that would alter the results.

.NET applications calling the same query multiple times see improved performance since query results are retrieved locally. Local client processing is faster than making a database round trip to re-execute a query and fetch results. If applications are frequently running the same queries, they will experience a significant performance improvement when their results are cached on the client, as well as a reduction in database server load.

On the database server, the client cache reduces the server CPU and network traffic load that would have been consumed for processing and returning the query results, thereby improving server scalability. ODP.NET statements from multiple sessions can match the same cached result set in the client process memory if they have similar schema, SQL text, bind values, and session settings. Otherwise, the query execution occurs on the server. This means that multiple ODP.NET users all have access to the same result cache, which minimizes cache redundancies and saves memory.

Because the client cache automatically stays consistent with database server data, developers do not need to write code to ensure the cache and server remain in synch. If a server change occurs that would invalidate the client-cached data, the Oracle client will automatically invalidate the cache and update it the next time the query is executed.

Faster LOB Fetching

ODP.NET improves the performance of small-sized LOB retrieval by reducing the number of round-trips to the database required for pre-fetching the LOB's metadata. This enhancement is available beginning with Oracle Database 11*g*. This enhancement is transparent to the developer, requiring no code changes to use this feature.

Enhanced Statement Caching

ODP.NET 11*g* enhances the existing caching statement caching infrastructure to now cache ODP.NET parameter contexts. This enhancement works with any currently supported Oracle database server version. .NET developers will see a performance improvement when executing statement-cached queries. This enhancement is transparent to developers, requiring no code changes.

A constant challenge for client side caches is keeping the data in synch with server data changes. Using database change notification, ODP.NET clients are alerted when data on the server is modified, even if there is no active connection back to the database. This allows the client to ensure its data cache stays synchronized with the database.

Database Change Notification

Database change notification enables client applications to receive notifications when DML or DDL changes are made to a database object of interest, even when the client no longer has a connection to the database server. .NET developers can now cache their data on the middle-tier without having to worry about the cached data becoming out of synch with the database. If a change happens to one of the cached data objects or rows of data, then ODP.NET will receive a notification from the database. This feature can be used in both the .NET Framework 1.x and 2.0 releases.

To use database change notification, the client application registers a query with the database. When a query has dependencies on underlying database objects and a change to an object is committed, the database publishes a change notification to the client application. The notification only contains metadata about what data or objects changed; it does not contain the changed data. .NET developers can create a client event handler to reissue the registered query to obtain the changed data.

Database change notification is particularly useful for applications that use cached results. Traditionally, data caching by itself is effective at improving application scalability by allowing rapid access to data without making expensive roundtrips to the database. But this scalability comes with a tradeoff, as there is no longer a guarantee that the data remains consistent with the database server after the initial query. Thus, the cached client data runs the risk of becoming stale.

Database change notification solves the stale data cache problem. Although database change notification is similar to a trigger in that it responds to a particular event, a trigger takes action immediately, whereas a database notification is just an alert, not an action. It is up to the application to determine what action, if any, to undertake and when. The application can immediately refresh the stale objects, postpone the refresh, or ignore the notification. Each .NET application may want to respond differently to a particular database change. Moreover, as additional applications are added to the database, it is often easier modifying the client application's event handler than modifying a database trigger. Modifying the trigger may require re-testing how existing applications work with the new database trigger code, while modifying just the new .NET application better isolates the testing boundaries.

Web applications often cache a variety of data, not all of which needs to be updated in real time. For example, a weather forecast may only be updated periodically. End users don't need to query the database every time the web page is visited. Since many people will be requesting the same data, application performance and scalability are greatly enhanced by caching the results and retrieving the data from the cache. At some point, the weather forecast is updated and the cache must be refreshed. This may be done the instant the current weather forecast in the database server has changed.

To receive database change notifications requires the database administrator to grant the CHANGE NOTIFICATION privilege to the application user. After connecting to the database, .NET users can then register their specific queries of interest for change notification. The developer creates a .NET client-side event handler to direct what the application should do upon receiving a database change notification. Usually, the event handler will re-query the database server and refresh the cache.

The following ODP.NET classes are used when building change notification applications:

- OracleDependency – Creates a dependency between an application and an Oracle database. It enables the application to receive a notification of a data change (for example, UPDATE statement), schema change (for example, ALTER TABLE), or global event (for example, database shutdown). The OnChange event handler in this class provides the client logic for what to do after the notification is received.
- OracleNotificationEventArgs – Provides all the event details when a change notification occurs.
- OracleNotificationRequest – Specifies the characteristics of a notification request and its notification, such as the notification registration's timeout value.

Oracle's database change notification has a number of features not available on SQL Server. Oracle supports all types of joins, whereas SQL Server does not support queries containing either outer joins or self-joins. SQL Server does not support notifications for statements using views, whereas Oracle database change notification supports views with the exceptions of fixed views (for example, V$ tables) and materialized views. SQL Server notifications also require explicit column references, while Oracle Database notifications support both SELECT * and explicit column references.

SQL Server notifications are not persistent. When a SQL Server notification is published, its notification handler is removed from the database. If the notification handler is still needed, the application must register a new notification handler. Oracle database change notification provides the option to persist the registration even after repeated changes. This is possible by setting OracleNotificationRequest.IsNotifiedOnce to false.

Real Application Clusters (RAC)

ODP.NET has two connection pooling features for RAC databases: ODP.NET connections are automatically load balanced across existing nodes based on real-time cluster metrics and severed database connections are automatically removed from the connection pool.

RAC is a cluster database with a shared cache architecture that overcomes the limitations of traditional shared-nothing and federated database approaches. The RAC cluster database is hosted on multiple computing nodes permitting better scalability and higher availability than a single computer server by itself. Because neither special hardware nor .NET application changes are required, RAC can be built with commodity hardware as the back end for existing applications without requiring any coding changes.

ODP.NET has always supported data access for RAC transparently. To improve ODP.NET connection management based on real-time database workload information, Oracle introduced two connection pool properties for ODP.NET 10.2. The first feature, runtime connection load balancing, improves load balancing workload across RAC instances, especially after nodes are added or removed from a cluster. The second feature, fast connection failover, automatically removes severed RAC connections from the connection pool.

With runtime connection load balancing, how ODP.NET connections are allocated is based on the database's load balancing advisory and service goal during runtime. Load balancing distributes work across all of the available RAC database instances.

In general, connections are created infrequently and exist for a long duration. Work comes into the system with high frequency, uses these connections from the pool, and exists for a relatively short duration. The load balancing advisory directs which RAC instances ODP.NET should allocate incoming work to provide the optimal service quality. Load balancing minimizes the need to relocate the work later to a different instance and ensures existing jobs can complete quickly.

The metric by which this work is distributed across instances is determined by the service goal. The database administrator sets the service goal for either service time or throughput.

The service time metric is based on how quickly the database can complete tasks, essentially its response time. The load balancing advisory data is based on the elapsed time for work done in the service as well as available bandwidth to the service. Service time is most useful for database applications where work tasks complete at different rates, such as an internet shopping system.

The throughput metric, too, tracks how quickly database tasks are completed, allocating more work to nodes where tasks are completing the fastest. Unlike service time, the throughput metric is intended to gauge the efficiency of each instance. It measures the amount of processor resources available to take on additional tasks. When a new operation comes in, it can be directed to the node with the most processor time available. This metric is best used for relatively uniform tasks, such as executing homogenous batch processes.

By using these service goals, feedback is built into the system. Work is routed to provide the best service times globally, and routing responds gracefully to changing

system conditions. When a RAC node is added to or removed from the cluster, load balancing allows work to more quickly distribute itself evenly across all the nodes to account for the system change. End users will then experience less disruption or slowed service. In a steady state, the system approaches equilibrium with improved throughput across all of the RAC instances.

To use this feature in ODP.NET, the application must enable connection pooling and set the connection pool parameter, Load Balancing, to true.

The second configurable feature for RAC connection pooling, fast connection failover, allows ODP.NET to automatically free resources associated with severed connections that were caused by a downed RAC service, service member, or node. Without this feature, if a RAC node failed, the connection pool would retain connection resources that were no longer valid. End users may attempt to use these severed connections from the pool. Without a way to identify these connections, administrators would have to reset all the ODP.NET connections in the pool every time some part of the RAC cluster failed.

These severed ODP.NET connections are now automatically cleaned up without requiring any administrator intervention at runtime. To enable this ODP.NET feature, use connection pooling and set the connection pool parameter, HA Events, to true.

XML Features

ODP.NET provides support for Oracle XML DB and interoperating with System.XML. ODP.NET can work with schema and non-schema-based XML with support for native Oracle XML data types using OracleXMLType and OracleXMLStream.

Because XML is a popular language for data integration and web services, many .NET programmers incorporate it as an integral part of their applications. XML is key part of both the Oracle database and .NET Framework. ODP.NET allows developers to exploit these two technologies together: Oracle XML DB and .NET System.XML services.

XML DB is Oracle's high-performance, native XML storage and retrieval technology available within the database server. It provides a unique ability to store and manage both structured and unstructured data under a standard W3C XML data model. XML DB provides complete transparency and interchangeability between the XML and SQL metaphors. ODP.NET makes available XML DB's functionality to .NET clients, allowing developers to share and make changes to XML between the database and .NET. This support extends to both schema and non-schema-based XML to provide flexibility for differing application requirements. Moreover, ODP.NET contains two native XML data types, OracleXMLType and OracleXMLStream, for easier client XML data management. XML can be shared easily between XML DB and Microsoft's System.XML services.

System.XML is a set of interfaces for manipulating XML data sets from .NET data providers. ODP.NET interoperates with the System.XML programming interfaces, feeding data via the ODP.NET DataAdapter interfaces. One of the key distinctions between XML DB and System.XML is that the former provides XML services where the data resides, on the database server; the latter manipulates XML

on the client side. As such, ODP.NET provides greater choice for programmers to pick the XML technology that best fits their project requirements.

With ODP.NET, relational and object-relational data can be accessed as XML from an Oracle database instance into a Microsoft .NET environment. XML changes can be made and saved back to the server as XML or relational data. ODP.NET includes support for schema-based OracleXMLType, as well as non-schema-based XML.

ADO.NET 2.0

ODP.NET supports ADO.NET 2.0's features, including provider factories, connection string builder, schema discovery APIs, and DataAdapter batch updates.

Beginning with ODP.NET 10.2.0.2, ODP.NET provides support for ADO.NET 2.0. ADO.NET 2.0 introduces a new level of abstraction for the Data Access Layer (DAL). Instead of using provider-specific classes that implement a generic interface, ADO.NET 2.0 offers the DbCommon classes, which inherit from the System.Data.Common namespace and allow developers to use factory classes. Database provider factory classes allow an easier way to create one set of generic data access code to any database.

The OracleClientFactory class supports the creation of all of the classes in System.Data.Common. Because these concrete classes inherit from the DbCommon abstract base classes, you can write generic DAL code by using DbCommon base class object names. There are some areas of the DAL that remain data source-specific, including the connection string, SQL, and stored procedure calls.

A key addition to ADO.NET 2.0 is improved connection string management. The DbConnectionStringBuilder class has a dictionary that maps generic parameter names to provider specific parameter names. The OracleConnectionStringBuilder class inherits from and extends the generic DbConnectionStringBuilder class to expose Oracle-specific connection string properties. Programmers can use OracleConnectionStringBuilder dynamically to set connection string parameters at run-time and/or obtain connection string parameters from the app.config file.

The ADO.NET 2.0 schema discovery APIs provide a generic way to retrieve metadata from any data source. Developers can obtain Oracle metadata by using an OracleConnection.GetSchema method call. There are five types of common metadata that can be exposed: MetaDataCollections, Restrictions, DataSourceInformation, DataTypes, and ReservedWords. In addition, there is additional ODP.NET-specific data source information that can be retrieved.

When working with large DataSets, developers will want to minimize the number of database round trips required to perform updates. By using the OracleDataAdapter.UpdateBatchSize property, programmers can specify the number of rows to update per roundtrip. To update all the modified rows in one roundtrip, UpdateBatchSize can be set to zero.

ODP.NET supports native Oracle database types, such as REF Cursors, within the .NET environment. Other major features include being able to use any type of PL/SQL; local and distributed transactions; and internationalized applications with Unicode.

Native Oracle Types

Microsoft has introduced a set of unified data types among the different .NET programming languages. With ODP.NET, .NET programmers have access to .NET data types as well as Oracle data types in .NET. Oracle types can be fully manipulated within a .NET application and interoperate with .NET data types. Oracle native types provide advanced functionality to store and manipulate data structures from the database, such as XML, result sets, images, or Microsoft Word documents. Even with scalar types, such as OracleDecimal, the equivalent to the .NET decimal type, Oracle types provide additional functionality. In the example of OracleDecimal, this data type provides a higher level of precision of 38 digits than the .NET decimal with a 28 digit precision.

ODP.NET supports the gamut of advanced Oracle types within the .NET environment, including REF Cursors, XMLType, LOBs (CLOBs, BLOBs, NCLOBs), BFILEs, LONGs, RAWs, LONG RAWs, and N-data types. One of the limitations of using other providers is that users may be limited in data type functionality. For example, in ODP.NET, multiple result sets returned from a stored procedure as REF Cursor output parameters can be accessed in an arbitrary way. One can read the results of the second REF Cursor without retrieving the results of the first. With other .NET providers, data may have to be accessed in a linear manner where the first result set's data must be retrieved prior to accessing the second's. This would negatively affect performance.

Beginning with ADO.NET 2.0, Oracle data types can be stored natively within the .NET DataSet. Previously, only .NET data types could be used in DataSet. By setting OracleDataAdapter.ReturnProviderSpecificTypes to true, the DataSet will be populated with ODP.NET-specific data types when OracleDataApdater.Fill is called.

Other Major Features

ODP.NET exposes many other Oracle database features, including PL/SQL, transactions, and Unicode support. ODP.NET users can fully execute PL/SQL stored procedures and functions in the database. PL/SQL can be packaged or non-packaged, or even exist as anonymous PL/SQL within .NET. Anonymous PL/SQL is generally employed to batch a set of SQL statements and execute the statements in one database round trip. Batching statements is a useful performance optimization technique.

ODP.NET can participate in transactional applications with the Oracle database as the resource manager. ODP.NET employs the Microsoft Distributed Transaction Coordinator (DTC) to oversee a transaction in a Windows environment. The Oracle Services for Microsoft Transaction Server (OraMTS) act as a proxy among ODP.NET, DTC, and the Oracle database in order to coordinate these transactions. OraMTS provides a robust architecture for ODP.NET programmers to have their transactional applications maintain high availability and high

scalability. In .NET Framework 2.0 and higher, ODP.NET supports both local and distributed transactions via the System.Transactions namespace.

ODP.NET has full Unicode support so that .NET users can globalize their applications easily in multiple written languages. This globalization support allows developers to create one set of code for multiple culture/language settings. ODP.NET globalization extracts the client computer's national language setting to display information in a locale-specific format. For example, a browser set to Japanese will have currency displayed in yen. Without additional coding, that same application can be deployed in Germany to show euros. This makes developing applications for multiple locales easier and faster since no additional coding is necessary.

Oracle Database Extensions for .NET

The Oracle Database Extensions for .NET is a feature of Oracle database on Windows that makes it easy to develop, deploy, and run stored procedures and functions written in a .NET managed language such as C# or VB.NET.

The Oracle Database Extensions for .NET is a feature of Oracle Database 11*g* on Windows that makes it easy to develop, deploy, and run stored procedures and functions written in a .NET managed language such as C# or VB.NET.

.NET Stored procedures or functions are developed using Visual Studio and then deployed using the tightly integrated .NET Deployment Wizard which is included with the Oracle Developer Tools for Visual Studio .NET. The deployment wizard loads the newly created .NET assembly into Oracle and registers the procedure or function with the database.

Stored procedures that require data access can use the Oracle Data Provider for .NET. The .NET Framework 2.0 and ADO.NET 2.0 are both supported in .NET stored procedures. The code used for the procedure or function is almost exactly the same as code written for a client side application, with the exception of an optional "context connection" in which the procedure uses the same connection as the caller. Code can also be written that can run both on the client side and in a stored procedure with no changes required.

After deploying, a .NET stored procedure can be called from within .NET code, from a SQL statement or trigger, from another .NET, PL/SQL or Java stored procedure, or from anywhere else a stored procedure or function call is allowed.

APPLICATION DEPLOYMENT

Oracle provides many different products that integrate Oracle applications into Windows environments, including Oracle Fail Safe, Windows security integration, and Active Directory integration.

Deploying applications on Windows requires close integration with native Windows services and middle-tier servers to ensure proper application interoperability. Oracle Database 11*g* provides this tight integration, allowing organizations to take advantage of core operating system and server functionality with Oracle's advanced feature set. Oracle's integration ensures transparent interoperability that is scalable, available, and secure. In this way, organizations can focus resources on building their application business logic, rather than accommodating product incompatibilities or inefficiencies.

Oracle Database 11*g* includes many sets of tools for ensuring optimized Windows application deployment environments. When deploying mission-critical Oracle

servers on Windows clusters, Oracle Fail Safe ensures that those servers remain available by providing robust fail over support through the Microsoft Cluster Service. Oracle integrates with native Windows security and directory tools that both improve database security and makes user management easier.

Oracle Database 11*g* on Windows introduces additional support for Active Directory, enhanced Kerberos support, and new Volume Shadow Copy Service integration.

Directory

Oracle's identity management platform allows Oracle Internet Directory (OID) to centralize synchronization and provisioning with other directories, including Active Directory. Oracle makes Windows single sign-on possible and user management easier by allowing user mappings to be stored in Active Directory or in OID. Moreover, Net naming (tnsnames.ora) can be stored in either Active Directory or OID for centralized management. New Active Directory features in Oracle Database 11*g* enhance security by disallowing anonymous access.

Directory servers often make user and resource management easier by centralizing their administration. Even with this centralization, a directory server must be able to integrate with other applications, departmental organizations, or trading partners, each of which may have their own separate directory server or user repository. As such, directory solutions must be able scalable to handle large user sets and interoperate with other third-party repositories.

Oracle Internet Directory (OID) allows enterprises to centralize their directory servers and leverage their existing directory servers, such as Active Directory (AD). Oracle Identity Management consists of a set of built-in services and interfaces that facilitate access control, single sign-on, user profile management, synchronization, and provisioning between Oracle and other repositories. These repositories include other directories, such as AD; application user repositories; or database tables containing human resources information.

With Oracle Identity Management, enterprises can manage end-to-end lifecycle of user identities across all enterprise resources both within and beyond the firewall. Administrators can deploy applications faster, apply the most granular protection to enterprise resources, automatically eliminate latent access privileges, and much more. Oracle Identity Management is a member of the Oracle Fusion Middleware family of products, which brings greater agility, better decision-making, and reduced cost to diverse IT environments today.

Without OID and Oracle Identity Management, Oracle provides services for direct database integration with AD. Oracle database includes two native integration features that leverage AD:

- Oracle single sign-on through native AD authentication
- Oracle Net naming management with AD

These features do not require OID, only AD and Oracle database. However, administrators have the option to use both these features with OID if needed.

Native Authentication and Active Directory

Oracle administrators often require enterprise-scale security and schema management. This is especially true for organizations with large user populations. OID is designed to handle these enterprise environments and interoperate with departmental directory servers, such as Active Directory. Oracle security and

administration are integrated with both these directory servers to provide seamless access and administration.

Oracle enables single sign-on through Microsoft authentication mechanisms, allowing Active Directory to perform user identification for Oracle databases. With native authentication enabled, database users can leverage single sign-on to access Oracle by logging onto Active Directory. This integration simplifies end-user administration and eliminates redundant security credentials.

Oracle enterprise user mappings allow multiple operating system users to access the database as a single global database user. In Windows-only environments, these enterprise user mappings may be stored in Active Directory. For instance, an entire LDAP organizational unit (OU) in Active Directory can be mapped to one database user.

Oracle stores enterprise role mappings in LDAP. On Windows, this LDAP storage feature is certified on OID as well as with Active Directory. With enterprise roles, privileges for multiple databases can be managed at the domain level through directories. This is accomplished by assigning Windows users and groups to the Oracle enterprise roles registered in the LDAP store.

Oracle Net Naming with Active Directory

Oracle leverages LDAP technology through OID and Active Directory to improve end user database connectivity information management. Traditionally, end users reference database servers with Oracle Net-style names resolved through the TNSNAMES.ORA configuration file. Traditionally, this file must be administered on each client machine. To ease manageability, Oracle Net names can be stored and resolved through OID or Active Directory. Centralizing such information in a directory eliminates administrative overhead and relieves users from configuring their individual client machines.

Furthermore, Windows tools, such as Windows Explorer and Active Directory Users and Computers, can connect to databases and test database connectivity. Oracle tools can also participate. The Oracle Database Configuration Assistant automatically registers database objects in an LDAP directory. The Oracle Net Manager, meanwhile, registers net service objects with the directory. These tools further simplify directory and database administration.

Active Directory Features – New for Oracle Database 11*g*

In Oracle Database 11*g*, administrators can choose to disallow anonymous access to a directory's database service information. Clients would be required to authenticate before performing LDAP directory-based name look-ups. With Microsoft Active Directory-based name lookups, Oracle Database uses the native Windows-based authentication. With Oracle Internet Directory (OID)-based name lookups, Oracle Database performs authentication by using wallets.

Security

The Oracle database supports a number of Windows security mechanisms, including single sign-on, Windows Kerberos, and Windows SSL. The ability to store Oracle wallets in the Windows registry eliminates insecure operating system file storage. Oracle integrates with the Microsoft Certificate Store, allowing Oracle PKI and Windows PKI applications to operate together seamlessly. New Kerberos features in Oracle Database 11*g* enhance security options and Windows interoperability.

Oracle Database 11*g* provides tight integration with the native Windows security model to better safeguard application security. In a client/server environment, there are three options for sites wishing to leverage Windows credentials or authentication services for signing on to the Oracle database:

- The Oracle database includes the Windows Native Authentication Adapter, which installs automatically with Oracle Net on the server and client. This adapter allows database users to have single sign-on capabilities by using Windows user credentials for database authentication. To use this feature, Windows users must be defined as external database users. These users can have external roles assigned to them in Microsoft Active Directory Services that will be respected by the database.
- Oracle Advanced Security, a database option, supports database authentication using Kerberos tickets issued by the Microsoft Key Distribution Center (MSKDC). This capability allows users who have been issued a valid Kerberos ticket in the Windows environment to sign-on to their database accounts without having to provide a username/password.
- The Oracle Advanced Security option also supports database authentication over SSL using an X.509v3 certificate. The Microsoft Certificate Store (MCS) may issue this certificate. To use this feature, the certificate must be contained in an Oracle Wallet configured on the client. The Oracle Wallet may be stored either in the user profile area of the Windows Registry or in an arbitrary file location on the client.

For web environments, integration with MCS is possible through Oracle Fusion Middleware (FMW). In addition to supporting password-based authentication mechanisms, Oracle FMW supports certificate-based authentication. This certificate can be generated by MCS. To use certificate-based authentication, a user certificate must be populated into the browser wallet and into the corresponding user entry in Oracle Internet Directory.

Oracle Wallets in Windows Registry

Oracle database wallets can be stored in the Windows registry, providing increased security on Windows clients. Without this feature, Oracle wallets are often stored on the Windows file system. If the operating system file permissions are not secured, the Oracle wallets are also not secured. Unsecured file permissions may come from improper administration or lack of operating system file security. On Windows systems, therefore, public key infrastructure (PKI) security is improved by storing Oracle wallets in the user profile area of the Windows registry. This registry area is accessible only to the properly logged-in user. Multiple Oracle wallets can be used and stored in the profile area.

Single sign-on for Oracle PKI applications is configurable through the Oracle Wallet Manager and Oracle Enterprise Login Assistant tools. The Wallet Manager

creates encrypted Oracle wallets, while the Enterprise Login Assistant creates the decrypted wallet. The decrypted wallet is then used by Oracle PKI applications for SSL authentication. Both of these tools are enhanced to support the storage and location of Oracle wallets in the registry, as well as the default file system if required.

Microsoft Certificate Store Integration

Oracle's database integration with the Microsoft Certificate Store allows Oracle PKI applications to interoperate with products that use Windows PKI. When this feature is enabled, Oracle PKI security uses Microsoft CryptoAPIs to access the Microsoft Certificate Store. The CryptoAPIs are used for operations, such as signing, encryption, decryption, verification, and validation. The Wallet Resource Locator (WRL) determines the PKI type and provides all the necessary information to locate the wallet.

With this feature, Oracle and non-Oracle applications can leverage the same set of PKI credentials (e.g. certificates, keys, revocation lists, and trustpoints) for public key security services, such as authentication and encryption.

Windows Security Features – New for Oracle Database 11*g*

The Oracle Database 11*g* Kerberos implementation now makes use of secure encryption algorithms like 3DES and AES in place of DES. This makes using Kerberos more secure. The Kerberos authentication mechanism in the Oracle database now supports the following encryption types:

- DES3-CBC-SHA (DES3 algorithm in CBC mode with HMAC-SHA1 as checksum)
- RC4-HMAC (RC4 algorithm with HMAC-MD5 as checksum)
- AES128-CTS (AES algorithm with 128-bit key in CTS mode with HMAC-SHA1 as checksum)
- AES256-CTS (AES algorithm with 256-bit key in CTS mode with HMAC-SHA1 as checksum)

The Kerberos implementation has been enhanced to interoperate more smoothly with the Microsoft Key Distribution Center. In addition, the Kerberos principal name can now contain more than 30 characters. The number of characters allowed in a database user name no longer restricts it. These improvements permit applications to be more easily secured with Oracle databases and Windows.

Backup and Recovery with Volume Shadow Copy Service – New for Oracle Database 11*g*

Volume Shadow Copy Service (VSS) on Windows server platforms enables volume backups to be performed while applications, such as Oracle, continue to write to the volumes. A shadow copy is a consistent snapshot of the data held on a volume or component at a well-defined point in time. A shadow copy set is a collection of

shadow copies that are all taken at the same time. Oracle now integrates with VSS to enable back up and recovery of Oracle database files.

VSS includes a Windows-specific interface that enables coordination between requestors that back up data, writers that update data on disk, and providers that manage storage. New in Oracle Database 11*g*, Oracle Database functions as a VSS writer that is integrated with VSS-enabled applications. As a writer, the Oracle database can be backed up and restored with VSS-enabled software and storage systems. A key benefit is the ability to use a VSS-enabled application to make an online backup of the whole database.

The Oracle VSS writer is a Windows service that coordinates an Oracle database instance and other VSS components. The writer service, which is started under a user account with SYSDBA privileges, runs separately from the database instance.

The Oracle VSS writer supports component-based shadow copies, which are sets of database files. During a backup, the Oracle VSS writer saves the redo generated during snapshot creation in a metadata document. During a restore operation, the writer automatically extracts the redo from the metadata document and applies it to files restored from a snapshot.

The Oracle VSS writer also supports volume-based shadow copies, which are snapshots of complete drives or volumes. The Oracle database will place its files in a state suitable to create shadow copies. For example, the datafiles are placed in hot backup mode and a new snapshot control file is created for a database in ARCHIVELOG mode. Oracle VSS writer excludes files such as the current control file and online redo logs from the shadow copies. The writer returns an error if the snapshot cannot be taken. For example, if a NOARCHIVELOG database is open in read/write mode, then the writer returns an error indicating that the snapshot is not possible.

With Oracle, VSS-enabled applications can easily work with the Oracle database to back up and recover snapshots.

High Availability with Oracle Fail Safe

Through integration with the Microsoft Cluster Service, Oracle Fail Safe ensures Oracle software remains highly available on Windows. Fail Safe supports the Oracle database, application server, and E-Business Suite,

Oracle Fail Safe is a core feature of Oracle Database 11*g* that provides high-availability for mission-critical solutions deployed on Windows clusters. A cluster eliminates individual host systems as points of failure. Oracle Fail Safe works with Microsoft Cluster Service (MSCS) to ensure that if a failure occurs on one cluster node, then the Oracle databases and applications running on that node will fail over (move) automatically and quickly to a surviving node. Unlike RAC, Oracle Fail Safe is an active-passive cluster, whereby the standby machine does not take over the workload until the active node fails.

Oracle Fail Safe is optimized for Windows customers with database and application workloads that can be handled by a single system. Oracle Fail Safe solutions can be deployed on all commodity Windows clusters. It supports up to eight-node

clusters on Windows Server 2003 Datacenter, the maximum node configuration available on Windows today. Supported products include:

- Oracle databases (Oracle 11*g*, Oracle10g, Oracle9*i*)
- Oracle Applications release 11*i*
- Oracle *i*AS, with the exception of cache products, including
  - Oracle Forms Server
  - Oracle Reports Server
  - Oracle HTTP Server
- Oracle Services for Microsoft Transaction Server
- Oracle Intelligent Agent
- Applications installed as Windows services

SAP, Baan, PeopleSoft, Lawson, J.D. Edwards, and other applications vendors also have validated their software solutions with Oracle Fail Safe.

Oracle Fail Safe includes two main components, a server and a manager. The server component, Oracle Services for MSCS, works with the cluster software to ensure fast automatic fail over during planned and unplanned outages. The manager, Oracle Fail Safe Manager, is an easy-to-use graphical interface that works with Oracle Fail Safe Server on one or more clusters to perform configuration, management, verification, and static load balancing. Together, these components provide a rich set of features and integrated troubleshooting tools that enable rapid deployment of highly available databases and applications—complete e-business solutions.

Logical/Physical Standby and Multi-cluster Manageability

Oracle Fail Safe supports disaster protection by enabling Logical/Physical Standby support. For improved manageability, multiple clusters can be managed from the same console.

With Logical/Physical Standby integration, the Oracle Fail Safe database can be run on a primary site and any remote standby sites. Fail Safe will monitor the database, bring it online, and take it offline. This allows customers to have single instance fail over within a cluster with minimal downtime, along with added benefits of maintaining standby sites. Oracle Fail Safe is commonly used for planned failovers, such as software and hardware upgrades common to Windows environments. With Logical/Physical Standby, Fail Safe now provides true disaster protection offered by standby. Together, the two products complement each other to provide higher availability and disaster protection than with either product alone.

For customers with multiple cluster deployments, Oracle Fail Safe Manager permits the administration of more than one cluster, allowing the user to manage all clusters from the same window, rather than from a separate window for each cluster. Administrators, for example, will be able to modify the database account credentials for databases across all their Fail Safe clusters all at once using a single wizard. Administrators will be able to avoid the tedious nature of comparing policies or modifying cluster display or alert preferences individually.

CONCLUSION

As technology in Windows application development and deployment has evolved, so has Oracle. Oracle Database 11*g* offers unprecedented flexibility for customers to choose the .NET application tools and architecture that fit their requirements without sacrificing performance, scalability, ease of use, and security. Oracle's commitment to the Windows developers includes integration with ADO.NET and Visual Studio. For Windows administrators, Oracle has closely integrated with many operating services, such as Active Directory, Internet Information Services (IIS), Windows security, and Microsoft Cluster Services. By designing Oracle Database 11*g* with Windows in mind, customer software applications can achieve seamless interoperability between the operating system and the database.

For more information about Oracle on Windows and .NET, visit

OTN Windows Technology Center: http://otn.oracle.com/windows

OTN .NET Developer Center: http://otn.oracle.com/dotnet

Oracle on Windows: http://www.oracle.com/windows

Oracle Database 11*g* on Windows: Development and Deployment
September 2007
Author: Alex Keh
Contributing Author: Christian Shay

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com