Oracle Waveset 8.1.1 Overview

Beta

Part No: 821–0762
August 2010

100729@24661

Contents

Preface

Oracle Waveset 8.1.1 Overview answers the question *What is Oracle Waveset and how does it work?* The book describes the Waveset product architecture, as well as information on how to plan a high-availability deployment.

Oracle Waveset was formerly known as Sun Identity Manager. This release of Oracle Waveset follows Oracle's acquisition of Sun Microsystems. Some parts of the product and documentation may still refer to the original Sun Identity Manager product name.

Who Should Use This Book

This guide is for IT professionals who are looking to better understand Oracle Waveset 8.1.1 and associated software. It will be of special value to IT professionals who are either in the process of evaluating Waveset, or who are in the beginning stages of planning a Waveset deployment.

How This Book Is Organized

This guide is organized into the following chapters:

Chapter 1, "Product Overview," describes the purpose of Waveset and highlights the application's major features.

Chapter 2, "Product Architecture," describes the Waveset architecture, the Service Provider architecture, and the web services architecture. Guidelines for system separation and physical proximity are also covered.

Chapter 3, "Clustering and High Availability," provides guidance on how to implement a high availability / fault tolerant (HA/FT) Waveset environment. It will also help you assess the amount of availability that your Waveset deployment requires.

Related Books

The Waveset 8.1.1 documentation set includes the following books.

Primary Audience	Title	Description
All Audiences	*Oracle Waveset 8.1.1 Overview*	Provides an overview of Waveset features and functionality and provides product architecture information.
	Oracle Waveset 8.1.1 Release Notes	Describes known issues, fixed issues, and late-breaking information not already provided in the Waveset documentation set.
System Administrators	*Installation Guide*	Describes how to install Waveset and optional components such as the Waveset Gateway and PasswordSync.
	Upgrade Guide	Provides instructions on how to upgrade from an older version of Sun Identity Manager to Oracle Waveset.
	System Administrator's Guide	Contains information and instructions to help system administrators manage, tune, and troubleshoot their Waveset installation.
Business Administrators	*Business Administrator's Guide*	Describes how to use Waveset provisioning and auditing features. Contains information about the user interfaces, user and account management, reporting, and more.

Primary Audience	Title	Description
System Integrators	*Deployment Guide*	Describes how to deploy Waveset in complex IT environments. Topics covered include working with identity attributes, data loading and synchronization, configuring user actions, applying custom branding, and so on.
	Deployment Reference	Contains information about workflows, forms, views, and rules, as well as the XPRESS language.
	Resources Reference	Provides information about installing, configuring, and using resource adapters.
	Service Provider 8.1 Deployment	Describes how to deploy Waveset Service Provider, and how views, forms, and resources differ from the standard Waveset product.
	Web Services Guide	Describes how to configure SPML support, which SPML features are supported (and why), and how to extend support in the field.

Documentation Updates

Corrections and updates to this and other Oracle Waveset and Sun Identity Manager publications are posted to a documentation updates website:

`http://blogs.sun.com/idmdocupdates/`

An RSS feed reader can be used to periodically check the website and notify you when updates are available. To subscribe, download a feed reader and click a link under Feeds on the right side of the page. Starting with version 8.0, separate feeds are available for each major release.

Related Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Oracle is not responsible for the availability of third-party web sites mentioned in this document. Oracle does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Oracle will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://docs.sun.com`)
- Support (`http://www.sun.com/support/`)
- Training (`http://education.oracle.com`) – Click the Sun link in the left navigation bar.

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to `http://docs.sun.com` and click Feedback.

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Meaning	Example
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX system prompt and superuser prompt for shells that are included in the Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Solaris release.

TABLE P–2 Shell Prompts

Shell	Prompt
Bash shell, Korn shell, and Bourne shell	`$`
Bash shell, Korn shell, and Bourne shell for superuser	`#`
C shell	`machine_name%`
C shell for superuser	`machine_name#`

CHAPTER 1

Product Overview

This chapter describes the purpose of Oracle Waveset and highlights the application's major features.

The chapter includes the following topics:

What is Oracle Waveset?

Oracle® Waveset makes it possible to automate the process of creating, updating, and deleting user accounts across multiple IT systems. Collectively, this process is known as *provisioning* (that is, creating and updating user accounts) and *deprovisioning* (deleting user accounts).

For example, when an employee joins a company, Waveset runs a workflow that retrieves the necessary approvals to grant the employee access. When these approvals are obtained, Waveset creates accounts for the employee in the company's human resources system (PeopleSoft), email system (Microsoft Exchange), and enterprise application (SAP). If the employee changes roles in the company, Waveset updates the user account and extends access to the necessary resources required in that new role. And when the employee leaves the company, Waveset automatically removes the user's accounts to prevent further access.

Waveset can also enforce audit policies on an ongoing basis. An *audit policy* specifies what types of access a user may or may not have. For example, in the United States it is a violation of Sarbanes-Oxley (SOX) for the same user to have access to both Accounts Payable and Accounts Receivable systems. This is known as a separation of duties violation. Waveset can conduct audit scanning to check for a variety of these types of violations and, depending on configuration, automatically remove access or send a notification to an administrator when a violation is detected. This process is known as *remediation*.

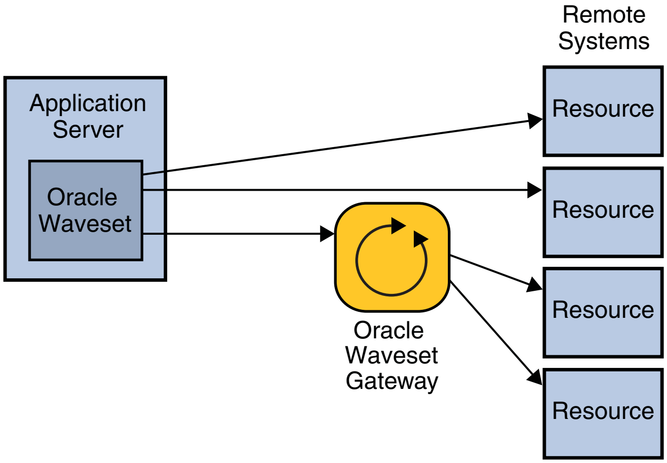
How Does Waveset Interface With Other IT Systems?

In Waveset, managed applications and other IT systems are called *resources*. Waveset uses either *adapters* or *connectors* to interface with resources.

Adapters and connectors are installed on the Waveset server. (Waveset does not require special software (called *agents*) to be installed on target resources.) Dozens of Waveset adapters and connectors are available, and new ones can be created to communicate with almost any resource using standard protocols or known application programming interfaces (APIs). Waveset ships with various adapters and connectors to communicate with many of the most common resources. In addition, templates and skeleton code is available to assist programmers in creating additional adapters and connectors.

Some resources cannot be communicated with directly and require the use of the Waveset Gateway. Examples of resources that require the Gateway include Microsoft products, such as Exchange and Windows Active Directory, Novell products, such as eDirectory (formerly Netware Directory Services), and several others. In such cases, Waveset communicates directly with the Gateway and the Gateway interfaces with the resource.

FIGURE 1–1 Waveset Interfaces with Some Resources Directly, While Other Resources Require the Waveset Gateway



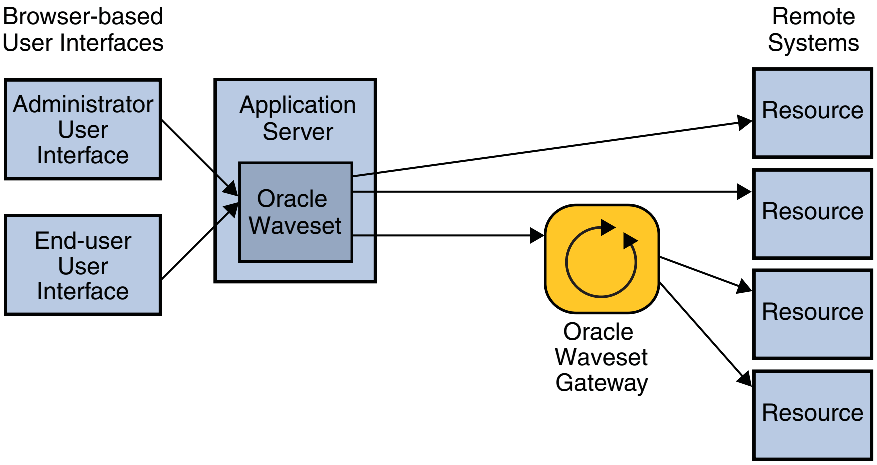
For a list of resources that Waveset supports, see "Supported Resources" in *Oracle Waveset 8.1.1 Release Notes*.

How Do Users Connect to Waveset?

Waveset has a user interface (UI) for administrators, and a separate interface for end users. To use Waveset, administrators and end users use a web browser to log on to Waveset.

- Administrators use the *administrator interface* to manage users, set up and assign resources, define rights and access levels, establish audit policies, manage compliance, and perform other business administrator and system administrator functions.
- End users use the *end-user interface* to perform a range of self-service tasks, such as changing passwords, setting answers to authentication questions, requesting access to IT systems, and managing delegated assignments.

FIGURE 1–2 Users Can Connect to Waveset Using the Administrator Interface and the End-User Interface



Companies can also use SPML (Service Provisioning Markup Language) to either create their own user interface, or integrate an existing front-end system with Waveset.

Other Waveset interfaces include the following:

- The IVR (Interactive Voice Response) telephone interface, which enables end users to perform Waveset functions using a telephone
- The Waveset IDE (Integrated Development Environment), which is used by software developers to customize Waveset
- The Waveset console, which is a command-line interface available to administrators

What is Waveset Service Provider?

Waveset Service Provider is a highly scalable, extranet-focused identity management feature that is capable of provisioning and maintaining millions of end user accounts that are stored on an LDAP directory server. The Service Provider feature can also manage thousands of administrator accounts and synchronize LDAP account data with other resources.

The Service Provider feature uses a subset of the features and functionality available in Waveset. For a detailed accounting of the differences between standard Waveset and the Service Provider feature, see "Service Provider Features" in *Oracle Waveset Service Provider 8.1.1 Deployment*.

Once available as a separate add-on product, Service Provider is now part of Waveset. Taking advantage of Service Provider functionality, however, requires special planning.

- For information on how the Waveset Service Provider system architecture, see "Understanding Waveset Service Provider System Architecture" on page 19.
- For information on planning a highly-available Waveset Service Provider architecture, see "Understanding the Recommended Service Provider HA Architecture" on page 30.
- For information on deploying Waveset to take advantage of the Service Provider feature, see *Oracle Waveset Service Provider 8.1.1 Deployment*.

CHAPTER 2

Product Architecture

This chapter provides an overview of the Oracle Waveset product architecture.

It includes the following topics:

- "Understanding Waveset Components" on page 15
- "Understanding the System Separation and Physical Proximity Guidelines" on page 18
- "Understanding SPML and the Web Services System Architecture" on page 19
- "Understanding Waveset Service Provider System Architecture" on page 19

Understanding Waveset Components

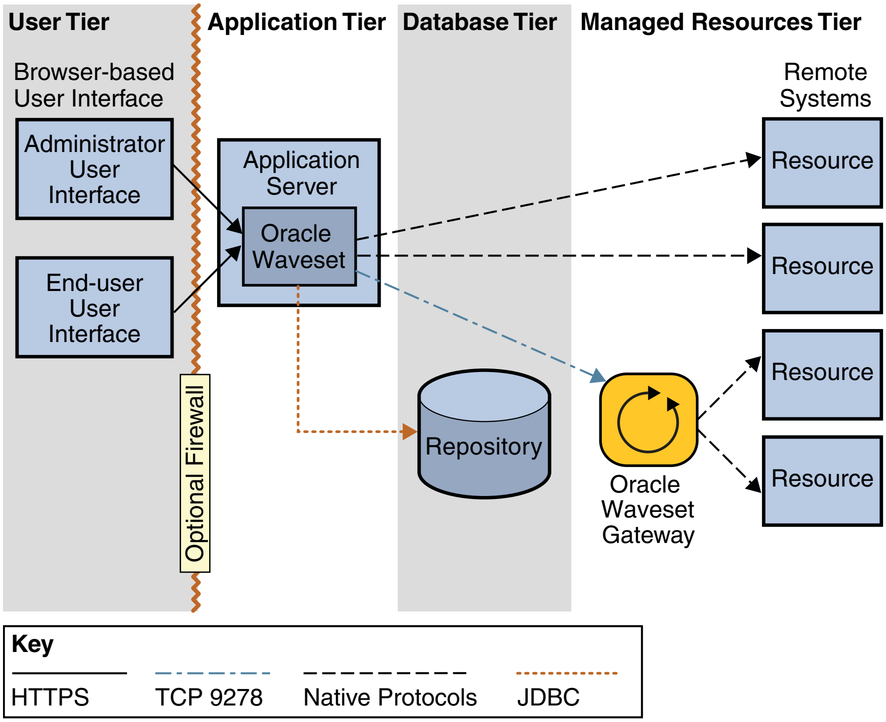
Waveset is a Java 2 Platform, Enterprise Edition (J2EE platform) web application. The J2EE platform consists of a set of industry-standard services, APIs, and protocols that provide the functionality for developing multitiered, web-based, enterprise applications.

The Waveset system architecture is distributed across four logical tiers:

- The user tier
- The application tier
- The database tier
- The managed resources tier

Each tier is discussed in the following sections, starting with the application tier.

FIGURE 2–1 Waveset System Architecture



Understanding the Application Tier

Waveset (also known as the Waveset server) is installed in a J2EE web container inside an application server. Waveset server consists of JSP files, HTML, images, and Java classes. Adapters and connectors, which interface with other IT systems (also known as *resources*), are also located in Waveset on the application server.

Note – See "Application Servers" in *Oracle Waveset 8.1.1 Release Notes* for a list of supported application servers.

Because Waveset is a web application, the user interface resides on the application server and pages are served to the user tier on a request-by-request basis.

Installing Waveset on the application server is straightforward: A graphical, wizard-based installer is provided, and, on UNIX systems, a command-line installer is also available. The application server must have a bundled or installed Java Development Kit (JDK) to run the Java classes that perform actions within Waveset.

Understanding the Database Tier

Waveset stores all of its provisioning and state information in the Waveset *repository*. The repository is comprised of tables that store all the configuration data about Waveset. It is a single point for Waveset to look up data and lock objects. The repository also contains an audit log, which is a history of actions taken in Waveset. Waveset data is stored as XML. The repository can reside in local files or a relational database, although in production, a relational database is required.

Note – See "Repository Database Servers" in *Oracle Waveset 8.1.1 Release Notes* for a list of supported database servers.

Note that, beyond a minimal amount of identity information about individual users, user data is not kept in Waveset. Instead, only those attributes that are needed to identify and differentiate users within Waveset (for example, *name* and *email address*) are saved in the repository.

Waveset can connect to the repository over a direct JDBC connection, or it can use data source functionality made available by your application server.

The Waveset Service Provider feature requires an additional LDAP repository for storing user information. See "Understanding Waveset Service Provider System Architecture" on page 19 for details.

Understanding the Managed Resource Tier

The managed resource tier consists of the applications and IT systems to which you provision and deprovision user accounts. It includes the Waveset Gateway, which is a helper application that allows Waveset to interact with certain resources.

Adapters and connectors provide user management functions, including creating, updating, deleting, and reading user accounts, and performing password change management functionality. Adapters and connectors can also extract account information from a remote system.

Note – In most cases, Waveset manages user data on the remote system and does not maintain it in its own data store.

Some common resources that require the use of the Waveset Gateway include Microsoft Exchange, Windows Active Directory, Novell eDirectory (formerly Netware Directory Services), Lotus Domino, and several others. (See "Oracle Waveset Gateway" in *Oracle Waveset 8.1.1 Release Notes* for a complete list.) The Gateway installs as a service in Windows and communicates with Waveset using TCP port 9278. Communication is initiated from Waveset using a proprietary encrypted protocol. The Gateway then interfaces with managed resources using the resources native protocols.

From an installation perspective, there are two type of adapters and connectors: *Waveset adapters and connectors* and *custom adapters and connectors*. Waveset adapters and connectors are pre-installed in Waveset. Custom adapters and connectors, however, need to be copied to a designated directory in the Waveset installation directory located on the application server.

Custom adapters are easy to create using the Waveset *Resource Extension Facility (REF)* kit. The REF kit provides the API and a number of template adapters that companies can use to jump start the development process. Simple resource functionality can be achieved by implementing only eight Java methods.

Understanding the User Tier

The user tier consists of administrators and end users who interact with Waveset through one of the user interfaces. The main user interface for the product is a web browser, which communicates with Waveset over HTTPS. The two browser-based UIs, the *administrator user interface* and the *end-user interface*, primarily consist of HTML pages, although some features may use Java applets.

For clarity, only the administrator user interface and the end-user user interface are shown in figure Figure 2–1. Other user interfaces, however, are also located in the user tier. These include the IVR telephone interface, the Waveset IDE, the SPML web services interface, and the Waveset console.

Understanding the System Separation and Physical Proximity Guidelines

This section contains basic guidelines on what Waveset components should run on what servers. It also contains recommendations on which components should be physically sited near one another in order to minimize performance issues that could arise due to latency and network congestion.

Note – Only basic guidelines are provided. For information on designing a high-availability Waveset architecture, see Chapter 3, “Clustering and High Availability.”

In a development environment, the application server and database can reside on the same machine. In testing and production environments, however, each Waveset instance should be installed on its own dedicated server. The relational database also requires a dedicated server.

The Waveset Gateway, if required, must be installed on one or more Windows machines. The Gateway is a lightweight component and does not require a dedicated server. All Windows domains managed by a Gateway must be part of the same forest. Managing domains across forest boundaries is unsupported. If you have multiple forests, install at least one Gateway in each forest. In production the Gateway must be made highly available. See "Making the Gateway Highly Available" on page 27 for details.

In a production environment, the highest amount of network traffic occurs between the database and application servers. These two environments must be on the same LAN with the shortest network hop possible. Gateway instances, as well as managed resources, do not need to be on the same network as Waveset.

If Waveset will be used for external users in a Service Provider configuration, a set of web servers should be setup in a DMZ. See "Understanding the Recommended Service Provider HA Architecture" on page 30 for details.

Understanding SPML and the Web Services System Architecture

Service Provisioning Markup Language (SPML) and Waveset Web Services can be used to implement a custom front-end for Waveset. Waveset sends and receives SPML messages and responses using the HTTPS protocol.

For more information about SPML and Web Services, see *Oracle Waveset 8.1.1 Web Services*.

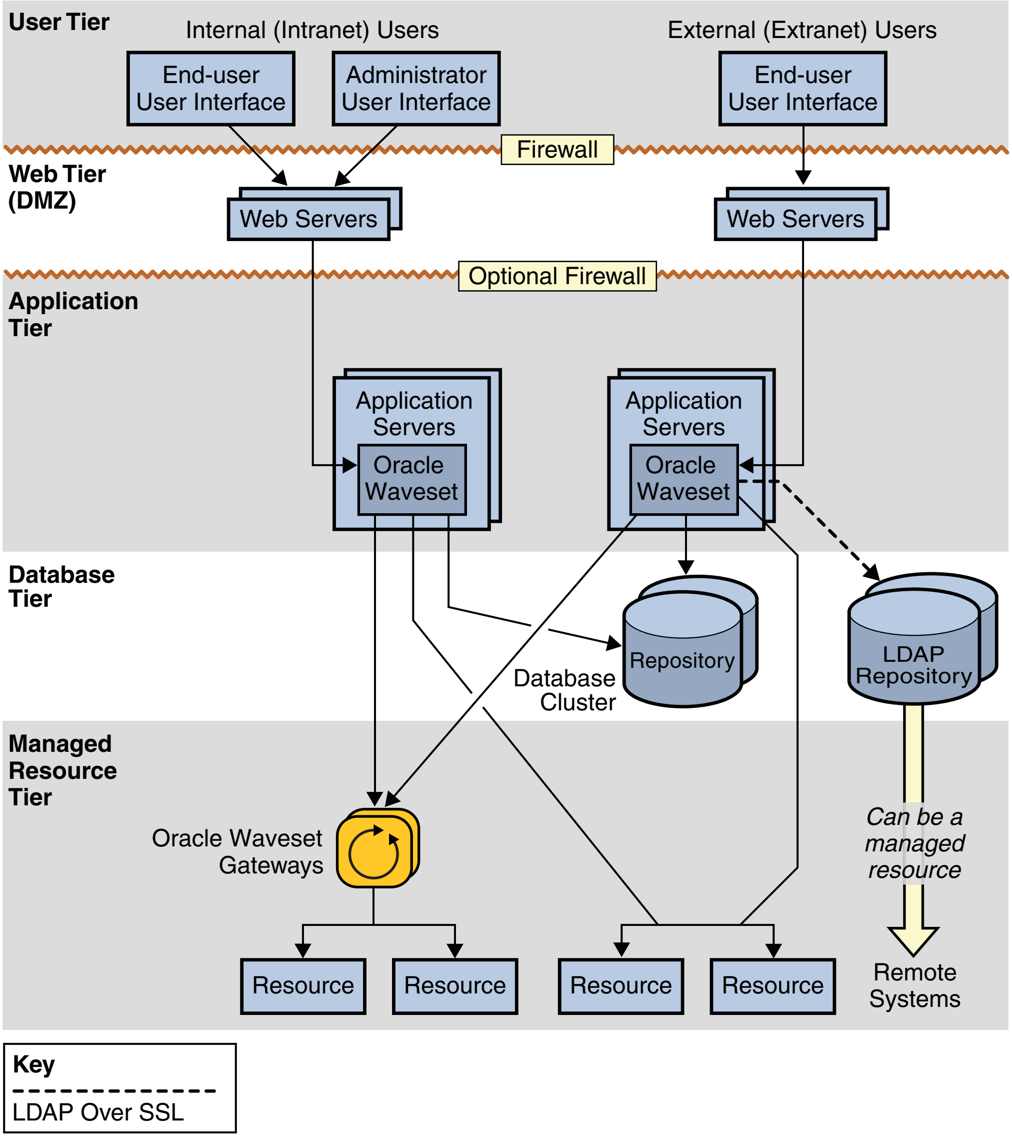
Understanding Waveset Service Provider System Architecture

If the Waveset Service Provider feature is implemented, a fifth tier is required. This tier is called the Web tier and it consists of one or more web servers located in a DMZ. No Waveset components are installed in the web tier. Instead, the web servers in the DMZ support one or more application servers in the application tier by responding to web page requests. Adding one or more web servers to the web tier provides enhanced scalability, and placing the web servers in a DMZ provides better network security.

The Service Provider feature also requires an LDAP repository. This repository resides in the database tier. Because the LDAP repository can be a managed resource, the LDAP server can be understood as residing in the managed resource tier, as well.

Note – In a service-provider-only implementation, a Waveset repository is required in addition to the LDAP repository in order to maintain configuration information, forms, rules, and other objects.

FIGURE 2–2 Waveset Service Provider System Architecture



CHAPTER 3

Clustering and High Availability

This chapter provides guidance on how to implement a high availability / fault tolerant (HA/FT) Waveset environment.

Note – Please consult your web server, application server, and database provider's documentation for best practices on ensuring a highly available deployment with each technology. This guide is not a substitute for the vendor-specific recommendations for web servers.

Assessing the Need for Availability

This section describes how to assess the amount of availability that your specific deployment requires.

Assessing the Cost of Downtime

Because Waveset is not in the transaction path between general users and the systems and applications that they already have access to, Waveset downtime is not the disaster that you might imagine. If Waveset is unavailable, end users are still able to access resources through their provisioned accounts.

The main cost of Waveset downtime is lost productivity. If Waveset is down, end users cannot use Waveset to gain access to systems that they are either locked out of or not provisioned to.

To calculate the cost of downtime, the first number that is needed is the average cost of lost productivity due to end users being unable to access computing resources within the enterprise. In our assessment, this number is called *productivity per person hour*.

The other major number that needs to be determined is the percentage of end users within the user population who need to use Waveset at any given time. This population usually includes new hires who need to be provisioned, and end users who have forgotten their password if password management is a part of the deployment.

Consider the following hypothetical situation:

Total number of employees	20,000
Number of password resets in a day	130
Number of new hires in a day	30
Number of hours in a work day	8

For this particular situation you can calculate the following:

- The number of employees needing Waveset at any given hour = (130 + 30) / 8 = 20
- The percentage of employees needing Waveset at any given hour = 20 / 20,000 = .1% or 1 in 1000

Using these numbers you can then estimate the cost of a Waveset outage:

Productivity per person hour	$100	
Loss in productivity	.5	(50% decrease in productivity due to inability to access system)
Number of people affected	20	
Subtotal	$1,000	
Duration of outage	2 hours	
Total immediate loss	$2,000	

This example shows that even though the number of users being managed by Waveset is high, the number of users needing Waveset to gain access to systems at any given time is usually low.

Another point to consider is that the time it takes to bring a system like Waveset back online is usually less than the time it takes to execute the manual provisioning processes that Waveset is automating. So while Waveset downtime exacts a cost, it is usually less than the cost of using manual processes to give users access to resources.

Understanding the Causes of Downtime

When planning for a Waveset highly-available deployment, it is worthwhile to consider the causes of downtime.

These causes include the following:

- Operator error
- Hardware failure
- Software failure
- Planned down time (Upgrades to hardware and software)
- Poor performance (Perceived downtime)

Calculating Return on Investment

Waveset automates processes and reduces lost productivity. The return on investing in a highly-available Waveset architecture is realized by minimizing downtime and averting lost productivity.

You can use the cost of downtime to determine the amount of availability that is ultimately needed for Waveset. In general, a moderate investment in making Waveset highly-available is worthwhile.

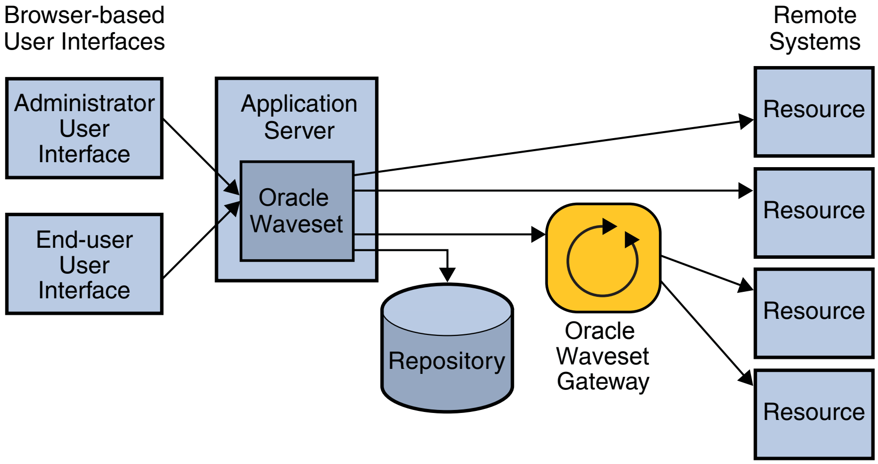
When calculating the cost of your investment, remember that purchasing HA/FT hardware and software is only one part of implementing an available solution. Having a knowledgeable staff to keep it up and running is another cost.

Understanding the Waveset High-Availability Feature Set

Waveset is designed to leverage HA infrastructure if it is available. For example, Waveset does not require an application server cluster to achieve high availability, but it can utilize a cluster if it exists.

The following diagram shows the major Waveset components deployed in a non-redundant architecture. The sections that follow will describe how the Waveset repository, application server, and gateway can be made highly-available.

FIGURE 3–1 Waveset Standard System Architecture



Note – See "Understanding the System Separation and Physical Proximity Guidelines" on page 18 for information on which components should be physically sited near one another in order to minimize performance issues that could arise due to latency and network congestion.

Making the Repository Highly-Available

Waveset stores all of its provisioning and state information in the Waveset repository.

The availability of the database instance storing the Waveset repository is the most critical piece to achieving a highly available Waveset deployment. The repository is the representation of the entire Waveset installation and the data within it must be protected as with other important database applications. At minimum, regular backups must be performed.

Note – Do not host the Waveset repository on a virtual platform such as a VMware virtual machine because performance (transactions per second) will be hindered significantly.

There can only be one image of the repository. It is not possible to have two separate databases for Waveset and attempt to synchronize them nightly. Oracle recommends using your database's clustering or mirroring capabilities to provide fault tolerance.

Making the Application Server Highly Available

Waveset can run within an application server cluster and take advantage of the added availability and load balancing that a cluster provides. Waveset does not use any J2EE features that require clustering, however.

Waveset uses the HTTP Session object that is available through the Servlet API. This session object tracks a user's visit as the user logs in and performs actions. In a cluster, you can optionally have multiple nodes handle a user's requests during a given session. This is usually not recommended, however, and most installations are configured to send a user's entire request for a given session to the same server.

It is possible to add additional availability and capacity to the application server running Waveset even if you do not set up a cluster. This is achieved by installing multiple application servers with Waveset, connecting them to the same repository, and putting a load balancer with *session affinity* in front of all the application servers.

Note – For more information on session affinity, see "Frequently Asked Questions Regarding Session Affinity and Session Persistence" on page 38.

Waveset runs certain tasks in the background—for example, scheduled reconciliation tasks. These tasks are stored in the database and can be picked up by any Waveset server to run. Waveset uses the database to ensure that these tasks are always run to completion, even if it has to fail over to another node.

Configuring Active Sync Clustering on Application Server Nodes

The `sources.hosts` setting in the `Waveset.properties` file controls which hosts in a multi instance environment are used for executing Active Sync requests. This setting provides a list of hosts that source adapters can run on. Setting this to *localhost* or *null* will allow source adapters to execute on any host in the web farm. (This is the default behavior.) By listing one or more hosts, you can restrict execution to that list. If you have inbound updates from another system that go to a particular host, use the `sources.hosts` setting to record the host names.

In addition, you can define a property named `sources.`*resourceName*`.hosts`, which controls where the resource's Active Sync task will run. Replace *resourceName* with the name of the resource object you wish to specify.

Making the Gateway Highly Available

Waveset requires a lightweight gateway to manage resources that cannot be directly accessed from the server. These include systems that require client-side API calls that are platform specific. For example, if Waveset is running on a UNIX-based application server, the ability to make NTLM or ADSI calls to managed NT or Active Directory domains is not possible. Because Waveset requires a gateway to manage these resources, it is important to ensure that the Waveset Gateway is made highly available.

To prevent the Gateway from becoming a single point of failure, Oracle recommends having multiple machines running a Gateway instance. A network routing device or load balancer should be configured to provide failover if the main Gateway instance dies. The failover device should be configured to use sticky sessions. A failover device without sticky sessions is not a supported configuration and will cause certain Waveset functions to fail.

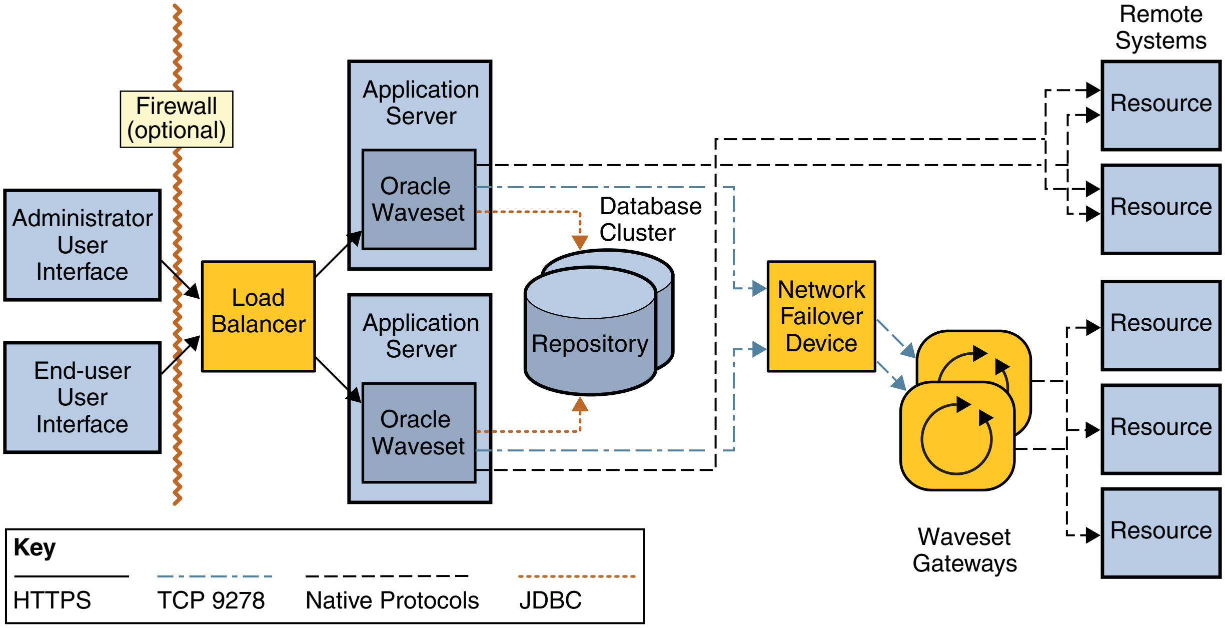
All Windows domains managed by a Gateway must be part of the same forest. Managing domains across forest boundaries is unsupported. If you have multiple forests, install at least one Gateway in each forest.

Win32 monitoring tools can be configured to watch the `gateway.exe` process on the Win32 host. In the event that `gateway.exe` fails, the process can be automatically restarted.

Understanding the Recommended HA Architecture

The following diagram shows the Waveset architecture Oracle recommends if there is no existing web application infrastructure.

FIGURE 3–2 Waveset High-Availability Architecture



In an actual deployment, existing redundant application server infrastructure should be utilized to the extent possible. The value of this architecture is that it only uses load balancers for achieving redundancy at the application server. Load balancers with session affinity detect failed application server instances and failover to active instances. Load balancers are also used to provide horizontal scaling in the web environment by spreading the user requests across a cluster of servers.

Though this is a straightforward architecture, the uptime characteristics are comparable to more complex deployments. Because of its simplicity, there are fewer pieces of software to maintain and monitor or fewer pieces that could fail. Because human error is the number one cause of downtime, a relatively simple solution may achieve better uptime characteristics than something more complex. There are no universal right answers. The point is to understand all of the causes of downtime and choose the architecture that will result in the best availability for the investment.

Note – It would be impossible to describe all of the different HA architectures that are possible with a web application like Waveset.

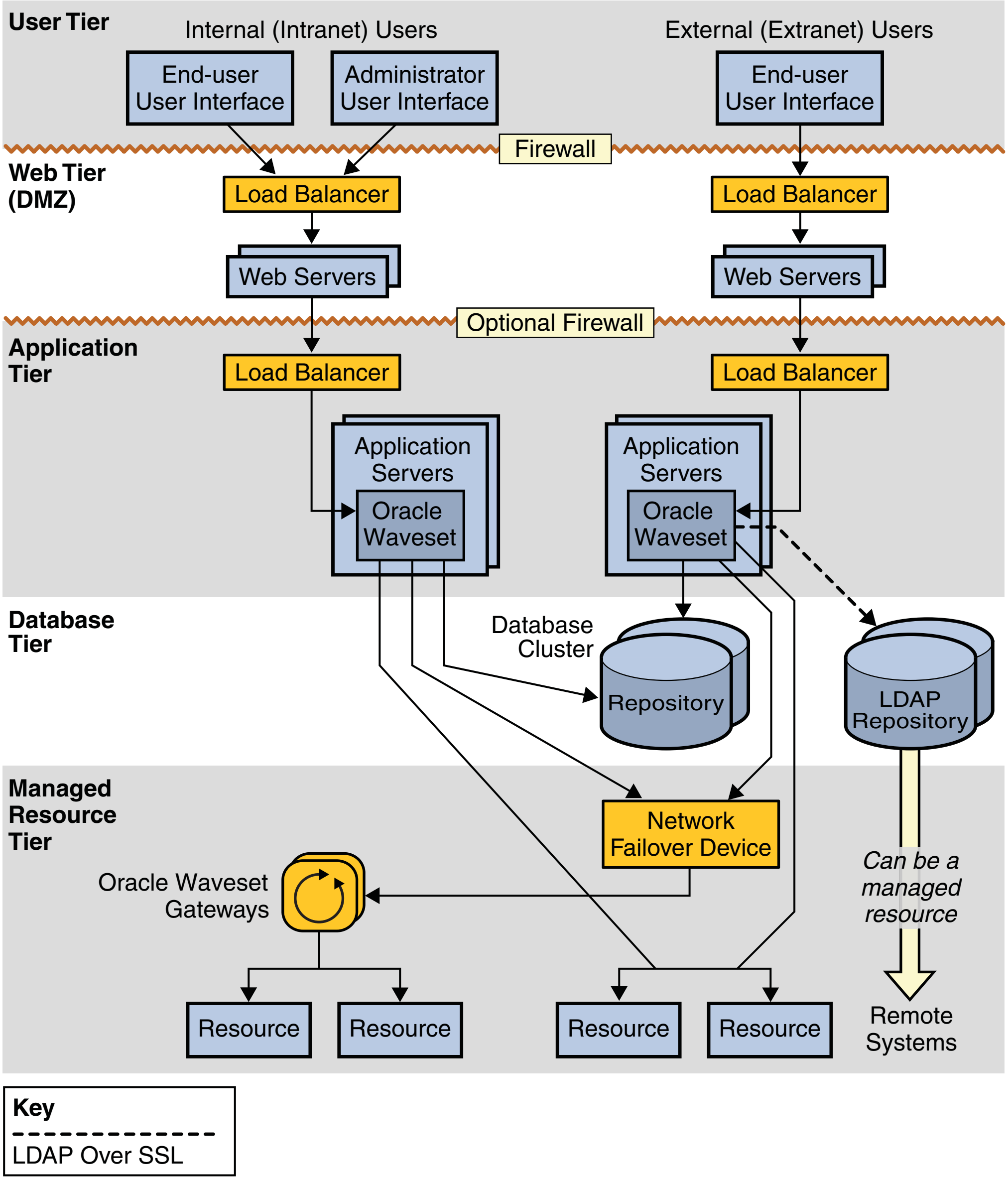
Because Waveset can be deployed in a variety of possible combinations, it may be most economical to identify existing infrastructure and utilize as much of it as possible when deploying Waveset.

Understanding the Recommended Service Provider HA Architecture

If Waveset Service Provider functionality is to be utilized, Oracle recommends adding a web tier between the user tier and the application tier. The web tier consists of one or more web servers that reside in a demilitarized zone (DMZ) that is separated by a firewall from the application tier.

An LDAP repository is required if Service Provider functionality is to be utilized. In addition, the standard Waveset repository is always required for Service Provider in order to maintain configuration information, forms, rules, and other objects.

FIGURE 3–3 Waveset Service Provider High-Availability Architecture



Understanding Failure Scenarios

This section lists eight failure scenarios and compares two deployments, one with session persistence, and one without.

- The deployment *with* session persistence has session affinity across a load balancer. The deployment has multiple instances in a cluster which have some form of session persistence turned on such that session changes are written to a physically distinct repository node.
- The deployment *without* session persistence has session affinity across a load balancer , and has multiple instances that are not part of a cluster.

Scenario 1: The No-Workflow Scenario

Scenario Description

The end user or the administrator is editing a form that is not a part of a workflow. The instance on which the user has an established session goes down.

Without Session Persistence

User experience: A nontransparent failover. Upon submitting the form, the user is returned to the login page.

Recovery steps: The user reenters his or her user name and password. Waveset then processes the form and presents the results as the next page immediately following the login.

With Session Persistence

User experience: The user's form is submitted and the results are returned without the user being logged off and required to log in again.

Recovery steps: No user action is needed.

Other Scenario Examples

- An end user has logged in and retrieved search results for users or other repository objects when the instance goes down.
- An administrator is about to submit a “reset password” or “edit user” request using the Administrator interface when the instance goes down.

Scenario 2: The Workflow-in-Progress Scenario

Scenario Description

The end user or the administrator has submitted a form that triggered a workflow. The instance on which the workflow is executing and on which the user's session exists is generally going to be the same except in case of some scheduled tasks where they can be different. This instances goes down while the workflow is in progress.

Without Session Persistence

User experience: A nontransparent failover. The form submit returns the user to the login page. The workflow task instance being executed should be in the repository, but because the node of execution is down, the workflow status will be “terminated.”

Recovery steps: The workflow has to be submitted again. This has to be done by going back to the same form and reentering the same information that was used to trigger the workflow before the node failed.

The submission of the same request data may work in some cases, but not all. If the workflow provisions to more than one resource during its execution and some of these resources were provisioned before the failure, the workflow resubmission from the user would have to account for the “already provisioned to” resources. Note that the terminated workflow sticks around in the repository until the `resultLimit` expires on the `TaskInstance` object.

With Session Persistence

User experience: A nontransparent failover. The user does not get logged out because his session is persisted and reestablished in the new instance. The form submit, however, will probably result in an error because the workflow will be terminated. This failover is nontransparent because recovery actions are needed.

Recovery steps: Same as in the Without Session Persistence mode. The user has to resubmit the request that triggered off the previous workflow with the same or modified parameters.

Other Scenario Examples

- An end user has just submitted a self-registration request to create a Waveset account and the instance goes down.
- An administrator has just submitted a “password reset” request that is in progress and the instances goes down.

Scenario 3: The Workflow-in-Abeyance-or-Sleep Scenario

Scenario Description

This scenario covers situations where the workflow has started, but is waiting for a manual action by an approver.

Without Session Persistence

User experience: Transparent failover with respect to the approver provided that the approver has not yet logged in. After the node failure, when the approver does log in, the approver will still see the approval request in his or her inbox, even though the request was triggered from a node that is no longer up.

Recovery steps: No user action is needed.

With Session Persistence

User experience: Same as in the Without Session Persistence mode.

Recovery steps: Same as in the Without Session Persistence mode.

Other Scenario Examples

- The workflow is in a sleep state, for example a manual action that sleeps until a sunrise or sunset date for an employee.
- An administrator submitted a user creation request that is waiting on an approver to log in and approve the request. The node from which the request was sent failed before the approver approved the request.

Scenario 4: The Work-Item-Edit Scenario

Scenario Description

This scenario includes cases where a user is editing a work item and the node upon which the user has a session goes down before the work item can be submitted.

Without Session Persistence

User experience: A nontransparent failover. When the work item edit form is submitted, the user is logged off and returned to the login page.

Recovery steps: Upon resubmitting login credentials, the user's work item is marked completed and the workflow can resume from that point. The workflow should be picked up by the new mode for execution from the point where the user's manual action is marked completed.

With Session Persistence

User experience: When the work item edit form is submitted, the user sees the effect of his submission—for example, either the next form in the custom workflow if there is one, or a success message.

Recovery steps: No user action needed.

Other Scenario Examples

- An end user is filling out a form associated with a manual action in a custom workflow, for example requesting access to specific resources. Before the user can submit the request, the node the user has a session on dies.
- An administrator has logged in to Waveset and has opened up an approval request for editing. Before the request can be submitted, the node the administrator has a session on fails.

Scenario 5: The Scheduled-Tasks-in-Progress Scenario

Scenario Description

These scenarios cover cases where a node failure occurs while a reconciliation is in progress or while a report is executing.

Without Session Persistence

User experience: The scheduled task terminates in process.

Recovery steps: The scheduled task that was in progress has to be restarted. The task will start from the beginning. (The task will not restart from the point of failure.) This is akin to creating and starting a new task.

With Session Persistence

User experience: Same as in the Without Session Persistence mode.

Recovery steps: Same as in the Without Session Persistence mode.

Other Scenario Examples

- An Active Sync adapter is configured to run on the failed node.

Scenario 6: The Scheduled-Task-in-Abeyance Scenario

Scenario Description

These scenarios cover cases where a user's custom workflow has scheduled a task for execution at a later date on a specific node. Before the scheduled date is reached, the node that the task was scheduled on fails.

Without Session Persistence

User experience: The failover is transparent with respect to the recovery actions required to ensure that this task executes at its scheduled time.

Recovery steps: The scheduled task is taken up by any live node when the scheduled execution time arrives.

With Session Persistence

User experience: Same as in the Without Session Persistence mode.

Recovery steps: Same as in the Without Session Persistence mode.

Other Scenario Examples

- In the process of a user's account creation, the Deferred Task Scanner is used to implement enabling an account on a sunrise date or to implement disabling the account on a sunset date. Before the sunrise or sunset date arrives, the node that the task was scheduled on fails.
- A report is scheduled to run at a future time, or a reconciliation is scheduled to run at a specific time and, before the time is reached, the node the task was scheduled on fails.

Scenario 7: The Web-Services-Workflow-Request-Not-Yet-Received-by-Waveset Scenario

Scenario Description

These scenarios cover those cases where the Waveset GUI is not used to launch provisioning. Instead, the user interface is provided by an application that internally calls to Waveset using the SPML or other custom web service interface. Here the user session related to the user going through the UI is managed by way of the calling application. For Waveset, the requests are all launched as the “soapadmin” subject.

In such a use case, this failure scenario covers the case where the request by way of the Waveset endpoint has not been received yet and the targeted node fails.

Without Session Persistence

User experience: A transparent failover. The SOAP administrator's credentials are passed in for each SOAP request, either over the wire or within Waveset through a `Waveset.properties` setting. As long as the node which was to receive this SOAP request has not received the request before going down, the failover is transparent with or without session persistence.

Recovery steps: No action needed. The SOAP request is sent to a live node that executes it.

With Session Persistence

User experience: Same as in the Without Session Persistence mode.

Recovery steps: Same as in the Without Session Persistence mode.

Scenario 8: The Web-Services-Workflow-Request-In-Progress-by-Waveset Scenario

Scenario Description

This scenario is similar to scenario seven. The only difference is that the workflow is in progress when the node fails, or the node has received the SOAP request when the node fails.

Without Session Persistence

User experience: This scenario is similar to scenario two (workflow in progress). The workflow is marked terminated and the user sees an error as a result of the SOAP request.

Recovery steps: The user has to resubmit the form with similar or modified parameters (based on where the failure occurs in the workflow) using the user interface in the third-party application.

With Session Persistence

User experience: Same as in the Without Session Persistence mode.

Recovery steps: Same as in the Without Session Persistence mode.

Frequently Asked Questions Regarding Session Affinity and Session Persistence

Should you have session affinity enabled when scaling application servers horizontally?

Yes.

Should you have session persistence when scaling application servers horizontally?

Unless your business requirements place a very high emphasis on having transparent failover in the limited situations where session persistence can make a difference, Oracle recommends against using session persistence. Session persistence has its own performance overhead and, unless transparent failovers are absolutely mandated by your business requirements, leave session persistence turned off.

If you study the failure scenarios documented in “Understanding Failure Scenarios” on page 32, in six of the eight scenarios there is no difference in the end-user experience or required recovery actions, regardless of whether session persistence is enabled. Only in scenarios one and four are there any difference between the session-persistence scenarios as opposed to the no-session-persistence scenarios.

In these two scenarios, session persistence can provide some failover transparency, but session persistence hurts performance. Based on the size of the session objects, the repository being used for session persistence, and the optimization of your specific application server's session management code, the performance overhead can range from 10 percent to 20 percent or even higher.

Should you have multiple application server instances in a cluster when scaling horizontally?

Multiple application server instances are not absolutely needed unless you want session persistence. Fail-over without session persistence can be achieved even if all application server nodes are not in a cluster.