Peregrine

Get-Answers

Administration Guide

PART NO: GET-ANS-3.0-01014-00195

Peregrine SYSTEMS®

The information in this document is subject to change without notice and does not represent a commitment on the part of Peregrine Systems, Inc. Contact Peregrine Systems, Inc., Customer Support to verify the date of the latest version of this document.

The names of companies and individuals used in the sample database and in examples in the manuals are fictitious and are intended to illustrate the use of the software. Any resemblance to actual companies or individuals, whether past or present, is purely coincidental.

If you have comments or suggestions about this documentation, please send e-mail to Peregrine Systems Customer Support.

This edition applies to version 3.0 of the licensed program.

Peregrine Systems, Inc.
Worldwide Corporate Campus and Executive Briefing Center
3611 Valley Centre Drive San Diego, CA 92130
Tel 800.638.5231 or 858.481.5000
Fax 858.481.1751
www.peregrine.com

Table of Contents

CHAPTER 1

Controlling Document Access

Organizations need to create and manage collections of individual documents, providing the ability to assign documents to or remove them from one or more collections. In Get-Answers, you publish documents into Domains. You also define Document Ownership Teams to manage the documents. Within Document Ownership Teams, you can assign Permissions to users and groups of users (called Roles) that determine their access rights to documents in the collection.

Topics in this chapter include:

- *Users* on page 8.
- *Roles* on page 8.
- *Domains* on page 8.
- *Document Ownership Teams* on page 8.
- *Anonymous users* on page 10.
- *Managing users* on page 10.
- *Managing roles* on page 13.
- *Managing domains* on page 14.
- *Managing document ownership teams* on page 17.

Users

Each user of Get-Answers will have a user account. Alternatively, a user can use the “anonymous” user account. For information on anonymous users, see *Anonymous users* on page 10.

Roles

Every user can occupy one or more Roles in Get-Answers. A Role is any collection of users. Permissions to access the system are assigned to users and/or Roles.

Roles are defined by their Capability collections. Each unique Capability gives a user access to a specific function. For example, a Capability to conduct searches gives the user access to the various search pages.

Domains

Documents are published into Domains. Domains can contain any number of documents. A document can belong to any number of Domains. However, a new document can only be submitted to one Domain initially. It then is managed by the Domain’s associated Ownership Team. A Domain is owned by one Ownership Team. Within Domains, you can control the document hierarchy by creating Categories for your documents.

Within Domains, there are two Permissions. A Domain Member has read access to any document published into the Domain. A Domain Submitter has read access to any document published into the Domain and can submit new documents to the Domain for review and possible publication by the owning Document Ownership Team.

Document Ownership Teams

Document Ownership Teams manage documents. One, and only one, Document Ownership Team owns a given document. You can transfer ownership of a document from one Ownership Team to another, but a document can never belong to more than one Document Ownership Team at a time.

You can associate Document Ownership Teams with any number of domains, but there is a single Document Ownership Team for each domain. This Document Ownership Team becomes the owner of any new documents submitted to the Domain.

When a new document is submitted to a Domain, the Domain's Ownership Team is assigned as the ownership team of the document. Users with the Owner Permission within the Document Ownership Team can transfer ownership of the document to another Document Ownership Team, if appropriate.

Seven Capabilities are defined within Document Ownership Teams. These Capabilities are used to define four Permissions for Document Ownership Teams. You can associate Users or Roles with one or more of the four Permissions in a Document Ownership Team.

The Capabilities for Document Ownership Teams include:

- Read — Can read working copies of documents
- Create — Can create new documents
- Update — Can check documents out, edit them, and check them back in
- Revert — Can undo an editing lock on a document they hold an editing lock on
- Retire — Can retire documents
- Publish — Can publish documents into any of the Domains associated with the Document Ownership Team
- Transfer — Can transfer ownership of a document to another ownership

The four Permissions and their Capabilities within a Document Ownership Team include:

- Owner — Read, Create, Update, Revert, Retire, Publish, and Transfer capabilities
- Author — Read, Create, and Update capabilities
- Reviewer — Read, Update, and Unlock capabilities
- Reader — Read capability only

In addition, Owners, Editors, and Reviewers have particular responsibilities in the Editorial and /or Triage workflows. Readers play no part in the workflows. For information on workflows, see the *Get-Answers Workflow Guide*.

Anonymous users

You can configure an “anonymous” user in Get-Answers. You will assign permissions to the anonymous user, which will define the access allowed to users who log in as anonymous users. The only difference between the anonymous user and any other user account is the name which is reserved and must be "anonymous".

Managing users

An Administrator can add, modify, and delete users from Get-Answers. You can also assign users to roles. Assign as many users as you want to a role and then you can give that role Domain and/or Document Ownership Team Permissions.

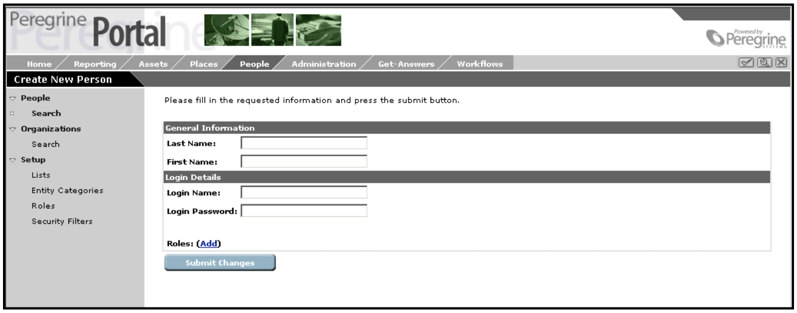
You will see menu options that are used by other Peregrine web applications in the People module, which is where Get-Answers users are managed. Only the menu options used by the Get-Answers web application are documented here.

Adding a user

To add a user:

1. Click the **People** tab.
2. Click **New** to add a new user.

   The Create New Person page displays.



Create New Person

Please fill in the requested information and press the submit button.

General Information

Last Name:

First Name:

Login Details

Login Name:

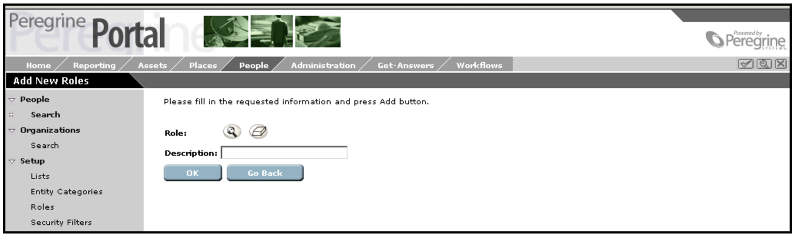
Login Password:

Roles: (Add)

3. Enter the **Last Name**, **First Name**, and **Login Name**, and **Login Password** for the user.

4 To assign a user to a role, click **Add** next to Roles.

The Add New Roles page displays.

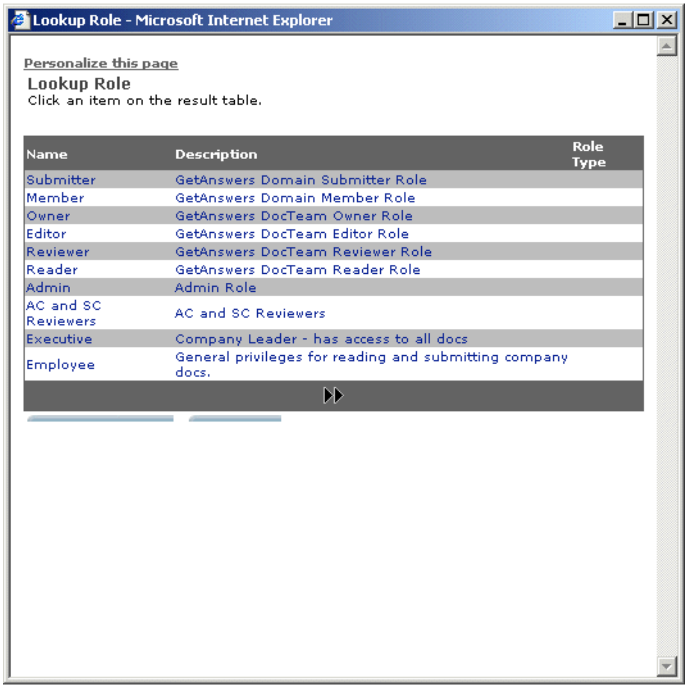


Please fill in the requested information and press Add button.

Role:

Description:

5 Click the **Magnifying Glass** icon to open the list of roles.



Lookup Role

Click an item on the result table.

Name	Description	Role Type
Submitter	GetAnswers Domain Submitter Role	
Member	GetAnswers Domain Member Role	
Owner	GetAnswers DocTeam Owner Role	
Editor	GetAnswers DocTeam Editor Role	
Reviewer	GetAnswers DocTeam Reviewer Role	
Reader	GetAnswers DocTeam Reader Role	
Admin	Admin Role	
AC and SC Reviewers	AC and SC Reviewers	
Executive	Company Leader - has access to all docs	
Employee	General privileges for reading and submitting company docs.	

6 From the list, select the desired role. The role will appear on the Add New Roles page.

Note: Do not add users to any of the Get-Answers roles directly on this page. These “roles” are placeholders for Domain and Document Ownership Team Permissions. Users must be given these permissions within the context of a specific Domain or Document Ownership Team. This is done in the Manage Domains and Manage Document Ownership Teams portions of the Get-Answers user interface.

7. Enter a description for the role and click **OK**.

   The role appears on the Create New Person page so you know the role (or group) that user belongs to.

8. Click **Submit Changes** to add the user.

Modifying a user

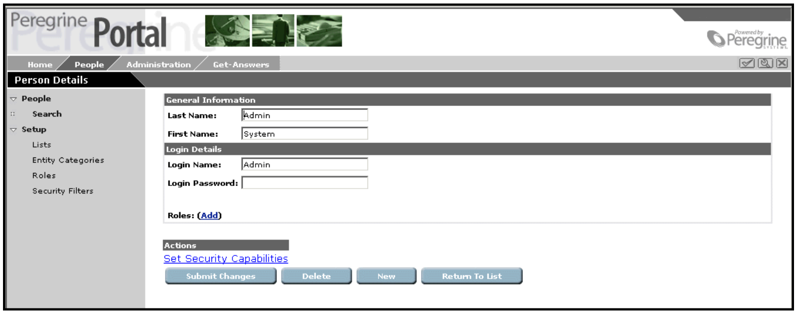
To modify a user:

1. Click the **People** tab.
2. You can search on the user’s First Name, Last Name, and Login Name. Then click **Search**.

   Note: You can click **View All** without entering the user’s name for a list of all users.

3. Select the desired user name you want to modify.

   The Person Details page displays.



General Information

Last Name: Admin

First Name: System

Login Details

Login Name: Admin

Login Password:

Roles: (Add)

4. Make the desired modifications.
5. Click **Submit Changes** when you finish.

Deleting a user

To delete a user:

1 Click the **People** tab.

2 You can search on the user’s First Name, Last Name, and Login Name. Then click **Search** to find the user you want to delete.

Note: You can click **View All** without entering the user’s name for a list of all users.

3 Select the user you want to delete.

4 In the Person Details page, click **Delete** to remove the user from Get-Answers.

Managing roles

An Administrator can create, modify, and delete groups of users by assigning them roles.

You will see menu options that are used by other Peregrine web applications in the People module, which is where Get-Answers users are managed. Only the menu options used by the Get-Answers web application are documented here.

Adding a role

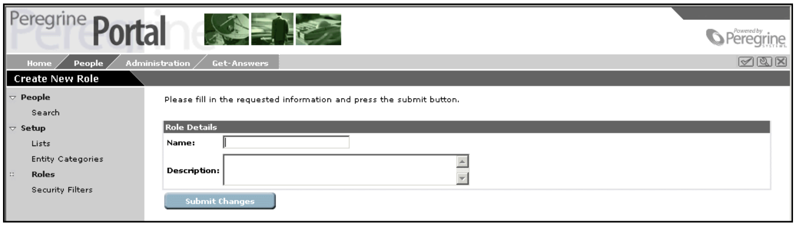
To add a role:

1 Click the **People** tab.

2 Click **Roles** in the menu options under Setup.

3 Click **New** to create a new role.

The Create New Role page displays.



Please fill in the requested information and press the submit button.

Role Details

Name:

Description:

4 Enter a name for the role.

5 Enter a description for the role.

6 Click **Submit Changes** when you finish.

Modifying a role

To modify a role:

1 Click the **People** tab.

2 Click **Roles** in the menu options under Setup.

3 Enter the role name and the description. Then click **Search**.

Note: You can click **View All** without entering the name for a list of all roles.

4 Select the role you want to modify.

5 In the Role Details page, make the desired modifications and click **Submit Changes**.

Deleting a role

To delete a role:

1 Click the **People** tab.

2 Click **Roles** in the menu options under Setup.

3 Enter the role name and the description. Then click **Search**.

Note: You can click **View All** without entering the name for a list of all roles.

4 Select the role you want to delete.

5 In the Role Details page, click **Delete** to remove the role.

Managing domains

Using the Manage Domains menu option, an Administrator can add, modify, and delete domains from Get-Answers.

Adding a domain

To add a domain:

1 Click the **Get-Answers** tab. Then click **Manage Domains** in the menu options.

2 Click **Create New Domain** to add a new domain.

3 Enter a **Name** for the domain.

4 Enter a description of the domain in the Description field.

5 Select the desired **Managing Document Team** for the domain.

6 Click **Add Domain**.

The Domain Detail page displays. The Name, Description, and Document Team fields are filled in.

7 Under Domain Permissions, click one of the default permissions to assign roles (group of users) or individual users to that particular permission, either a Submitter or a Member.

The Permission Detail page displays.

8 Click **Add Role** to assign a role (group of users) to this permission.

9 In the Search for Roles page, enter the role name you want to add and click **Search**.

10 Click the role and it will be appear on the Permission Detail page. Do this for as many roles as you want to add.

11 Click **Add User** to assign an individual user to this permission.

12 In the Search for Users page, enter the name of the user you want to add and click **Search**.

13 Click the user name and it will appear on the Permission Detail page. Do this for as many users as you want to add.

14 Return to the Domain Detail page.

15 To control document hierarchy, you can add categories to the domain. To add a new category, enter a name in the Category field and click **Add Category**.

The new category will appear in the list of Domain Categories.

16 To delete a category, select the desired category from the list and click **Delete Category**.

17 Click **Save Changes** when you finish.

Modifying a domain

To modify a domain:

1 Click the **Get-Answers** tab. Then click **Manage Domains** in the menu options.

2 Enter the domain name. Then click **Search** to find the domain you want to modify.

Note: You can click **Search** without entering the domain name for a list of all domains.

3 Select the desired domain. In the Domain Detail page, make the desired modifications and click **Save Changes**.

Note: For details on the domain options, see *Adding a domain* on page 14.

Deleting a domain

To delete a domain:

1 Click the **Get-Answers** tab. Then click **Manage Domains** in the menu options.

2 Enter the domain name. Then click **Search** to find the domain you want to delete.

Note: You can click **Search** without entering the domain name for a list of all domains.

3 Select the desired domain. In the Domain Detail page, click **Delete Domain** to remove the domain.

Managing document ownership teams

Using the Manage Document Ownership Teams menu option, an Administrator can add, modify, and delete ownership teams from Get-Answers.

Adding a document ownership team

To add a document ownership team:

1. Click the **Get-Answers** tab. Then click **Manage Document Ownership Teams** in the menu options.
2. Enter a Name for the document team. Then click **Add Document Team**.

   The Document Team Detail page displays. The Name field is filled in.
3. Under Document Team Permissions, click one of the default permissions (Owner, Editor, Reviewer, Reader) to assign roles (group of users) or individual users to that particular permission.

   The Permission Detail page displays.
4. Click **Add Role** to assign a role (group of users) to this permission.
5. In the Search for Roles page, enter the role name you want to add and click **Search**.
6. Click the role and it will be appear on the Permission Detail page. Do this for as many roles as you want to add.
7. Click **Add User** to assign an individual user to this permission.
8. In the Search for Users page, enter the name of the user you want to add and click **Search**.
9. Click the user name and it will appear on the Permission Detail page. Do this for as many users as you want to add.
10. Return to the Document Team Detail page.
11. Click **Save Changes** when you finish.

Modifying a document ownership team

To modify a document ownership team:

1. Click the **Get-Answers** tab. Then click **Manage Document Ownership Teams** in the menu options.

2 Enter the document team name. Then click **Search** to find the document team you want to modify.

Note: You can click **Search** without entering the document team name for a list of all ownership teams.

3 Select the desired document team. In the Document Team Detail page, make the desired modifications and click **Save Changes**.

Note: For details on document team options, see *Adding a document ownership team* on page 17.

Deleting a document ownership team

To delete a document ownership team:

1 Click the **Get-Answers** tab. Then click **Manage Document Ownership Teams** in the menu options.

2 Enter the document team name. Then click **Search** to find the document team you want to delete.

Note: You can click **Search** without entering the document team name for a list of all document teams.

3 Select the desired document team. In the Document Team Detail page, click **Delete Document Team** to remove the document team.

2 CHAPTER Customizing Get-Answers

This chapter discusses how to customize specific areas of Get-Answers. You can modify the stop word list, create custom dictionaries, and customize the spelling checker used with the RTF editor.

Topics in this chapter include:

Modifying the stop words list

Get-Answers uses a standard stop words file (rw_english.slx) containing the most commonly used function words in the English language. Get-Answers automatically removes these stop words from the indexes and all user queries because they don't add much value in locating relevant documents.

Stop words are removed from indexes and queries with the following exceptions:

- During query processing, if the query type is Boolean, the words AND, OR, NOT, and BUT will not be removed (since they are recognized as Boolean operators).
- Stop words are skipped only during indexing if the word in the input file is exactly the same word in the stop words file (if "in" is in the stop words file, "INS" will still be indexed).
- Stop words in a particular database field are not removed if the NO_STOP_WORDS flag is set in the configuration file (`rware.cfg`) for that field. For example, the NO_STOP_WORDS flag is set for the file name and the file extension fields, so that a file named 'begin.out' would not be reduced to a single period if the stop words 'begin' and 'out' were removed.
- Stop words contained within idioms (for example, "by" and "all" within "by all odds" or the idiom "to be") are not removed as long as the idiom exists in the dictionary. If an idiom with stop words is indexed, an indicator for the idiom is indexed, not the individual words contained in the idiom. This makes idioms independent of the stop words file. You can, however, set up entire idioms in the stop words file (stop idioms) for any idioms contained in an indexing dictionary. This removes the idiom from the indexing and query processes.

You can modify the standard stop words file by adding or removing stop words. Words that exist in the stop words file at the time document indexes are created will not be indexed.

If you **add** stop words to the file:

- Newly added stop words (after initial indexing) are not removed from the existing indexes. However, because these new stop words will be removed from all queries, the fact that they exist in the indexes is normally irrelevant (they just take up some space and might match wildcard or pattern searches).

- Stop words you add will be removed from all queries the next time the query program is started.
- Added stop words will not be included in any future document indexes you create.

If you **remove** stop words from the file:

- Newly removed stop words are not automatically added back into the document indexes, and queries on existing documents won't find those words. To add former stop words back into existing document indexes, you must delete the index files and then re-index the library.
- Queries on removed stop words will not retrieve any documents indexed prior to the change. (A query using the removed stop words would not retrieve anything because the indexes for existing documents are still missing those words.)
- Removed stop words will be included in any future document indexes you create.

Stop words file format

If you want to make changes to the stop words file, use the following syntax rules. You may want to make a copy of the standard file and save it elsewhere first.

The rw_english.slx file is a standard ASCII text file. You can edit it using any word processing program or ASCII text editor. The syntax of the file is as follows:

- One stop word is allowed per line.
- The words are not case-sensitive (everything is converted to uppercase).
- The stop words do not need to appear in any particular order (no sorting is required).
- "Stop idioms" are allowed, as long as the idiom exists in an indexing dictionary. Words in the idiom must be separated by a single space (multiple spaces or tabs are not allowed).
- Stop words must be exactly as they appear in the text because they are not removed from the processing pipeline until after morphological analysis has taken place. For example, if "begin" is a stop word, you might also want to include "beginning," "began," and "begins."

- Blank lines in the file are allowed.
- Leading spaces are not allowed. Trailing spaces are automatically removed.

To modify the stop words list:

1. Open the `rw_english.slx` file in a text editor. The file is located in the \getanswers\resource\rw_english directory.
2. Make the desired additions or deletions to the file using the guidelines mentioned above.
3. Save the file.

Custom Dictionaries

Get-Answers uses the dictionary during both indexing and queries. The dictionary enables Get-Answers to perform “concept” searching, which means that it automatically finds words related to query terms. You can help determine which related terms are found by using Expert mode to choose to expand certain word meanings, and by setting the word expansion level to tell Get-Answers what types of related terms to find.

Get-Answers provides a baseline English dictionary. This section describes how to create a custom dictionary of your own to use along with the English dictionary for indexing and/or querying.

The baseline English dictionary was compiled from the following electronic sources:

- American Heritage® Dictionary of the English language, Third Edition
  © Copyright 1992 by Houghton Mifflin Company
- Enhanced Roget’s US Electronic Thesaurus
  © Copyright 1995 by Inso Corporation
- Enhanced Roget’s UK Electronic Thesaurus
  © Copyright 1995 by Inso Corporation
- WordNet 1.5
  © Copyright 1991, 1993, 1995 by Princeton University

A Get-Answers “dictionary” consists of the six files listed below, contained in a single directory:

- semantic.db — Semantic database of meanings and the words related to each one
- irreg.db — Lists of related words with irregular morphology or alternate spellings
- semantic.xrf — Concept cross-reference network of the semantic.db file
- irreg.xrf — Concept cross-reference network of the irreg.db file
- morph.rul — Morphology rules for finding the roots of words
- expand.wgt — Weights for words expanded via the semantic database

Each dictionary’s files are totally independent from those of any other dictionary. For example, you could create a corporate dictionary with morphology rules that are different from those in the English baseline dictionary. For details on creating a set of customized files, see *Creating a custom dictionary* on page 24.

Using a query-only dictionary

The baseline English dictionary is used for both indexing and queries, but you can specify an additional dictionary you want to use for queries only. Typically, if you’re using a custom dictionary that you created to handle special words or idioms, you will set it up as a query-only dictionary.

Queries using words that exist only in query-only dictionaries are the same as for words in query/index dictionaries with the following two exceptions:

- No morphological variants of the word will be found (for example, plurals and past tense) unless you add all variations of the word to the query-only dictionary as separate words in the irreg.db file.
- Idioms (any multiple word entry in the dictionary) are treated as exact phrases because they are not looked up and indexed as true idioms during indexing.

Using the System Administration wizards, specify a query-only dictionary in the Server Setup wizard in the Search Servers section (the wizard actually adds an argument to the cqserv program in the exec.cfg configuration file). The query-only dictionary opens only when a user executes a search.

Creating a custom dictionary

We highly recommend you do not modify the Get-Answers baseline dictionaries, since you might lose your changes in a future update. Instead, you can create a custom dictionary to use along with the baseline dictionaries for indexing and/or querying.

The following sections describe the purpose and format of each dictionary file you will need to create a custom dictionary. The samples provided are from the baseline English dictionary.

The semantic.db file

This file contains the "semantic network" of words related to each other by common meanings. The syntax is as follows:

- Each group of related words is preceded by a period character (.) alone on a line.
- The line following the period offset contains a definition or word meaning that defines the word list. This definition is one line of unlimited size, headed by an exclamation character (!). You can leave the meaning for a word blank, but you must have the ! character.
- The part of speech of the meaning that defines the word list comes next, on a separate line. This identifier is an "at" sign character (@) followed by a single uppercase character denoting the part of speech.

  Part of speech identifiers:

  @A = Adverb

  @C = Conjunction

  @E = Preposition

  @J = Adjective

  @N = Noun

  @P = Pronoun

  @R = Proper Name

  @V = Verb

@U = Unknown

- The word list contains words (or idioms) in all uppercase letters, one term per line. Each terms is preceded by a word relation number that indicates how the term is related to the meaning (synonym, similar to, antonym, etc.). Be sure you only have one space between the word relation number and the term. The word relation numbers are defined in the **expand.wgt** file. See *The expand.wgt file* on page 27.
- Comment lines can appear in the file, as a line headed by a semicolon (;).
- Do not leave trailing spaces on any lines and do not use colons (:).

SAMPLE:

```
;Semantic network of meanings, each with a group of related words and a
;numeric code defining the word relationship (defined in exapnd.wgt).
.
!a member of the Religious Society of Friends
@N
10 FRIEND
10 QUAKER
12 PROTESTANT
14 SOCIETY OF FRIENDS
14 QUAKERS
14 RELIGIOUS SOCIETY OF FRIENDS
26 QUAKERESS
.
! FOE, CHALLENGER, COMPETITOR, CONTENDER
@N
```

10 FOE

12 CHALLENGER

12 COMPETITOR

12 RIVAL

11 CONFEDERATE

11 FRIEND

11 ALLY

26 OPPOSITION

26 OPPONENT

26 ENEMY

.

!use explosives on; “The enemy has been shelling us all day”

@V

10 STRAFE

10 SHELL

10 BLAST

12 BOMB

12 THROW BOMBS AT

12 BOMBARD

26 CRUMP

.

```
!dry outer covering of a fruit or seed or nut

@N

10 HULL

12 HUSK

26 SHELL
```

An individual word may appear under multiple meanings, with different word relation numbers. In the sample above, the word “friend” is linked to two different meanings: in the first case, it’s a synonym of the meaning (word relation 10), but in the second case, it’s an antonym (word relation 11).

The expand.wgt file

This file contains the weights that are applied to query expansion terms for each word relation number. In the example below, the following line

```
    3   0.70   ;ahd   synonym
```

defines word relation number “3” to be weighted at 0.70 (70% of a full match), and indicates that words with this code are synonyms from the American Heritage Dictionary.

During a query, the “expansion level” the user sets determines which of these word relation numbers will be used for the expansion. There’s a built-in weight range associated with each expansion level; for example, level 2 includes weights 1.00-0.84, and level 3 includes weights 0.83-0.74. Therefore, using the expand.wgt file shown below, only words with relation numbers 2, 5, and 8 (all weighted at 0.90) will be added to a query at expansion level 2. Words with the relation number 6 (weighted at 0.82) will be added to the query at level 3 (in addition to words with numbers 2,5, and 8).

Also, the file defines which of the relation types are “entry terms” for the semantic.db file. In the default expand.wgt file (shown below), relations 3, 4, 7, and 10 identify entry terms. Entry terms have the following significance:

- The morphology module (used during querying and indexing) only morphs words that are entry terms in the semantic.db file. For example, the index program would first look up the word “shells” in semantic.db, but wouldn’t find it. Next, during morphology processing, the program would apply the rule for removing the ending “s,” and look up the word “shell” in semantic.db. Since “shell” is listed there as an entry term, the word “shells” would be indexed as “shell.”

  For the name, “thomas,” the index program would first look up the name in semantic.db, but wouldn’t find it. Next, during morphology processing, the program would apply the rule for removing the ending “s,” and look up the word “thoma” in semantic.db. Since “thoma” is not in semantic.db either, the word would be indexed as “thomas.”
- The query program only uses entry terms as the beginning points for expansion. For example, the word “shell” is an entry term (word relation 10) for the meaning “use explosives on.” Therefore (provided you were using Concept mode and expansion level 4 or greater), a query on “shell” would use this meaning, and it would add the words “strafe” and “blast” to the query, as well. However, “shell” is not an entry term for the meaning “dry outer covering of a fruit or seed or nut” (it’s word relation 26, which does not identify an entry term). Therefore, that meaning would not be used for the expansion, and the words “hull” and “husk” would not be added to the query.

The file has the following format:

- The first non-comment line contains the word relation numbers that are cross-referenced as entry terms into the semantic.db file.
- Word relation numbers and their weights on a 0.0 to 1.0 scale are listed one per line. The word relation type numbers should start with 1, and you can use any numbers you want. Be aware, however, that memory will be allocated based on the highest number you use. Therefore, you should keep these numbers as low as possible.
- Comment lines can appear anywhere in the file, as a line headed by a semicolon (;) character.

SAMPLE:

```
;  dict60  Expansion weight table
;
;expansion_type  weight
;the next line has the types which are index entry points into semantic.db
3  4  7  10
;
;next line contains the expansion thresholds for levels 1-9
;keyword THRESHOLDS must be first, followed by a space
;then each weight is listed, separated by spaces
;
THRESHOLD  1.00  0.84  0.74  0.62  0.55  0.43  0.27  0.16  0.06
;  1  2  3  4  5  6  7  8  9
1  0.00
2  0.90  ;british spellings
3  0.70  ;ahd synonym
4  0.80  ;ahd alternate word
5  0.90  ;variant spellings
6  0.82  ;neighboring words
```

```
7  0.57  ;roget's synonym

8  0.90  ;ahd irreg variants

9  0.00

10  0.73  ;wn synset

11  0.29  ;wn antonym

12  0.18  ;wn hypernym

13  0.45  ;wn hyponym

14  0.19  ;wn member meronym

15  0.08  ;wn substance meronym

16  0.09  ;wn part meronym

17  0.47  ;wn member holonym

18  0.37  ;wn substance holonym

19  0.35  ;wn part holonym

20  0.07  ;wn part attribute

21  0.20  ;wn entailment

22  0.09  ;wn cause

23  0.60  ;wn also see

24  0.45  ;wn similar to

25  0.78  ;wn pertains to

26  0.46  ;wn hyponym

27  0.18  ;wn verb participle
```

The irreg.db file

This file contains words that have irregular morphology (catch/caught), words that have variations that cannot easily be defined with rules (theater/theatre), and words that should be considered the same as another entry term or terms in the semantic.db file (swim/swimming).

Each word in a group is linked to each other word in the group. This means that a semantic expansion of one word will expand to the other words, and to the expansions of the other words (at a slightly lower strength). Also, if you were to look up the meanings of one of the words, you would see meanings for the other words in addition to meanings for the original word you looked up.

Note that each group has a part if speech associated with it, which prevents inaccurate expansions. In the sample below, the verb “caught” would not expand to any noun meanings of “catch,” such as “a number of fish caught by a fisherman.”

The format of this file is similar to the format of the semantic.db file:

- Word lists are preceded by a period character (.) on a line by itself.
- The part of speech line consists of an “at” sign character (@) followed by a single uppercase letter denoting the part of speech.
- The list of linked irregular variants contains words in all uppercase letters, one word per line.
- The number to the left of the word corresponds to an expansion weight line in the expand.wgt file. The root word has a “1.” Other words linked to that word have another number, according to which expansion weight should be used.
- Comment lines can appear anywhere in the file, as a line headed by a semicolon character (;).

SAMPLE:

```
; Table of irregular morphological variants and spelling variations.
; The first word in each group is considered the "root word."
.
@V
1 CATCH
8 CAUGHT
.
@V
1 SWIM
8 SWAM
8 SWUM
8 SWIMMING
.
@N
1 THEATER
2 THEATRE
```

The morph.rul file

This file contains a list of the morphology rules used to find the roots of words during indexing and querying.

- Each morhpology rule appears on a separate line, with three fields separated by a colon character (:).
  - The first field is the suffix to be removed from the morpheme.

  - The second field is an ending to be added on to the base after removing the suffix from the first field.
  - The third field is the part of speech that the root word must have in order for the rule to be applied. A rule can have more than one part of speech. The parts of speech are the same letters as used in the semantic.db file described in *The semantic.db file* on page 24.

    For example, given the word “rallies” (which has both noun and verb meanings), the fourth line of the file below (IES:Y:NV) would first remove the “ies,” then add a “y,” and finally index or query on the word “rally.”

    Notice the line for a doubled consonant with an “ed” ending (ZED: :V). Given the word “batted” (which has a verb meaning), this rule would remove “ted.”

    The part of speech is important in Concept-Expert mode for determining which meanings will be used for expansion. For example, if you queried on “talked,” the morphology module would apply the second “ed” rule (ED: :V) to the word, removing the “ed.” The word “talk” would be used in the query and would be marked as being a verb. The program would only list verb meanings for selecting meanings to expand, eliminating noun meanings such as “idle gossip or rumor.”

- Comment lines can appear anywhere in the file, as a line headed by a semicolon character (;).

SAMPLE:

```
; Baseline English dictionary morphology rules

;

S: :NV

ES: :NV

‘S: :N

IES:Y:NV

ERS:E:N
```

```
ERS: :N

IZES: :N

SSES:S:N

INGS:E:V

INGS: :V

2INGS: :V

TIONS:TE:V

NESS: :J

;

ED:E:V

ED: :V

IED:Y:V

2ED: :V
```

Using the RECORD_MORPH_RULES flag

Besides changing the morph.rul file, another way to change the standard Get-Answers morphology is to use the RECORD_MORPH_RULES flag for a library. During indexing, this flag causes the system to store the following numbers for words with the corresponding endings:

1) nothing

2) ing

3) ings

4) s

5) ed

6) en

7) er

8) est

This means, for example, that the word looking is indexed as look, and a “2” is stored with it because it matches rule 2. A search for the term ‘looking’ (enclosed in single or double quotes), matches only occurrences of the exact term looking. A search for the term looking (with no quotes) ignores the stored rule numbers, and matches occurrences of look, looks, looked, and looking.

This feature is only useful if an integrator enables the EXACT_MATCH_PROPERTY using the cqhl_set_property function in your query interface.

SAMPLE:

```
INDEXES {

DIRECTORY “/getanswers/work/indexes/business_lib”;

QUERY_MEMORY  8MB;

INDEX_MEMORY  8MB;

FLAGS QUERY_WHILE_INDEXING RECORD_MORPH_RULES;

}
```

The semantic.xrf and irreg.xrf files

The semantic.db and irreg.db files need to have cross-reference (.xrf) files, which are compressed indexes. To make the semantic.xrf and irreg.xrf files, run the dict_xrf program. For details, see *Running the dict_xrf program* on page 36. Do not change either of the .xrf files manually.

Running the dict_xrf program

This program creates the cross-reference files for the semantic.db and irreg.db files in whichever directory you specify (the files are named semantic.xrf and irreg.xrf, respectively). You must run this program after you modify either .db file.

To run the program, enter dict_xrf followed by the directory of the dictionary that needs to be cross-referenced. For example, from the install directory, you might type the following:

```
$ bin/dict_xrf  dict/query/corporate
```

Updating indexes for dictionary changes

If you edit a dictionary that you used for indexing (one that you set up in the DICTIONARY block in the rware.cfg file), we recommend you re-index the affected libraries. If you don't re-index, morphological variants of any new query terms will not be found, and new idioms will be ranked lower than they should be ranked.

To re-index an entire library, from the System Utilities menu, choose Indexing and index utilities. From the next menu, choose Index RDBMS records (rdbindex_), and choose the Index all RDBMS records in your library. This will delete and recreate the indexes.

Customizing the spelling checker

You can customize the spelling checker used in the RTF editor of the Authoring activity.

To customize the spelling checker:

1 Add the spelling checker library to the HTML Applet tag:

```
<APPLET code="EditorApplet.class" archive="edit-on-pro.jar,ssce.jar"
name=MyEditor>
  <PARAM name="CODEBASE" value="http://www.yourdomain.com/eopro/">
  <PARAM name="CABBASE" value="edit-on-pro.cab,ssce.cab">
       .
       .
       .
</APPLET>
```

2 Add the parameter SPELLCHECKPROPERTIES to the HTML Applet tag to specify the URL address of the spelling checker properties file relative to the applet code base. The spelling checker properties file defines which options to set and which dictionaries/lexicons to open. Change the parameter value to change the dictionary/lexicon.

```
  <PARAM name="SPELLCHECKPROPERTIES" value="sc - americanenglish.txt">
```

3 Add the parameter SPELLCHECK in the button ordering file to make the spelling checker icon visible.

4 Copy the dictionary/lexicon files specified in the spelling checker properties file to the proper location on the web server.

Spelling checker properties

A spelling checker properties file might contain the following keys.

Lexicon:

- MainLexicon
- UserLexicon

Options:

- CASE_SENSITIVE_OPT

- IGNORE_ALL_CAPS_WORD_OPT
- IGNORE_CAPPED_WORD_OPT
- IGNORE_MIXED_CASE_OPT
- IGNORE_MIXED_DIGITS_OPT
- IGNORE_NON_ALPHA_WORD_OPT
- REPORT_DOUBLED_WORD_OPT
- REPORT_MIXED_CASE_OPT
- REPORT_MIXED_DIGITS_OPT
- REPORT_UNCAPPED_OPT
- SPLIT_CONTRACTED_WORDS_OPT
- SPLIT_HYPHENATED_WORDS_OPT
- SPLIT_WORDS_OPT
- STRIP_POSSESSIVES_OPT
- SUGGEST_SPLIT_WORDS_OPT
- IGNORE_DOMAIN_NAMES_OPT
- ALLOW_ACCENTED_CAPS_OPT

MainLexicon and UserLexicon

The spelling checker opens main lexicons using keys named MainLexicon, where *n* is a number ranging from 1 to 99. The first main lexicon must be specified by MainLexicon1, and any other main lexicons must be numbered sequentially starting with MainLexicon2. The lexicon will be stopped if the MainLexicon sequence numbers break.

UserLexicon and MainLexicon property settings have the following form:

{Main|User}Lexicon*n*=*name* [, *access method]* [, *format*] n: Sequence number of the main or user lexicon, ranging from 1 to 99. Lexicons open in the specified sequence.

name: Name of the lexicon file, resource, or URL (depending on the access method).

access method: Method used to access the named lexicon:

- file: The lexicon is opened as a local disk file (default).
- resource: The lexicon is opened as a resource using java.lang.Class' getResourceAsStream method.
- url: The lexicon is opened as a URL using java.net.URL's openStream method.

format: The format of the lexicon. The format defaults to "t" unless the extension part of the name parameter is "clx" or "uclx," in which case the default format is "c." "t" represents a text lexicon. "c" represents a compressed lexicon.

The spelling checker accesses the lexicon files relative to the code base. The lexicon files can reside in the same directory as the applet or in a sub-directory.

Currently available MainLexicons include:

- American English MainLexicon
- British English MainLexicon

Spelling checker options

You can make the option value "true" or "false." The option is set if the value is "true." The option's default value is used if the corresponding key is not included in the Properties.

The options affect the way the spelling checker operates. In most cases, options are enabled by setting their values to true and disabled by setting their values to false.

- ALLOW_ACCENTED_CAPS_OPT: If set to true, capital letters containing accents are considered acceptable. If set to false, words containing accented capitals are considered misspelled. Setting this option to false will degrade the performance. Default: true.
- CASE_SENSITIVE_OPT: Set to true if words with different letter-case patterns should be treated as different words. Set to false if words containing different case patterns should be treated as identical. Setting this option to false will degrade performance. Example: If set to true, treat *Canada* and *canada* as two different words; if set to false, treat *Canada* and *canada* as the same word. Default: true.
- IGNORE_CAPPED_WORD_OPT: Set to true if words should be ignored (skipped) if they begin with an uppercase letter. Set to false if the words should be checked for spelling errors. Example: If set to true, ignore *Clarkson*; if set to false, check *Clarkson*. Default: false.
- IGNORE_ALL_CAPS_WORD_OPT: Set to true if words consisting entirely of uppercase letters should be ignored (skipped). Set to false if the words should be checked for spelling errors. Example: If set to true, ignore *ASAP*; if set to false, check *ASAP*. Default: false.
- IGNORE_DOMAIN_NAMES_OPT: Set to true if words that appear to be Internet domain names should be ignored (skipped). Set to false if the words should be checked for spelling errors. Words are considered to be Internet domain names if they contain at least one “dot” (.) and at least two alpha-numeric characters in a row. Example: If set to true, ignore *wintertree-software.com*; if set to false, check *wintertree-software.com*. Default: false.
- IGNORE_MIXED_CASE_OPT: Set to true if words containing an unusual mixture of uppercase and lowercase letters should be ignored (skipped). Set to false if such words should be checked for spelling errors. Example: If set to true, ignore *PrintScreen*; if set to false, check *PrintScreen*. Default: false.
- IGNORE_MIXED_DIGITS_OPT: Set to true if words containing a mixture of letters and digits should be ignored (skipped). Set to false if such words should be checked for spelling errors. Example: If set to true, ignore *Win95*; if set to false, check *Win95*. Default: false.
- IGNORE_NON_ALPHA_WORD_OPT: Set to true if words that contain no alphabetic characters should be ignored (skipped). Set to false if the words should be checked for spelling errors. Example: If set to true, ignore *12345*; if set to false, check *12345*. Default: true.

- REPORT_UNCAPPED_OPT: Set to true if uncapitalized words which exist in the lexicons in capitalized form only should be reported (via UNCAPPED_WORD_RSLT). Set to false if uncapitalized words should not be reported. Example: If set to true, report *canada*; if set to false, do not report *canada*. Default: true.
- REPORT_MIXED_CASE_OPT: Set to true if words containing an unusual combination of upper and lowercase letters should be reported (via MIXED_CASE_WORD_RSLT). Set to false if such words should not be reported. Example: If set to true, report *TUesday*; if set to false, do not report *TUesday*. Default: false.
- REPORT_MIXED_DIGITS_OPT: Set to true if words containing a combination of letters and digits should be reported (via MIXED_DIGITS_WORD_RSLT). Set to false if such words should not be reported. Example: If set to true, report *June5*; if set to false, do not report *June5*. Default: false.
- REPORT_DOUBLED_WORD_OPT: Set to true if two occurrences of the same word in a row should be reported (via DOUBLED_WORD_RSLT). Set to false if doubled words should not be reported. Example: If set to true, report *the the*; if set to false; do not report *the the*. Default: false.
- SPLIT_HYPENATED_WORDS_OPT: Set to true if hyphens (-) should be treated as word separators, and each sub-word checked individually. This word splitting is done only if the hyphenated form of the word does not exist in any open lexicon. The word is considered correctly spelled if all sub-words are correctly spelled. Set to false if hyphenated words should be checked in their entirety. Example: If set to true, and *bright-blue* was not found in the open lexicons, check both *bright* and *blue* and treat *bright-blue* as correctly spelled if both words are found; if set to false, report *bright-blue* as misspelled if not found in the open lexicons. Default: true.
- SPLIT_CONTRACTED_WORDS_OPT: Set to true if apostrophes should be treated as word separators, and each sub-word checked individually. This word splitting is done only if the contracted form of the word does not exist in any open lexicon. The word is correctly spelled if all sub-words are correctly spelled. Default: false.
- SPLIT_WORDS_OPT: Set to true if words should be treated as a series of concatenated sub-words, and each sub-word checked individually. This word splitting is done only if the original word is not found in any open lexicon. The word is correctly spelled if all sub-words containing two or more characters are correctly spelled. Default: false.

- STRIP_POSSESSIVES_OPT: Set to true if possessives of the form ‘s and s’ should be removed from words before checking their spelling. The main lexicons included with the Sentry SDKs contain no possessive word forms, so this option should be enabled when using these lexicons. Set to false if words should be checked with their possessives intact. Default: true.
- SUGGEST_SPLIT_WORDS_OPT: Set to true if the suggest method should attempt to split words into two valid sub-words. Set to false if split words should not be suggested. Example: If set to true, suggest *the boy* as a replacement for *theboy*; if set to false, do not suggest *the boy*. Default: false.

Example properties file

```
#Sentry Spelling Checker Engine edit-on Pro
UserLexicon1=userdic.tlx, url, t
UserLexicon2=correct.tlx, url, t
MainLexicon1=ssceam.tlx, url, t
MainLexicon2=ssceam2.clx, url, c
REPORT_UNCAPPED_OPT=false
IGNORE_CAPPED_WORD_OPT=false
SPLIT_WORDS_OPT=false
IGNORE_NON_ALPHA_WORD_OPT=true
REPORT_MIXED_CASE_OPT=true
REPORT_DOUBLED_WORD_OPT=true
IGNORE_ALL_CAPS_WORD_OPT=false
ALLOW_ACCENTED_CAPS_OPT=true
MinSuggestDepth=30
SPLIT_HYPHENATED_WORDS_OPT=true
Suggestions=typographical
IGNORE_MIXED_CASE_OPT=false
STRIP_POSSESSIVES_OPT=true
REPORT_MIXED_DIGITS_OPT=true
SuGGEST_SPLIT_WORDS_OPT=true
IGNORE_DOMAIN_NAMES_OPT=false
IGNORE_MIXED_DIGITS_OPT=false
SPLIT_CONTRACTED_WORDS_OPT=false
Comparator=Typographical
CASE_SENSITIVE_OPT=true
```

A Architecture Overview

APPENDIX

Important: Get-Answers is built on top of the Peregrine OAA Platform. The information in the following appendices is generci information for all web applications built on the platform.

This chapter introduces the architecture behind Peregrine OAA Platform. Peregrine OAA Platform offers a simple and extensible way of using Web applications to interface with Peregrine's existing systems or with other databases.

Peregrine OAA Platform Architecture

Peregrine OAA Platform applications and interfaces are implemented using basic building blocks that include:

HTTP	A simple and widely supported protocol for sending client requests to a server. Variations such as HTTPS provide security as well.
XML	Extensible Markup Language. A documentation meta-language that allows you to format data, which can then be displayed through a Web browser. Unlike HTML, you create your own XML tags and define them any way you want.

Commercial web servers	The services provided by the Archway architecture can be served from any commercial Web server, including IIS, Apache, Netscape Enterprise Server, or the Java Web Server.
Application servers	Peregrine OAA Platform supplies Apache Tomcat for an application server with the installation. JRun, WebSphere, and WebLogic are also supported.
Common clients	Applications can be deployed via Web browsers (IE, Netscape), handheld devices (Palm Pilot), or mobile phones (through HDML).

The application server processes data (JSP pages, XML, and so forth) that it receives from the database or client that is specifically related to the Peregrine Systems Web applications. The Web server converts the data into a form (HTML) that can be displayed in a Web browser.

The Archway component listens to HTTP requests from clients, routes the requests to an appropriate server, and returns data or documents. The requests supported by Archway can vary, but they fundamentally consist of queries, data updates, or system events.

For example, a client can contact Archway and ask to query a database for a list of problem tickets. Another client could contact Archway and supply it with a new purchase request to be entered into the database.

All requests and responses are formatted using XML. For example, a problem ticket expressed in XML could appear as follows:

```
<problem>
 <number> PM5670 </number>
 <contact> Joe Smith </contact>
 <description> My printer is out of paper </description>
</problem>
```

Clients that interact with Archway can do anything they need with the XML that is returned as a response. Very frequently, the client initiating the request is a user interface such as a Web browser. Such a client could easily display the XML documents returned by Archway. However, to be of better use, the XML documents are often displayed within a formatted HTML page. This is accomplished by using Java Server Pages (JSP).

JSP provides a syntax for creating HTML pages that is pre-processed by the Web server before being sent to the browser. During this processing, XML data obtained from Archway is merged into the HTML page.

Archway's architecture includes special support for automatically generating the HTML and JSP pages that make up a Web application.

Archway Internal Architecture

Archway is implemented as a Java servlet. The Java servlet is an application executed by a Web server that processes HTTP requests from a client through a Web browser and sends the request, by way of an adapter, to a database. It then retrieves the requested information from the database and returns it to the client. Archway requires both a Java environment and a Web server.

Each request is interpreted to determine its destination. Archway is able to communicate with a variety of back-end systems, including the AssetCenter or ServiceCenter products from Peregrine.

Requests can be handled in one of three ways:

- A request can be sent directly to an adapter that talks to a back-end server. For instance, a query request for opened tickets could be forwarded to an adapter capable of communicating with ServiceCenter.
- A request can be sent to a script interpreter hosted by Archway. This enables you to define your own application-specific services. Within a script, calls can be made back to Archway to access the back-end system with database operations and events.
- Finally, a request can be sent to a component known as a Document Manager. This component provides automated services for combining logical documents.

Archway communicates with back-end systems with the help of specialized adapters that support a predefined set of interfaces for performing connections, database operations, events, and authentication. All adapters use DLLs to communicate with each application.

Messages can be routed to a script interpreter hosted by Archway. The interpreter supports ECMAScript, a European standard based on the Core JavaScript language used by Netscape (JavaScript) and Microsoft Internet Explorer (JScript).

Messages can be routed to the Document Manager component. This component reads special schema definitions that describe application documents for logical entities such as a purchase request, problem ticket, or product catalog. The script interpreter uses these schemas to automatically generate database operations that query, insert, or update such documents.

Each form displayed by a Web application using Peregrine OAA Platform has a related JSP. A virtual directory tells the URL the location of the JSP pages the Web browser will use to display the Web application forms.

Archway Requests

Archway supports a variety of requests, all of which are based on two basic technologies: HTTP and XML. The HTTP protocol defines a simple way for clients to request data from a server. The requests are stateless and a client/server connection is maintained only during the duration of the request. All this brings several advantages to Archway, including the ability to support a large number of requests with the help of any of today's commercial Web servers.

Another important advantage is that any system capable of making HTTP requests can contact Archway. This includes Web browsers, of course. But in addition, all modern programming environments support HTTP. This makes it very simple to write new adapters that communicate with Peregrine servers without the need of specialized APIs.

An HTTP connection consists of:

- A client request
- A server response

The messages exchanged normally have a number of header lines and some content lines. For example, consider the following two principal parts of a request:

Query String — The parameters sent with the URL for the HTTP connection.

For example:
http://prgn/servlet/archway?hello&world

This URL is made up of a server locator (http://prgn/servlet/archway) and a query string (hello&world).

Content — The data appended to the request. This data can be in any format, but for Archway, the data is always formatted as XML.

Archway uses the query string of a request to determine what it has been asked to do. The following query string syntax is expected:

```
archway?target.command&param=value&param=value&...
```

Let's consider each part of the request:

Target
The name of the target object that should handle the request. Archway forwards requests to a system and returns the response. Thus, the target could be ServiceCenter, AssetCenter, or another database. The target may also be the name of a Script Object that contains customizable logic for handling the request.

Command
The action that the target object should take. By default, five basic actions are supported: query, update, insert, delete, and event. However, when the target is a Script Object, the action can be any function defined by the script.

Param=Value
Parameter values included in the request. An arbitrary number of parameters can be passed along with the request. The encoding of these parameters is the same as that used by CGI (Common Gateway Interface). As with CGI, data sent by a browser is provided by fields embedded in an HTML form. This data is automatically formatted as a CGI request in a way that Archway understands.

The following are sample URLs that query Archway with HTTP requests. These queries return data in XML documents.

- host name/servlet/archway?sc.query&_table=probsummary&priority.code=1

  This sends a query request to ServiceCenter for all records in the probsummary table with a priority code of 1.

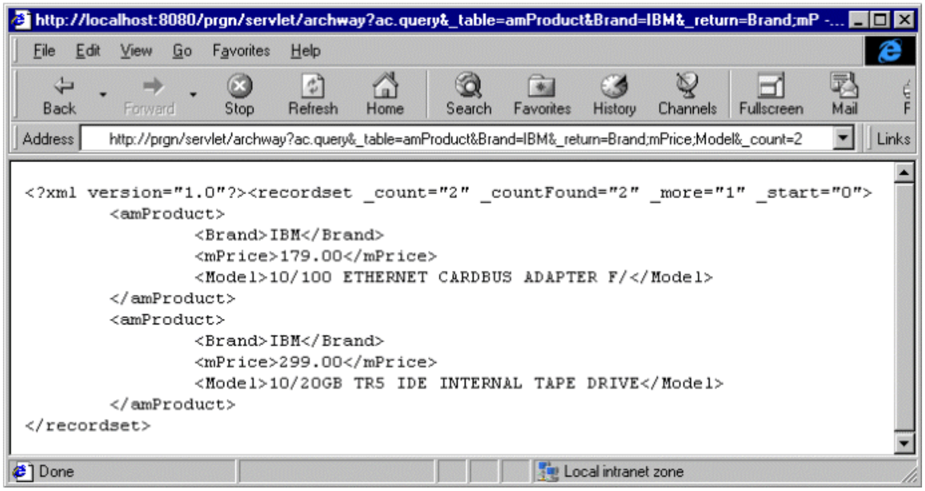
- host name/servlet/archway?ac.query&_table=amAsset&_return=Brand;mPrice;Model&_count=2

  This sends a query request to AssetCenter for the first two records in the amProduct table. Only the Brand, mPrice, and Model fields are returned for each record.

- host name/servlet/archway?test.helloWorld&greeting=Hello

  This sends a *helloWorld* request to a script object named *test*.

The screen below shows the XML results of a query for products from AssetCenter.



http://localhost:8080/prgn/servlet/archway?ac.query&_table=amProduct&Brand=IBM&_return=Brand;mP -...

Address http://prgn/servlet/archway?ac.query&_table=amProduct&Brand=IBM&_return=Brand;mPrice;Model&_count=2

```
<?xml version="1.0"?><recordset _count="2" _countFound="2" _more="1" _start="0">
        <amProduct>
                <Brand>IBM</Brand>
                <mPrice>179.00</mPrice>
                <Model>10/100 ETHERNET CARDBUS ADAPTER F/</Model>
        </amProduct>
        <amProduct>
                <Brand>IBM</Brand>
                <mPrice>299.00</mPrice>
                <Model>10/20GB TR5 IDE INTERNAL TAPE DRIVE</Model>
        </amProduct>
</recordset>
```

The Document Manager

Archway uses XML to exchange data and documents between clients and the supported back-end systems. Fundamentally, the XML data returned by Archway is obtained by executing queries against one or more systems. The queries can be executed by a direct URL request or indirectly within an ECMAScript.

Simple queries are only capable of returning record sets of data. However, clients are more often interested in exchanging documents. A Document is a logical entity built up of several pieces of data that can come from various physical database sources.

The Document Manager uses schemas to determine which XML elements to use and what data should be contained in the elements. The data used by the Document Manager depends on the back-end system being used.

B Configuring the Application Servers

APPENDIX

This chapter includes instructions for configuring the alternate application servers supported with Peregrine OAA Platform 1.0.

The following alternate application servers are supported:

- JRun 3.1
- WebSphere 4.0.1 or 4.0.2
- WebLogic 6.1 SP1 or SP2

The procedures in this chapter will only configure Peregrine OAA Platform on the application servers. Refer to the documentation for the individual servers for installation instructions.

JRun

The JRun application server requires a Java run-time environment. The Peregrine OAA Platform installer includes the Java 2 SDK Standard Edition v1.3.1_01. However, you can also use JRE 1.3.1 if you already have it installed.

Configuring JRun

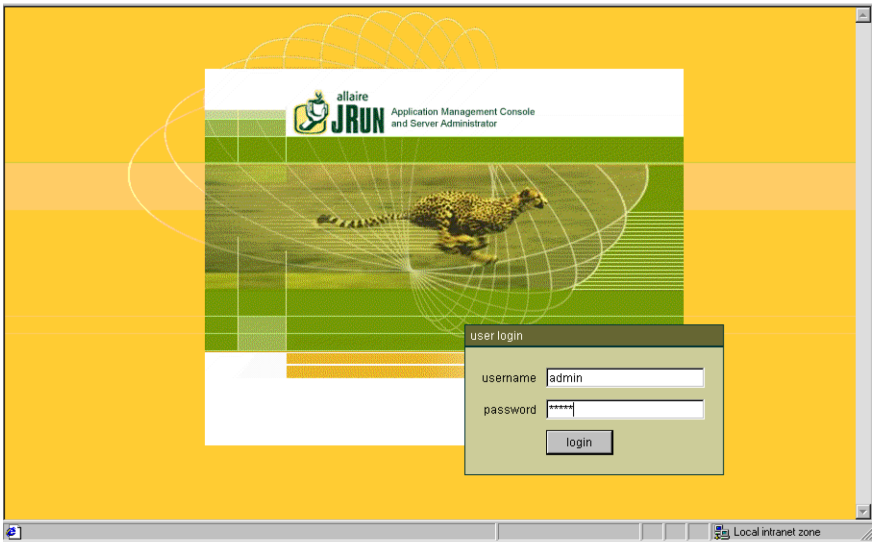
Before you configure JRun, do the following:

- If you do not have the appropriate Java run-time environment as described above, install the RTE provided with the Peregrine Web application installer. Refer to the documentation for your Peregrine Web application for instructions.
- Install JRun from the Allaire Web site.
- Return to the Peregrine installer and install Peregrine OAA Platform and your Web application. The Zip files will be copied to your system, but will not be deployed.

The following procedure will deploy the oaa package, portal.<version>.war, into the JRun Default Server.

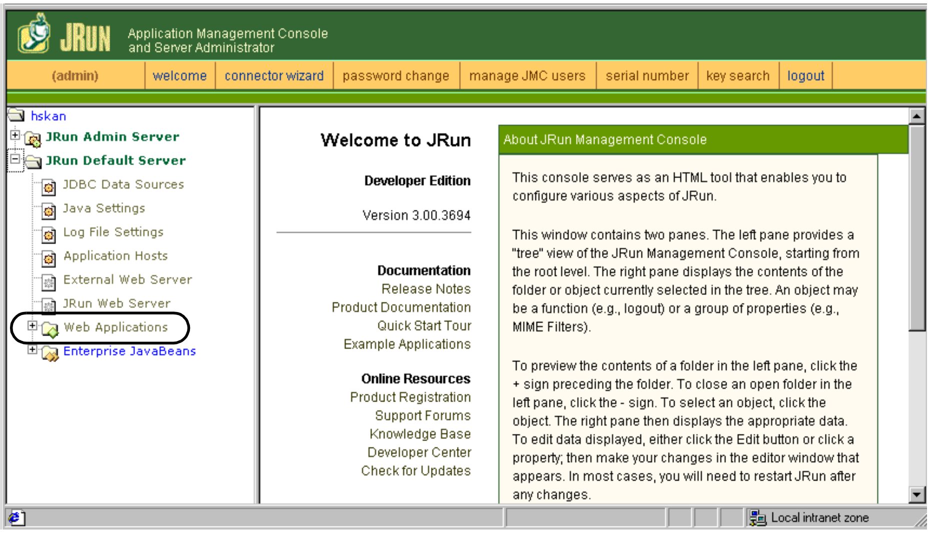
To deploy the Peregrine OAA Platform package into JRun:

1 Open the JRun Management Console and log in.



username: admin
password: *****

2 Select JRun Default Server>Web Applications.



Welcome to JRun

Developer Edition

Version 3.00.3694

Documentation
Release Notes
Product Documentation
Quick Start Tour
Example Applications

Online Resources
Product Registration
Support Forums
Knowledge Base
Developer Center
Check for Updates

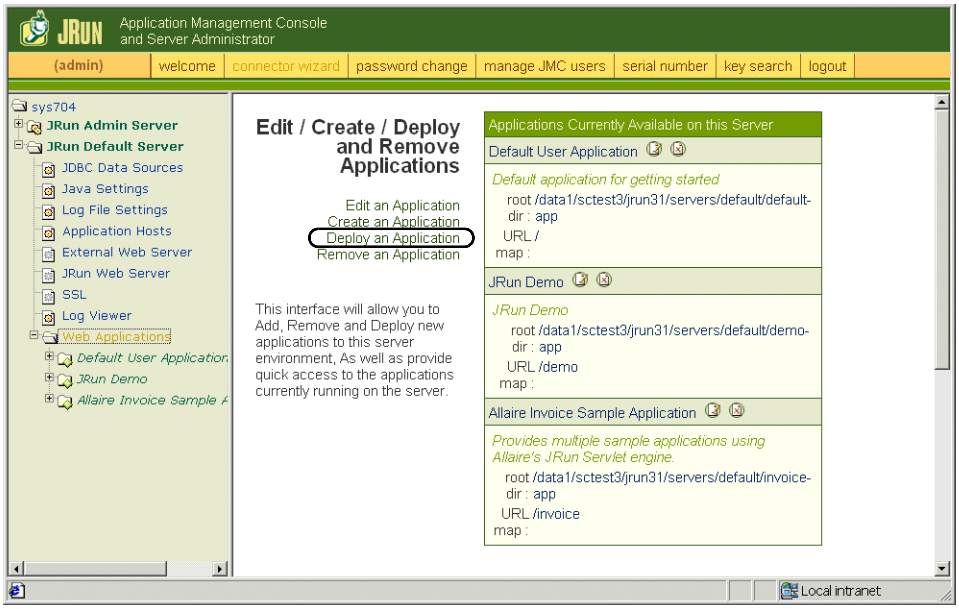
About JRun Management Console

This console serves as an HTML tool that enables you to configure various aspects of JRun.

This window contains two panes. The left pane provides a "tree" view of the JRun Management Console, starting from the root level. The right pane displays the contents of the folder or object currently selected in the tree. An object may be a function (e.g., logout) or a group of properties (e.g., MIME Filters).

To preview the contents of a folder in the left pane, click the + sign preceding the folder. To close an open folder in the left pane, click the - sign. To select an object, click the object. The right pane then displays the appropriate data. To edit data displayed, either click the Edit button or click a property; then make your changes in the editor window that appears. In most cases, you will need to restart JRun after any changes.

A screen is displayed from which you can edit, create, deploy, or remove applications.



Edit / Create / Deploy and Remove Applications

Edit an Application
Create an Application
Deploy an Application
Remove an Application

This interface will allow you to Add, Remove and Deploy new applications to this server environment, As well as provide quick access to the applications currently running on the server.

Applications Currently Available on this Server

Default User Application

Default application for getting started

root dir : /data1/sctest3/jrun31/servers/default/default-app
URL : /
map :

JRun Demo

JRun Demo

root dir : /data1/sctest3/jrun31/servers/default/demo-app
URL : /demo
map :

Allaire Invoice Sample Application

Provides multiple sample applications using Allaire's JRun Servlet engine.

root dir : /data1/sctest3/jrun31/servers/default/invoice-app
URL : /invoice
map :

3 Click the **Deploy an Application** link.

4 On the screen displayed, fill out the fields as follows:

- Servlet War File or Directory:

  Browse to c:\oaa\packages\portal.<version>.war.

  Select this file, and then click **Accept**.

- JRun Server Name:

  Select JRun Default Server.

- Application Name:

  Type oaa.

- Application URL:

  Type /oaa.

- Application Deploy Directory:

  This directory is generated internally by JRun. Make a note of this path. You will need this information later in the procedure.

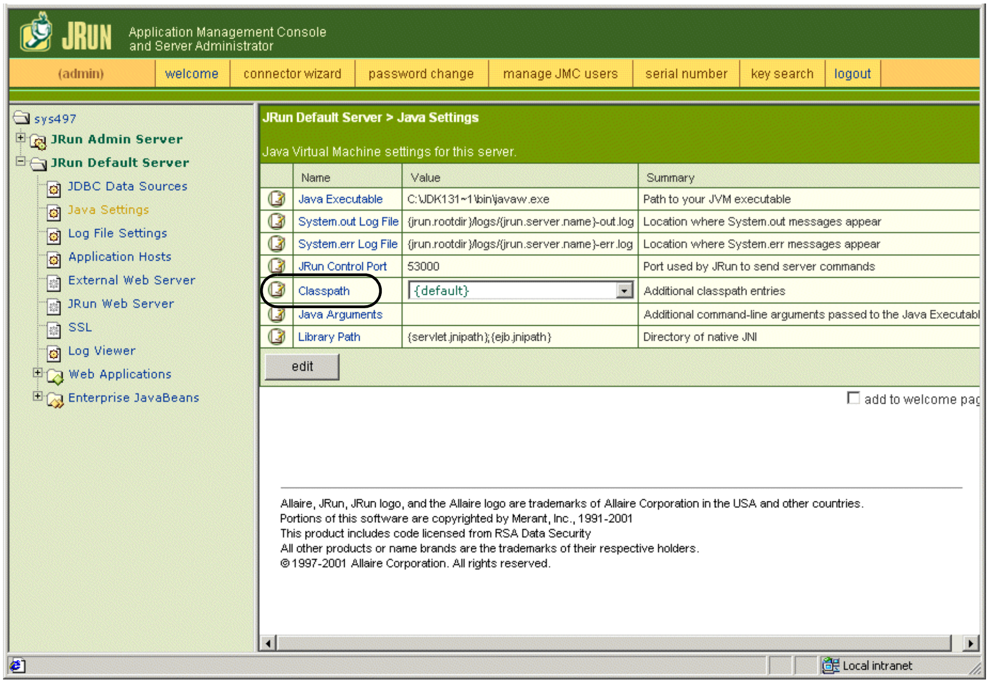
5 Click **deploy**.

A message is displayed that oaa has been successfully deployed. The next step is to edit the class path.

6 Copy all of the .jar files in the c:\oaa\external folder to c:\jdk131\jre\lib\ext.

Note: This assumes that you have installed the Java RTE supplied with the Peregrine OAA Platform installer. If you are using an existing RTE, copy the .jar files into the equivalent directory location.

7 From the Management Console activity menu, select JRun Default Server>Java Settings. Click **Classpath**.



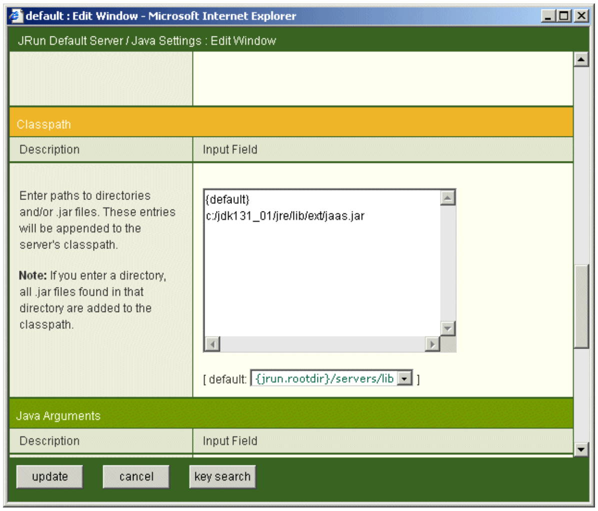
JRun Default Server > Java Settings

Java Virtual Machine settings for this server.

Name	Value	Summary
Java Executable	C:\JDK131~1\bin\javaw.exe	Path to your JVM executable
System.out Log File	{jrun.rootdir}/logs/{jrun.server.name}-out.log	Location where System.out messages appear
System.err Log File	{jrun.rootdir}/logs/{jrun.server.name}-err.log	Location where System.err messages appear
JRun Control Port	53000	Port used by JRun to send server commands
Classpath	{default}	Additional classpath entries
Java Arguments		Additional command-line arguments passed to the Java Executabl
Library Path	{servlet.jnipath};{ejb.jnipath}	Directory of native JNI

Allaire, JRun, JRun logo, and the Allaire logo are trademarks of Allaire Corporation in the USA and other countries.
Portions of this software are copyrighted by Merant, Inc., 1991-2001
This product includes code licensed from RSA Data Security
All other products or name brands are the trademarks of their respective holders.
© 1997-2001 Allaire Corporation. All rights reserved.

8 In the edit window displayed, type the path to the directory to which you copied the JAR files in step 6 (c:/jdk131/jre/lib/ext/jaas.jar).



default : Edit Window - Microsoft Internet Explorer

JRun Default Server / Java Settings : Edit Window

Classpath

Description	Input Field
Enter paths to directories and/or .jar files. These entries will be appended to the server's classpath. **Note:** If you enter a directory, all .jar files found in that directory are added to the classpath.	{default} c:/jdk131_01/jre/lib/ext/jaas.jar [default: {jrun.rootdir}/servers/lib]

Java Arguments

Description	Input Field

9 Click **update**.

10 Log out of the JRun Management Console.

11 Stop the JRun Default Server from the Windows Services dialog box.

12 In the OAAdeploy.properties file, located at c:\oaa\packages, replace the path to the deployment directory with the path you noted in step 4. Save the file.

13 Open a command prompt window and change directories to c:\oaa\packages.

This directory contains all of the Zip files that were installed onto your system.

14 To deploy all packages, type java -jar OAADeploy.jar, and then press Enter. If you want to deploy only some of the packages, add the appropriate parameters to the command.

Tip: For a list of the optional parameters available and instructions for using them, use the command java -jar oaadeploy.jar -help.

15 Start the JRun Default Server.

16 Make sure that your Web server has an oaa virtual directory pointing to the Application Deploy Directory noted in step 4.

The JRun configuration for Windows is now complete and you can log in to the Peregrine Portal.

Continue to the next section for Solaris or Linux systems.

Additional Steps for Configuring JRun on Solaris or Linux

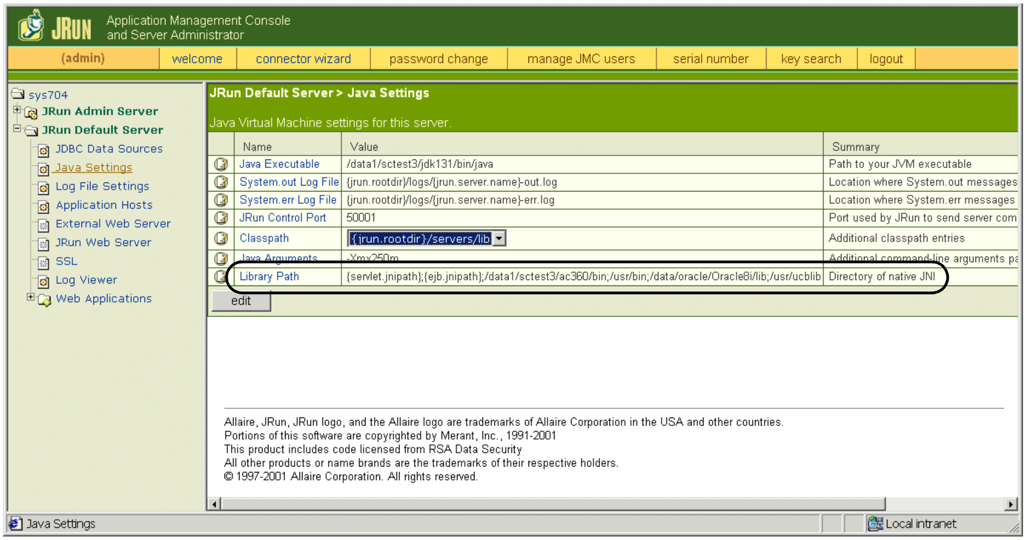
1 Open the JRun Management Console and log in.

2 On the menu at the left, select JRun Default Server>Java Settings.

3 Click **Library Path.**

4 If you are using AssetCenter, add the AssetCenter bin directory and the Oracle lib directory to the path. On Solaris, also add /usr/bin and /usr/ucblib to the path.

5 Click **update.** The updated path is shown in the following screen.



JRun Default Server > Java Settings

Java Virtual Machine settings for this server.

Name	Value	Summary
Java Executable	/data1/sctest3/jdk131/bin/java	Path to your JVM executable
System.out Log File	{jrun.rootdir}/logs/{jrun.server.name}-out.log	Location where System.out messages
System.err Log File	{jrun.rootdir}/logs/{jrun.server.name}-err.log	Location where System.err messages
JRun Control Port	50001	Port used by JRun to send server com
Classpath	{jrun.rootdir}/servers/lib	Additional classpath entries
Java Arguments	-Xmx250m	Additional command-line arguments pa
Library Path	{servlet.jnipath};{ejb.jnipath};/data1/sctest3/ac360/bin;/usr/bin;/data/oracle/Oracle8i/lib;/usr/ucblib	Directory of native JNI

Allaire, JRun, JRun logo, and the Allaire logo are trademarks of Allaire Corporation in the USA and other countries.
Portions of this software are copyrighted by Merant, Inc., 1991-2001
This product includes code licensed from RSA Data Security
All other products or name brands are the trademarks of their respective holders.
© 1997-2001 Allaire Corporation. All rights reserved.

6 Click **logout** to exit the Management Console.

7 To stop and restart the JRun Default Server, type the following commands from the jrun/bin directory:

```
./jrun -stop default &
./jrun -start default &
```

WebSphere

The following procedures will set up WebSphere on Windows, Solaris, or Linux.

To configure WebSphere:

1 Install your Peregrine Web application using the installation procedure in your Web application documentation.

Note: If you already have a Java run-time environment installed on your system, skip this option on the installer CD browser. Skip the Tomcat installation option and go directly to the option to install your Web application.

2 Verify that the WebSphere Admin Server has been started.

3 Open the WebSphere Advanced Administrator's Console (Start>Programs>IBM WebSphere>Application Server>Administrator's Console).

4 On the menu at the left side of the console, right-click on Enterprise Applications and select Install Enterprise Application.

5 On the screen displayed, do the following:

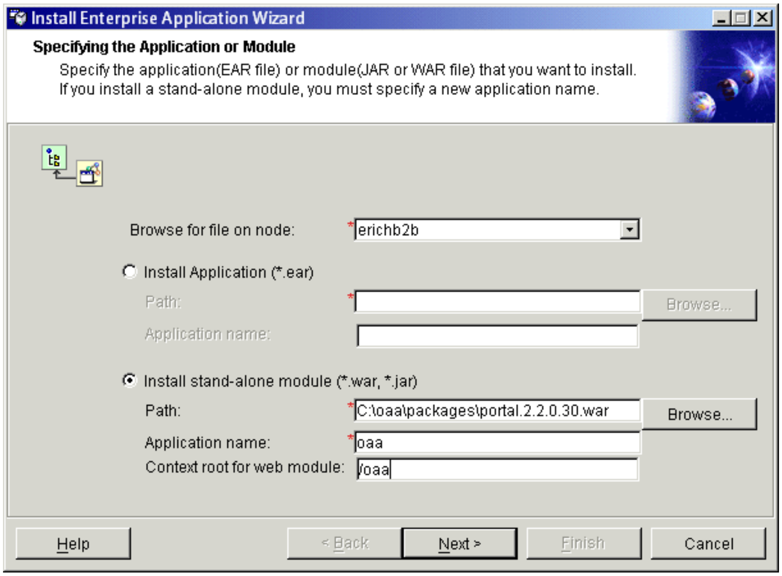
a Select **Install stand-alone module**.

b In the Path field, browse to the path to the `portal<version #>.war` file. The default is C:\oaa\packages\portal<version #>.war.

c In the Application Name field, type `oaa`.

d In the Context Root field, type `/oaa`.

The following screen shows the completed form.



Install Enterprise Application Wizard

Specifying the Application or Module

Specify the application(EAR file) or module(JAR or WAR file) that you want to install.
If you install a stand-alone module, you must specify a new application name.

Browse for file on node: *erichb2b

○ Install Application (*.ear)

Path: *

Application name:

● Install stand-alone module (*.war, *.jar)

Path: *C:\oaa\packages\portal.2.2.0.30.war

Application name: *oaa

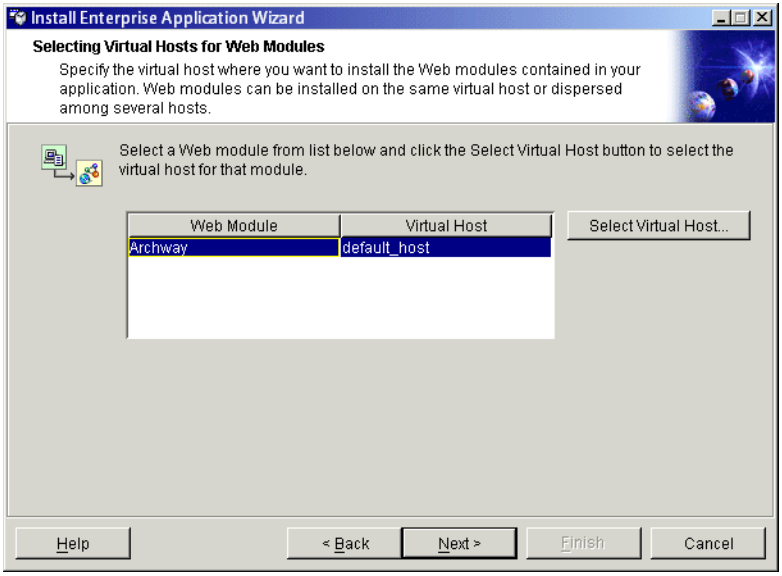
Context root for web module: /oaa

6 Click **Next**.

7 Click **Next** on the following dialog boxes. These screens will not be used.

- Mapping Users to Roles
- Mapping EJB RunAs Roles to Users
- Binding Enterprise Beans to JNDI Names
- Mapping EJB References to Enterprise Beans
- Mapping Resource References to Resources
- Specifying the Default Datasource
- Specifying Data Sources for Individual CMP Beans

8 In the Selecting Virtual Hosts for Web Modules, verify that the default is selected, and then click **Next**.



Install Enterprise Application Wizard

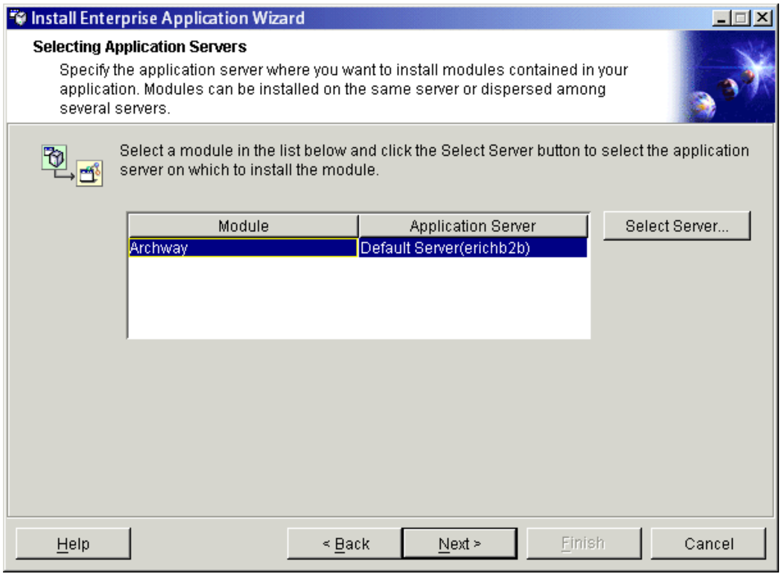
Selecting Virtual Hosts for Web Modules

Specify the virtual host where you want to install the Web modules contained in your application. Web modules can be installed on the same virtual host or dispersed among several hosts.

Select a Web module from list below and click the Select Virtual Host button to select the virtual host for that module.

Web Module	Virtual Host
Archway	default_host

9 In the Selecting Application Servers dialog box, verify that the default is selected, and then click **Next**.



Install Enterprise Application Wizard

Selecting Application Servers

Specify the application server where you want to install modules contained in your application. Modules can be installed on the same server or dispersed among several servers.

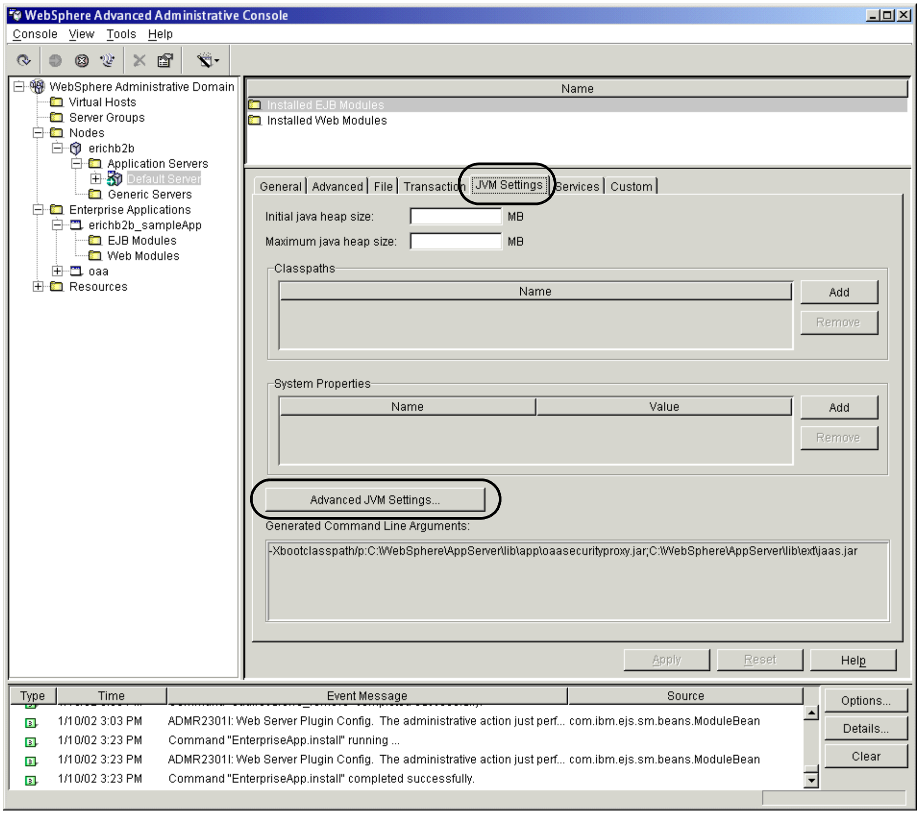
Select a module in the list below and click the Select Server button to select the application server on which to install the module.

Module	Application Server
Archway	Default Server(erichb2b)

10 On the dialog box displayed, click **Finish**.

11 On the left menu, select Nodes>system name>Application Servers>name of application server (for example, Default Server).

12 In the screen displayed, click the JVM Settings tab.



Generated Command Line Arguments:

```
-Xbootclasspath/p:C:\WebSphere\AppServer\lib\app\oaasecurityproxy.jar;C:\WebSphere\AppServer\lib\ext\jaas.jar
```

```
Type  Time            Event Message                                                              Source
      1/10/02 3:03 PM ADMR2301I: Web Server Plugin Config.  The administrative action just perf... com.ibm.ejs.sm.beans.ModuleBean
      1/10/02 3:23 PM Command "EnterpriseApp.install" running ...
      1/10/02 3:23 PM ADMR2301I: Web Server Plugin Config.  The administrative action just perf... com.ibm.ejs.sm.beans.ModuleBean
      1/10/02 3:23 PM Command "EnterpriseApp.install" completed successfully.
```

13 Set the initial and maximum Java heap size. Recommended settings are as follows:

a In the **Initial Java heap size** field, type `60`.

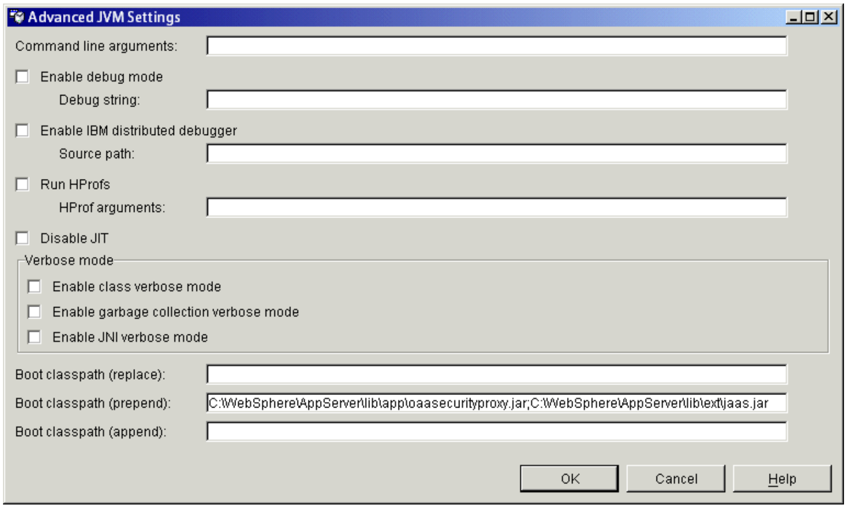
b In the **Maximum Java heap size** field, type 256. This setting should always be at least 256, but should never be so large that it exceeds physical RAM or causes the operating system to swap.

14 Click the **Advanced JVM Settings** button.

15 In the dialog box displayed, add the full paths to the **oaasecurityproxy.jar** and **jaas.jar** files, separated by a semicolon. For a default WebSphere installation, this will be (all on one line):

```
C:\WebSphere\AppServer\lib\app\oaasecurityproxy.jar;C:WebSphere\AppServer\lib\ext\jaas.jar
```

The following screen shows the completed field.



Advanced JVM Settings

Command line arguments:

□ Enable debug mode

Debug string:

□ Enable IBM distributed debugger

Source path:

□ Run HProfs

HProf arguments:

□ Disable JIT

Verbose mode

□ Enable class verbose mode

□ Enable garbage collection verbose mode

□ Enable JNI verbose mode

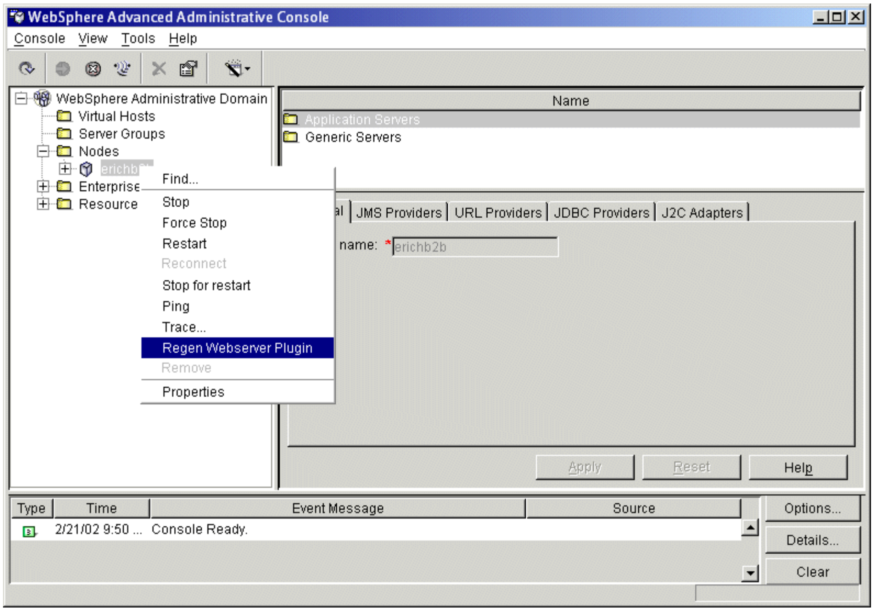
Boot classpath (replace):

Boot classpath (prepend): C:\WebSphere\AppServer\lib\app\oaasecurityproxy.jar;C:\WebSphere\AppServer\lib\ext\jaas.jar

Boot classpath (append):

16 Click **OK**, and then click **Apply**.

17 On the left menu, right-click on your Node, which will have the same name as your computer. Select **Regen Webserver Plugin**.



name: *erichb2b

Type	Time	Event Message	Source
	2/21/02 9:50 ...	Console Ready.	

This updates the configuration file for WebSphere's connector to your Web server so that the Web server knows that files under /oaa are handled by WebSphere.

18 Minimize the Console window.

19 Restart your Web server.

To complete the configuration of WebSphere:

1 Verify that the WebSphere Default Server is stopped.

2 Open the oaadeploy.properties file (located at c:\oaa\packages) in any text editor. Set the deployment directory to the path to the location of the oaa files inside your application server. The default path is:

```
C:/WebSphere/AppServer/installedApps/oaa.ear/portal.2.2.0.xx.war/
```

3 In the ...\oaa\packages directory, delete portal.2.2.0.xx.war.ear.

4 Open a command prompt window and do the following:

a Change directories to the location of the packages directory. The default location is C:\oaa\packages.

This directory contains all of the Zip files that were installed onto your system.

b To deploy all packages, type java -jar OAADeploy.jar, and then press Enter. If you want to deploy only some of the packages, add the appropriate parameters to the command.

Tip: For a list of the optional parameters available and instructions for using them, use the command java -jar oaadeploy.jar -help.

5 Go to the ...oaa\WEB-INF\lib folder and copy the following file to the C:\WebSphere\AppServer\lib\app directory:

```
oaasecurityproxy.jar
```

6 Go to c:\oaa\external and do the following:

a Copy the following files to the C:\WebSphere\AppServer\lib\app directory:

```
crimson.jar
xalan.jar
xerces.jar
jsse.jar
```

b Copy the following files to the C:\WebSphere\AppServer\lib\ext directory:

```
jaas.jar
jai_codec.jar
jai_core.jar
jaxp.jar
jcel_2_1.jar
jcert.jar
jnet.jar
mail.jar
mlibwrapper_jai.jar
sunjce_provider.jar
```

7 Open the archway.xml file (located at C:\WebSphere\AppServer\installedApps\oaa.ear\portal.2.2.0.xx.war\WEB-INF\default\) in any text editor, and do the following:

a Add the following three lines above the line that says </settings>:

```
<SSLProvider>com.ibm.jsse.JSSEProvider</SSLProvider>
<HTTPSHandlerPkg>com.ibm.net.ssl.internal.www.protocol</HTTPSHandlerPkg>
<CryptoProvider>com.ibm.crypto.provider.IBMJCE</CryptoProvider>
```

This will change the SSL implementation to use IBM instead of Sun.

b Save the file.

8 If you are using WebSphere with IIS as your Web server, create an /oaa virtual directory in IIS to point to websphere/appserver/installedapps/oaa.ear/portal.2.2.0.xx.war.

9 Open the WebSphere Advanced Administrative Console that you minimized earlier. In the menu on the left, right-click on the server name and select Start.

To test the configuration:

1 After the server is started, open a Web browser and type the following in the Address field:

```
http://<server name>:9080/oaa/admin.jsp
```

where 9080 is the port number for WebSphere 4.0.1's built-in Web server. If you are using a different version of WebSphere, enter the correct port number for that version. If you are using IIS as your Web server, omit the port number.

2 Press Enter.

If everything is configured properly, the Administrator login page will be displayed.

WebLogic

The following procedure configures WebLogic for Peregrine OAA Platform on Windows, Solaris, or Linux.

1 Install Peregrine OAA Platform and your Web application using the installation procedures in your Web application installation guide.

Note: If you have already have a Java run-time environment installed on your system, skip this option on the installer CD browser. Skip the Tomcat installation option and go directly to the option to install Peregrine OAA.

2 After the installation is complete, open a command prompt window and change directories to the location of the packages directory. The default location is C:\oaa\packages.

This directory contains all of the Zip files that were installed onto your system.

3 To deploy all packages, type java -jar OAADeploy.jar, and then press Enter. If you want to deploy only some of the packages, add the appropriate parameters to the command.

   Tip: For a list of the optional parameters available and instructions for using them, use the command java -jar oaadeploy.jar -help.

4 In the WebLogic applications directory, create a directory called oaa.

5 Copy the contents of the deploy directory to the WebLogic applications\oaa directory.

6 Verify that the following directory exists. If it does not exist, create it: C:\bea\jdk131\jre\lib\ext

7 Go to the Peregrine OAA Platform lib folder (typically bea\wlserver6.1\config\
<my domain>\applications\oaa\WEB-INF\lib), where <my domain> is the WebLogic domain of the system on which WebLogic is installed.

   Move the following files to the \bea\jdk131\jre\lib\ext folder:

```
jaas.jar
jai_codec.jar
jai_core.jar
jcel_2_1.jar
jcert.jar
jnet.jar
jsse.jar
local_policy.jar
mlibwrapper_jai.jar
oaasecurityproxy.jar
sunjce_provider.jar
US_export_policy.jar
```

8 Add the WebLogic system password to WebLogic's startup script, startWebLogic.cmd. To do this:

   a Open the startWebLogic.cmd file (located at c:\bea\wlserver6.1\config\mydomain\), in any text editor.

   b Scroll to the following section of the script:

```
echo **************************************************
echo * To start WebLogic Server, use the password *
echo * assigned to the system user. The system *
echo * username and password must also be used to *
echo * access the WebLogic Server console from a web *
echo * browser. *
echo **************************************************
@rem Set WLS_PW equal to your system password for no password prompt server
startup.
set WLS_PW=password
```

c. In the last line, change the word "password" to your WebLogic system password.

d. Save the file.

9. Create a virtual directory named oaa on your Web server and point it to the WebLogic applications oaa directory.

   This will create a connection between the application and Web servers.

10. Restart the Web server.

11. Restart WebLogic as follows:

    The first time you start WebLogic after the installation, you will need to start it in development mode for it to find the Web applications that have been deployed.

    a. Open the `startweblogic` (.cmd or.sh) file.

    b. Locate the line that sets the STARTMODE variable. Set this variable to STARTMODE=false.

    c. Start WebLogic.

    After you have done this at least once, you can change the setting to STARTMODE=true, which is the usual setting for a production system.

To test the configuration:

1. After the server is started, open a Web browser and type the following in the Address field:

   ```
   http://<server name>:7001/oaa/admin.jsp
   ```

   If you are using IIS as your Web server, omit the port number.

2. Press Enter.

If everything is configured properly, the Administrator login page will be displayed. In the Admin module Control Panel, verify that the adapters are connected. For instructions for using the Admin module, refer to your Web application guide.

If you need to reinstall Peregrine OAA Platform on WebLogic, first uninstall your current installation by deleting the oaa directory from WebLogic. Recopy the Peregrine OAA Platform files, and then restart WebLogic and your Web server.

Updating the scriptpoller.ini Files

If you will be using script pollers, you will need to update the script poller INI files with the name of the Java virtual machine (JVM) for the application server you are using.

The scriptpoller.ini files are configured to use the default Java virtual machine (JVM). If you are using an alternate application server, you will need to tell Archway where to run the script pollers. Script poller files are located at <application server>\webapps\oaa\WEB-INF\apps in the common, notifications, and oaaworkflow folders.

In the scriptpoller.ini file, add the ArchwayJVMName parameter, as shown in bold type in the example below, substituting jvm_name with the appropriate name for the application server you are using:

- WebSphere: oaa_bin
- WebLogic: oaa_wlserver60 (This will be different if WebLogic is not installed in the default installation directory.)
- Tomcat: oaa_bin

```
<poller>
    <name>KeepAliveSC</name>
    <interval>600</interval>
    <parms>
        <ArchwayJVMName>jvm_name</ArchwayJVMName>
    </parms>
</poller>
```

C

APPENDIX

Load Balancing Your System

After completing your installation of Peregrine OAA Platform and your Web application, decide whether or not you will need multiple application server instances.

Server Processing Considerations—Multiple Application Server Instances

A Peregrine OAA Platform server running a Web application such as Peregrine's Get-Services or Get-Resources consumes approximately 60 to 100 MB of memory per application server instance. Care should be taken not to set the maximum heap size of the JVM in excess of the free RAM available to the application server(s). Exceeding the amount of available RAM causes the JVM processes to swap to disk, reducing overall performance.

Unlike other Adapters, the AssetCenter and ServiceCenter Adapters each create a single connection to the respective back end. Therefore, the memory consumed on the AssetCenter database server is the same as that consumed by a single client connection. The memory consumed on the ServiceCenter server is also the same as that of a single ServiceCenter client process.

Note that memory usage does not increase significantly per session, because the architecture is based on the sharing of a set of resources and database connections among all sessions handled by the same application server instance. The small amount of memory consumed for session-specific information is released as the users log off or as their sessions expire. Note that server sessions do not expire unless the browser is closed or the user navigates to a different domain.

Because ServiceCenter and AssetCenter adapters maintain a single connection to the back end, adding extra application server instances brings the added benefit of concurrent access to the back-end data store.

The need for extra application server instances and therefore JVMs is directly related to three variables:

- The number of concurrent users.
- The processing power of the machine hosting the Peregrine OAA Platform Web server.
- The number of processors on the machine.

Each deployment may make different demands of the software and hardware, but, in any case, optimal back-end throughput for ServiceCenter and AssetCenter is achieved with the maximum number of application server instances that the server can handle without degraded performance due to lack of CPU headroom, file system swapping and context switching.

Cache synchronization with Symmetric MultiProcessing (SMP) servers can, in most cases, be ignored as a performance tuning factor except in the case of the extremely large-scale systems.

To serve as a control guideline, low-end processors, such as a Pentium 450, should be capable of producing acceptable load handling for around 100 concurrent sessions on a single application server process. A dual Pentium 1000 with 2 gigabytes of RAM (a common data center configuration) should be capable of handling 400+ concurrent sessions using multiple application server instances. When using adapters capable of pooling, the JDBCAdapter, BizDocAdapter, and so forth, performance beyond the 400-concurrent-user benchmark can be achieved.

Creating Multiple Instances of Tomcat

Multiple instances of Tomcat are installed as services. Although this is not required, it makes the instances easier to manage and provides extra functionality, including restarting the service if it fails or if the machine on which the instances are installed needs to be restarted.

Important: The following procedures use Tomcat with IIS as the Web server. This configuration is required if you are using Windows NT Challenge/Response. However, if you are not using Windows NT Challenge/Response, it is recommended that you use Tomcat with Apache as your Web server. The process for both configurations is similar and differences in configuring multiple instances of Tomcat with Apache are noted in the instructions. Any procedures that are not designated specifically for IIS or Apache should be completed for both.

Creating multiple JVMs using Tomcat requires the following:

- Editing the workers.properties file to set the values (port number, host, and so on) for each Tomcat instance.
- Editing the uriworkermap.properties file to define a default worker name. (For Tomcat with Apache, you will be editing a copy of the mod_jk.conf-auto file and the Apache httpd.conf file.)
- Copying and modifying the wrapper.properties file to specify the parameters used by Windows to start the Tomcat JVM as a Windows service. There is a separate file for each Tomcat instance.
- Copying and modifying a server.xml file for each Tomcat instance.
- Installing multiple instances of Tomcat as a service using jk_nt_service.exe. This file can be found on the installation CD.
- Testing the configuration.

The following procedures describe an example of an installation of four Tomcat instances on a system using IIS.

Editing the workers.properties File

For each server on which Tomcat instances are installed, there is only one workers.properties file, located in the config directory of your Tomcat installation. This file is shared by all Tomcat instances on the server.

The workers.properties file specifies the worker threads that the Web server connector will create to communicate with the Tomcat instances. Each Tomcat instance must communicate on a different port. The host should be set to the name of the server running the instances or localhost if they are running on the same server as IIS.

Cache size is the maximum number of user sessions that IIS should direct to the Tomcat instance at one time.

Lbfactor is a number greater than or equal to 1 that IIS uses to load balance the workers. If all the workers are running on servers that have equal performance strengths, the lbfactor numbers should be equal. Workers with a lower lbfactor will be assigned fewer user sessions by the loadbalancer worker in IIS.

The following example configures four load-balanced Tomcat instances, identified by port numbers 8009, 8011, 8013, and 8015.

To edit the workers.properties file:

1. Open the workers.properties file (located in the config directory of your Tomcat installation) in any text editor.
2. Edit the following lines as shown. The paths for workers.tomcat_home and workers.java.home are the locations of your Tomcat installation and Java SDK installations.

```
workers.tomcat_home=c:\jakarta-tomcat-3.2.3
workers.java.home=c:\jdk1.3.1_01
ps=\
worker.list=loadbalancer, w8009, w8011, w8013, w8015
worker.loadbalancer.type=lb
worker.loadbalancer.balanced_workers=w8009, w8011, w8013, w8015
```

Note: You can define the worker names any way you want as long as you continue the same naming convention throughout the procedure.

3 Add the following lines for each Tomcat instance you have installed, incrementing the port number for the values shown in step 2:

```
worker.w8009.port=8009
worker.w8009.host=localhost
worker.w8009.type=ajp13
worker.w8009.cachesize=40
worker.w8009.lbfactor=10
```

4 Comment out the following lines. These default workers will not be used.

```
worker.ajp12.port=8007
worker.ajp12.host=localhost
worker.ajp12.type=ajp12
worker.ajp12.lbfactor=1

worker.ajp13.port=8009
worker.ajp13.host=localhost
worker.ajp13.type=ajp13
worker.ajp13.lbfactor=1
worker.ajp13.cachesize=10
```

5 Save the file.

Editing the uriworkermap.properties File (IIS only)

Note: For systems using the Apache Web server, go to the next section to edit the mod_jk.conf-auto file.

For each server on which Tomcat instances are installed, there is only one uriworkermap.properties file, located in the config directory of your Tomcat installation. This file is shared by all Tomcat instances on the server.

To edit the uriworkermap.properties file:

1 Open the uriworkermap.properties file in any text editor.

2 Change the default worker to loadbalancer.

Example:

```
default.worker=loadbalancer
```

3 Change all oaa contexts to use loadbalancer instead of default worker ajp12. Make sure the oaa contexts do not include patterns for files that IIS is more suited to serve such as pictures.

Usage: <file(s) or directory context pattern> = <worker name>

Examples:

```
/oaa/servlet/*=$(default.worker)
/oaa/presentation/*.jsp=$(default.worker)
```

4 Save the file.

Editing the mod_jk.conf-auto File (Apache only)

If you are using Apache as your Web server, use the following procedure. This file is shared by all Tomcat instances on the server. It is important that you do this step after you have already installed Peregrine OAA Platform, otherwise mounts, locations, and directories will not be included in the mod_jk.conf-auto file and you will have to manually add them.

To edit the mod_jk.conf-auto file:

1 Make a copy of the mod_jk.conf-auto file and rename it to mod_jk.conf-local. Both of these files are located in the Tomcat conf directory.

2 Open the mod_jk.conf-local file in any text editor.

3 Change JkWorkersFile to point to the correct worker.properties file.

Example:

```
JkWorkersFile “C:\Jakarta-tomcat-3.2.3\worker.properties”
```

4 Change all JkMounts to use *loadbalancer* instead of *default worker ajp12*.

Usage: JkMount<file(s) or directory> <worker name>

Examples:

```
JkMount/oaa/servlet/* loadbalancer
JkMount/oaa/* .jsp loadbalancer
```

5 Save the file.

Editing the httpd.conf File (Apache only)

The httpd.conf file must include mod_jk.conf-local.

To edit the httpd.conf file:

1. Open the **httpd.conf** file in any text editor.
2. Add the following line:

```
include "C:/<tomcat directory>/conf/mod_jk.conf-local"
```

3. Save the file.

Editing the wrapper.properties Files

You will need a separate **wrapper.properties** file for each instance of Tomcat that will be running concurrently. These files will specify the parameters used by Windows to start the Tomcat JVMs as a Windows service.

To edit the wrapper.properties files:

1. Make a copy of the **wrapper.properties** file for each Tomcat instance and give each file a unique name.
2. Open one of the **wrapper.properties** files in any text editor.
3. Set wrapper.tomcat_home to point to the Tomcat home directory.

   Example:

```
wrapper.tomcat_home=C:\jakarta-tomcat-3.2.3
```

4. Set wrapper.java_home to point to the JDK home directory.

   Example:

```
wrapper.java_home=C:\jdk1.3.1_01
```

5. Set wrapper.server_xml to point to the **server.xml** file for this instance of Tomcat.

   Example:

```
wrapper.server_xml=$(wrapper.tomcat_home)\conf\8009server.xml
```

6. Add the -Xrs switch (required) and the -Xms and -Xmx switches (optional) to the wrapper.cmd_line execution statement.
   - -Xrs—required when using Tomcat as a Windows NT service. This switch tells the JVM not to exit when the current user logs off the system.
   - -Xmx—a memory switch. This switch tells the JVM the maximum amount of memory it is allowed to use.

- -Xms—a memory switch. This switch gives the JVM a minimum amount of memory to allocate when starting.

The -Xmx switch can be added to optimize the usage of RAM on the server by Tomcat. A good way to determine usage is to start the server and IIS, but not Tomcat. Observe how much memory is used at idle with no user load (or the normal load level experienced by the server when under normal pre-Peregrine OAA usage).

The following example outlines how to analyze and configure the -Xms and -Xmx switches:

Example:

The server has 2 Gb RAM hardware. At idle it uses about 180 MB. That leaves 1916 MB left for Tomcat usage. Subtracting 116 MB for just-in-case extra overhead leaves 1800 MB. Divided by 4 for the four Tomcat instances gives each Tomcat instance a maximum of 450 MB of RAM to use without worrying about the OS paging the memory.

```
wrapper.cmd_line=$(wrapper.javabin) -Xrs -Xmx450m -Xms150m
-server -classpath $(wrapper.class_path) $(wrapper.startup_class) -config
$(wrapper.server_xml) -home $(wrapper.tomcat_home)
```

7. Save the file.
8. Repeat steps 2 through 7 for each copy of the wrapper.properties file you made.

Editing the server.xml Files

You will need a separate server.xml file for each Tomcat instance that will be run concurrently. The name of each file must match the name specified in the wrapper.properties file for this Tomcat instance. This file contains the information Tomcat needs to connect to the Web server as well as to find the Peregrine OAA Platform Web application files.

To edit the server.xml files:

1. Make a copy of the server.xml file for each Tomcat instance and give each file a unique name.
2. Open one of the copied files in any text editor.
3. Set the log file locations and verbosity levels. The log file names in the path should be distinct for each Tomcat instance.

   Examples:

```
<Logger name=”tc_log” path=”logs/8009TC.log”
verbosityLevel=”INFORMATION” />
<Logger name=”servlet_log” path=”logs/8009servlet.log”
verbosityLevel=”INFORMATION” />
<Logger name=”JASPER_log” path=”logs/8009Jasper.log”
verbosityLevel=”INFORMATION” />
```

4 Comment out the Tomcat HttpConnectionHandler connector.

This is the port that Tomcat uses to communicate with a browser for direct HTTP requests. Since IIS will be serving the static data, Tomcat does not need to listen on this connector. It will also prevent a user from directly accessing Tomcat instances.

Example:

```
<Connector className="org.apache.tomcat.service.PoolTcpConnector">
        <Parameter name="handler"
         value="org.apache.tomcat.service.http.HttpConnectionHandler"/>
        <Parameter name="port" value="8080"/>
 </Connector>
```

5 Change the port number for the Tomcat Ajp13 listener. This is the port that Apache and Tomcat use to communicate when processing HTTP requests. This port number needs to be unique for each Tomcat instance on a server.

Example:

```
<Connector className="org.apache.tomcat.service.PoolTcpConnector">
    <Parameter name="handler"
          value="org.apache.tomcat.service.connector.Ajp13ConnectionHandler"/>
    <Parameter name="port" value="8009"/>
    <Parameter name=”max_threads” value=”100”/>
    <Parameter name=”max_spare_threads” value=”30”/>
    <Parameter name=”min_spare_threads” value=”10”/>
 </Connector>
```

6 Change the OAA context so that it is not reloadable.

This prevents Tomcat from reloading the servlet without restarting the service. This improves performance and helps keep the JSP code that the Tomcat instances are serving in sync during an update. All other contexts should be set to reload=false.

Example:

```
<Context path=”/oaa”
    docBase=”webapps/oaa”
    crossContext=”false”
    debug=”0”
    reloadable=”false”>
</Context>
```

7 Save the file.

8 Repeat steps 2 through 7 for each copy of the **server.xml** file you made.

Installing Additional Tomcat Instances

After you have edited the Tomcat files, you are ready to install additional instances of Tomcat as Windows services using **jk_nt_service.exe**.

To install additional Tomcat instances:

1 Open a DOS command prompt and change directories to your Tomcat bin directory.

2 Use the following syntax to create each Tomcat instance:

```
jk_nt_service.exe -I <ServiceName><Path to wrapper file>
```

Example:

```
jk_nt_service.exe -I 8009Tomcat ..\conf\8009wrapper.properties
```

where **8009wrapper.properties** is the name you gave each copy you made of the **wrapper.properties** file.

3 Repeat step 3 for each **wrapper.properties** file, changing the port number values for each Tomcat instance.

Testing the Load Balancing

After you have created the additional Tomcat instances, test the system by starting the services and logging in to your Web application.

To start the JVM services:

1 Open your Windows Control Panel>Services dialog box.

2 Scroll to the Tomcat entry, and then click **Start**.

3 Log in to your Peregrine OAA Platform Web application.

D Security

APPENDIX

This chapter includes information about how security is handled by Peregrine OAA Platform, including the following:

- User registration
- User authentication
- Installing and using Windows NT Challenge/Response

Peregrine OAA Platform operates transparently over Secure Sockets Layer (SSL) when the virtual directory of the Web server is configured to use SSL. No certificate-based authentication is used. Passwords are sent over the wire as plain text unless SSL is employed. Passwords are stored as plain text in a browser cookie if the user selects **Enable automatic login**.

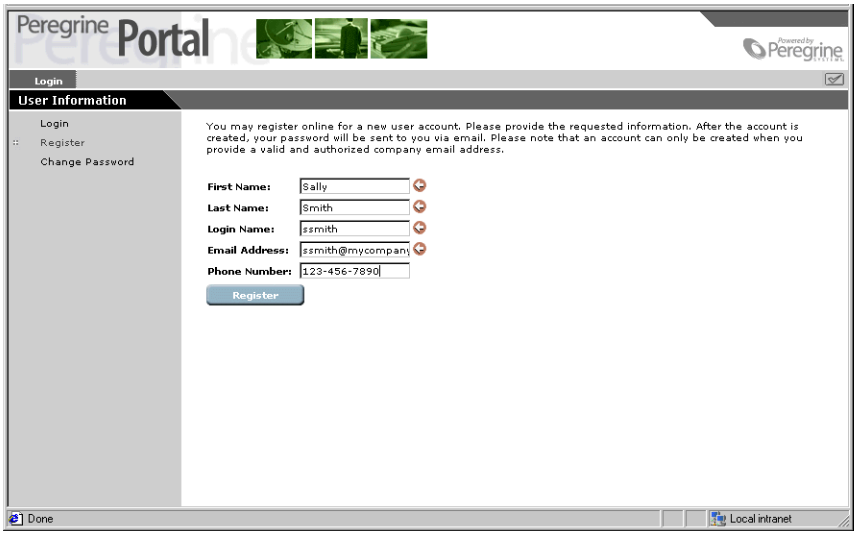
SSL uses the public-and-private key encryption system, which includes the use of a digital certificate. If your Web server has a certificate, then your URL would be something like `https://webserver/oaa`. Most browsers (Internet Explorer 4.0 or higher and Netscape 4.x) recognize this and know to send data encrypted when the site to which it is connecting is a secured site (https).

Peregrine OAA Platform also supports Windows NT Challenge/Response authentication. When this form of authentication is used, passwords are not actually exchanged between the browser and Web server, and the authentication process is kept secure. However, Windows NT Challenge/Response is only supported by Internet Explorer browsers on Windows systems. Instructions for configuring Windows NT Challenge/Response begin on page 87.

User Registration

If a user has a login name already established in the back-end system you are using, the user does not need to register in the Web application. For example, in ServiceCenter the appropriate capability words would be defined in the user's Operator record. In AssetCenter, the appropriate user rights would be defined in the user's Profile. Similar access rights can be defined in whichever back-end system you are using. The user login will automatically be authenticated in the back-end system.

However, if a user is attempting to log in for the first time without back end authentication, the user is prompted for certain default information as shown in the following screen. Note that the first four fields are required, as indicated by the arrows to the right of each field.



You may register online for a new user account. Please provide the requested information. After the account is created, your password will be sent to you via email. Please note that an account can only be created when you provide a valid and authorized company email address.

First Name: Sally
Last Name: Smith
Login Name: ssmith
Email Address: ssmith@mycompany
Phone Number: 123-456-7890

When the user clicks **Register**, the information is stored in the appropriate database. If you are using AssetCenter, Peregrine OAA Platform will transform this data into a Profile record that will then be passed to your AssetCenter system. An amEmplDept record is created with the user-supplied data and a default Profile will be assigned, getit.default.

If you are using ServiceCenter, an Operator record is created. A contact record is also created for the new user.

Note: The adapter must be defined before the capability words will be recognized. For example, if no adapter is defined for Service Center, the ServiceCenter capability words will not be used.

Your AssetCenter and ServiceCenter documentation include instructions for establishing user rights and using capability words. The access right, getit.admin, can be defined for any user who will need administrative access. Other access rights, which are specific to the web applications, are explained in the guides for the applications you have purchased.

Basic registration information and login scripts are stored in the .../oaa/apps/common/jscript/ directory. Login scripts are in the file named login.js. If you want to make changes to the registration process, such as changing the way a user's password is defined, you can change the scripts in this directory.

Authenticating Users

All user authentication in Peregrine OAA Platform is provided through the Java Authentication and Authorization Service (JAAS). When a user attempts to log in to a Web application, the user name and password entered are validated against the set of configured JAAS login modules. By default, a JAAS login module is configured for each registered adapter. For example, if you are using both AssetCenter and ServiceCenter, the user attempting to log in will be validated against *both* the ACAdapter and the SCAdapter.

The set of JAAS login modules used during authentication is defined in the local.xml file. However, by default, for each adapter deployed, a JAAS LoginModule is defined. A LoginModule can have one of four behaviors:

- REQUIRED—If the user cannot be authenticated against the adapter, the login will fail. Whether it succeeds or fails, authentication continues to the next LoginModule in the list.

- REQUISITE—If the user cannot be authenticated against the adapter, the login will fail. If it succeeds, authentication continues to the next LoginModule in the list.
- SUFFICIENT—Authentication can proceed even if this LoginModule fails. It it succeeds, authentication does not continue to the next LoginModule in the list. If it fails, authentication continues to the next LoginModule in the list.
- OPTIONAL—Authentication can proceed even if this LoginModule fails. Whether it succeeds or fails, authentication continues to the next LoginModule in the list.

The overall authentication succeeds only if all Required and Requisite LoginModules succeed. If a Sufficient LoginModule is configured and succeeds, then only the Required and Requisite LoginModules prior to that Sufficient LoginModule need to have succeeded for the overall authentication to succeed. If no Required or Requisite LoginModules are configured for an application, then at least one Sufficient or Optional LoginModule must succeed.

By default, the behavior of all LoginModules used in Peregrine OAA Platform Web applications is OPTIONAL. For most enterprises, this will be the desired configuration. However, you can change the way users are authenticated by changing the parameter settings in the local.xml file, as described in *Defining LoginModule Parameters* on page 83.

JAAS is used by Peregrine OAA Platform to authenticate users for login purposes only. Once logged in, all user rights are defined by Operator or Profile records in the back-end systems (for example, ServiceCenter or AssetCenter). These rights determine which modules the user can access and what tasks they can perform within those modules. For example, one user may be able to open tickets only, while another may have rights to approve tickets as well. For instructions for defining user rights, refer to *Defining User Rights* on page 86 and to the guides for the Web applications you are using.

Defining LoginModule Parameters

This section describes how the default JAAS configuration works and how to change the default behavior if needed.

Default configuration

A custom JAAS Configuration implementation is used by default. This Configuration subclass is only installed when the property:

```
<jaas_config>
   <useStandardJAASConfiguration> false</useStandardJAASConfiguration>
</jaas_config>
```

is unset or is set to false (the default).

When using the custom JAAS configuration, a default login module is created for each adapter deployed. The default settings are:

- loginModule=com.peregrine.OAA.security.OAALoginModule
- control flag=OPTIONAL
- options=<none>

You can modify the control flag option setting as needed.

Changing the default behavior

Using the Sun configuration file

When set to true, the *useStandardJAASConfiguration* property instructs Peregrine OAA Platform to use the standard JAAS configuration file defined by Sun Microsystems. The command line properties required for use of the standard file-based configuration are as follows:

```
java -classpath <some list of jars> \
   -Djava.security.manager \
   -Djava.security.policy==java2.policy \
   -Djava.security.auth.policy==jaas.policy \
   -Djava.security.auth.login.config==jaas.config \
      MyMainClass
```

Defining a custom JAAS configuration using local.xml

The following XML code is an example of a customized JAAS configuration defined in the local.xml file using the Sun JndiLoginModule to authenticate through an LDAP adapter.

```
<jaas_config>
   <jaasConfiguration>myconfig</jaasConfiguration>
   <myConfig>ldap;ac</myConfig>
   <ldap>
      <loginModule>com.sun.security.auth.module.JndiLoginModule</
      loginModule>
      <controlFlag>REQUIRED</controlFlag>
      <options>debug= user.provider.url="ldap://sys708/dc=gluttony,dc=com"
      group.provider.url="ldap://sys708/dc=gluttony,dc=com";</options>
   </ldap>
</jaas_config>
```

A JAAS configuration must be contained within tags as follows:

```
<jass_config>
.....
</jass _config>
```

In the example, the tag <myConfig> encloses a semicolon-delimited list of *all* login modules to be used during authentication. The default values for the Peregrine OAA adapter, ac, will be used. However, because JNDI is a third-party login module, the module class name must be specified with the <loginModule> tag. Note that the logical module names referenced by <jaasConfiguration> are used in the order supplied. If no AdapterPool is registered for the adapter name, then the name is assumed to be the logical name of a non-OAA LoginModule.

The tag <controlFlag> establishes the login authentication status, as defined by JAAS.

The following table includes some of the valid settings for the <options> tag.

Option	Type	Description
debug=true	Standard JAAS option	Instructs a LoginModule to output debugging information. The OAALoginModule logs debugging information to stdout and not to archway.log.
tryFirstPass=true	Standard JAAS option	The first LoginModule in the list saves the password entered and this password is used by subsequent LoginModules. If authentication fails, the LoginModules prompt for a new password and repeats the authentication process.
useFirstPass=true	Standard JAAS option	The first LoginModule in the list saves the password entered and this password is used by subsequent LoginModules. If authentication fails, LoginModules do not prompt for a new password.
storePass=true	Standard JAAS option	Stores the password for the user being authenticated.
clearPass=true	Standard JAAS option	Clears the password for the user being authenticated.
storeIdentity=true	Specific to OAALoginModule	Stores a reference to the User object representing the User being authenticated.
clearIdentity=true	Specific to OAALoginModule	Clears a reference to the User object representing the User being authenticated.

The following table shows some sample scenarios and how the login process works.

Module Name	Status	Scenario 1	Scenario 2	Scenario 3
LoginModule1	required	pass	pass	fail
LoginModule2	sufficient	fail	fail	fail
LoginModule3	requisite	pass	pass	pass
LoginModule4	optional	pass	fail	fail
Final Authentication		pass	pass	fail

In Scenario 1, the authentication succeeded even though LoginModule2 failed. This is because the status was defined as "sufficient" and authentication can proceed to the next loginModule in the list, even if authentication at this level fails.

In Scenario 2, the authentication succeeded because the loginModules that failed had a status of "sufficient" and "optional."

Scenario 3 authentication did not succeed because a loginModule with a status of "required" failed.

Overriding the Login Script

As an alternative to writing a custom JAAS login module, you can override the login script function, login.init(), to obtain login credentials from another source.

To accomplish this, you will need to create a function that obtains the user's login name.

For example: msg.set("loginuser", strUser);

Implementing this functionality requires that you use a different login screen for the initial Web application page. This screen does authentication, such as Windows NT Challenge/Response, and passes the authentication to the login.jsp file. The login.jsp file then sets up the initial frame set, and the server script functions, login.init() and login.login() are called. s

Defining User Rights

Once a user has been authenticated, the modules to which the user has access are defined by the back-end system. For example, if you are using AssetCenter and a user does not have access rights to a particular table in AssetCenter, the user will not be able to access the corresponding module in the Web application.

The same is true if you are using ServiceCenter for the back-end system. The user must have the appropriate capability words set in the Operator record in ServiceCenter in order to see the corresponding module in the web application.

For specific information about defining user rights, refer to the documentation for the Web application you have purchased.

HTTP Authentication

Peregrine OAA Platform uses HTTP basic authentication. When you query the Archway servlet using a URL, you will be prompted for your user name and password.

The default setting for this parameter is *true*. The setting can be changed in the archway.xml file, located at <tomcat installation>\webapps\oaa\WEB-INF\default.

Warning: If you change the HTTP authentication setting to *false*, you will open up the URL query being used to any user on the system. If this protection is turned off, you will need to protect this URL by restricting access through your Web server.

Windows NT Challenge/Response

Windows NT Challenge/Response is one of the ways Windows NT facilitates the authentication of users on a Web server. The process consists of a secure handshake between the browser (IE) and the Web server (IIS). The handshake lets the Web server know exactly who the user is, based on how they logged in to their workstation. This allows the Web server to restrict access to files or applications based on who the user is. Applications running on the Web server can use this information to identify users without requiring them to log in.

Peregrine OAA Platform uses Windows NT Challenge/Response as follows:

- The user logs in to an NT workstation.
- The user starts an IE browser and navigates to the login.asp page.
- IE automatically sends user authentication information to IIS. The user’s password is not transferred, but the Windows NT Challenge/Response handshake between IE and IIS is enough for the server to recognize the user.

- The Web application login automatically detects the user by using the Windows NT Challenge/Response/IIS server data.
- The user is logged in without requiring that a name and password be entered.

During this process, Archway authenticates and impersonates the NT user with each of its adapters.

The following circumstances must be handled during this process:

- The NT user is not yet registered with an Archway adapter. When this occurs, the web application asks the user to register and enter profile information. The application then lets the user log in and stores this information for future login attempts.
- The NT user name is already registered as an Administrator in the back-end system. When this occurs, the web application does not proceed with automatic login. The user is presented with another login screen and is asked to verify their password. This step is an added security measure to prevent a user from accidentally logging in with administrative rights.

Setting up Windows NT Challenge/Response

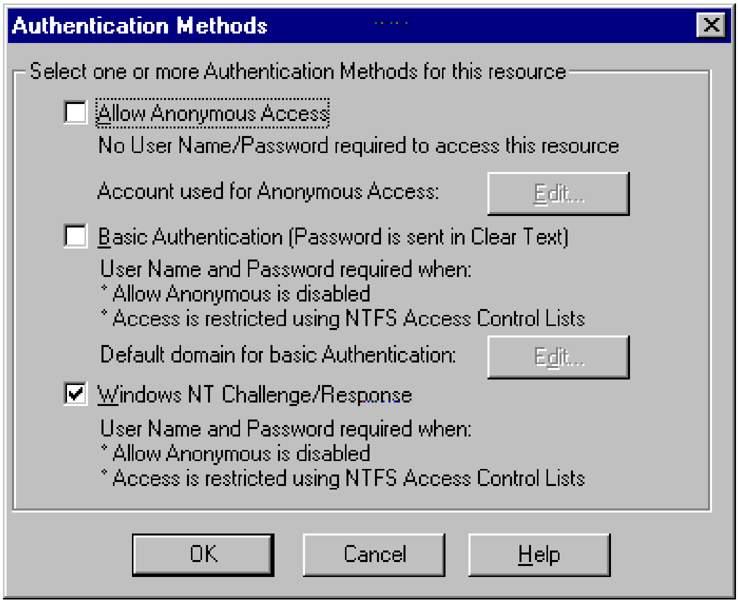
The following procedure uses Windows NT as an example.

If you are using Windows 2000, the overall procedure is the same. However, in Windows 2000, Challenge/Response is called Integrated Windows Authentication, and the IIS Management Console is called Internet Information Services.

To configure Windows NT Challenge/Response:

1. Open the IIS Management Console (Start>Programs>Administrative Tools>Internet Services Manager).
2. Click on the oaa virtual directory.
3. Right-click on `login.asp` and select **Properties**.
4. Select the File Security tab.

5 Click **Edit** in the “Anonymous Access and Authentication Control” section.



Authentication Methods

Select one or more Authentication Methods for this resource

□ Allow Anonymous Access
No User Name/Password required to access this resource

Account used for Anonymous Access: Edit...

□ Basic Authentication (Password is sent in Clear Text)
User Name and Password required when:
* Allow Anonymous is disabled
* Access is restricted using NTFS Access Control Lists

Default domain for basic Authentication: Edit...

☑ Windows NT Challenge/Response
User Name and Password required when:
* Allow Anonymous is disabled
* Access is restricted using NTFS Access Control Lists

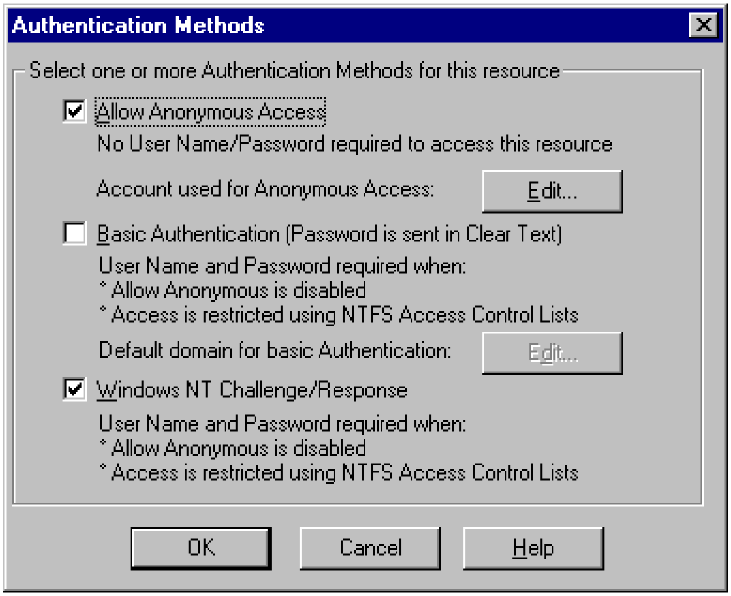
6 Check **Windows NT Challenge/Response**. Make sure this is the only option checked. Click **OK**.

7 Click **OK** on the other windows displayed until you return to the Microsoft Management Console.

Updating the loginverify.asp File

1 Repeat the steps above for loginverify.asp. Follow steps 3 through 5 as they are written above except select loginverify.asp instead of login.asp.

2 In the Authentication Methods window, check the **Allow Anonymous Access** and **Windows NT Challenge/Response** options. Click **OK**.



Authentication Methods

Select one or more Authentication Methods for this resource

☑ Allow Anonymous Access

No User Name/Password required to access this resource

Account used for Anonymous Access: Edit...

□ Basic Authentication (Password is sent in Clear Text)

User Name and Password required when:
* Allow Anonymous is disabled
* Access is restricted using NTFS Access Control Lists

Default domain for basic Authentication: Edit...

☑ Windows NT Challenge/Response

User Name and Password required when:
* Allow Anonymous is disabled
* Access is restricted using NTFS Access Control Lists

3 Click **OK** on the other windows until you return to the Microsoft Management Console.

4 Close the Management Console.

Setting Permissions for the Presentation Folder

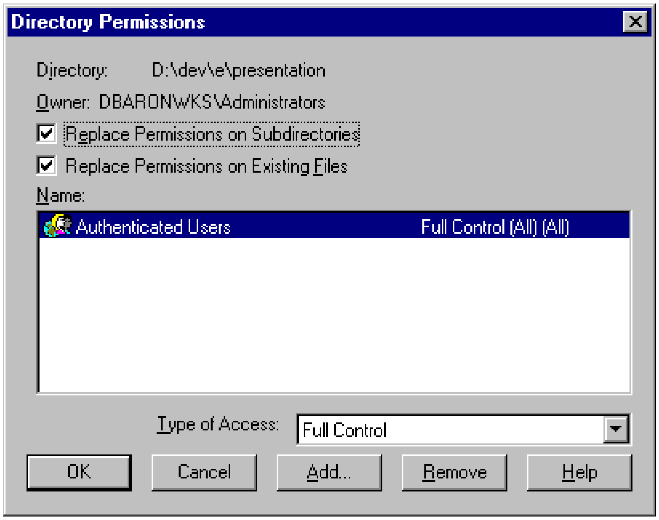
1 Use Windows NT Explorer to navigate to the ...oaa\presentation folder.

2 Right-click on **presentation** and select **Properties**.

3 On the Security tab, click **Permissions**.

Note: If you do not see a Security tab, verify that Peregrine OAA Platform is installed on an NTFS partition.

4 Click **Add** to change the user groups that have permission to access the folder. Change the permission to a named authenticated group. For example, you could change permissions to all "Authenticated Users".



Directory Permissions

Directory: D:\dev\e\presentation

Owner: DBARONWKS\Administrators

☑ Replace Permissions on Subdirectories

☑ Replace Permissions on Existing Files

Name:

Authenticated Users	Full Control (All) (All)

Type of Access: Full Control

5 If the user group called "Everyone" has permissions, highlight the entry, and then click **Remove** so that only the group you selected in the previous step can access the Peregrine OAA Platform.

6 Click **OK.** Close all remaining windows.

Testing the Settings

Log in to your Peregrine Web application to make sure the access permissions are set correctly. The Windows NT Challenge/Response settings are activated when you log in through a special login page named login.asp. Accessing your applications through the standard login.jsp page results in the users needing to log on as usual.

To test the settings:

1 Open a Web browser.

2 Enter the following URL: http://webserver/oaa/login.asp in the browser address field (where webserver is the name of your Web server and oaa is the name of the virtual directory created during installation).

3 Verify that access to Peregrine OAA Platform is what you expected based on the settings you chose for the login.asp and loginverify.asp files.

Index

Peregrine
SYSTEMS®

March 22, 2002