



Siebel Brightware™

Knowledge Engineer Handbook

Version 8.1.6

ORACLE®

Copyright © 2005, 2006, Oracle. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

PRODUCT MODULES AND OPTIONS. This guide contains descriptions of modules that are optional and for which you may not have purchased a license. Siebel's Sample Database also includes data related to these optional modules. As a result, your software implementation may differ from descriptions in this guide. To find out more about the modules your organization has purchased, see your corporate purchasing agent or your Siebel sales representative.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS. Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Table of Contents

About This Guide	1
How this Guide is Organized	1
Related Documentation	1
Conventions	2
Help	2
If You Need Help	3
Information to Provide	3
Contacting Oracle's Siebel Technical Support	4
 Knowledge Engineering Concepts	 5
About This Chapter	5
Intelligence Engine	5
Knowledge Bases	5
Knowledge Manager	6
Natural Language Processing	6
Requests	7
Request Sets	7
Fields	7
Features	8
Regular Expressions	9
Feature Hierarchy	9
Intents	11
Intent Scoring and Matching	11
Intent Overrides	12
Explicit Overrides	12
Implicit Overrides	12
Rules	14
E-mail Rules	14
E-mail Responses	14
E-mail Response Modes	15
E-mail Rule Conditions	15
E-mail Additional Actions	16
E-mail Rules Firing	16
Chat Rules	17
Request Sources	17
Web Rules	18
Simple Web Rules	18
Web Responses	19
Advanced Web Rules	21
Rule Firing for Web	22
Business Units	22
Additional NLP Processing	23
NLP Processes	23
Morphology	23
Editing NLP files	24

Global Knowledge Base Rules	25
Auto-Matched Intents	25
Global Knowledge Base Rule Example	25
Siebel Brightware and Web User Interaction	26
The Presentation Layer	27
Creating Knowledge Bases	29
About this Chapter	29
The Knowledge Manager	29
Menu Bar	30
Toolbar	31
Status Bar	32
Work Areas	32
Editors	32
Knowledge Base Management	33
The Intents Frame	36
Creating Intents	37
Automatic Object Association	38
Creating Intent Overrides	38
Intents Advanced Settings	39
The Requests Editor	40
Requests Editor Procedures	41
The Feature Editor	44
Creating Features	44
Creating Phrases and Regular Expressions	45
Associating Features with Intents	47
The Testing Editor	48
Test Results Table	49
Filtering and Unit Testing	49
Testing Match Tree	50
The Web Responses Editor	51
Creating Web Responses	51
Creating Web Response Questions	56
The Integrator Editor	58
The Web Rules Editor	60
Simple Rules	60
Advanced Web Rules	63
The E-mail Rules Editor	67
Creating E-mail Rules	67
The Chat Rules Editor	70
Creating Chat Rules	70
Printing the Knowledge Base	72

About This Guide

This manual describes how to use the Knowledge Manager. It is intended for anyone involved in creating or maintaining Knowledge Bases. This includes, but is not limited to the:

- Knowledge Engineer
- Database Administrator
- Web Administrator
- Siebel Brightware Managers

How this Guide is Organized

[Chapter 2, “Knowledge Engineering Concepts”](#) provides information about the structure and concepts used in the Knowledge Manager.

[Chapter 3, “Creating Knowledge Bases”](#) provides instructions and procedures detailing the creation of Knowledge Bases.

Related Documentation

For more information about Oracle’s Siebel Brightware please refer to these documents, included in PDF format, on the installation CD:

- *Installation Guide*
- *Agent Guide*
- *Contact Center Console Guide*
- *Implementation Guide*
- *Integration Development Kit Guide*
- *Analytics Overview Guide*
- *DB Administrator Guide*
- *Report Developer Guide*

Conventions

The following typographic conventions are used in this document:

- Items that you are instructed to click or select, such as button names and hyperlinks, are bold:
 - Select **Add Response**.
 - Click the **OK** button.
- Documents, headings, and chapter titles are italicized:
 - “Refer to the *Reference Manual* for more information.”
- Notes are flagged along the left margin:



This icon indicates noteworthy information.

- Cautions are flagged along the left margin:



This icon indicates critical information.

- Programming code and system messages appear in a fixed-width font:
`Set-request-condition (<condition>)`
- [Hyperlinks](#) and Cross References - If viewing a document online, you can navigate through it using hyperlinks, which appear in blue text, and cross references. Although not displayed in blue, the Table of Contents and Index entries are also hyperlinks. Cross references are specific page number references. Click the page number to navigate to that page:
 - Refer to “[Contacting Oracle’s Siebel Technical Support](#)”, on page 4.
- The term Type usually refers to typing information on your keyboard:
 - Type the number of decimal places you want displayed.
- The term Enter typically refers to the “Enter” key on your keyboard:
 - Type the number of decimal places you want displayed and press the **Enter** key.
- When a directory path is given, the hard drive letter is omitted since it is unknown what hard drive the system is installed on. Only the default install path is supported:
 - Documents are available under edocs\Brightware\docs\.

Help

You can access this document from the Help menu on the Knowledge Manager menu bar. The file is in Adobe Acrobat Portable Document Format (PDF). You need Acrobat Reader installed on your computer to view the file.

If You Need Help

Technical Support is available to customers who have an active maintenance and support contract with Oracle. Technical Support engineers can help you install, configure, and maintain your Oracle application.

This guide contains general troubleshooting guidelines intended to empower you to resolve problems on your own. If you are still unable to identify and correct an issue, contact Technical Support for assistance.

Information to Provide

Before contacting Oracle's Siebel Technical Support, try resolving the problem yourself using the information provided in this guide. If you cannot resolve the issue on your own, be sure to gather the following information and have it handy when you contact technical support. This enables your Oracle support engineer to more quickly assess your problem and get you back up and running more quickly.

Please be prepared to provide Technical Support the following information:

Contact information

- Your name and role in your organization.
- Your company's name
- Your phone number and best times to call you
- Your e-mail address

Product and platform

- In which Oracle product did the problem occur?
- What version of the product do you have?
- What is your operating system version? RDBMS? Other platform information?

Specific details about your problem

- Did your system crash or hang?
- What system activity was taking place when the problem occurred?
- Did the system generate a screen error message? If so, please send us that message. (Type the error text or press the Print Screen button and paste the screen into your email.)
- How did the system respond to the error?
- What steps have you taken to attempt to resolve the problem?
- What other information would we need to have (supporting data files, steps we'd need to take) to replicate the problem or error?

Problem severity

- Clearly communicate the impact of the case (Severity I, II, III, IV) as well as the Priority (Urgent, High, Medium, Low, No Rush).

- Specify whether the problem occurred in a production or test environment.

Contacting Oracle's Siebel Technical Support

You can contact Technical Support online, by email, or by telephone.

Worldwide Support Center

Telephone: 800-214-0400 or 650-341-0700

Oracle's Siebel SupportWeb

<https://ebusiness.siebel.com/supportweb/>

Email Support

mailto:siebelsupport@oracle.com

Knowledge Engineering Concepts

About This Chapter

This chapter is dedicated to the structure and concepts used in Knowledge Engineering. The following sections cover the concepts, objects, and definitions important to understanding Knowledge Engineering. No previous experience with Knowledge Engineering is required.

Intelligence Engine

The Intelligence Engine is a set of software components. It analyzes requests and determines the appropriate responses and routing for each request.

Knowledge Bases

In the beginning, the Intelligence Engine is equipped with a brain and some basic instincts, but is devoid of knowledge. As the Knowledge Engineer, it is your task to create the knowledge the Intelligence Engine will need. This is accomplished by creating a Knowledge Base.

Knowledge Bases are collections of information that tell the Intelligence Engine how to respond to user requests. The information in a Knowledge Base is divided into a set of building blocks called features, intents, responses, and rules.

- Features are built from phrases (PHZ) and regular expressions (RE) and each represents a unique concept.
- An intent is built from a collection of features, defining the relationship between those features. That relationship is used by the Intelligence Engine to understand what a user intends.
- Responses are content or actions that address a specific user need.
- Rules associate an intent with an appropriate response.

For example a simple Knowledge Base might contain:

Halibut	Keyword
Albacore Tuna	A Phrase
ab{3,5}c	A Regular Expression
Salt Water Fish	A Feature
Is It A Fish?	An Intent
IF the intent “Is It a Fish?” is found	A User Request
THEN RESPOND with	A Rule
Route to: http://www.It-Is-a-Fish.com/Index.html	A Web Response
Or	
Send the e-mail: Dear Sir, yes that is a fish.	An E-mail Response

The features, intents, responses, and rules defined in the Knowledge Base echo your business information. By limiting a Knowledge Base to specific business needs, you can create a Knowledge Base that is both easy to set up and maintain. In other words, you do not need to create a Knowledge Base that understands every possible word and idea. A Knowledge Base performs better when it has a very specific use (e.g. buying a ticket at a train station kiosk), rather than a more general use (e.g. asking your friend a question).

Knowledge Manager

The Knowledge Manger is the primary tool used for the creation and maintenance of Knowledge Bases. It contains several editors that allow you to create, edit, and manage Knowledge Bases. The bulk of this handbook is dedicated to the Knowledge Manager and the tasks performed with it. However, some knowledge engineering tasks for routing user requests are completed using other applications, such as the Contact Center Console. These tasks and their associated documentation will be indicated.

Natural Language Processing

“Natural” language is the language you use in everyday conversation. The Intelligence Engine accepts user requests that use natural language. When the Intelligence Engine receives a request, a technique called Natural Language Processing (NLP) is employed to extract the pertinent information. This allows the Intelligence Engine to “understand” the meaning of the user request. NLP compares the content of the user request with the information stored in a Knowledge Base. This allows the Intelligence Engine to determine the appropriate response to the user request.

Requests

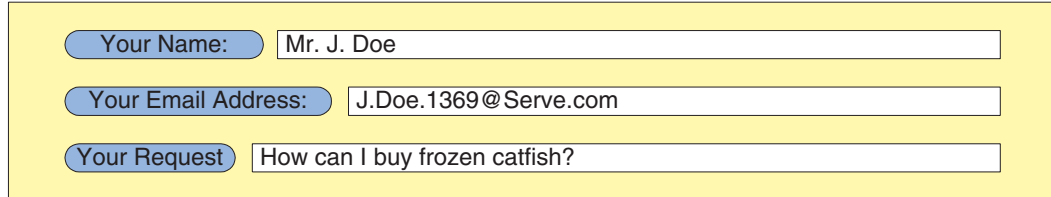
The term “request” refers to any query submitted to Siebel Brightware by an end user. E-mail requests are the contents of the e-mail. Web requests are the text contained within the fields of the web user’s submission. Chat requests contain the first entry made by the user. Requests may also originate from the Server e-mail portal. A portion of each request will undergo NLP.

Request Sets

Prior to creating a Knowledge Base, it is a good idea to gather information about previous requests customers have submitted. This information can be acquired from e-mail, web logs, web search-engine logs, customer service, technical support databases, and personnel. The Knowledge Manager allows you to import these requests into your Knowledge Base in the form of a request set. By reviewing the request set, you can determine the types of intents and features that you will need in your Knowledge Base. In addition, by linking the request set to intents, the Knowledge Base can be tested for accuracy and coverage within the Knowledge Manager.

Fields

Fields are pieces of a request that contain information. For example, if a web user submitted a query on a web form that contained a box for name, e-mail address, and an inquiry they might respond:



Your Name:	Mr. J. Doe
Your Email Address:	J.Doe.1369@Serve.com
Your Request	How can I buy frozen catfish?

Figure 2-1. An Example Request

In the example shown in [Figure 2-1](#), the contents of the *Your Request* field, may receive NLP. Any number of fields can undergo NLP, however, it is not required for all fields. The two other fields also contain information which can be used by the Intelligence Engine, but these fields will not undergo NLP. These non-NLP fields are important for a robust Knowledge Base because they supply information used to determine the most appropriate response to a request. Only the Subject and Body fields in an e-mail request undergo NLP. Whether a web request will undergo NLP or not is determined in the Integrator Editor, see [“The Integrator Editor”, on page 58](#).

Fields may be visible, such as the Subject, Body, and Address Fields in an e-mail, or hidden to the user. Hidden fields can be information that is contained in a web page, or may be appended to e-mails. Hidden fields may even contain information retrieved when the user sends their request. These fields can include a wide variety of information, such as the user’s account status, personal preferences, or information about the web page or e-mail address from which the request was issued. For example, in [Figure 2-1](#) there may be a hidden field that tells the Intelligence Engine that Mr. Doe is a Gold Star Customer. This information was retrieved when Mr. Doe logged on to the web site. There may also be a hidden field that tells the Intelligence Engine that Mr. Doe is making his request from the “How To Cook Fish” web page.

Features

Features are collections of *keywords* and *phrases* that represent common concepts. A feature is said to be “matched” when a phrase or keyword in a user’s request is found as a phrase or keyword in that feature. In the example shown in [Figure 2-1](#) the keyword “buy” appears. Mr. Doe could also have typed “How can I *purchase* frozen catfish”, since the keywords “buy” and “purchase” are largely synonymous. A feature containing keywords and phrases that represent “buying” should be created to allow the Intelligence Engine to understand the concept of “buying”. Let’s create one called “*BuyFeature*.” The list and picture in [Figure 2-2](#) show what the contents of BuyFeature would look like.

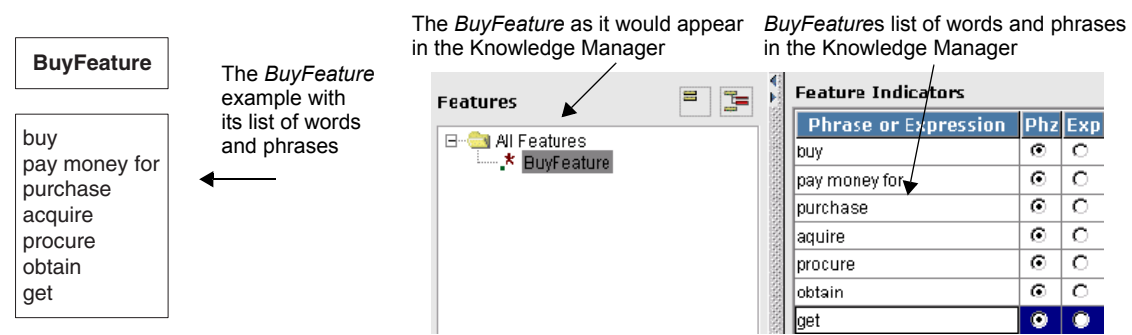


Figure 2-2. BuyFeature Contents

Now, any time a user types one of these phrases or keywords in a request, the Intelligence Engine will match the BuyFeature and will “understand” that the customer wants to buy something. Once a feature is marked as found, the Intelligence engine does not search for any other phrases/expressions in that feature. [Figure 2-3](#) shows some further feature examples to help clarify how features may be composed:

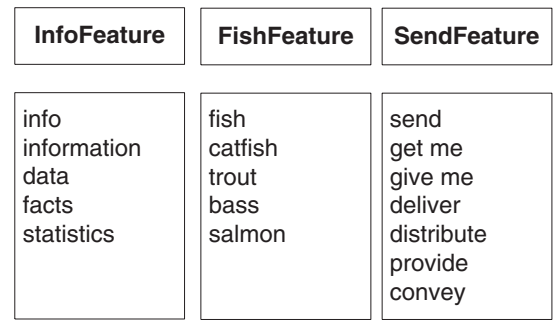


Figure 2-3. Additional Feature Examples

If you have examined the features in [Figure 2-2](#) and [Figure 2-3](#) you might have noticed that some features contain verbs and verb phrases, while others contain nouns and noun phrases. This is considered “well styled.” It provides a Knowledge Base with maximum flexibility while keeping the Knowledge Base small. We could create a feature with verb-noun phrases as follows:

buy trout, purchase bass, get salmon, etc.

However, the feature would have to be massive to contain all of the permutations. In addition, we would need to create similar features for every product line. Features function best as narrowly

defined concepts. To access the power of combining features you will use the Knowledge Manager to create objects called *intents*.



Features are not case sensitive.

Regular Expressions

In addition to phrases and keywords, features can utilize objects called *regular expressions*. Regular expressions are strings of characters that specify a pattern. [Table 2-1](#), provides some examples.

Table 2-1. Regular Expression Examples

Name	Example	Pattern
Phone number	(555) 884-4744	<code>\(\d{3}\) \d{3}-\d{4}</code>
Account number	289302-14	<code>\d{6}-\d{2}</code>
Credit card number	5268 1987 1245 3412	<code>\d{4} \d{4} \d{4} \d{4}</code>

The Regular Expression notation was designed to locate, extract, and manipulate data stored in ASCII character strings. Regular expressions allow the Intelligence Engine to recognize text when a specific pattern appears in a request. In Knowledge Bases, regular expressions are written in Perl Regular Expression syntax. Regular Expressions are described in detail by programming experts on the Internet. Their web sites provide more depth and guidance than could be given here. Offerings include language specifications, tutorials, examples, and a Java application that tests your RegExp code. Here are a few of the best:

- “Perl Regular Expression Tutorial,” by Carl Franklin and Gary Wisniewski.
Web page: <http://virtual.park.uga.edu/humcomp/perl/regex2a.html>.
- “Regular Expressions,” The Single UNIX ® Specification, Version 2, Copyright © 1997 The Open Group.
Web page: <http://www.opengroup.org/onlinepubs/7908799/xbd/re.html>.
- Regular Expression Translator at <http://lightning.prohosting.com/~tbates/regex.html>.
- “PERL5 Regular Expression Description,” by Tom Christiansen.
Web page: <http://virtual.park.uga.edu/humcomp/perl/regexps.html>.

Please use these resources when constructing Regular Expressions.

Feature Hierarchy

When you are creating features in the Knowledge Manager you can organize them in a conceptual hierarchy. This allows you to group and categorize your features by their commonality and relationships. Consider the example in [Figure 2-4](#):

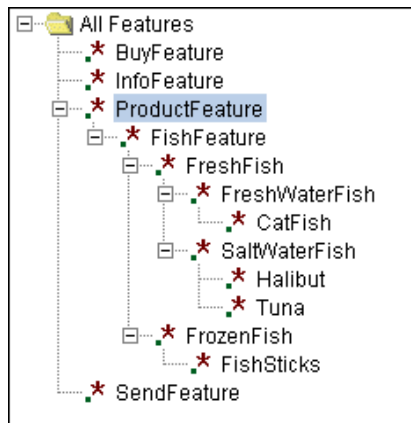


Figure 2-4. Feature Hierarchy

In the example, FishFeature is categorized as a sub-feature or child of ProductsFeature, while ProductsFeature is categorized as the super-feature, or parent, of FishFeature. It is important to note that whenever the Intelligence Engine finds a child feature in a request the parent feature is also found. Feature hierarchies allow you to conceptualize your knowledge model easily. In addition, feature hierarchies are used within Knowledge Bases to correctly determine intents.

Intents

An intent is a collection of features. Each intent represents one thing, and is structured to reflect your understanding of “what the customer wants.” When some or all of the features in an intent match a user’s request, that intent is said to be “found.”

In [Figure 2-1](#) the user presented the request “How can I buy frozen catfish?” Upon examination, we can see that the keyword “buy” matches the BuyFeature, and the keyword “catfish” matches the FishFeature. We need to create an intent that combines the BuyFeature and the FishFeature so the Intelligence Engine can understand that users want to “buy a fish.” To accomplish this we will create an intent called *BuyFishIntent* and provide it with the structure in [Figure 2-5](#).



Figure 2-5. BuyFishIntent Structure

Now, when a user’s request matches both the BuyFeature and the FishFeature the Intelligence Engine will “find” the BuyFishIntent. In other words, the Intelligence Engine will “understand” the user wants to buy a fish. By combining other features together we can create other useful intents. See [Figure 2-6](#) for example:

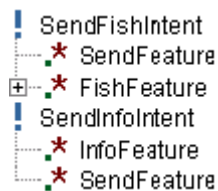


Figure 2-6. Additional Intent Examples

Once the Intelligence Engine has determined which intents have been found, it uses the information to determine the appropriate response.

Intent Scoring and Matching

When an intent is found, the Intelligence Engine assigns a score based on the number of features matched. The scoring formula for intents is as follows:

$$\text{Intent Score} = \frac{\text{The number of features in the intent which match features in the user request}}{\text{The total number of features in the intent}}$$

Figure 2-7. Intent Scoring

Given this formula the maximum possible score is 1, and the minimum possible score is 0. Intent scores are used to determine whether or not a found intent should be ignored (refer to [“Intents Advanced Settings”](#), on page 39 for more information about settings that impact how the

Intelligence Engine uses intents). Generally, you will set up your Knowledge Bases to ignore intents that have a low score (0.5 or less). A low score indicates that a found intent is probably not the correct intent. For example, *BuyFishIntent* would be found if only the *BuyFeature* matches, but using the formula we would find the intent score would be 0.5. This means there was only a 50% match, which is low. When an Intent score equals 1, that intent is called ‘certain’.

Intent Overrides

Sometimes it is desirable to have one intent “override” another. Sometimes multiple intents will be found, but one intent is more preferable than the other intent. This most frequently occurs when one intent is more specific. For example, we can have two intents *BuyFishIntent* and *BuyFreshFishIntent*. If our user asks “How do I buy fresh fish?” both of these intents will be found. Obviously we would prefer that *BuyFreshFishIntent* be used. This is accomplished by using *overrides* in the Knowledge Base. Overrides force the Intelligence Engine to choose one intent over another if both are found. There are two types of overrides in a Knowledge Base: explicit, and implicit.

Explicit Overrides

Explicit overrides are overrides that are created by you in the Knowledge Manager. This is accomplished by selecting an intent, and then selecting the intent that will override it. To achieve the desired result when the user asks the “fresh fish” question all we have to do is make the *BuyFreshFishIntent* override the *BuyFishIntent*. Refer to [“The Intents Frame”, on page 36](#) for more information.

Implicit Overrides

Implicit overrides are created automatically by the Knowledge Manager. There are three types of implicit overrides: Category overrides, Feature-Hierarchy overrides, and Intent-Feature overrides.

Category Overrides

The Knowledge Manager provides three default categories for intents: high priority, medium priority, and low priority. All intents are created under one of these categories. All intents in the high priority category automatically override any intents in the medium and low priority categories. All intents in the medium priority category override any intents in the low priority category.

Feature-Hierarchy Overrides

Feature hierarchies can create implicit overrides. Consider the feature hierarchy in [Figure 2-8](#). We might have two Intents that use this hierarchy, “FishInfoIntent” and “FreshFishInfoIntent.” We can give these intents the following structure:

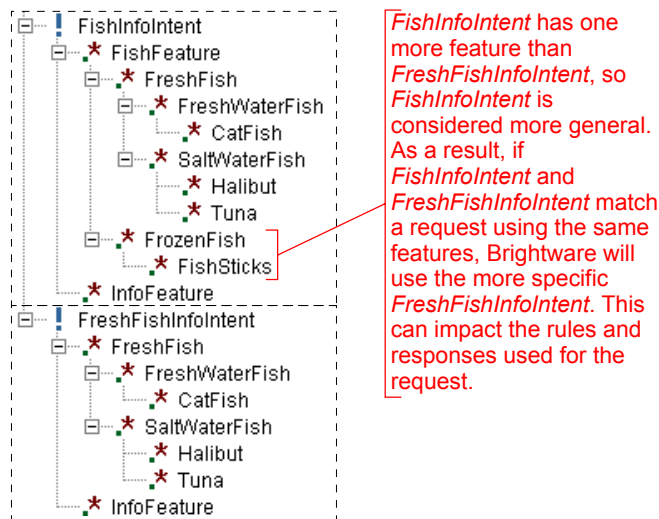


Figure 2-8. Feature-Hierarchy Overrides

Since whenever the Intelligence Engine finds a child feature in a request the parent feature is also found, child features are always considered to be “more specific” than their parents. With the structure shown in [Figure 2-8](#), the Intelligence Engine will cause the *FreshFishInfoIntent* to override the *FishInfoIntent* because *FreshFishInfoIntent* is a more specific intent. Generally, it is considered desirable when a more specific intent (meaning fewer features) overrides a more general intent. It allows the system to more accurately respond to the user’s requests.

Intent-Feature Overrides

When one intent has all of the features of another intent, and it has additional features, it is considered a more specific intent. Consider the following intents:

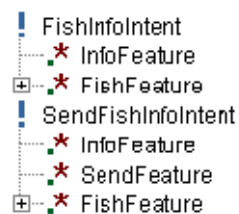


Figure 2-9. Intent Feature Overrides

In this example we can see that the *SendFishInfoIntent* contains all of the features that are found in the *FishInfoIntent*, but it also contains the *SendFeature*. From this it should be clear that the *SendFishInfoIntent* is the more specific of the two intents. In such cases the Intelligence Engine will “understand” that intents like *SendFishInfoIntent* override intents like *FishInfoIntent*. In essence if all of the features in intent (A) are in intent (B), and intent (B) contains additional features, then intent (B) will override intent (A).

Rules

Rules govern how the Intelligence Engine behaves when an intent is found. In other words, a rule connects an intent with an appropriate response. Every rule may be associated with only one intent, but multiple rules can be associated with each intent. Rules can also contain conditions. A rule is said to “fire” when the associated intent is found, and all the conditions are met.

When the Intelligence Engine analyzes rules there are three possible outcomes:

- **Zero Rules Fired** – This outcome occurs if the Intelligence Engine fails to find any intents in a user’s request. This may occur if the user submits a question that contains unusual terminology, excessive misspellings, or a lack of meaningful content.
- **Single Rule Fired** – This outcome occurs when one, and only one intent is found.
- **Multiple Rules Fired** – This outcome occurs when multiple intents are found, or if multiple rules for a single intent fire.

There are three types of rules available in a Knowledge Base: E-mail Rules, Web Rules, and Chat Rules. Each Type of rule is used to handle a request originating from its related channel.

E-mail Rules

E-mail rules are associations between intents and responses. Each e-mail rule contains the following components:

- **Rule Name** – The name of the rule. It is created when the rule is created.
- **Intent Name** – The intent with which the rule is associated. It is assigned when the rule is created.
- **Response Mode** – This specifies how the rule is to react when fired. It is selected after the rule is created.
- **Response** – This specifies the response which will be used by the rule. It is selected after the rule is created.
- **Business Unit** – This specifies the Business Unit where the associated response is located. It is assigned at the same time the response is selected.

In addition, each e-mail rule may also contain:

- **E-mail Rule Conditions** – These are used to create conditions which must be satisfied for the rule to be found. They are specified after the rule is created.
- **Additional Actions** – These are used to specify additional actions which will be enacted if the rule is found. They are specified after the rule is created.

E-mail Responses

An e-mail response is a template, or pre-formatted reply, used to respond to customer e-mail. E-mail responses are created in the Contact Center Console and stored in response libraries. There is one global response library, and each business unit contains its own response library. The Knowledge

Manager allows you to select any response from any available response library for use in an e-mail rule.

E-mail Response Modes

There are three types of Response Modes:

- **Suggest** – If this is used, the request will be sent to an agent along with the associated response.
- **Auto Reply** – If this is used, and the intent is certain, then an automatic reply will be sent to the request. The reply will contain the associated response and the Intelligence Engine will then close the request. If the intent is not certain the mode will act as if it were set to Suggest.
- **Close the Request** – If the intent is certain, the request will be closed with no further action. If the intent is not certain, the response mode will act as if it were set to Suggest.

E-mail Rule Conditions

Conditions are requirements that you specify for your rule. In order for the rule to fire, the conditions must be satisfied. Conditions use the fields contained in the request. There are three parts to any condition:

- **Field** – This is a field in the e-mail.
- **Value** – This is any piece of text.
- **Condition** – The relationship between the contents of the specified e-mail field and the specified value which must be true for the condition to be satisfied.

There are six valid field options:

- From
- Reply To
- To
- CC
- Subject
- Body

There are four types of conditions:

- = (equals)
- CONTAINS
- <> (unequal)
- DOES NOT CONTAIN

Lets say we get the following e-mail:

FROM: TonyMonopthalmus@general.non
REPLY TO: tm.per@me.non
TO: you@yours.non

CC: him@his.non

SUBJECT: Acquisition of Petra

BODY: The main force operation has rendered only moderate results.

If our rule has a condition as follows:

Table 2-2. E-mail Rule Condition - Example One

Field	Condition	Value
From	Contains	Tony

If the intent is found the rule will fire because the contents of the From Field contains “Tony”, but if the condition was set up as follows:

Table 2-3. E-mail Rule Condition - Example Two

Field	Condition	Value
From	=	Tony

The rule would not fire. This is because the contents From Field is “TonyMonophthalmus@general.non” which is not equal to the text “Tony” specified in the Value field.

Multiple Conditions

In order for a rule to fire, all conditions associated with the rule must be satisfied.

E-mail Additional Actions

Siebel Brightware allows rules to trigger additional actions beyond those specified by Response mode. These additional actions are written in the Java language by your internal developers.

If a rule with an additional action is fired, any additional actions associated will always be carried out automatically.

E-mail Rules Firing

Having a single rule fired is not always the case. In the alternate cases the following actions occur:

- **Zero Rules Fired** – In this situation the request is automatically forwarded to an agent with no suggested responses.
- **Multiple Rules Fired** – When multiple E-mail rules are fired, the Intelligence Engine uses the response modes to determine the appropriate course of action. The following rules:
 - When any combination of multiple rules includes a ‘Close the Request’ rule where the intent for that rule is not certain, the request will be sent to an agent with all responses attached, and no Auto Replies will be made.

- When any combination of multiple rules includes a ‘Close the Request’ rule where the intent for that rule is certain, the request will be closed with no further action.
- When any combination of ‘Suggest’, and ‘Auto Reply’ rules are found, the request will be sent to an agent with the suggested response related to the first rule fired, and no Auto Replies will be made. Responses for the other fired rules will be shown as suggested, but not selected.

Chat Rules

Chat rules are similar to e-mail rules with the three main exceptions:

- There are no Response Modes. The request is always forwarded to an agent (unless the chat is taking place after hours and the auto reply “Server down for maintenance” is used).
- There are no Conditions.
- There are no additional Actions.

Request Sources

A request source is the web page or site where a web user’s request originates. As discussed earlier in [“Fields”, on page 7](#), this value can be set in a hidden field on the web page. Request sources are used as conditions within rules. Using request sources, the Intelligence Engine can adjust the response to a request based on where the request originated. To illustrate this we can create an intent called *ManagerInfoIntent*, which returns the name of the department manager. We will have three pages with different managers: Sales, Customer Service, and Technical Support.

Table 2-4. Site Managers for ManagerInfoIntent

Page	Dept. Manager
Sales	Sally ‘Sales-Siren’ Saltimbocca
Technical Support	Tim ‘Tech-Man’ Talbot

If our web user is on the Sales web page and asks “Who is the manager of this department?” we want the Intelligence Engine to respond with “Sally Saltimbocca.” If the web user is on the Technical Support page the Intelligence Engine should reply “Tim Talbot.” In order to accomplish these results we will create three rules each using the *ManagerInfoIntent*, but with three different request sources.

Table 2-5. Setting Up Request Sources

Intent	Request Source	Manager
ManagerInfoIntent	Sales page	Sally
ManagerInfoIntent	Tech. Support page	Tim

Table 2-5. Setting Up Request Sources

Intent	Request Source	Manager
ManagerInfoIntent	Cust. Service page	Chuck

With the structure presented in [Table 2-5](#) the web user will receive the correct response to the request. In most cases no specific request source will be defined. This can be handled by setting the request source value for a rule to “Any” which is used by default.



You do not need to create request sources if you are not using multiple fields in the request. If you do not create request sources, the Intelligence Engine uses a default field called “question.”

Web Rules

Web rules are somewhat more complicated than E-mail rules. This is because web interaction requires the system to be more dynamic.

Simple Web Rules

Simple Web rules use Universal Resource Locators (URLs) or explanation text or both as a response. URLs are the addresses for web pages and sites on the Internet. When a URL is used as the response, the web user is routed to the web page designated by the URL. This is known as a route.

Explanation text is used to explain to the web user the action that the Intelligence Engine has taken. When explanation text is used as the response, the Intelligence Engine can send the text to the web user without routing them to a web page.

Simple web rules may also contain disambiguation text and a request source.

Table 2-6. Example of Simple Web Rules

Intent	Request Source	URL/Response	Explanation Text	Disambiguation Text
BuyFishRule	Any	http://www.FishiesRus.com/shopping/cart.html	Contact Center has routed you to the Fish shopping cart.	Do you want to buy some fish?
CeoInfoRule	Any		We do not have a CEO, we are a co-operative.	Do you want information on our Co-op?

Disambiguation Text

Disambiguation text is used when a multiple rule outcome has occurred. The disambiguation text for each of the multiple web rules is sent back to the web user. This allows the web user to select the desired action. In the example “Send me info on how I can buy catfish?” the Intelligence Engine finds both the rule associated with the *BuyFishIntent* and the rule associated with the *SendInfoIntent*. You can see the disambiguation text for these rules in [Table 2-6](#). When both of these

rules are fired we can make the Intelligence Engine respond with a web page as seen in [Figure 2-10](#). This type of web page is called a disambiguation page. In a standard implementation of the presentation layer, disambiguation text appears as hyperlinks. When one of the hyperlinks is selected the Intelligence Engine carries out the associated rule with that disambiguation text.

**We are unable to determine which of these actions you would prefer.
Please select one:**

[1.\) Do you want to buy some fish?](#)

[2.\) Do you want us to send you information?](#)

Figure 2-10. An Example of a Simple Disambiguation Page

Default Rules

The Intelligence Engine fires the default rule whenever zero rules fire. The default rule can be a simple or advanced rule. A blank default rule is automatically created in every Knowledge Base. If the default rule is not filled out, the Intelligence Engine will respond with an error message when zero rules fire.

Web Responses

Technically, every time the Intelligence Engine replies to a web user's request it uses a response. In ["Simple Web Rules", on page 18](#) you learned about using URLs and Explanation text as responses. The Knowledge Manager provides the ability to create more "advanced" web responses as objects in the web Responses Editor. These responses can be associated with one or more rules. Web Responses contain three major elements: Actions, Questions, and Explanation text.

Response Based Explanation Text

Explanation text created in the Web Responses Editor functions identically to explanation text in simple rules. If a response is added to a simple rule, the explanation text from the response will be used instead of the existing text.

Actions

Actions are behaviors. In simple rules the actions are 'routes', 'explanations' or both. Rules which use responses also can use these actions and others. Most importantly, responses allow multiple actions to be used when responding to a web user's request.

Action Types

Web Responses can use several types of actions. There are two classes of actions: Integral, and Custom. Each action type has a unique set of action parameters. These are used to define and expand the function of an action. [Table 2-7](#) shows the actions with their classifications, and parameters. Refer to [“Creating Web Response Actions”, on page 52](#) for more information about setting up actions.

Table 2-7. Action Types and Parameters

Action Type	Classification	Parameters
Route	Integral	URL, Link Text, URL Parameter, Value
Escalate to E-mail	Integral	To, Acknowledgement, Subject
Escalate to Chat	Integral	Acknowledgement, URL

Integral Actions

Integral actions are actions that are inherent to the structure of the Intelligence Engine. Currently there are three integral actions available in a Knowledge Base: Route, Escalate to E-mail, and Escalate to Chat.

- **Route** – In essence, route is a method of sending a web user to a specific web page. This is also known as “pushing” the page. Route actions will support one or more URLs for routing. Generally, the first URL in a table of URLs is the only page that will be “pushed.” Pushing of a page occurs on the presentation layer. When multiple URLs are sent, the Intelligence Engine will provide the first URL as a “primary” URL, and the rest as “secondary” URLs.
- **Escalate to E-mail** – Escalate to e-mail is used when the Intelligence Engine determines that an issue should be handled by e-mail rather than through web interaction. Escalate to e-mail uses three parameters: To, Acknowledgment, and Subject.
- **Escalate to Chat** – Escalate to Chat is used to handle live customer interaction on the web through the familiar “chat” type interface. Escalate to Chat requires two parameters: Acknowledgement, and URL.

Custom Actions

Custom actions are actions that can be added to the Intelligence Engine. These additional actions are created using the Java programming language. While Intelligence Engine documentation does not cover Java itself, the *Implementation Guide* does cover some example code.

Questions

Web Responses provide a mechanism that allows you to create questions that can be asked of the web user. The answers given by the web user can be used as data passed to the URL in route actions, or they can be used as arguments passed into custom actions. Multiple questions can be created in each response. Questions require three parameters: Question, Question attribute, and Question type.

- **Question** – This is the text of the question itself.
- **Attribute** – The attribute is a variable that is used to hold the web user’s answer to your question. Simply put, the attribute holds the “value” that the web user enters in response to your question. Attributes can pass the values to response actions. For more details see [“Actions”, on page 19](#) and [“Creating Web Response Actions”, on page 52](#).
- **Type** – The question type establishes the format and values that are expected by the posed question. The Knowledge Manager provides five different types as seen in [Table 2-8](#).

Table 2-8. Question Types and Parameters

Type	Format	Usage
String	Min. / Max. length	Enter a minimum and maximum length for a string of alphanumeric characters.
Number	Min. / Max. value	Enter a minimum and maximum value for a number.
List	List of legal values	Create a list of items that are legal values for the web user to enter.
Date	Start / End date	Enter a start date and an end date in the format (mm/dd/yyyy)
Boolean	N/A	This is always a True or False value, no formatting is required.

Advanced Web Rules

Advanced web rules are rules that use conditions or use Web responses created in the Web Response Editor.

Conditions

Conditions can be added to a rule to refine how it behaves. Each condition uses a field that is contained on the request source. When a condition is used, the content of a field is compared to a value defined in the condition. If the comparison evaluates to true, the condition is valid. A rule will only fire if all of its conditions evaluate as valid. For example, if there was an *Age* field on the request source, and it was desirable to validate that the web user was over 21, the condition would look similar to [Table 2-9](#).

Table 2-9. A Simple Condition

Field	Condition	Value
Age	>	21

If the content of the Age field is greater than 21, the condition will evaluate as true. Conditions require three parameters:

- **Field** – This is the field that the condition compares to the value. Fields are filtered based on the selected request source. If “Any” is selected as the request source, all of the fields from all of the request sources will be available to use.
- **Condition** – The condition sets the type of comparison that will be used between the field and the value. The condition options change depending upon the type of field selected. See [Table 2-10](#) for specifics on what conditions are available for each type.
- **Value** – This is the value that the condition compares to the field.

Table 2-10. Condition Types

Field Type	Available Conditions
String	Contains, <, >, <>, <=, >=
Number	<, >, <>, <=, >=
List	<, >, <>, <=, >=
Date	<, >, <>, <=, >=
Boolean	None

Rule Firing for Web

Having a single rule fired is not always the case:

- **Zero Rules Fired** – In this situation Knowledge Bases are designed so that the Intelligence Engine fires a default rule if no intents are found.
- **Multiple Rules Fired** – When multiple web rules are found, the Intelligence Engine presents a *disambiguation* page to determine the appropriate rule. When the determination is made the single rule is carried out.

Business Units

Business units represent any logical organizational entity within a business. In most cases, each Business Unit will have its own e-mail address, and be associated with a Knowledge Base(s) that provides the routing mechanisms for its incoming messages. Business Units are created in the Contact Center Console, and may be deployed on separate computers.

Additional NLP Processing

When user requests are processed, the Intelligence Engine uses three similar processes to correct and clean up the contents of each request. These processes, used in order, are Spelling, Normalization, and Morphology.



If you are attempting to diagnose why a feature is not being fired, check these NLP files for possible conflicts. For example, if you have a “CFO” feature, it probably won’t fire because the `USEnglishAbbreviationNormalization.xml` file contains an entry to change CFO to Chief Financial Officer.

NLP Processes

Spelling

The request is first reviewed by the Spelling process. In the Spelling process, words in a request are compared to the words in a spelling dictionary. If an incorrectly spelled word is found it is replaced with the correct word.

The Spelling dictionary is a third-party tool. The contents cannot be edited.

Normalization

Once the request is spell checked it is reviewed by the Normalization process. The Normalization process replaces common abbreviations, contractions, and units of measure with the full version of words or phrases. For example, the word “can’t” is replaced with “can not.” To do this, the Intelligence Engine references XML files that contain lists of abbreviations, contractions, and units of measure along with the full version of the word or phrase.

The Normalization files for English are:

- `USEnglishAbbreviationNormalization.xml`
- `USEnglishContractionNormalization.xml`
- `USEnglishUnitNormalization.xml`

Morphology

After the request is Normalized it is reviewed by the Morphology process. Morphology is used to replace words that occur frequently with different word forms. For example, the words “am,” “are,” “is,” “was,” and “were” are all forms of the word “be.” Similar to the Spelling and Normalization processes, a dictionary is used to compare common words with their desirable word forms.

Custom Morphology Dictionary

A custom Morphology Dictionary is also available for morphing specific terms and for tracking and organizing entries specific to your company’s implementation.

The Morphology Files for English are:

- `USEnglishMorphology.xml`
- `CustomUSEnglishMorphology.xml`

Here is a sample of the USEnglishMorphology.xml file:

```
<entry key="accelerated" value="accelerate" />
<entry key="accelerates" value="accelerate" />
<entry key="accelerating" value="accelerate" />
```

, Use single words for the Key attribute. Phrases such as “who am I” are ignored.

Determining the Language of Incoming Email Messages

In addition to the previously discussed NLP processes, another file is used to help determine the language of incoming email. The language-detector.xml file contains information used to detect a number of incoming languages as well as extensive documentation on the language detection process.

This file should be edited for content to further discern the language of an incoming email message, especially in relation to the terms commonly used for your particular business.

Editing NLP files

The contents of the Normalization, Morphology, and language detection files requires understanding of the NLP requirements for your company. Adding certain terms may interfere with how the Knowledge Manager infers intents. For example, an internet company would include the word “net” (for internet) in their abbreviation file, whereas a sporting goods company would not. Likewise, most businesses would include the word “Wed” (for Wednesday) in their abbreviation file, but a wedding planning company most likely would not.

The Normalization, Morphology, and language detection files can be edited or expanded to include industry or company specific abbreviations, contractions, units, and terms. It is important to understand that these files are word filters and processing time can be affected if the contents become too large. Therefore, only common terms should be included. A default set has been provided.

The files are located at \edocs\Brightware\lib\common\dictionaries.

Global Knowledge Base Rules

The Siebel Brightware product supports the specification and use of “global rules” in the Knowledge Base. Global rules apply equally to email, chat, and web and are fired appropriately in the context in which they were received. This feature is possible using “automatically matched intents”, or “Auto-Matched Intents”. Auto-Matched Intents are intents that are matched for every incoming request — Email, Web, and Chat — regardless of what features are associated with them. Any rules associated with such intents will in effect be treated as “global rules”.

Auto-Matched Intents

Auto-matched intents do not need to be associated with any features to be matched in the context of a request. However, if a feature is not associated with an intent, a warning message may be displayed on the server console when the Knowledge Base is first loaded. In the case of auto-matched intents, this warning could either be ignored or removed by attaching a feature with no associated phrases. Features tied to an auto-matched intent should not have any phrases.

The name of the intent plays an important role in determining whether or not it is treated as an auto-matched intent. By default, if an intent is given the specific name “AutoMatchedIntent” it will acquire this behavior. This keyword is defined in the configuration parameters, and can be changed if necessary (for example, it might be changed to be more appropriate to the language of the installation). To modify this keyword, alter the value of the parameter named “auto.matched.intent.name” in the config.cml file (this file contains configuration parameters for the entire system).



If the **configure.cmd** is run to regenerate the **config.cml** file, the same change needs to be made in **components-config.xml** file, located at edocs\Brightware\fpconfig.

Global Knowledge Base Rule Example

One such use of Global Knowledge Base Rules and Auto-Matched Intents is in the case of a large Siebel Brightware implementation which receives requests from a number of different web sites. Instead of requiring the Agent to infer the correct response and then attend to the actual request, the system can be set up to recognize which site the request is being sent from, and automatically select an appropriate response form, allowing the Agent to be one step closer to processing the request and fulfilling the SLA.

Siebel Brightware and Web User Interaction

Siebel Brightware interacts with your web users in an intuitive way. This allows web users to help themselves in navigating and obtaining information from your web site. In a standard application, Siebel Brightware evaluates incoming inquiries, then either directs the web user to the web page that answers the question, or provides direct responses to the specific query.

In order to produce maximum flexibility with minimum complexity, Siebel Brightware separates the information presentation from knowledge processing. Knowledge management and processing are contained within the Knowledge Manager, Knowledge Bases, Contact Center Console, and Siebel Brightware Server. Acquisition of requests and the presentation of responses is handled by the presentation layer.

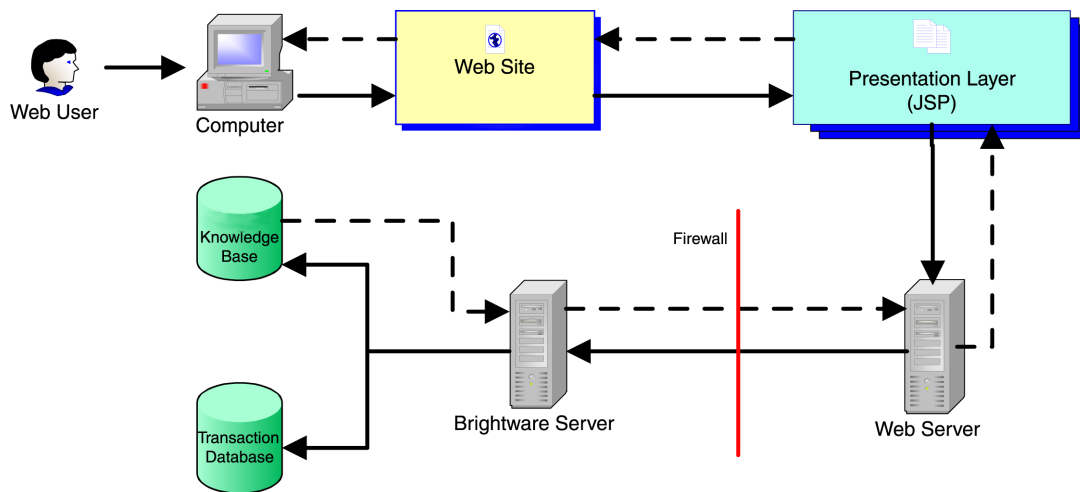


Figure 2-11. Web User Interaction

The steps of the simplified workflow are as follows:

1. The web user submits a request on the web site (Presentation Layer).
2. The request is passed to the Knowledge Base via the Contact Center Java Bean (Presentation Layer).
3. The Intelligence Engine analyzes the request using the Knowledge Base and determines the web user's intent (Siebel Brightware Server).
4. The Intelligence Engine uses rules to determine the appropriate response to the identified intent (Siebel Brightware Server).
5. The Intelligence Engine returns the response with associated data via the Contact Center Java Bean (Siebel Brightware Server).
6. The response is received and the correct presentation is determined (Presentation Layer).
7. The response is presented to the web user (Presentation Layer).

The Presentation Layer

In standard Siebel Brightware implementations, the presentation layer contains HTML pages, Java Server Pages (JSP), and JavaScript. It resides on your company's web server and integrates Siebel Brightware with your web site. Generally, the bulk of coding for the presentation layer is written in JSP. However, other methodologies are available (XML/XSL, ASP for example). Details on the JSP methods are presented in the *Implementation Guide*.

The interaction between the presentation layer and the Knowledge Base is handled using the Contact Center Java Bean. The Contact Center Java Bean provides a Java based API to Contact Center functionality. Information on its use is presented in the *Implementation Guide*.

Creating Knowledge Bases

About this Chapter

This chapter is dedicated to the Knowledge Manager and how it is used to create and edit Knowledge Bases. Every step and element of Knowledge Base creation will be detailed in easy to follow procedures.

The Knowledge Manager

The Knowledge Manager will be your primary tool when working with Knowledge Bases. The Knowledge Manager contains a number of editors which allow you to create and edit all of the Knowledge Base elements. The Knowledge Manager provides tools which are available globally in every editor. In addition, each editor provides its own unique tools contained in one or more work areas. The tools which are available globally are listed in [Table 3-1, on page 30](#), and can be seen in [Figure 3-1](#).

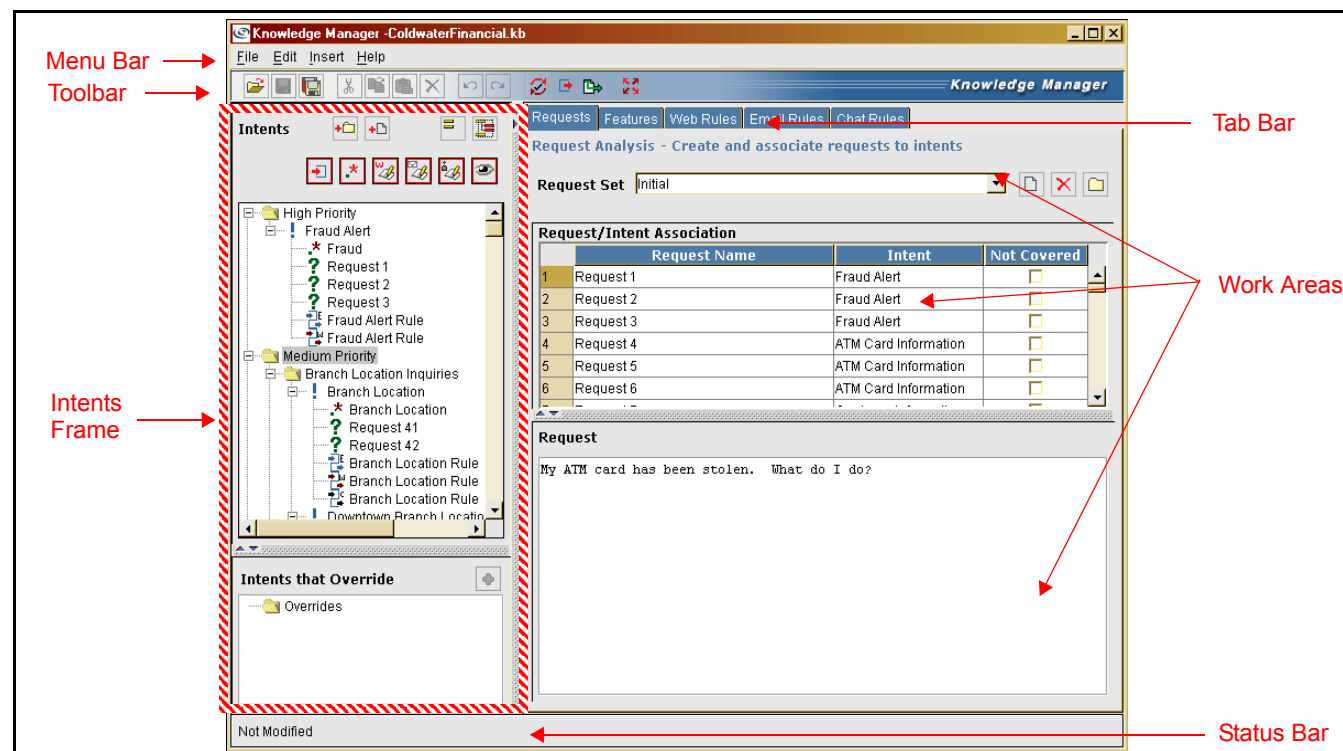


Figure 3-1. The Knowledge Manager

Table 3-1. Global Tools

Tool	Function
Menu Bar	Used for common actions and functions.
Toolbar	Used for frequent and common actions.
Tab Bar	Used for navigating between editors.
Work Areas	The primary work areas for editing the Knowledge Base.
Status Bar	Displays information about the working Knowledge Base.
Intents Frame	An Intent centric editor.

Menu Bar

The Menu bar contains the following structure and commands. Some of these commands are available through the shortcut menu when you right-click an item in the Knowledge Manager.

File

- **New KB** – Creates a new blank unnamed Knowledge Base.
- **Open KB** – Open a Knowledge Base that has been referenced by the database.
- **Delete KB** – Deletes a selected Knowledge Base.
- **Load KB from Disk** – Import a new Knowledge Base that has not been referenced in the database.
- **Save KB** – Saves the current Knowledge Base.
- **Save KB As** – Saves the current Knowledge Base to a different file.
- **Generate HTML for Printing** – Saves the Knowledge Base in a readable and printable HTML format.
- **Exit** – Exits the Knowledge Manager.

Edit

- **Undo** – Un-does the last action taken.
- **Redo** – Re-does the last action that was undone.
- **Cut** – Cuts the currently selected item, and puts it in the clipboard.
- **Copy** – Copies the currently selected item to the clipboard.
- **Paste** – Pastes the current contents of the clipboard into the selected location.
- **Delete** – Deletes the currently selected item.

Insert (right-mouse click only)

- **Intent** – Inserts an intent under the current folder.
- **Request** – Inserts a request under the current intent.

- **Feature** – Inserts a feature under the current intent.
- **Web Rule** – Inserts a Web rule under the current intent.
- **E-mail Rule** – Inserts an e-mail rule under the current intent.
- **Chat Rule** – Inserts a chat rule under the current intent.

Help

- **About** – Displays info about the Knowledge Manager.
- **User Guide** – Opens this document.

Toolbar

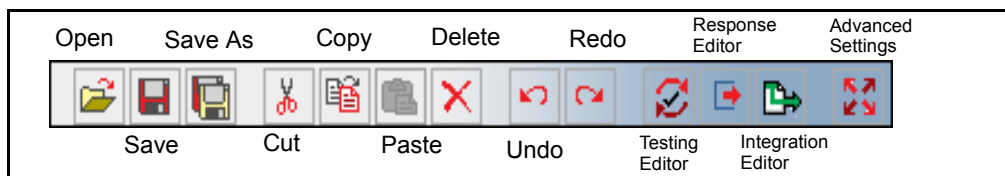


Figure 3-2. The Toolbar

The toolbar contains of the following buttons:

- **Open** – Loads a Knowledge Base from disk or database.
- **Save** – Saves the current Knowledge Base.
- **Save As** – Saves the Active Knowledge Base as a different version.
- **Cut** – Cuts the currently selected item, and puts it in the clipboard.
- **Copy** – Copies the currently selected item, and puts it in the clipboard.
- **Paste** – Pastes the current contents of the clipboard into the selected location.
- **Delete Item** – Deletes the currently selected item.
- **Undo** – Un-does the last action taken.
- **Redo** – Re-does the last action that was undone.
- **Testing Editor** – Displays the Testing Editor.
- **Response Editor** – Displays the Web Response Editor.
- **Integration Editor** – Displays the Integrations Editor.
- **Advanced Settings Editor** – Displays the Advanced Settings dialog box.

Tree Expand and Collapse Buttons

Throughout the Knowledge Manager there are numerous tree controls. Each of these controls are equipped with buttons that allow you to expand and collapse the tree.



Figure 3-3. The Tree Expand and Collapse Buttons

New Buttons

Throughout the Knowledge Manager there are numerous tree controls. Each of these controls is equipped with buttons that allow you to create new folders or objects.



Figure 3-4. The Tree Expand and Collapse Buttons

Status Bar

The status bar shows the status of the Knowledge Base (modified, or not modified).

Work Areas

The work area always contains the user interface for the functional area selected in the tab controls. You can easily navigate between work areas by simply selecting different tabs.

Intents Frame

The Intents frame creates an “intent centric” editing experience. The frame is used to create, edit, and structure intents, and intent overrides. It is also used to create and organize all of the other main elements of a Knowledge Base, including: rules, features, and requests. The elements within the frame are organized by their associated intents.

Editors

The Knowledge Manager contains eight editors. The editors are used to create the various items within a Knowledge Base. The following list provides a brief description of each editor:

- **Requests** – This editor is used to import, analyze and create request sets.
- **Features** – This editor is used to create features, feature hierarchies, phrases, and regular expressions.
- **Web Rules** – This editor is used to create Web Rules.
- **E-mail Rules** – This editor is used to create simple and advanced rules.
- **Chat Rules** – This editor is used to create simple and advanced rules.
- **Web Responses** – This editor is used to create and edit responses that can be used in rules.
- **Testing** – This editor is used with request sets to interactively test the coverage and accuracy of the Knowledge Base.
- **Integrator** – This editor is used to set up request sources and request source fields. These can be used to determine the web user’s situation.
- **Intents Frame** – This frame is used to manage the all of the Knowledge Base elements.

Navigation to editors is handled in the Tab bar and the Toolbar. To navigate to an editor, select the tab or button with the editor’s name or image.

Knowledge Base Management

The File menu includes two commands that help you manage the Knowledge Bases that are stored as files or in a database. You can view the name given to the Knowledge Base as well as the version, creation date, and modification date.

- **Open** – Used to open/load a selected Knowledge Base (Menu & Toolbar).
- **Delete** – Used to Delete, or Un-delete a selected Knowledge Base (Menu Only).

By storing Knowledge Bases in the Siebel Brightware database they can be accessed across a network. This allows you to create Knowledge Bases on one computer, and use them somewhere else. For more information about setting up databases refer to the *Installation Guide*.

You may also save Knowledge Bases as KB files so they can be easily transferred to other systems or worked on off-line. This is the easiest method to use when you are working on your local computer.

You can only work on one Knowledge Base at a time. The active Knowledge Base is displayed on the title bar.



Only Knowledge Bases which are stored in the database will generate transactional data. File Knowledge Bases do not store transactional data when used, and cannot be used to support reporting.

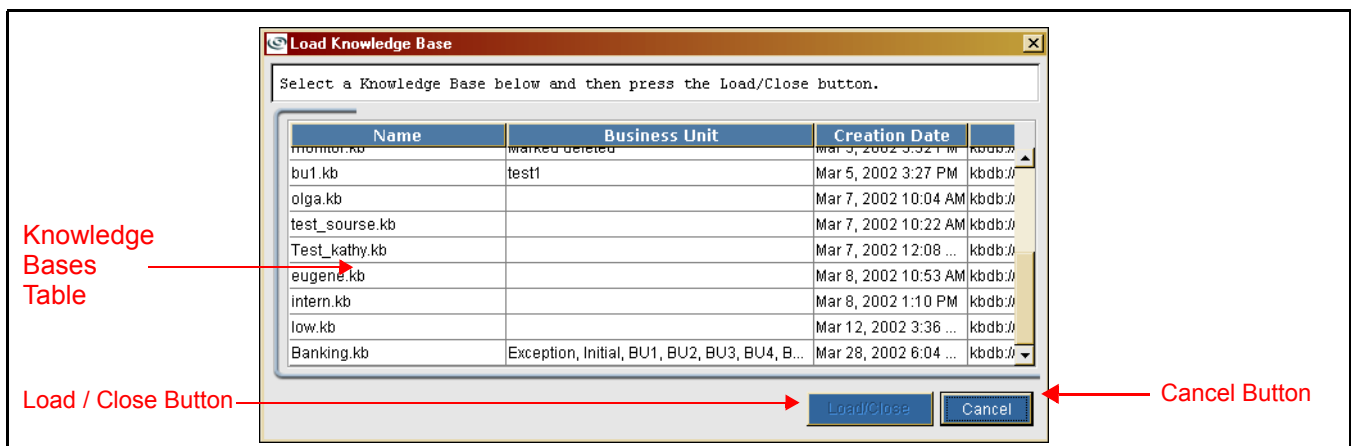


Figure 3-5. The Open Dialog

To create a new Knowledge Base:

1. From the Menu bar select **New KB** from the **File** menu.
2. Save the Knowledge Base.

To load or open an existing Knowledge Base:

1. Double-click the Knowledge Base in the Knowledge Bases Table.

Or

1. Select the desired Knowledge Base in the **Knowledge Bases** table.
2. Click the **Load** button

Or

1. Select **Open** from the **File** menu.
2. Select a file in the **Open** dialog box.
3. Click **OK**.

To save a Knowledge Base:

1. Click the **Save** button.

To save an unnamed or file Knowledge Base to the database:

1. Click the **Save As** button.
2. Click the **Database** option.
3. If the Knowledge Base is unnamed, assign a name in the **Save KB As** dialog.
4. Click **OK**.

Or

1. Select **Save** from the **File** menu.
2. Click the **Database** option.
3. If the Knowledge Base is unnamed, assign a name in the **Save KB As** dialog.
4. Click **OK**.

To save a unnamed or database Knowledge Base as a file:

1. Select **Save KB** from the **File** menu.
2. Click the **File** option.
3. Click **Save KB**.

To save a Knowledge Base with a new name:

1. Click the **Save As** button.
2. Type the Knowledge Base name into the **Save KB As** dialog.
3. Specify the desired **Database**, **File** or **Other** option.
4. Click **OK**.

To delete a Knowledge Base from the database:

1. Click **Delete** in the **File** menu.
2. Select the desired Knowledge Base in the **Knowledge Bases** table.
3. Click the **Delete** button.
4. Click **OK** to delete.

To save a Knowledge Base file with a new name:

1. Select **Save KB As** from the **File** menu.
2. Click the **File** option.
3. Type the Knowledge Base name into the **File Name** field of the **Save KB As** dialog.
4. Click **Save**.

To load a new Knowledge Base from disk:

1. Select **Load KB from Disk** from the **File** menu.
2. Navigate to the appropriate directory.
3. Select the Knowledge Base file.
4. Click **Open**.

The Intents Frame

The Intents frame is the primary editor for creating and managing objects in the Knowledge Base. There are three primary areas in the Intents frame:

- **Intents Tree** – Used for creating and organizing intents.
- **Filter Buttons** – Used to show or hide elements in the Intents tree. A red border indicates that all of the related items will be shown in the tree.
- **Intent Overrides Tree** – You can specify intents that override the selected intent.

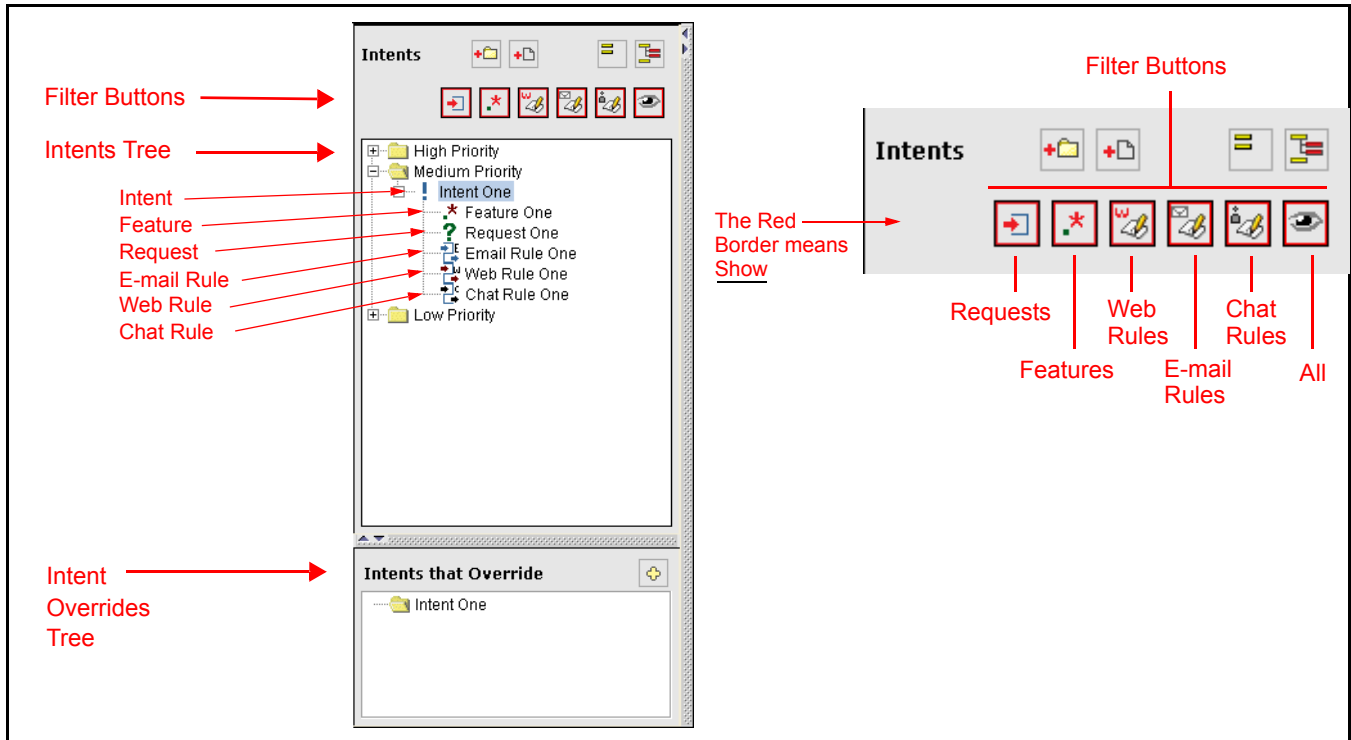


Figure 3-6. The Intents Editor

Creating Intents

Intents are created to represent the requests or intents of the web user. Intents can be organized by using folders. The intents tree allows you to create and organize the intents in a Knowledge Base. Use the following procedures to add and organize intents in your Knowledge Base.

To add intents to the Intents tree:

1. In the Intents tree, select the category under which you want to add the intent.
2. Press the **Enter** key.
3. Type in the name of the new intent.
4. Press the **Enter** key again to finish.

Or

1. In the **Intents** tree, select the folder under which you want to add the intent.
2. Click **Intent** in the **Insert** menu.
3. Type in the name of the new intent.
4. Press the **Enter** key to finish.

To add an intents category folder:

1. In the **Intents** tree, select the category under which you want to add the new category folder.
2. Click the **Add Folder** button.
3. Type in the name of the category folder into the **Folder Name** dialog.
4. Click **OK**. The new category folder will be placed under the category you selected in step1.

To delete an intent:

1. Select the intent you want to delete.
2. Click the **Delete** button.

Or

1. Select the intent you want to delete.
2. Click the right mouse button to open the local menu.
3. Select **Delete**.



You can use drag-and-drop to arrange your intents. However, any explicit intent overrides associated with an intent will be deleted when you drag-and-drop.

Automatic Object Association

The Intents frame is not limited to the creation of intents, it can also be used to create requests, features, and rules that are automatically associated with specific intents. The type of objects that can be created are dependant upon which editor tab is open. When the Features editor is open features can be created, and when the web rules editor is open Web rules can be created.

To create a automatically associated object:

1. Select the intent to which you want to associate the object.
2. Open the editor appropriate for the object type (e.g. Web Rules).
3. Press the **Enter** key.
4. Type in the desired name for the object.
5. Press the **Enter** key to finish.

The new object will be created, and automatically associated with the selected intent.

Creating Intent Overrides

Creating explicit overrides forces Siebel Brightware to choose one intent over another if both are found. This gives added power and flexibility to your Knowledge Bases. Use the following procedures to manage and create overrides.

To add an intent override to an intent:

1. Select an intent that you want to override in the **Intents** tree.
2. Click the **Add Override** button. The **Overriding Intent Selection** dialog appears.
3. Select the intent that will override the intent selected in step 1.
4. Click **OK**.

Or

1. Select an intent that you want to override in the **Intents** tree.
2. Select a second intent and drag-and-drop it into the **Override** tree.

To delete an intent override:

1. In the **Intents** tree select an intent from which you want to remove an override.
2. In the **Override** tree select the override you want to remove.
3. Click the **Delete** button.



At this point it is important to remember: When one of the sub-features is recognized by the NLP, the parent feature is also recognized. For example, when the request enters the system: "I'd like Information about Contact Center" the Product Information intent is found because the Contact Center feature is a sub-feature of the Product Feature.

Intents Advanced Settings

The Intents Advanced Settings dialog allows you to set advanced parameters for intent matching and overrides in your Knowledge Base. To display the Intents Advanced Settings dialog, click the **Advanced** button in the toolbar.

- **Intent Ignore Threshold** – A number from 0 to 1 that sets the minimum score for an intent to be found (refer to [“Intent Scoring and Matching”](#), on page 11 for more information about intent scoring).
- **Intent Override Discrimination** – A number from 0 to 1 that sets the maximum difference allowable for overriding to be valid when the overriding intent has a lower score than the intent it overrides. If intent (B) overrides intent (A) and the score for intent (A) is greater than the score for intent (B) then the intent (B) override will be ignored if the difference between the intent scores is greater than the *Intent Override Discrimination* value.
- **Intent Validity Range** – A setting that affects whether or not Siebel Brightware finds an intent. In order for an intent to be valid it must be within the Intent Validity Range. By default, the range is set at (0.2). To be valid, an intent must have a score equal to or greater than the score of the highest scoring intent minus the range. For example, if the score for the highest scoring intent equals 1, the intent range will be 1 to 0.8 (1 minus 0.2), and to be valid any other intent must have a value of 0.8 or greater.
- **Language** – The language used by the server to determine the intents within the Knowledge Base.

The Requests Editor

The Requests Editor allows you to enter or import requests that you expect to come into the web site. You want to get a good understanding of the types of requests that are coming into the web site so you can determine how best to respond. You can use the Requests Editor to analyze the requests and determine how to develop the Knowledge Base.

Generally, Knowledge Base development begins in the Request Editor. This is done by creating or importing a large sampling of typical requests and then categorizing them by intent. Categorizing requests by intent is accomplished by “Linking” requests to intents. The following example shows how intents might be categorized.

The following questions:

- How much is Contact Center?
- What’s the price of Contact Center?
- What does Contact Center do?
- Please give me some information about Contact Center?
- Will Contact Center fit into my budget?

May be grouped under just two intents:

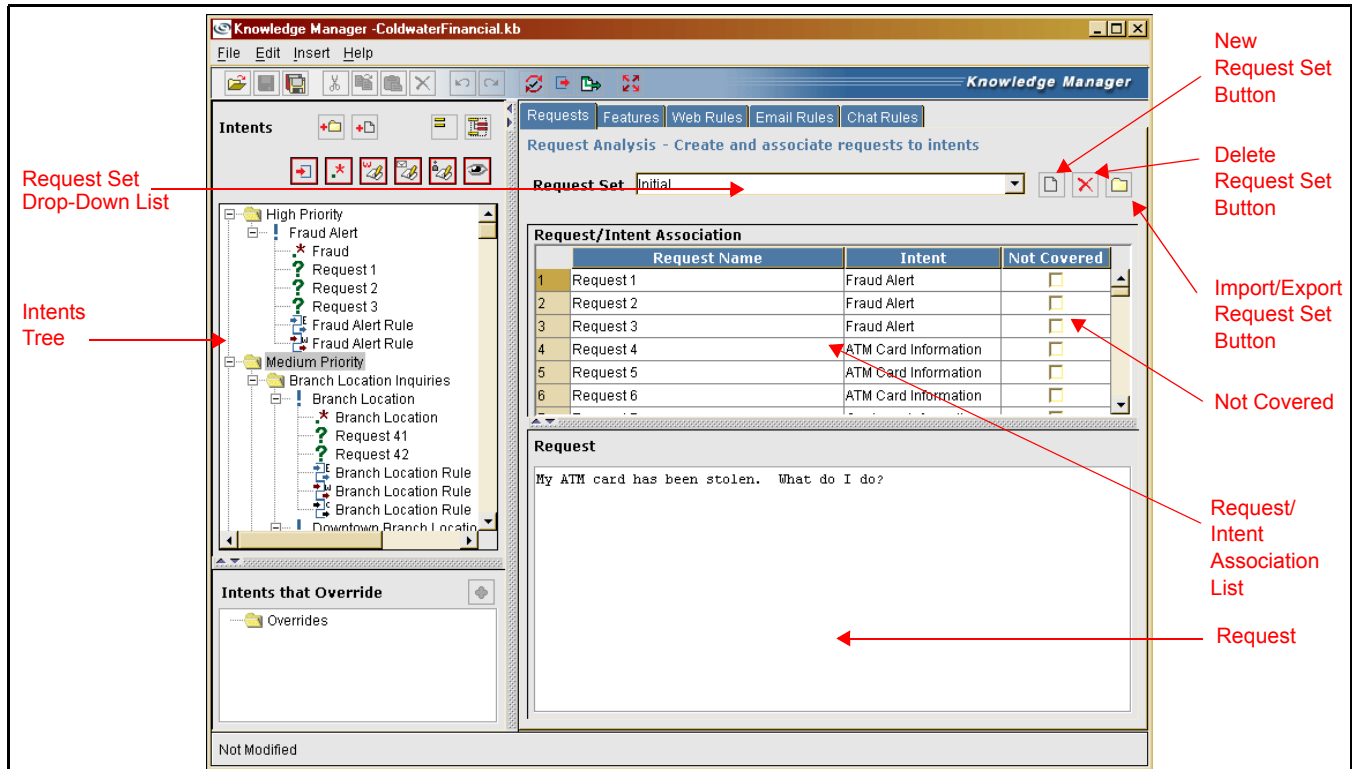
- Contact Center Price Request
 - How much is Contact Center?
 - What’s the price of Contact Center?
 - Will Contact Center fit into my budget?
- Contact Center Info Request
 - What does Contact Center do?
 - Please give me some information about Contact Center?

This linking process is referred to as “Labeling the Requests.”

There are many potential sources you can use to determine the content of the requests you create:

- Web logs which track how web users utilize your web site.
- E-mails from customers.
- Reports from Customer Service and Technical support agents.
- Web Site maps which indicate the resources available on your site.

These and other sources can provide you with the critical information necessary to create Knowledge Bases. Before creating a Knowledge Base for your business, it is suggested that you gather as much of this information as possible. It will make your Knowledge Bases more effective, and make them easier to create.



Requests Editor Procedures

The following set of procedures cover all of the steps necessary to create or edit a request set.



Each new Knowledge Base is provided with an initial request set.

To add a request set:

- Import an existing request set
Or
- Create a request set

To import a request set:

The file must be in a Comma Separated Value (CSV) format.

1. Click the **Import/Export Request Set** button.
2. Click the **Import** button.
3. Select a file in the **Open** dialog.
4. Click **Open**.
5. Click **Close** when finished.

To Export a request set to a CSV file:

1. Click the **Import/Export Request Set** button.
2. Click the **Export** button.
3. Type a file name into the **Save** dialog.
4. Click **Save**.
5. Click **Close** when finished.

To create a request set:

1. Create a new request set name.
2. Add requests to the request set.
3. Add intents to the Knowledge Base.
4. Link requests to intents.

To create a new request set name:

1. Click the **New Request Set** button.
2. Type the request set name into the **Request Set Name** dialog.
3. Click **OK**.

To add requests to a request set:

1. Click in the blank **Request Name** field of the **Request/Intent Association** list.
2. Type new request names into the *Request Name* field.
3. Press the **Enter** key to move to the next request field.

To modify the content of a request:

1. Select the request to be modified in the **Request/Intent Association** list.
2. Select the **Request** field.
3. Edit the text of the request.



By default the content of a request is the same as the request's name.

To delete a request set from the Knowledge Base:

1. Select a request set in the **Request Set** drop-down list.
2. Click the **Delete Request Set** button.

To create an automatically associated Intent/Request:

1. In the Intent tree, select the intent to which you want to associate the object.
2. Press the **Enter** key.
3. Type in the desired name for the Request.

4. Press the **Enter** key to finish.

To link unassociated requests to intents:

1. Select an intent from the **Intent** tree.
2. Select a request by clicking its number from the **Request/Intent Association** list.
3. Click the right mouse button and select **Add Intent**.

Or

1. Select a request by clicking its number from the **Request/Intent Association** list.
2. Drag-and-drop the request onto an intent.



Requests can be linked to an unlimited number of intents. Conversely intents may be linked to an unlimited number of requests.



Careful consideration should be used when linking requests to intents. If requests are shared by too many intents it creates an unreadable tangle of links. If Intents are carefully created most requests will only link to one or two intents.

To unlink requests to intents:

1. Select a request from the **Intent** tree.
2. Press the **Delete** key.

Not Covered Requests

When a request is not being used (it is not linked to an intent) it is best to mark it as Not Covered by marking the **Not Covered** check-box adjacent to the request in the **Request/Intent Association** list. When a request is marked Not Covered it is ignored during testing. This prevents false results from creeping into your test routines.

The Feature Editor

The Features Editor is for creating and maintaining Features, Phrases, and Regular Expressions. The Features editor area consists of the following two panels. The rest of this section will discuss these panels in detail.

Features Tree – Used for creating features and defining the feature hierarchy.

Feature Indicators – Used to define the phrases and regular expressions for each feature.

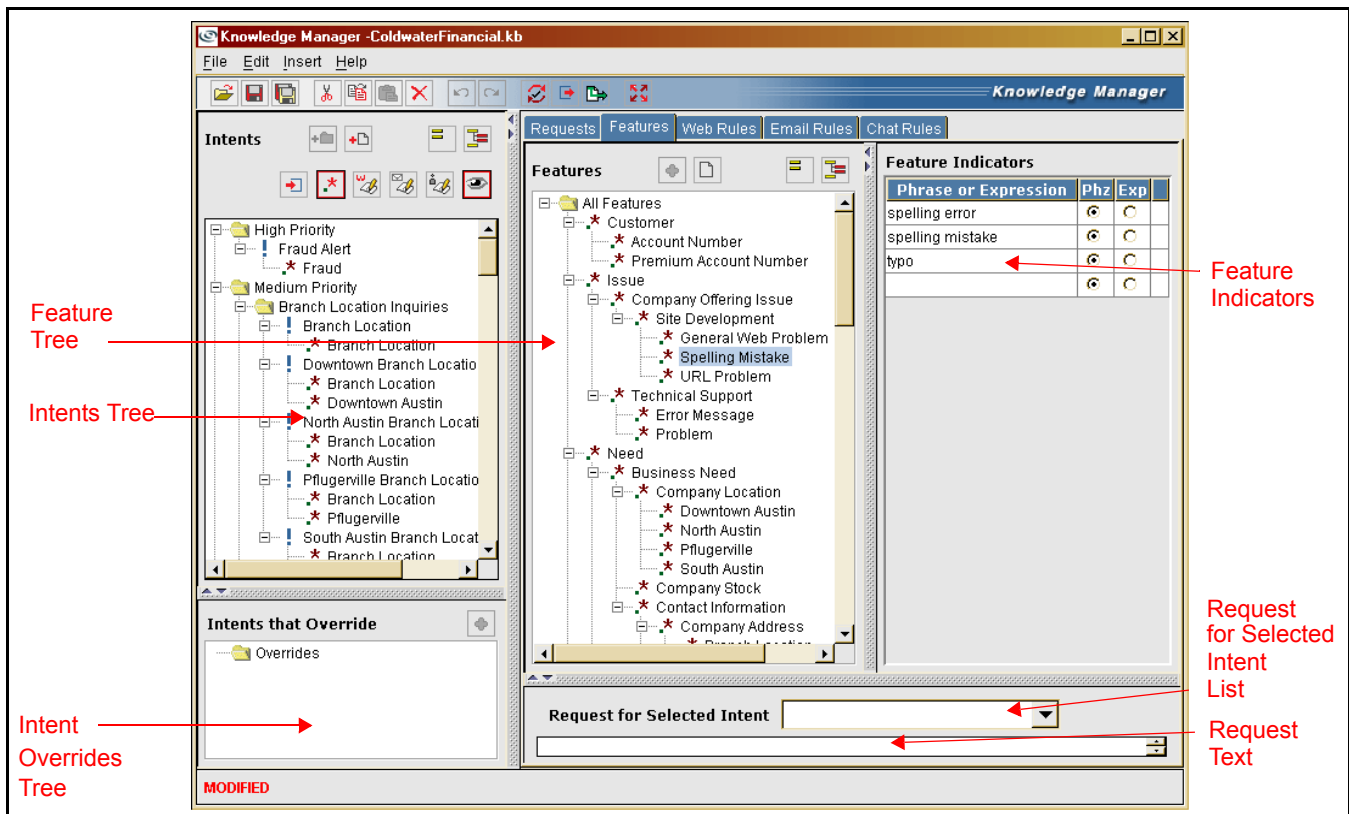


Figure 3-8. The Feature Editor

Creating Features

Features are the elements from which intents are composed. Each feature is a collection of phrases and/or regular expressions that encapsulate an idea. Features can be created before or after intents, but features must be created before the phrases and regular expression they contain. Use the following procedures to add and organize features in your Knowledge Base.

To add a new feature:

1. Select the *All Features* folder.
2. Press the **Enter** key.
3. Type in the name of the new feature.
4. Press the **Enter** key again to finish.

To create an automatically associated feature:

1. In the Intent tree, select the intent to which you want to associate the feature.
2. Open the Feature editor.
3. Press the **Enter** key.
4. Type in the desired name for the feature.
5. Press the **Enter** key to finish.

To add a sub-feature:

1. In the **Feature** tree, select the feature under which you want to add the new sub-feature.
2. Press the **Enter** key.
3. Type in the name of the new feature.
4. Press the **Enter** key again to finish.

To delete a feature:

1. Select the feature you intend to delete.
2. Click the **Delete** button.

Or

1. Select the feature you intend to delete.
2. Click the right mouse button to open the local menu.
3. Select **Delete**.

Creating Phrases and Regular Expressions

Phrases and regular expressions give features their meaning. Use the following procedures to add phrases and regular expressions to your Knowledge Base.

To add phrases to a feature:

1. Select a feature in the **Features** tree.
2. Click in the first open field of the **Feature Indicators** list.
3. Type the phrase into the field.
4. Press the **Enter** key if you want to enter another phrase.

To paste phrases into a feature:

1. Highlight and copy a list of phrases from any text editor (notepad or emacs for example).
2. Select a feature in the **Features** tree.
3. Click in the first open field of the **Feature Indicators** list.
4. Click the right mouse button to open the local menu.
5. Select one of the following paste options:

- *Paste - Break between words* – This will paste the copied text into the list. Each word in the text will be added as a separate phrase.
 - *Paste - Break between lines* – This will paste the copied text into the list. Each line in the text will be added as a separate phrase.
6. Press the **Enter** key if you want to enter another phrase.

To add regular expressions to a feature:

1. Select a feature in the **Features** tree.
2. Click in the first open field of the **Feature Indicators** list.
3. Type the regular expression into the field.
4. Select the **Exp** option button adjacent to the field to indicate that the item is a regular expression.
5. Press the **Enter** key if you want to enter another regular expression.

To paste regular expressions into a feature:

1. Highlight and copy a list of regular expressions from any text editor (notepad or emacs for example).
2. Select a feature in the **Features** tree.
3. Click in the first open field of the **Feature Indicators** list.
4. Click the right mouse button to open the local menu.
5. Select *Paste - Break between lines* – this will paste the copied text into the list. Each line in the text will be added as a separate regular expression.

Make sure that the text you use for regular expressions separates each regular expression onto separate lines. Do not use text that contains multiple regular expressions per line, and do not use the *Paste - Break between lines* option. The composition of regular expressions can cause breakage if this is done incorrectly.

6. Select the **Exp** option button adjacent to the field to indicate that the item is a regular expression.

To delete a phrase or regular expression:

1. Select the phrase or regular expression you intend to delete.
2. Click the **Delete** button.

Or

1. Select the phrase or regular expression you intend to delete.
2. Click the right mouse button to open the local menu.
3. Select **Delete**.

Using the Request for Selected Intents List

The **Request for Selected Intent** list allows you to select each of the requests associated with the selected intent. When a request is selected, the associated text is shown in the Request text field. Part or all of this text may be copied to compose phrases or regular expressions.

Associating Features with Intents

Sometimes it may be necessary to add unassociated features to an intent. Use the following procedures to add features to intents.

To add a feature to an intent:

1. Select the feature in the **Features** tree.
2. Drag-and-drop the feature onto the appropriate intent.

Or

1. Select an Intent.
2. Select a feature.
3. Click the right mouse button to open the local menu.

To delete a feature from an intent:

1. Select the feature in the **Intents** tree.
2. Click the **Delete** button.

Or

1. Select the feature in the **Intents** tree.
2. Click the right mouse button to open the local menu.
3. Select **Delete**.

The Testing Editor

Testing allows you to verify the accuracy and coverage that your Knowledge Base provides. To perform testing, it is necessary that you have a request set, and that intents and features have been built. The results of testing can be examined in the Test Results table, Filter Results table, and the Match tree.

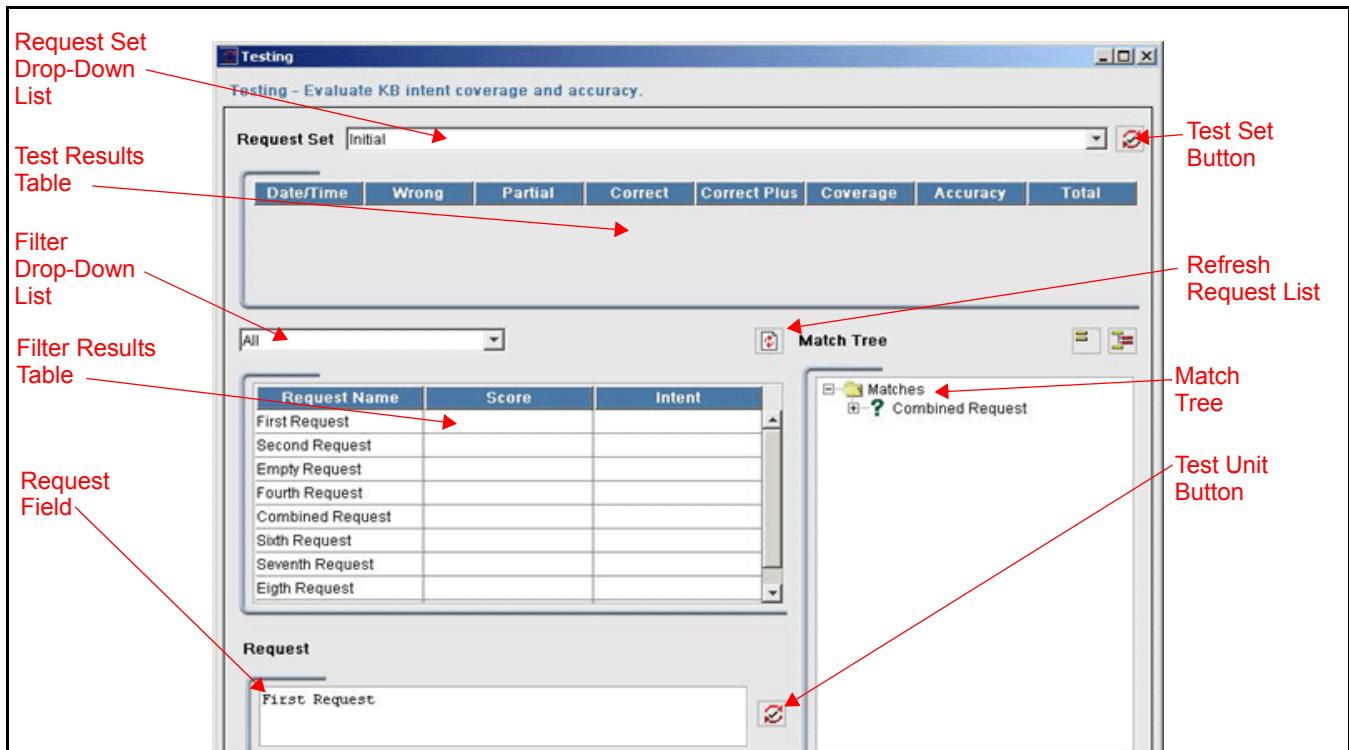


Figure 3-9. The Testing Editor

To test a request set:

1. Select the request set to be tested from the **Request Set** drop-down list.
2. Click the **Test Set** button.

Test Results Table

The Test Results Table shows the results of each request set test. Within the table, the coverage and accuracy columns give the quickest summary as to how the Knowledge Base is performing.

- **Coverage** – A measure of the types of questions that the Knowledge Base is designed to handle vs. the actual types of requests that come into the sites. For example, if 90% of all of the web user requests were to change their password, and the Knowledge Base implements only 1 intent (*Password Change*), then the coverage of this Knowledge Base would be 90%. $\text{Coverage} = ((\text{Total} - \text{Unclassified}) / \text{Total}) * 100$.
- **Accuracy** – Given the same example above, if you take 100 examples of Password Change requests and the *Password Change* Intent is found 80 times, the accuracy of the Knowledge Base would be 80%. Knowledge Base development is about making trade-offs between coverage and accuracy, as coverage goes up, accuracy will go down. $\text{Accuracy} = ((\text{Correct} + \text{Partial} + \text{Correct Plus}) / \text{Total}) * 100$ (percent accuracy).

The other columns; wrong, partial, correct, and correct plus, give some insight into where to focus in improving the Knowledge Base.

- **Wrong** – The NLP either identified the wrong intents, or didn't find any intents at all for labeled questions.
- **Partial** – Some of the correct intents were found by the NLP engine, but not all of them were found.
- **Correct** – The intent(s) found by the NLP engine were the same as the intent(s) assigned by the user in the Request editor. This is clearly the best situation to be in.
- **Correct Plus** – All of the same intent(s) were found by the NLP engine as specified by the user, but some additional intent(s) were also found. In this situation, the web user would be provided with a disambiguation page and could choose the correct situation. This number also includes intents that would have been correct except for the fact that they are implicitly overridden.

Filtering and Unit Testing

The Testing Editor provides you with the ability to sort testing results with filters. The following filters are provided:

- **All** – Shows all requests.
- **Wrong** – Shows the requests that were scored as wrong.
- **Partial** – Shows the requests that were scored as partial.
- **Correct** – Shows the requests that were scored as correct.
- **Correct Plus** – Shows the requests that were scored as correct plus.

To filter the results:

1. Select a filter from the **Filter** drop-down list.
2. Review the results in the **Filter Results** table.

To test a single request:

1. Select the request set to be tested from the **Request Set** drop-down list.
2. Select a request by name from the **Filter Results** table.
3. Click the **Test Unit** button.

To test a modified single request:

1. Select the request set to be tested from the **Request Set** drop-down list.
2. Select a request by name from the **Filter Results** table.
3. Edit the content of the request in the Request field.
4. Click the **Test Unit** button.



Individual requests that are modified for testing are not permanently modified in the request set. Modifications remain only for the duration of the Unit testing.

Filter Results Table

The Filter Results table shows the results of your test organized by a filter. It displays the request and the intent which matches that request and the score of that intent.

Testing Match Tree

The Match tree shows the structure of how intents were or were not found for each request. The Match tree uses the following icons to represent results:

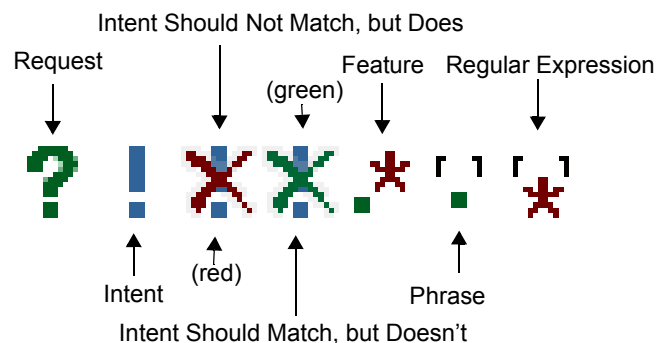


Figure 3-10. Match Tree Icons

By reading the icons in the Match tree you can find why an intent matched a request. The tree will show you which features, phrases, and regular expressions caused a particular intent to be found. When you run a test on a single unit, the tree will show you the results for only that unit.

The Web Responses Editor

Advanced Web rules often use Web responses to handle web user requests. It is therefore necessary to create the responses before creating the associated rule. For this reason, responses are discussed before rules. If you are not going to be creating advanced rules you may skip to the section on rules. The Web Responses Editor contains three main editing areas: Web Response tree, Question tab, and Actions tab. The rest of this section will deal with how to use each of these areas. To open the Web Response Editor, click the **Web Responses** button in the toolbar.

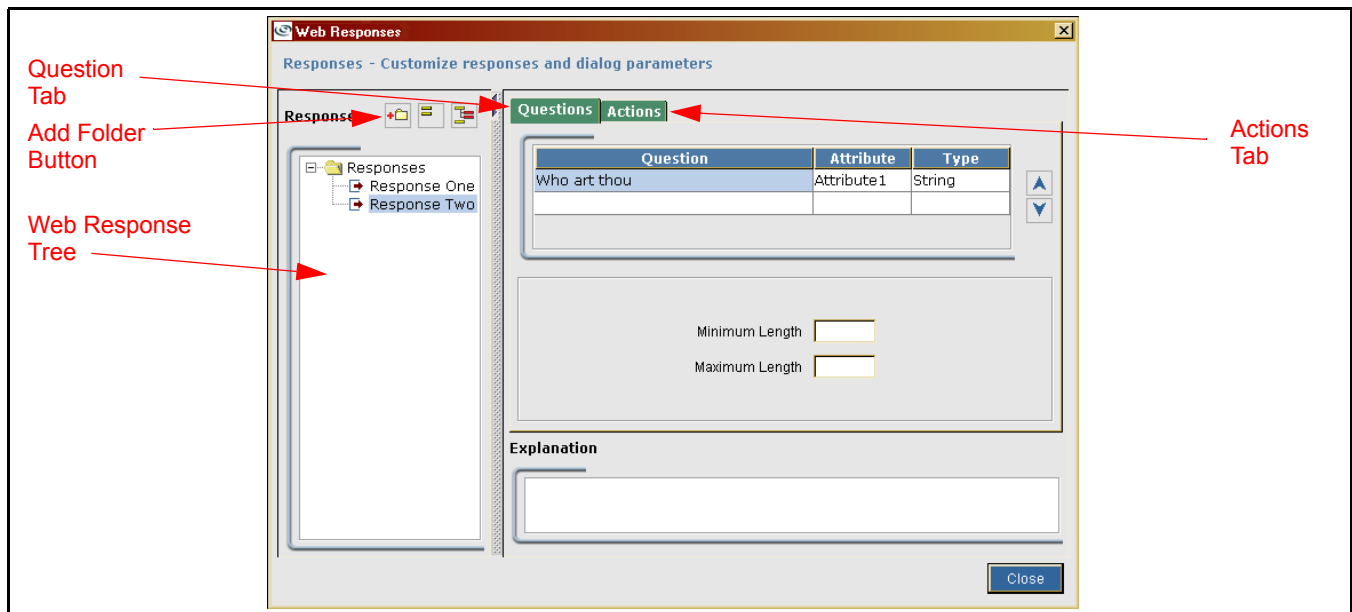


Figure 3-11. The Web Responses Editor

Creating Web Responses

Web Responses are a major part of Knowledge Base interaction with web users. The creation of web responses is a two step procedure. First the response must be created, then the content of the response may be added. The Knowledge Manager also allows you to create web response categories. The categories are maintained as folders in the Web Response tree.



Web Response categories are only for organizing your responses. They have no impact on the processing of the Knowledge Base.

The following procedures detail the steps necessary for creating and maintaining responses and response categories:

To create a response category folder:

Click the **Add Folder** button.

Type the folder name into the **Folder Name** dialog.

Click **OK**.

To create a new web response:

1. Select a category folder in the **Web Response** tree.
2. Press the **Enter** key.
3. Type the response name directly into the **Web Response** tree.
4. Press the **Enter** key again to finish.

Or

1. Select the appropriate category folder in the **Web Response** tree.
2. Click the right mouse button to open the local menu.
3. Select **New**.
4. Type the response name directly into the **Web Response** tree.
5. Press the **Enter** key again to finish.

To delete a response:

1. Select the response in the **Web Response** tree.
2. Press the **Delete** key.

Or

1. Select the response in the **Web Response** tree.
2. Click the right mouse button to open the local menu.
3. Select **Delete**.

Web Response Explanation Text

Web Responses contain their own explanation text. The explanation text in a response will be used in place of explanation text created in the Web Rule Editor. As a result, it is necessary to create explanation text in every response.

Creating Web Response Actions

Providing dialog support and URL parameter bindings is one reason to use the Web Response editor. Another reason is to provide additional types of actions besides URL routing. This is done in the Actions tab of the Web Response editor.

The screenshot shows the 'Actions' tab in the Web Response Editor. It contains several tables and a text field:

- Actions table:** A table with two columns: 'Description' and 'Action Type'. The 'Action Type' column has a dropdown menu currently set to 'Route'.
- URL Table:** A table with two columns: 'URL' and 'Link Text'. The 'URL' column contains the text '/financialDemo/privacySecurity.html' and the 'Link Text' column contains 'Privacy & Security'.
- URL Parameter Table:** A table with two columns: 'URL Parameter' and 'Value'.
- Response Explanation Field:** A text area labeled 'Explanation' containing the text: 'Your privacy and security of your personal information is our top priority at Coldwater Financial.'

Red arrows point from labels on the right to each of these components.

Figure 3-12. Web Response Editor Action Tab

Siebel Brightware provides three integral action types:

- **Route** – Routes the web user to a URL. The URL must use the format http://. URL parameters can be included with the URL.
- **Escalate to E-mail** – Prompts the user to escalate their request to E-mail.
- **Escalate to Converse** – Prompts the user to escalate their request to a chat session.

To create an action:

1. Select the **Actions** tab.
2. Select the first blank **Description** field in the **Actions** table (optional).
3. Type in a description of the action.
4. Select an action type in the **Action Type** drop-down list.

To delete an action:

1. Select the action in the **Actions** table.
2. Click the **Delete** button.

Route Actions

Route actions can take the following parameters:

- **URL** – This is the web address. It may be used to send a user directly to the page, or it may appear as a link in a web page. The use of the URLs is handled on the presentation layer. In a standard implementation, if one URL is entered then the web user will be sent to the associated web page. If multiple URLs are entered then the web user will be sent to the first one, and the others will be displayed as hyperlinks.
- **Link Text** – If a URL appears as a hyperlink in a web page this text will appear as the link.
- **URL Parameter** – Some URLs use URL parameters to control page content. For example, if an entered URL takes parameters:

<http://www.quicken.com/investments/charts/?period=3MONTHS&symbol=MSFT>

then you can specify the required parameters (e.g. period and symbol) for the web user to enter.

- **Value** – For URL parameters you can specify a constant value, or select one of the attributes defined in the question area. When an attribute is used, the answer that the web user gives to the question is used as the value for the parameter.

To create a route action:

1. Type a description of the action into the first blank **Description** field (optional).
2. Select the *Route* type in the **Action Type** drop-down list.
3. Select the first blank **URL** field in the **URL** table.
4. Type in the URL.
5. Press the **Tab** key twice to jump to the associated **Link Text** field.
6. Type in link text for the URL.
7. Press the **Tab** key and then the **Enter** key if you want to add more URLs.

To add URL parameters to a URL:

1. Select a URL in the **URL** table.
2. Select the first blank **URL Parameter** field in the **URL Parameter** table.
3. Type in the URL parameter.
4. If a value is required, press the **Tab** key twice to jump to the associated **Value** drop-down list.
5. Type in a default value, or select a value from the question attributes.
6. Press the **Tab** key and then the **Enter** key if you want to add more URL parameters.

To delete a URL:

1. Select the URL in the **URL** table.
2. Click the right mouse button and select **Delete**.

To delete an URL parameter:

1. Select the URL in the **URL** table.
2. Select the URL parameter in the **URL Parameter** table.
3. Click the right mouse button and select **Delete**.

Escalate to E-mail Actions

Escalate to E-mail is used when you determine that an issue should be handled by e-mail rather than through web interaction. Escalate to E-mail uses three parameters as follows:

- **Acknowledgment** – The “Acknowledgement” parameter is used to hold the explanation text about the escalation. In general it is used to explain that the issue is being escalated and why it is being escalated.
- **Subject** – The “Subject” parameter holds the subject line for the e-mail.

To create an Escalate to E-mail action:

1. Type in a description of the action into the first blank **Description** field (optional).
2. Select the *Escalate to E-mail* type in the **Action Type** drop-down list.
3. Select the **Acknowledgement** field.
4. Type in the acknowledgement.
5. Select the **Subject** field.
6. Type in the subject.

Escalate to Chat Actions

Chat is used to handle live customer interaction on the web. Escalate to Chat requires two parameters as follows:

- **Acknowledgement** – The “Acknowledgement” parameter is used to hold the explanation text about the escalation. In general it is used to explain that the issue is being escalated and why it is being escalated.
- **URL** – The “URL” parameter is used to hold the URL location of the Converse implementation.

To create a Escalate to Chat Action:

1. Type in a description of the action into the first blank **Description** field (optional).
2. Select the *Escalate to Chat* type in the **Action Type** drop-down list.
3. Select the **Acknowledgement** field.
4. Type in the acknowledgement.
5. Select the **URL** field.
6. Type in the URL.

Custom Actions

Custom actions are actions that can be added to Siebel Brightware. These additional actions are created using Java. While the Siebel Brightware documentation does not cover Java itself, the *Implementation Guide* does cover some example code.



Custom actions cannot include a response to the web user. Therefore, the web user will receive a “response not selected” error when a custom action is used. To avoid the error, use an additional action which responds to the web user in the Knowledge Base responses that use custom actions.

Multiple Actions

Siebel Brightware allows multiple actions to be assigned to web responses. However, there is a restriction on the use of multiple actions. Only the first action in the actions table can use the Contact Center Java Bean. All other actions will be ignored by the Java Bean. This means that you cannot use multiple integral actions. For example, you cannot assign a route action and an escalate to e-mail action to the same response. Multiple custom actions can be used if the additional actions do not use the Java Bean. You can use an integral action with additional custom actions to create

multiple actions for a response. However, the integral action must be the first action in the action table.

Creating Web Response Questions

Web Responses provide a mechanism that allows you to create questions that can be asked of the web user. This creates a dialog interface between the web user and your Knowledge Base. The answers given by the web user can be used as data passed to the URL in route actions, or they can be used as arguments passed into custom actions. Multiple questions can be created in each response.

Figure 3-13. Web Response Editor Question Tab

Questions require three parameters: question, question attribute, and question type.

- **Question** – The question parameter is where you create the question itself. In other words this is where you create the text of the question.
- **Attribute** – The attribute is the variable you use to hold the web user’s answer to your question. Simply put, the attribute holds the “value” that the web user enters in response to your question. Attributes can be used to pass these values to response actions. See [“Creating Web Response Actions”](#), on page 52 for more information.
- **Type** – The question type establishes the format and values that are expected by the posed question. There are five different types. Their description and required parameters are as follows:

Table 3-2. Web Response Question Types

Type	Format	Usage
String	Min. / Max. length	Enter a minimum and maximum length for a string of alphanumeric characters.

Table 3-2. Web Response Question Types

Type	Format	Usage
Number	Min. / Max. value	Enter a minimum and maximum value for a number.
List	List of legal values	Create a list of items that are legal values for the web user to enter.
Date	Start / End date	Enter a start date and an end date in the format (mm/dd/yyyy).
Boolean	none	This is always a True or False value, so formatting is not required.

To create a response question:

1. Select the **Question** tab.
2. Select the first blank **Question** field in the **Question** table.
3. Type in the name of the question.
4. Press the **Tab** key twice to jump to the associated **Attribute** field.
5. Type in attribute name.
6. Select a question type in the associated **Type** field.
7. Set the parameters for the question type.

To delete a question:

1. Select the action in the **Questions** table.
2. Click the right mouse button and select **Delete**.

The Integrator Editor

Requests can come in from many locations on a web site. For example, the home page or the support page. You can control Siebel Brightware's interaction with the web user based upon the origin of the request. In order to accomplish this you can integrate your web site with your Knowledge Base. To perform this integration, you simply create a *request source* for each page that will be configured to submit requests to Siebel Brightware. You then specify the information required to integrate with the web site. In addition, custom code must be added to each web page that acts as a request source.

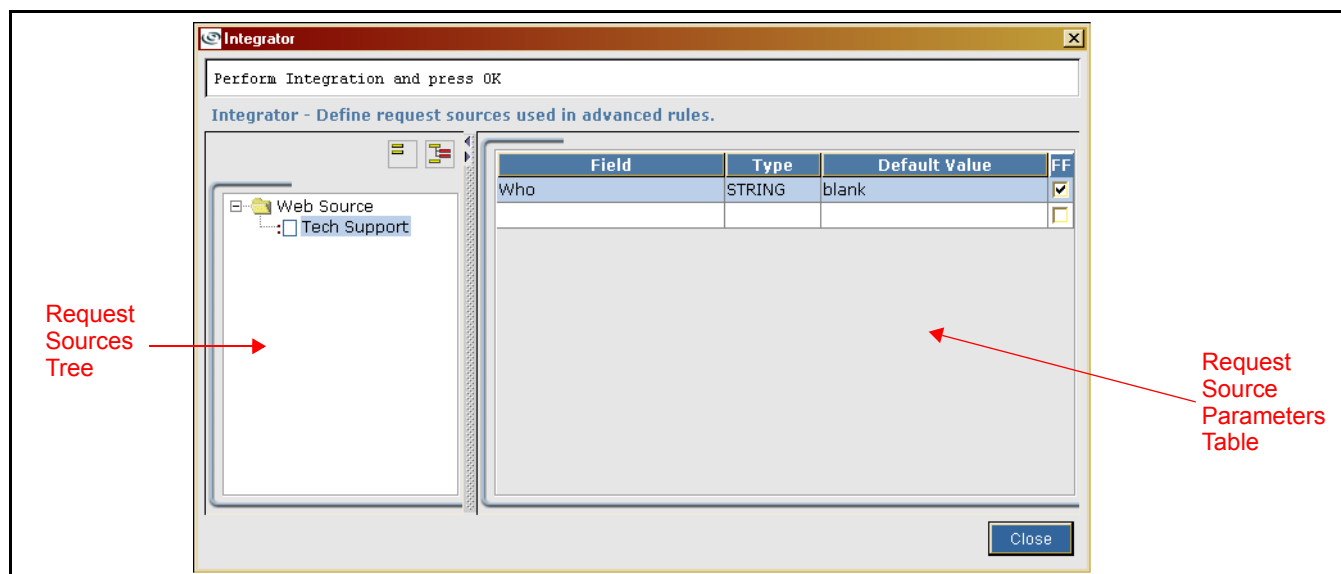


Figure 3-14. The Integrator Editor

The five parameters required for each request source are as follows:

- **Field** – The various fields that are present from the request source.
- **Type** – The Type value is used to specify the data type of each field. Siebel Brightware accepts five standard data types: string, number, date, and Boolean.
- **Default value** – The default value parameter is used to set the default value of each field.
- **FF** – The FF (Free Form) checkbox is used to specify which fields Siebel Brightware is to treat as the “request”. Remember that NLP is only carried out on these fields. If you are using request sources in your Knowledge Base you must create at least one “request” field for NLP. The field can be named anything you want, but if it is not created, Siebel Brightware will not process the request. You do not have to create request sources if you are not using fields other than the standard “question” field. If you do not create request sources, Siebel Brightware will assume the use of a default request field called “question”.

To create a new request source:

1. Select the request source folder in the **Request** tree.
2. Press the **Enter** key.
3. Type the request source name directly into the **Request** tree.
4. Press the **Enter** key again to finish.

Or

1. Select the request source folder in the **Request** tree.
2. Click the right mouse button to open the local menu.
3. Select **New**.
4. Type the request source name directly into the **Request Source** tree.
5. Press **Enter**.

To set the parameters for a request source:

1. Select the request source in the **Request** tree.
2. Click in the first blank **Field** field of the **Request Source** table.
3. Type in the name of the field.
4. Press the **Tab** key twice.
5. Select the Type in the **Type** drop-down list.
6. Press the **Tab** key twice.
7. Type in the default value.
8. Select the **FF** checkbox if the field is the free-form field to be used for NLP.

To delete a request source:

1. Select the request source in the **Request Source** tree.
2. Click the right mouse button and select **Delete**.

Or

1. Select the request source in the **Request Source** tree.
2. Click the right mouse button to open the local menu.
3. Select **Delete**.

The Web Rules Editor

Web Rules are used to determine the appropriate action that should be taken when an intent originating from the web is found. Because Web rules can be complex, we will divide them into two types: simple and advanced. The Web Rules Editor is used to set up both types.

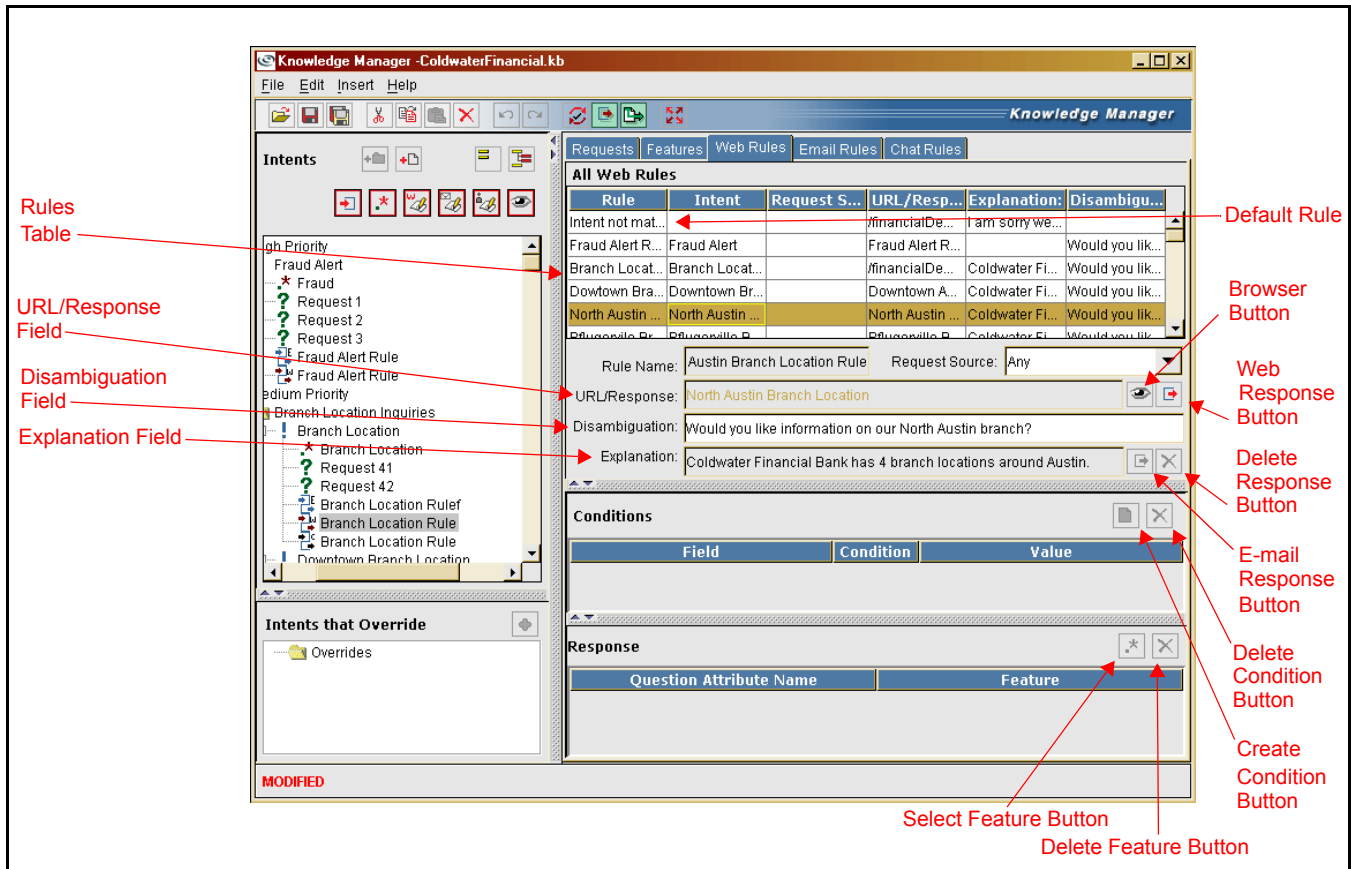


Figure 3-15. The Web Rules Editor

Simple Rules

A simple Web rule connects an intent to a URL. There are two types of simple Web rules: route and answer. Simple route rules route the user to a URL. Simple answer rules send a text reply to the web user.

To create a simple route rule:

1. Open the Web rule editor.
2. Select the intent to which you want to associate the rule.
3. Press the **Enter** key.
4. Type in the desired name for the rule.
5. Press the **Enter** key to create the rule.
6. Select the **URL/Response** field.
7. Type or paste in a URL.

8. Press the **Tab** key to move to the adjacent **Disambiguation Text** field.
9. Type in the disambiguation text.
10. Press the **Tab** key to move to the adjacent **Explanation Text** field.
11. Type in the explanation text.

To create a simple answer rule:

1. Select the intent to which you want to associate the rule.
2. Open the Web rule editor.
3. Press the **Enter** key.
4. Type in the desired name for the rule.
5. Press the **Enter** key to create the rule.
6. Select the **Disambiguation Text** field associated with that intent.
7. Type in the disambiguation text.
8. Press the **Tab** key to move to the adjacent **Explanation Text** field.
9. Type in the explanation text.



Notice that the only difference between a route rule and an answer rule is that the answer rule does not contain a URL.

To duplicate a rule:

1. Select the rule in the Intents tree.
2. Click the right mouse button and select **Copy**.
3. Select the Intent with which you want to associate the copy.
4. Click the right mouse button and select **Paste**.

Setting the Rule Request Source

The request source is the web page or site where a request originates. You can set the request set for any rule to any existing request source in the Knowledge Base. By default every Knowledge Base starts with one request source called “any.” Additional request sources have to be created.

To set a rules request source:

1. Select the rule for which you want to set the request source.
2. Select the **Request Source** drop-down list associated with that rule.
3. Select the request source from the drop-down list.

Request Sources as a Simple Rule Conditions

Within a Knowledge Base it is possible to have multiple Web rules which are all based on the same intent. To differentiate between these rules you can create conditions. Using request sources, you can cause Siebel Brightware to adjust a rule’s behavior based on where the request originated.

To set a request source itself as a condition:

1. Determine the rule for which the reply to the web user should be dependant on the origin of the request.
2. Make copies of that rule.
3. Set a unique request source for each copy of the rule.

The Default Rule

Since it is possible that Siebel Brightware will fail to find an intent in the web user's request, it is necessary to have a means of dealing with that situation. When Oracle's Siebel Brightware fails to find an intent, the Default Rule is used. The first rule in the Web rules table is always the default rule. It cannot be deleted or moved.

To setup the default rule:

1. Select the first row of the Web Rule table.
2. Type a URL into the Default Rule **URL/Response** field.
3. Type explanation text into the **Explanation** field.

To setup the default rule with a response:

1. Type a URL into the Default Rule **URL/Response** field.
2. Type explanation text into the Default Rule **Explanation** field.
3. Use the Response button to open the **Web Response Selector** dialog.
4. Click on a response to select it.
5. Click **OK**.

Using Responses (e-mail) as Explanation Text

It is possible to use the body text from an e-mail response as the explanation text for a Web rule. This ensures that web users and e-mail users receive identical information.

To use an e-mail response for explanation text:

1. Select the Web rule.
2. Click the **E-mail Response** button. This will open the Set Web Rule Explanation dialog.
3. Select the **Get Explanation from Response Library** option.
4. Click the **Choose Domain Response from Library** button.
5. Select the desired e-mail response from the Response tree.
6. Click **OK**.
7. Click **OK**, again.

To delete an e-mail response for explanation text:

1. Select the Web rule.
2. Click the **Delete e-mail response for explanation text** button.

Advanced Web Rules

Advanced Web Rules use Web Responses, conditions, and features.

Advanced Rules with Conditions

The most basic of type of advanced rules are rules with conditions. In order to use conditions, a Request Source with fields must be selected. There are three parameters for each condition:

Field – This is the field that you want to use in the condition. Fields are filtered based on the selected request source. If “Any” is selected as the request source, conditions cannot be used.

Condition – The condition setting is the method of comparing the content of the field with the set value. The condition options change depending upon the type of field selected, the following table specifies what conditions are available based on type.

Table 3-3. Advanced Rule Condition Settings

Field Type	Conditions
String	Contains, <, >, <>, <=, >=
Number	<, >, <>, <=, >=
List	<, >, <>, <=, >=s
Date	<, >, <>, <=, >=
Boolean	None

Value – The value is the setting to which the content of the field is compared.

To create a condition:

1. Select the simple rule you want to change to an advanced rule.
2. Click the **Create Condition** button. This opens a dialog.
3. Select a field from the **Field** drop-down list.
4. In the **Conditions** table, select a condition from the **Condition** drop-down list.
5. Type the desired value into the **Value** field.

To delete a condition:

1. Select the condition in the **Conditions** table.
2. Click the **Delete Condition** button.

Advanced Rules with Web Responses

The user can select a Web Response to be used when a rule occurs. Only one response can be used with any rule.

To create an advanced rule with responses:

1. Select the simple rule you want to change to an advanced rule.
2. Use the **Web Responses** button to open the **Web Response Selector** dialog.
3. Click on a response to select it.
4. Click **OK**.

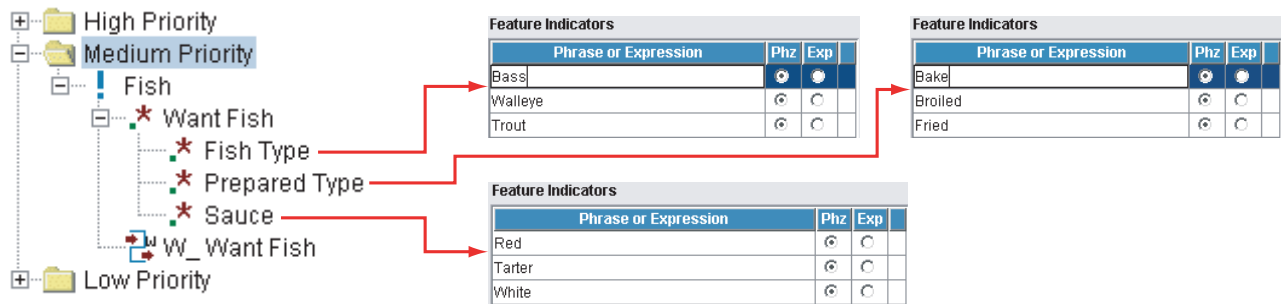
Web Rules with Web Responses

When a rule uses a response, the response's question attributes are available through the Advanced Rule Editor. These attributes can be associated with features in the Knowledge Base.

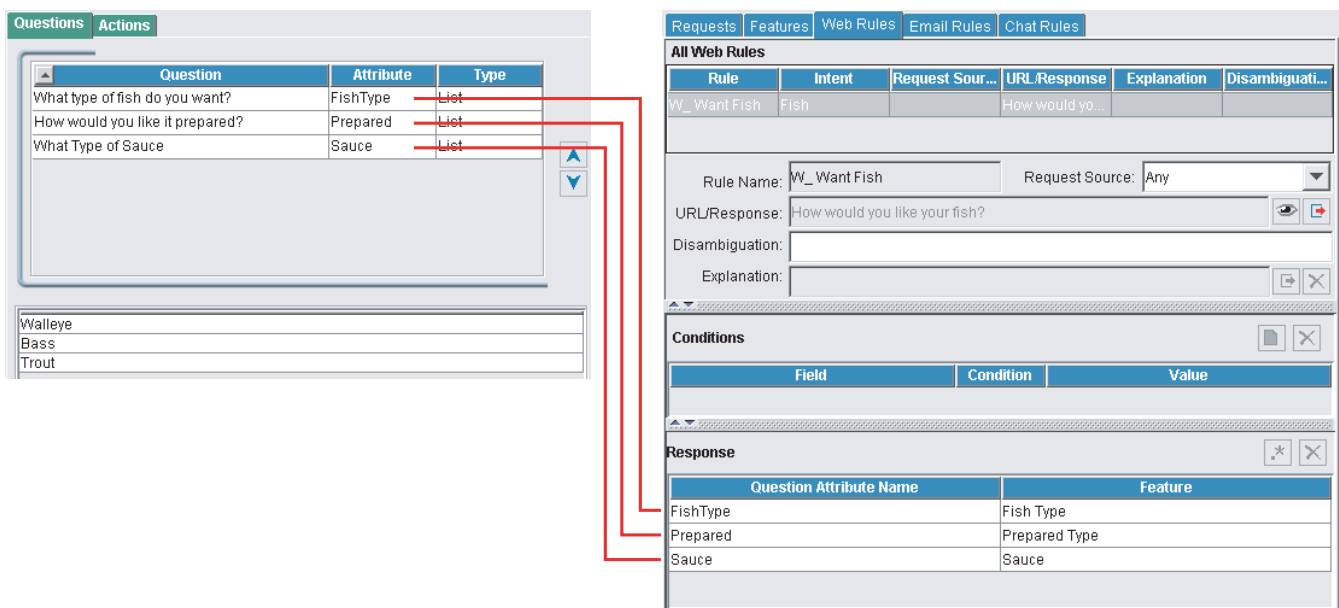
In [Figure 3-16, “Want Fish Example”, on page 65](#), if the web user were to request, “How can I buy some pan fried bass?” or “How can I buy fish?” the *Fish* intent would be found for either question. If we set up the Knowledge Base to respond with the questions “What type of fish?” (with an associated list of types), “How would you like it prepared?” and “What type of sauce?”, the response would be appropriate for “How can I buy fish?” but not for “How can I buy some pan fried bass?”. After all, the web user has already stated that they want pan fried bass.

To make the system look a little smarter to the web user, we can use matched phrases in the *Fish Type*, *Prepared Type*, and *Sauce* features. The matched phrases can be tied to the follow up questions of the web response on the Web Rules Editor. Now when the web user asks his fish question, any provided information (such as bass, fried, and tarter sauce) in the request will be filled in for the user on the follow up questions.

1 Set up Features to identify what the user wants.



2 Write questions that will help clarify any missing information. Tie the questions to the features.



3 When the user makes a request, the follow up questions are displayed, if needed, with the known answers already filled in.

I would like some bass, pan fried.

Concierge™ your question

Concierge requires some additional information before it can answer your requests. Please answer the following questions:

What type of fish do you want?

How would you like it prepared?

Do you want it with sauce?

SUBMIT

By associating responses with features, the answers to follow up questions will be supplied if the request already contained those answers.

Figure 3-16. Want Fish Example

To create an advanced rule with features:

1. Select the simple rule you want to change to an advanced rule.
2. Use the **Web Responses** button to open the **Response Selector** dialog.
3. Click on a response to select it.
4. Click **OK**.
5. Select the question attribute, in the Response section, to which you want to associate a feature.
6. Click the **Associate Feature** button. This will open the **Feature Chooser** dialog.
7. Click on a feature to select it.
8. Click **OK**.
9. Repeat steps 6 through 9 for each attribute/feature association you want to create.

To delete a response associated feature:

1. Select the feature.
2. Click the **Delete Feature** button.

The E-mail Rules Editor

E-mail Rules are used to determine the appropriate action that should be taken when an intent originating from an e-mail is found.

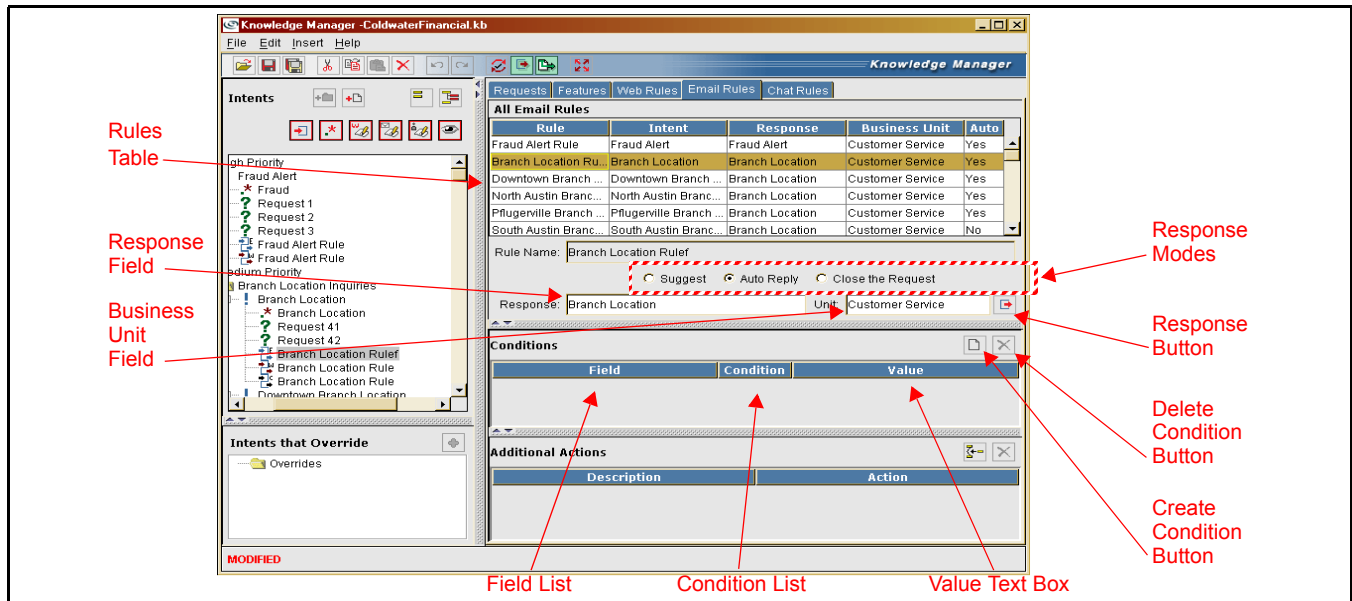


Figure 3-17. The E-mail Rules Editor

Creating E-mail Rules

An e-mail rule connects an intent to a response. Depending on the response mode, the e-mail is then closed, forwarded with the response to an agent, or generates an automatic reply.

To create an e-mail rule:

1. Select the intent to which you want to associate the object.
2. Open the E-mail Rule editor.
3. Press the **Enter** key.
4. Type in the desired name for the rule.
5. Press the **Enter** key to finish the blank rule.
6. Select the rule in the Rule Table, or the Intents tree.
7. Select a Response Mode for the Rule.
8. Add a response.
9. Add a condition (if desired).

To duplicate a rule:

1. Select the rule in the Intents tree.
2. Click the right mouse button and select **Copy**.
3. Select the Intent with which you want to associate the copy.

- Click the right mouse button and select **Paste**.

Selecting a Response

In order for an e-mail rule to work, a response must be assigned. The responses for e-mail rules are created in the Contact Center Console and are contained within response libraries. There is one global library, and each business unit has its own library.

To select a response for an e-mail rule:

- Click the **Response** button. This will open the Responses dialog.
- If the response library for a desired Business Unit is not showing, select the desired business unit from the Business Unit list.
- Select the desired response from the response in the response tree.
- Click **OK**.

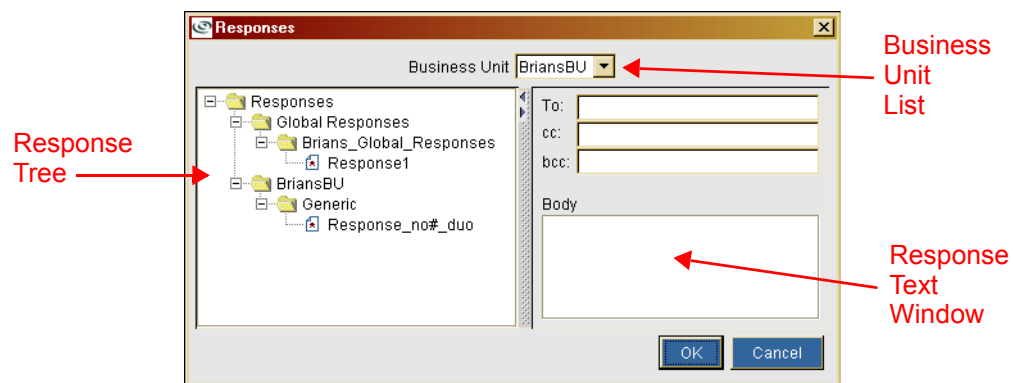


Figure 3-18. Responses Dialog

Setting E-mail Rule Conditions

There are three parameters for each condition:

Field – This is the field that you want to use in the condition. There are six valid field options:

- From
- Reply To
- To
- CC
- Subject
- Body

Condition – The condition setting is the method of comparing the content of the field with the set value. The condition options change depending upon the type of field selected, the following table specifies what conditions are available based on type. There are four types of conditions:

- = (equals)
- CONTAINS

- \neq (unequal)
- DOES NOT CONTAIN

Value – The value is the setting to which the content of the field is compared.

To create an e-mail rule condition:

1. Click the **Create Condition** button.
2. Select the field type from the field list.
3. Select the Condition from the Condition list.
4. Type the value into the Value text box.

To delete an e-mail rule condition:

1. Select the condition in the condition table.
2. Click the **Delete Condition** button.

The Chat Rules Editor

Chat Rules are used to determine the appropriate action that should be taken when an intent originating from Chat is found.

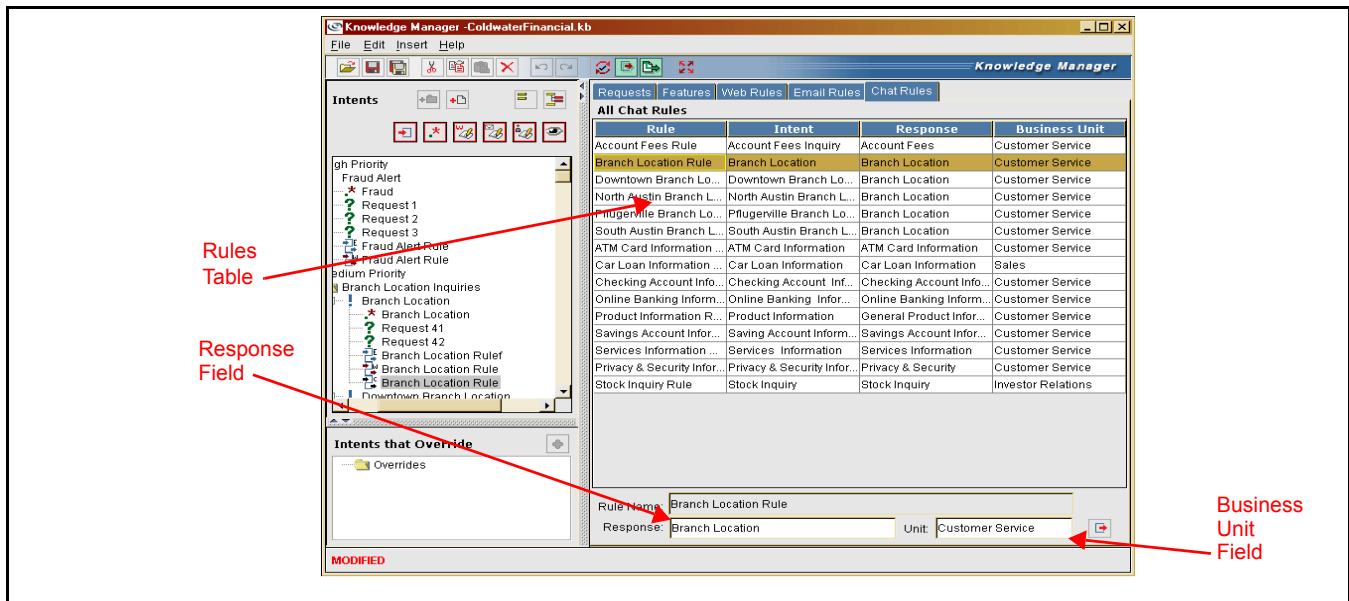


Figure 3-19. The Chat Rules Editor

Creating Chat Rules

A chat rule connects an intent to a response. Depending on the response mode, the chat request is forwarded with the suggested response to an agent, or generates an automatic reply.

To create an Chat rule:

1. Select the intent to which you want to associate the object.
2. Open the Chat Rule editor.
3. Press the **Enter** key.
4. Type in the desired name for the rule.
5. Press the **Enter** key to finish the blank rule.
6. Select the rule in the Rule Table, or the Intents tree.
7. Add a response.

To duplicate a rule:

1. Select the rule in the Intents tree.
2. Click the right mouse button and select **Copy**.
3. Select the Intent with which you want to associate the copy.
4. Click the right mouse button and select **Paste**.

Selecting a Response

In order for a chat rule to work, a response must be assigned. The responses for chat rules are created in the Contact Center Console and are contained within Response libraries. There is one global library, and each business unit has its own library.

To select a response for an e-mail rule:

1. Click the **Response** button. This will open the Responses dialog.
2. If the response library for a desired Business Unit is not showing, select the desired business unit in the Business Unit list.
3. Select the desired response from the response in the response tree.
4. Click **OK**.

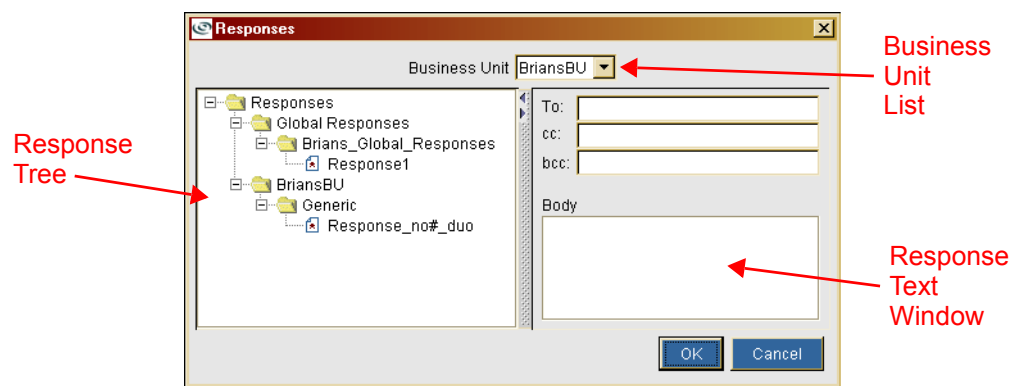


Figure 3-20. Responses Dialog

Printing the Knowledge Base

Once you have created a Knowledge Base, you may find it useful to print out its contents. Printing is handled in a two stage process: first an HTML file is generated, second you print the HTML file. This allows the Knowledge Base to be printed on any platform.

To print a Knowledge Base:

1. Select **Generate HTML for Printing** from the **File** menu.
2. Specify the file name and location in the **Save** dialog.
3. Click **OK**.
4. Open the file in your favorite browser and print.

Index

A

- Accuracy 49
- Action
 - Custom 20
 - Integral 20
 - Types 20
- Actions 19
 - Custom 55
 - Multiple 55
 - Route 53
- Additional NLP processing 23
- Advanced
 - Rules 63
- Advanced Rules 21
 - with Conditions 63
 - with Responses 64
- All 49
- Associating Features with Intents 47
- Attribute 21, 56

C

- Category Overrides 12
- Condition 22
- Conditions 21
- Conventions Used in this Manual 2
- Copy 31
- Correct 49
- Correct Plus 49
- Coverage 49
- Creating
 - Features 44
 - Intent Overrides 38
 - Intents 37, 44
 - Phrases 45
 - Regular Expressions 45
 - Response Actions 52
 - Response Questions 56
 - Responses 51
- Custom Actions 20, 55
- Cut 31

D

- Default
 - Rule 62
- Default Rules 19
- Default value 58
- Delete Item 31

E

- Edit
 - Menu 30
- Editors 32
- Escalate to Converse 53
- Escalate to Converse Actions 55
- Escalate to Email 53
- Escalate to Email Actions 54
- Explanation Text
 - Response Based 19
- Explicit Overrides 12

F

- Feature
 - Hierarchy 9
 - Indicators 44
 - Tree 44
- Feature-Hierarchy Overrides 13
- Features 8
 - Creating 44
 - Editor 32
- FF 58
- Field 22, 58
- Fields 7
- File
 - Menu 30
- Filter Results Table 50
- Filtering and Unit Testing 49

G

- Global KB Rules 25

H

- Help

- Menu 31
- How
 - This Guide is Organized 1

I

- Implicit Overrides 12
- Integral Actions 20
- Integrator
 - Editor 32, 58
- Intent
 - Ignore Threshold 39
 - Override Discrimination 39
 - Overrides 12
 - Overrides Tree 36
 - Scoring and Matching 11
 - Validity Range 39
- Intent-Feature Overrides 13
- Intents 11
 - Advanced Settings 39
 - Creating 37, 44
 - Editor 32, 36, 44
 - Tree 36

K

- KBs
 - Editor 33
- Knowledge
 - Manager 29

L

- Link Text 53

M

- Menu
 - Bar 30
 - Edit 30
 - File 30
 - Help 31
- Morphology 23
- Multiple Actions 55

N

- Normalization 23
- Not Covered Requests 43

O

- Open 31
- Overrides
 - Category 12
 - Explicit 12
 - Feature-Hierarchy 13
 - Implicit 12
 - Intent 12
 - Intent-Feature 13

P

- Partial 49
- Paste 31
- Phrases
 - Creating 45
- Presentation Layer 27
- Printing the KB 72

Q

- Question 21, 56
- Questions 21

R

- Redo 31
- Regular Expressions 9
 - Creating 45
- Related Documentation 1
- Request
 - Sets 7
 - Sources 17
- Request Sets 7
- Request Sources 17
- Request Sources as a Simple Rule Conditions 61
- Requests
 - Editor 32, 40
 - Procedures 41
- Response
 - Explanation Text 19, 52
- Response Questions
 - Creating 56
- Responses 19, 32
 - Creating 51
 - Editor 51
 - Rules
 - Features 64
- Route 53

Route Actions 53

Rules 14

 Advanced 21

 Default 19

 Editor 32, 60, 67, 70

S

Save 31

Save As 31

Setting the Rule Request Source 61

Simple

 Rules 60

Spelling 23

Status Bar 32

T

Test Results Table 49

Testing

 Editor 32, 48

Testing Match Tree 50

Toolbar 31

Tree Expand and Collapse Buttons 31

Type 21, 56, 58

U

Undo 31

URL 53

URL Parameter 53

V

Value 22, 54

W

Work Areas 32

Wrong 49

