Genesys®

eServices 8.1

User's Guide

About Genesys

Genesys is the world's leading provider of customer service and contact center software - with more than 4,000 customers in 80 countries. Drawing on its more than 20 years of customer service innovation and experience, Genesys is uniquely positioned to help companies bring their people, insights and customer channels together to effectively drive today's customer conversation. Genesys software directs more than 100 million interactions every day, maximizing the value of customer engagement and differentiating the experience by driving personalization and multi-channel customer service - and extending customer service across the enterprise to optimize processes and the performance of customer-facing employees. Go to www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Technical Support website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

Although reasonable effort is made to ensure that the information in this document is complete and accurate at the time of release, Genesys Telecommunications Laboratories, Inc., cannot assume responsibility for any existing errors. Changes and/or corrections to the information contained in this document may be incorporated in future versions.

Your Responsibility for Your System’s Security

You are responsible for the security of your system. Product administration to prevent unauthorized use is your responsibility. Your system administrator should read all documents provided with this product to fully understand the features available that reduce your risk of incurring charges for unlicensed use of Genesys products.

Trademarks

The Crystal monospace font is used by permission of Software Renovation Corporation, www.SoftwareRenovation.com.

Technical Support from VARs

If you have purchased support from a value-added reseller (VAR), please contact the VAR for technical support.

Technical Support from Genesys

If you have purchased support directly from Genesys, please contact Genesys Technical Support. Before contacting technical support, please refer to the ***Genesys Care Program Guide f***or complete contact information and procedures.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the ***Genesys Licensing Guide.***

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 81mm_us_10-2012_v8.1.202.00

Table of Contents

Glossary

Supplements

Index

List of Procedures

Preface

Welcome to the eServices 8.1 User's Guide. This guide provides instructions on using eServices 8.1 features and functions.

This guide is valid only for the 8.1.x releases of this product.

Note: For releases of this document created for other releases of this product, please visit the Genesys Technical Support website, or request the Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com.

This preface contains the following sections:

For information about related resources and about the conventions that are used in this document, see the supplementary material starting on page 361.

eServices and the CIM Platform

Genesys eServices (formerly Multimedia) is a cover term for Genesys components that work together to manage interactions whose media is something other than traditional telephonic voice (for example, e-mail or chat).

eServices includes some parts of the Genesys Customer Interaction Management (CIM) Platform, plus certain of the media channels that run on top of the Platform.

CIM Platform

The CIM Platform consists of the following:

- Management Framework
- Reporting (CC Analyzer, CCPulse+)
- Interaction Management, which in turn consists of:
  - Universal Routing
  - Interaction Workflow
  - Knowledge Management
  - Content Analysis
  - Universal Contact History

On top of the CIM Platform are various media channels. Some, such as Genesys Network Voice, handle traditional telephony. Others, such as Genesys E-mail, handle other media.

eServices

eServices, then, consists of the following:

- From the CIM Platform, all of Interaction Management except for Universal Routing:
  - Interaction Workflow—centralized handling of interactions irrespective of media type
  - Knowledge Management—creation and maintenance of standard responses and screening rules
  - Content Analysis—optional enhancement to Knowledge Management, applying natural language processing technology to categorize interactions
  - Universal Contact History—storage of data on contacts and on interactions (linked as threads)

  Universal Routing is not considered part of eServices because it deals with both traditional telephonic interactions and the nontraditional interactions that are handled in eServices.
- From the media channels, at least one of the following:
  - Genesys E-mail
  - Genesys Chat (formerly Genesys Web Media)
  - Genesys SMS (Short Message Service)
  - Genesys MMS (Multimedia Messaging Service)
  - Genesys Web Callback
  - Genesys 3rd Party Media—ability to add customized support for other media (fax, for example)
- Optionally, Web Collaboration—the ability for agents and customers to co-browse (simultaneously navigate) shared web pages. This is an option that you can add to either Genesys Chat or Inbound Voice.

See Figure 1.

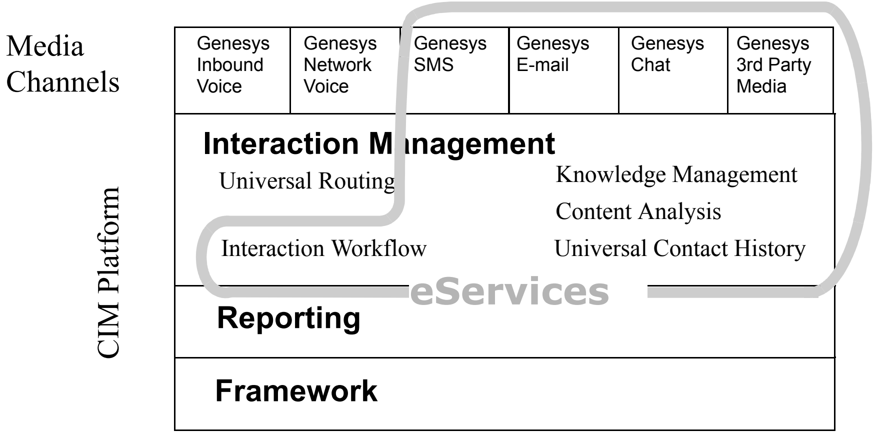


Figure 1: eServices in Relation to the CIM Platform and Media Channels

Note: Although Universal Routing is not considered part of eServices, any functioning solution (platform plus channels) that includes any part of the Interaction Management sector requires Universal Routing.

Licensing

Licensing requirements are:

- For each agent: one eServices Agent seat.
- For each media option: one media channel (E-mail and/or Web Media and/or SMS and/or custom media).
- For Genesys Content Analyzer: NLP Content Analysis license.

See also the *Genesys Licensing Guide.*

Reporting

Reporting templates are available for eServices. For details see the *Reporting Technical Reference Guide for the Genesys 7.x Release.*

Intended Audience

This guide, primarily intended for all users involved in setting up Genesys eServices, assumes that you have a basic understanding of:

- E-mail and web technology.
- Network design and operation.
- Your own network configurations.

You should also be familiar with:

- Genesys Framework architecture and functions.
- Computer-telephony integration (CTI) concepts, processes, terminology, and applications.

Making Comments on This Document

If you especially like or dislike anything about this document, please feel free to e-mail your comments to `Techpubs.webadmin@genesyslab.com`.

You can comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this document. Please limit your comments to the information in this document only and to the way in which the information is presented. Speak to Genesys Technical Support if you have suggestions about the product itself.

When you send us comments, you grant Genesys a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Contacting Genesys Technical Support

If you have purchased support directly from Genesys, please contact Genesys Technical Support.

Before contacting technical support, please refer to the *Genesys Care Program Guide* for complete contact information and procedures.

Document Change History

This section lists topics that are new or that have changed significantly since the first release of this document.

New in Document Version 8.1.202.00

- In Chapter 7, “Searching the UCS Database,” on page 237, the following section has been added:
  - “Using Full Text Search in a Primary/Backup Environment” on page 245

New in Document Version 8.1.201.00

- In Chapter 9, “Capture Points Functionality in Interaction Server,” on page 265, the following sections have been added:
  - “Configuring JMS Capture Point to use SSL” on page 293
  - “Web Service Capture Point” on page 308
- In Chapter 5, several modifications were made to “Required Queries” on page 185.

Genesys

Chapter

1 Genesys Knowledge Management: Basics

This chapter describes the use of the basic Knowledge Management functionalities. It covers these topics:

Use this chapter in conjunction with *eServices 8.1 Knowledge Manager Help.*

The use of Genesys Content Analyzer, which is an optional enhancement to Knowledge Management, is described in Chapter 2, “Genesys Knowledge Management: Content Analyzer,” on page 89.

Overview

This section provides an overview of Knowledge Management: its components, its functionalities, and how they are displayed in the user interface.

Functionality

Knowledge Management functionalities fall into the following four groups:

- **Categories/standard responses/field codes.** A system of categories, organized in a tree structure, provides the means of organizing *standard responses,* which are prewritten responses to interactions. Field codes provide a way to particularize the standard response to individual interactions. Category trees are also integral to the classification functionality of Genesys Content Analyzer (see the third item in this list). You use Knowledge Manager to create category trees, and to create and edit the standard responses and the field codes that they can contain.
- **Screening rules.** Screening rules perform pattern matching on incoming interactions. The results of the pattern matching are then available for use in subsequent steps in routing and in interaction workflows. You use Knowledge Manager to create and edit the screening rules.
- **Genesys Content Analyzer.** This optional functionality uses natural language processing to analyze incoming interactions and assign them to categories in a category tree. Content analysis uses *models,* which are statistical representations of category trees. Models are produced by *training* on a collection of precategorized text objects (e-mails and other types). Knowledge Manager controls the training process and displays information about models. For details see Chapter 2, “Genesys Knowledge Management: Content Analyzer,” on page 89.
- **FAQ.** With Content Analyzer, you can convert your category structure and standard responses into an FAQ list. You can either post the resulting FAQ list as text on your web site or use it as the source for an automatic question-answering facility.

Note: Many of the terms and concepts relevant to Genesys Knowledge Management are also defined in the glossary, which begins on page 353.

Components

Knowledge Management consists of the following components:

- **Classification Server** applies screening rules when triggered to do so by a routing strategy. In Genesys Content Analyzer, Classification Server also applies models to categorize incoming interactions. Both screening rules and models are stored in the Universal Contact Server database.
- **Training Server** trains the system to recognize categories. It is active only in the Genesys Content Analyzer.
- **Knowledge Manager** is the user interface.

Figure 2 shows the Knowledge Management components.

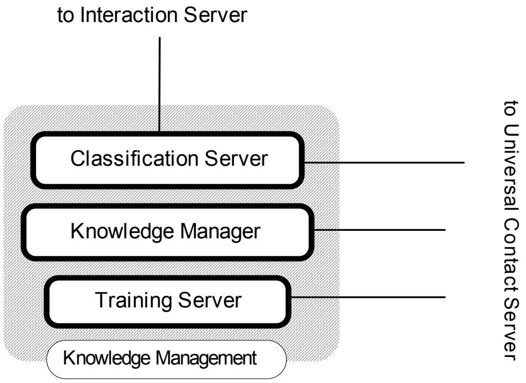
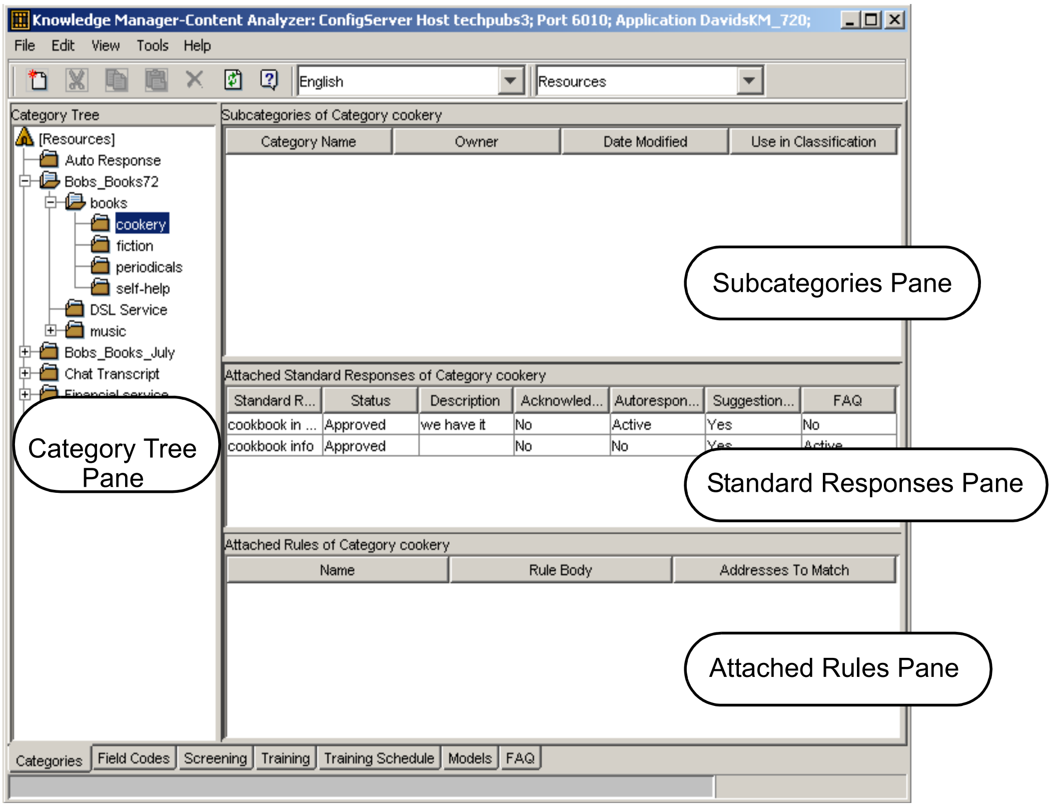


Figure 2: Knowledge Management and Other Relevant Components

User Interface

The Knowledge Manager user interface is a window with three (basic Knowledge Manager) or seven (Genesys Content Analyzer) tabs. Figure 3 shows the interface with seven tabs. It also shows the four panes Category Tree, Subcategories, Standard Responses, and Attached Rules.



Subcategories of Category cookery

Category Name	Owner	Date Modified	Use in Classification

Attached Standard Responses of Category cookery

Standard R...	Status	Description	Acknowled...	Autorespon...	Suggestion...	FAQ
cookbook in ...	Approved	we have it	No	Active	Yes	No
cookbook info	Approved		No	No	Yes	Active

Attached Rules of Category cookery

Name	Rule Body	Addresses To Match

Figure 3: Knowledge Manager Window

Note: All Knowledge Manager screenshots in this guide show the Genesys Content Analyzer version. This differs from the version without Genesys Content Analyzer, even on tabs that are common to both. For example, without Genesys Content Analyzer, there is no `Tools` menu.

Table 1 briefly describes each tab and what it does. Note that the first three tabs are relevant to all users of Knowledge Manager, whereas the last four tabs are relevant only to users of Genesys Content Analyzer. For details see *eServices 8.1 Knowledge Manager Help*.

Table 1: Knowledge Manager Tabs

Relevance	Tab	Description
General	Categories	Displays and gives access to category trees and the standard responses and screening rules associated with categories. Provides access to the `Standard Response Editor`.
	Field Codes	Displays the set of field codes and provides access to the `Field Code Editor`
	Screening	Displays the set of screening rules and provides access to the `Screening Rules Editor`
Genesys Content Analyzer only	Training	Displays the set of training objects and provides access to the `Mail Editor`
	Training Schedule	Displays the start time, status, and other information about training objects
	Models	Displays information about models
	FAQ	Displays information about FAQ objects

Note: In most Knowledge Manager windows, lists of items can be sorted by clicking on the column headings. The major exceptions are lists of test results on the `Models` tab (see “Testing Models” on page 125 and “Using and Rating Models” on page 134).

Introduction to Category Structure

In general terms, a *category* is a unit of knowledge. Categories are organized in a tree structure; Figure 4 shows an example.

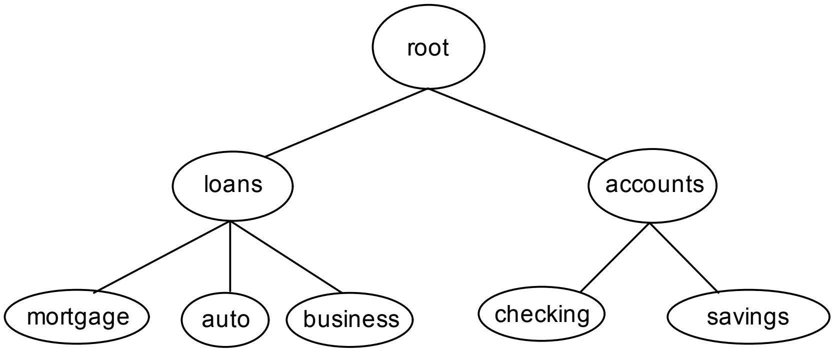


Figure 4: Example Category Tree

Genesys eServices uses category trees in the following ways:

- **Standard Responses.** The category tree is the means of organizing and providing access to the library of standard responses. Each standard response must be associated with one category. One category can have zero or many standard responses associated with it.
- **Classification and Routing.** Genesys Content Analyzer can classify an incoming e-mail in terms of the category tree. You can use the resulting classification in three ways:
  - To supply a standard response as an acknowledgment or an automatic response to an e-mail.
  - To suggest a standard response to the agent handling the e-mail.
  - To route the e-mail.

Categories with no associated standard responses may be of use in grouping other categories together.

Note these definitions:

- A *terminal* category is one that has no subcategories: a leaf on the category tree.
- A *nonterminal* category is one that has subcategories.
- *Child* is another term for subcategory. For example, in Figure 4, `savings` is a child of `accounts`, and `accounts` has the two children `checking` and `savings`.

A category tree is specific to a tenant and a language. Each tenant/language pair can have multiple category trees. Select tenant and language using the drop-down lists near the top of the Knowledge Manager window.

Without Content Analyzer, language is simply a label. You can design different sets of screening rules (for example) for different languages within a single tenant. But the screening rules operate the same way regardless of which language they are grouped under. With Content Analyzer, choice of language affects the way the system operates; for example, when classifying interactions. See "Notes on Language" on page 121 for more information on this topic.

Category membership is inherited. That is, if Category 1 includes Categories 10 and 11, and Category 10 includes Categories 100 and 101, then Category 1 also includes Categories 100 and 101.

Using Basic Knowledge Manager Objects

The basic Knowledge Manager objects are categories, standard responses, and screening rules. Table 2 presents an overall task flow for creating and managing these objects, and for importing and exporting them.

Table 2: Task Flow for Basic Knowledge Manager Objects

Objective	Related Procedures and Actions
1. Create a category tree.	See “Creating a category tree” on page 25.
2. Create and manage standard responses.	1. Open the Standard Response Editor and use the `General` tab to create a name and plain text content for a standard response: page 29. 2. Use the `HTML part` tab to create an HTML version of the standard response: page 33. 3. Use the `Additional` tab to specify possible uses and other attributes of the standard response: page 36. 4. Use the `Attachments` tab to add and manage attachments: page 38. 5. Use the `History` tab for version control: page 39. 6. Search for standard responses: page 41 7. Create standard responses for use with SMS gateways: page 43
3. Create field codes and use them in standard responses.	1. Create field codes: page 49. 2. Insert field codes in standard responses: page 51. 3. Study two examples of the use of field codes: page 56.
4. Create and manage screening rules.	1. Create screening rules: page 59. 2. Test screening rules: page 65 and page 67. 3. Search for screening rules: page 69.
5. Import and export Knowledge Manager objects.	1. Export: page 79. 2. Import: page 81.

Using Categories and Standard Responses

This section describes the creation and editing of categories and standard responses. It also provides some explanation of how they work.

Creating a Category Tree

You create categories on the `Categories` tab of Knowledge Manager.

Procedure:
Creating a category tree

Purpose: To create a category tree for organizing and providing access to standard responses (and to additional functions in Genesys Content Analyzer).

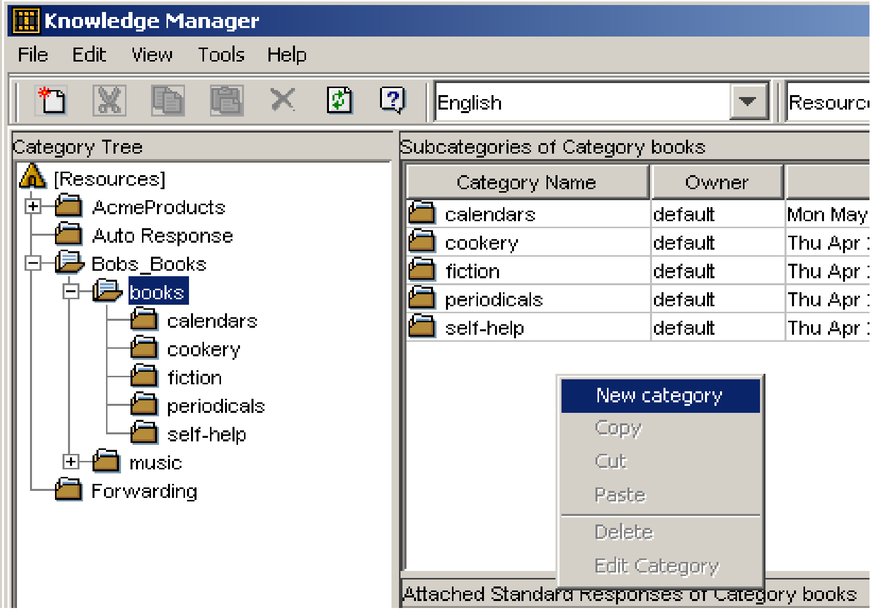
Summary

Create categories on the `Categories` tab of Knowledge Manager.

Start of procedure

1. Decide whether you want a new root category or a subcategory of an existing category:
   a. For a root category, select the tenant, which is the topmost node in the category. Be sure that you also select the correct language.
   b. For a subcategory of an existing category, select the existing category on the Category Tree pane.
2. Select `New category` from the `File` menu (or right-click in the Category Tree pane or the Subcategories pane).

See Figure 5 on page 26, where the new category being created will be a subcategory of `books`.



Category Name	Owner	
calendars	default	Mon May
cookery	default	Thu Apr
fiction	default	Thu Apr
periodicals	default	Thu Apr
self-help	default	Thu Apr

Figure 5: Creating a New Category

Note: Names of categories, like those of all Knowledge Manager objects, can consist only of alphanumeric characters (A–Z, a–z, 0–9), plus the characters shown in Table 3.

Table 3: Characters Allowed in Object Names

Name	Character
Hyphen	-
Exclamation point	!
Number sign, pound	#
Dollar sign	$
Caret	^
Asterisk	*
Underscore	_
Curly brackets	{ }
Angle brackets	< >
Period, full stop	.

Names must also be no more than 64 characters long. For a reason to keep name length well below 64 characters, see page 82.

3. Selecting `New category` produces different dialog boxes, depending on whether your new category is a root or a subcategory.
   a. The `New root category` dialog box, shown in Figure 6, requires that you enter a name for the category:

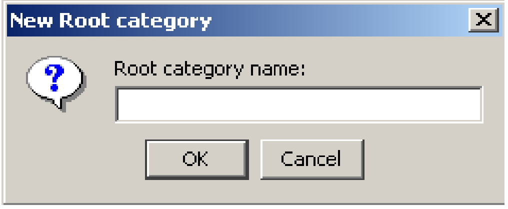


Figure 6: New Root Category Dialog Box

   b. The `New category` dialog box, shown in Figure 7 on page 27, creates a subcategory. Enter a name for the category.

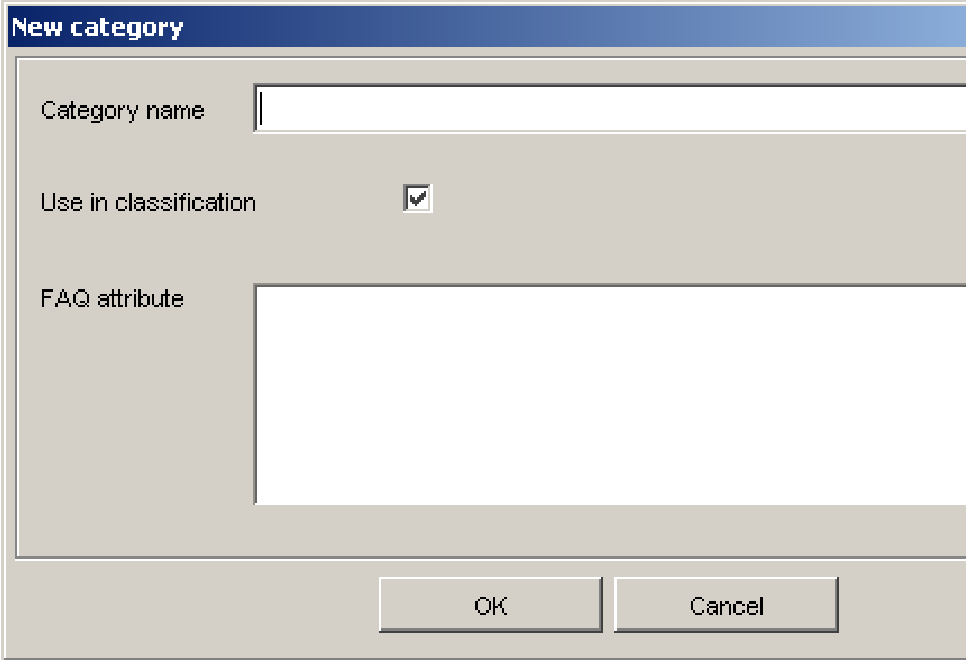


Figure 7: New Category Dialog Box

The `Use in Classification` check box enables you to choose whether Genesys Content Analyzer uses the category in classification. You may wish to have some categories that are used only for organizing other categories or standard responses, not for classification.

In the `FAQ attribute` box you can enter the text of a question to which a standard response associated with this category can serve as the answer. See "FAQ Objects" on page 150.

You can also edit an existing nonroot category by right-clicking it and selecting `Edit category`.

End of procedure

Next Steps

- Read further information about "Copying, Pasting, and Deleting" categories and category trees (this page), and about "Considerations in Designing a Category Tree" (page 28).
- Create standard responses (page 28).

Copying, Pasting, and Deleting

You can copy, paste, edit, and delete categories and category trees and their associated standard responses. Any of these operations on a category also applies to all of its subcategories.

If a category tree is used by a training object, you cannot delete the entire tree unless you first delete the training object. You can, however, delete any nonroot category from the tree (you cannot delete the root category). Training objects are described in "Training" on page 90.

Considerations in Designing a Category Tree

Without Genesys Content Analyzer, the sole function of categories is to organize the library of standard responses. With Genesys Content Analyzer, additional considerations come into play, as described in "Design and Use Considerations" on page 144.

Creating Standard Responses

You create standard responses on the `Categories` tab (see Figure 3 on page 21). If you want to use field codes in a standard response you must create them on the `Field Codes` tab, as described in "Field Codes" on page 45.

You can create two versions of a standard response, one in plain text format and one in HTML format (see "Filling out the HTML part tab" on page 33). When E-mail Server uses the standard response to create an e-mail (for example, when generating an acknowledgment), it creates a multipart e-mail that includes both plain text and HTML versions. Then the settings of the e-mail client that receives the e-mail determine which version displays.

You should be aware that e-mail clients may display multipart e-mails in varying ways. For example, if Microsoft Outlook has AutoPreview turned on, the preview may show the plain text version whereas the full display shows the HTML version. For this reason you should be careful that the plain text and HTML versions have identical content.

Note: To create and edit an HTML version of a standard response, you must have Internet Explorer 5.5 or later on the same host as Knowledge Manager.

Procedure: Creating or editing a standard response

Purpose: To create prewritten content that can be used in responding to interactions.

Prerequisites

- You must first create a category tree.
- If you want to use field codes in your standard responses, you must first create the field codes (page 49).

Start of procedure

1. On the Category Tree pane, select the category that you want to add a standard response to.
2. On the Standard Responses pane, right-click and select `New Standard Response`.The Standard Response Editor displays, as shown in Figure 8.

   Note: The title of the Standard Response Editor dialog box is `New Standard Response` if you are creating a standard response and `Edit Standard Response` if you are editing an existing standard response. The contents are otherwise identical.

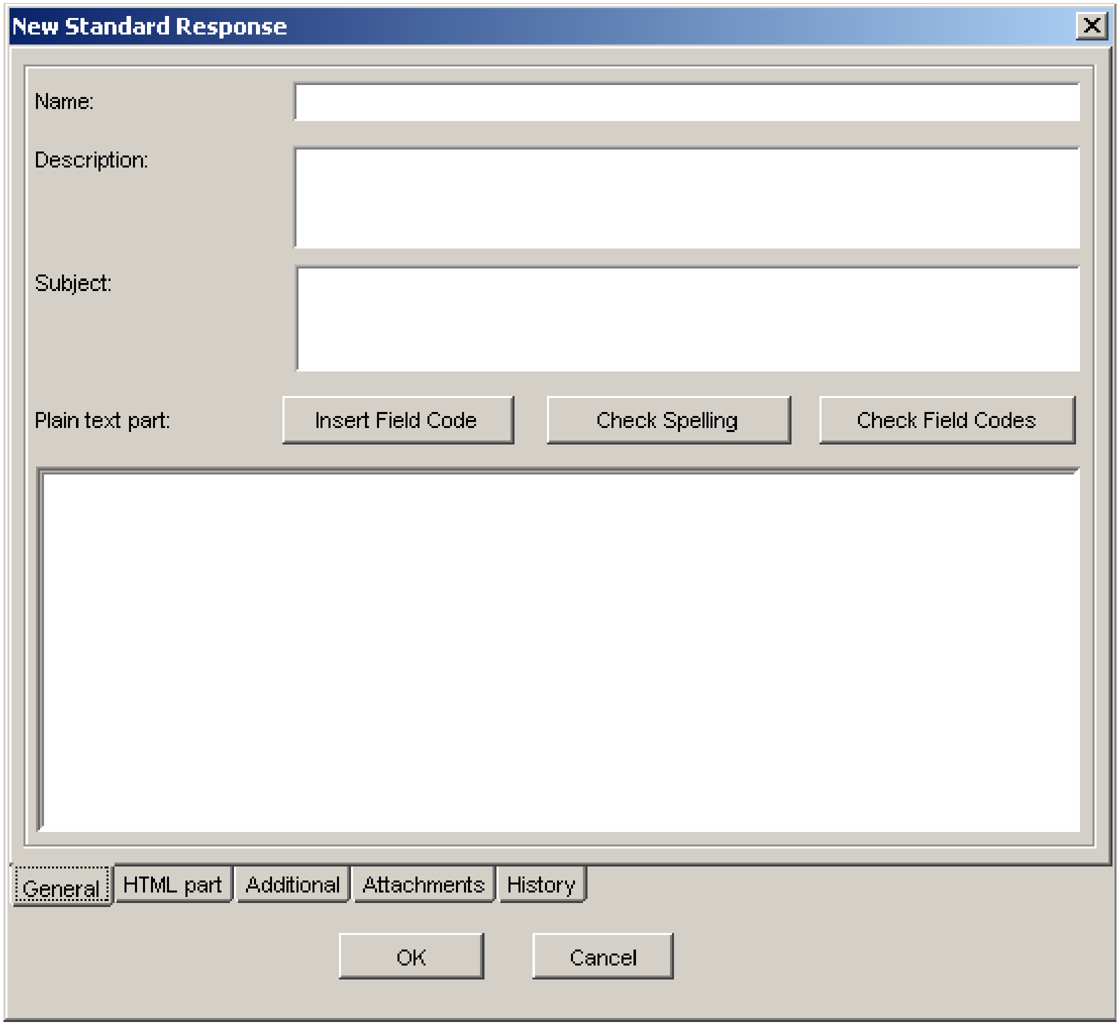


Figure 8: New Standard Response

The display opens on the `General` tab.

3. Enter a name (required) and description (optional) for the standard response. Remember that the name can use only alphanumeric characters (A–Z, a–z, 0–9), hyphen, underscore, and space.

4. Enter a subject for the standard response.

   What you enter here appears as the Subject line in an e-mail (such as acknowledgement or autoresponse) generated from this standard response.

   You can also specify a Subject line in any IRD strategy object that has a Format tab (see the *Universal Routing 8.1 Reference Manual*). If you do, this overrides any Subject line that is specified for the standard response in Knowledge Manager.

5. Enter text for the plain text version of the standard response, using the `Insert Field Code` button to insert field codes (see "Field Codes" on page 45). If you have not yet created field codes, you can continue creating the standard response, then return to it and insert field codes after you have created them.
6. Click `Check Field Codes` to show the result of applying (rendering) the field codes in this standard response. The values rendered are taken from a collection of generic properties that exists for this purpose.
7. Click `Check Spelling` to check spelling for the language selected in the adjacent drop-down list. See "Language and Dictionary Names" below for more information on this.

End of procedure

Next Steps

- Read further information about the names of languages and dictionaries (this page).
- Go on to the `HTML part` tab (page 33).

Language and Dictionary Names

For each language with spelling checker support, Knowledge Manager maintains a number of files in the `\lex` subdirectory of the Knowledge Manager installation directory. This section explains several features of the following file types:

- `<language_name>_spllchk.pro` file—Properties file, contains settings for options such as case sensitivity. Also points to the dictionary and user dictionary files.
- `userdic_<language_name>_spllchk.tlx` file—User dictionary file, contains words that you add by selecting `Add` in the `Spelling` dialog box.

For a language to display in the `Check Spelling` drop-down list, it must have both of the following:

- `<language_name>_spllchk.pro` file
- A Business Attribute, of type `Language`, whose Attribute Value name is `<language_name>`, exactly matching the name of the `.pro` file.

For example, the file name `BrazilianPortuguese_spllchk.pro` shows that the name of the Business Attribute for this language must be `BrazilianPortuguese`, not `Brazilian Portuguese` or `BrazPort` or anything else.

Renaming a Language

It is not possible to rename an existing language Business Attribute. You can, however, alter the language name that displays in the `Check Spelling` drop-down.

For example, if you build a German-language category tree, you may want the name of the language to appear as the German word *Deutsch* rather than the English word *German.* So you can create a Business Attribute called `Deutsch` and select it when creating your category tree. But when it comes to checking the spelling, Genesys Knowledge Management uses the name *German,* as shown by the filename `German_spllchk.pro`. With the `.pro` file and the Business Attribute having nonmatching names, German will not be available on the drop-down list. You can rectify this situation by renaming the dictionary file:

1. Locate the file `German_spllchk.pro` and rename it `Deutsch_spllchk.pro.`
2. Refresh the view in Knowledge Manager (select `Refresh` from the `View` menu or click the refresh icon).

Saving a User Dictionary

If you have customized your spelling checking by adding items to a user dictionary, you will want to avoid overwriting your user dictionary when installing a new version of Knowledge Manager (or reinstalling the existing one).

Procedure: Saving an existing user dictionary

Start of procedure

1. Make a copy of the existing `userdic.tlx` file, located in the existing Knowledge Manager installation directory.

> **Note:** Prior to release 7.6.1, Knowledge Manager had spell checking for English only, so there was only one user dictionary file, named `userdic.tlx.` This section describes saving the contents of this English user dictionary.

2. Install or reinstall Knowledge Manager.
3. Locate the new `userdic_English_spllchk.tlx` (in the new Knowledge Manager installation directory) and replace its contents with the contents of `userdic.tlx.`

End of procedure

Procedure:
Filling out the HTML part tab

Prerequisites

For the HTML version of a standard response, you must first create the standard response as a standalone HTML file, using whatever authoring tool or text editor you prefer.

Start of procedure

1. Start on the `HTML part` tab, as shown in Figure 9.

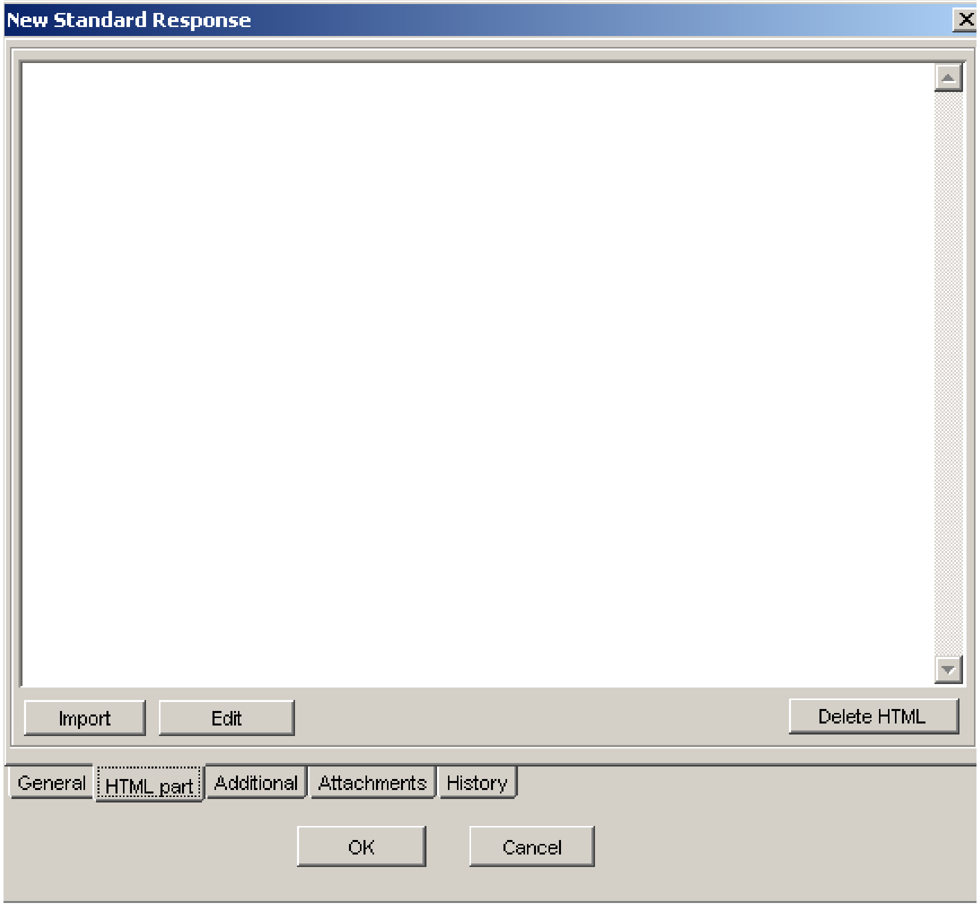
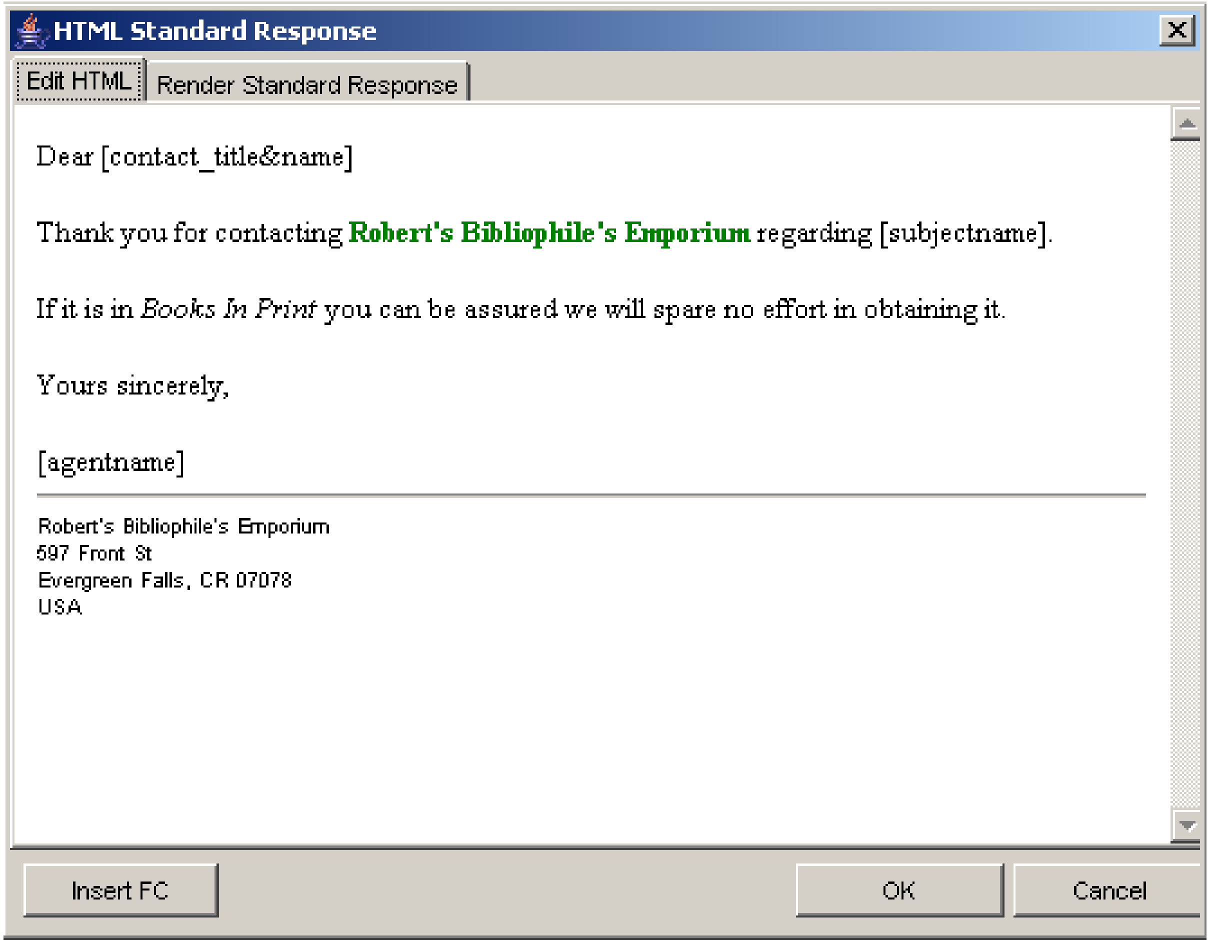


Figure 9: HTML Part Tab

2. On the `HTML part` tab, click `Import` and browse to the HTML file that you want to import, then click OK. The `HTML Standard Response` dialog box appears, as shown in Figure 10.



Dear [contact_title&name]

Thank you for contacting **Robert's Bibliophile's Emporium** regarding [subjectname].

If it is in *Books In Print* you can be assured we will spare no effort in obtaining it.

Yours sincerely,

[agentname]

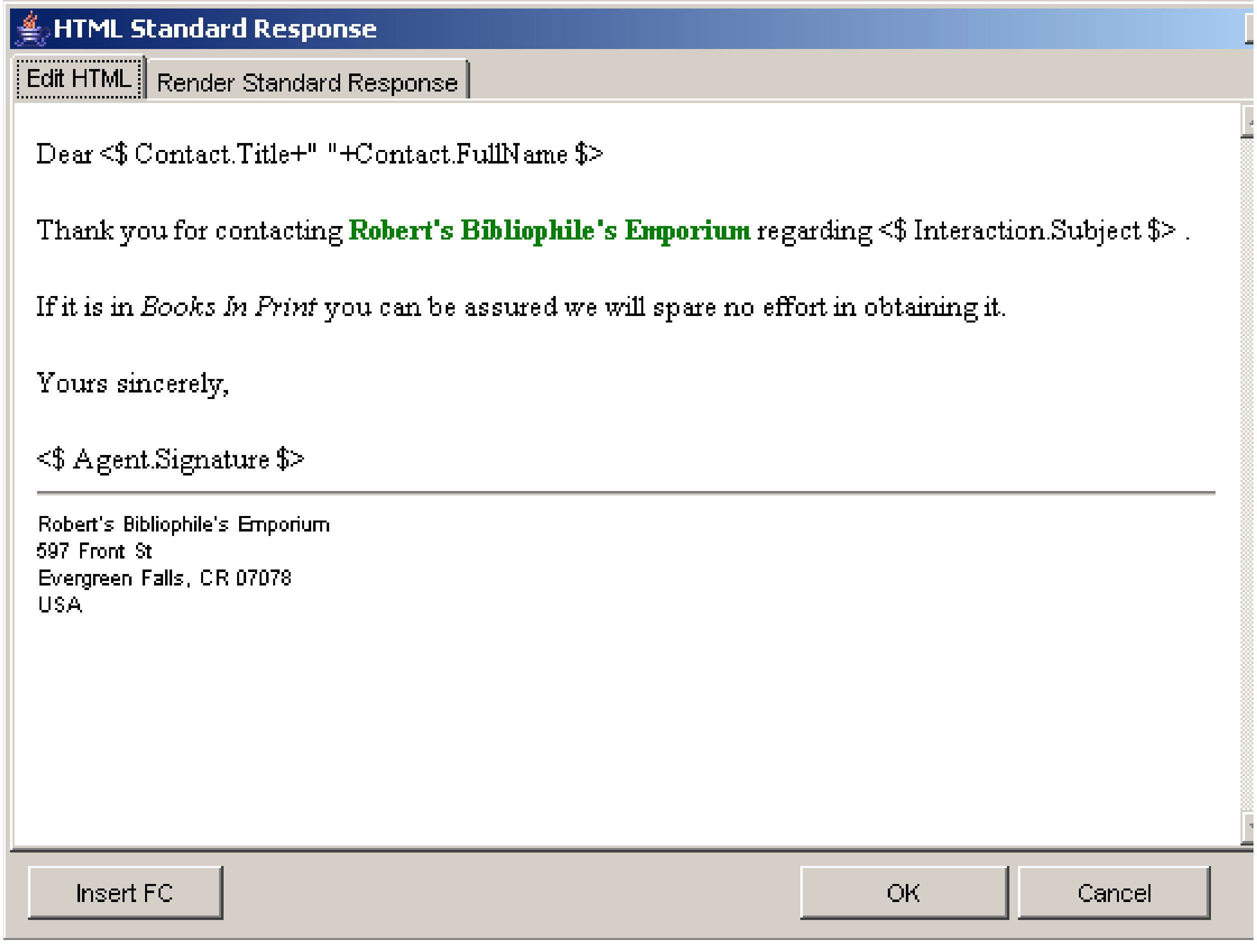
Robert's Bibliophile's Emporium
597 Front St
Evergreen Falls, CR 07078
USA

Figure 10: HTML Standard Response

Note: In the HTML version of a standard response, links to resources that are used in the content, such as image files, must use absolute URLs; embedded graphics or relative links are not supported. Also, these resources must be available on the web through an HTTP server at the time that the standard response containing this HTML is sent to the customer.

This dialog box opens with its `Edit HTML` tab. On this tab you can do the following:

- Insert, delete, copy, and paste text. Pasted text retains its attributes. Inserted text takes on the attributes of its insertion point.
- Insert field codes. Note that the imported file in Figure 10 includes bracketed expressions like [subjectname] to indicate where field codes should be inserted. Using the `Insert FC` button to insert field codes produces the result shown in Figure 11.



Dear <$ Contact.Title+" "+Contact.FullName $>

Thank you for contacting **Robert's Bibliophile's Emporium** regarding <$ Interaction.Subject $> .

If it is in *Books In Print* you can be assured we will spare no effort in obtaining it.

Yours sincerely,

<$ Agent.Signature $>

Robert's Bibliophile's Emporium
597 Front St
Evergreen Falls, CR 07078
USA

Figure 11: HTML Standard Response with Field Codes Inserted

Note: Be careful that the content of the HTML version, including field codes, matches the plain text version that you created on the `General` tab. See the explanation of the relation between plain text and HTML versions on page 28.

Go to the `Render Standard Responses` tab to see the standard response with the field codes rendered in the same way as when you click `Check Field Codes` on the `General` tab, as described in "Inserting field codes into a standard response" on page 51.

End of procedure

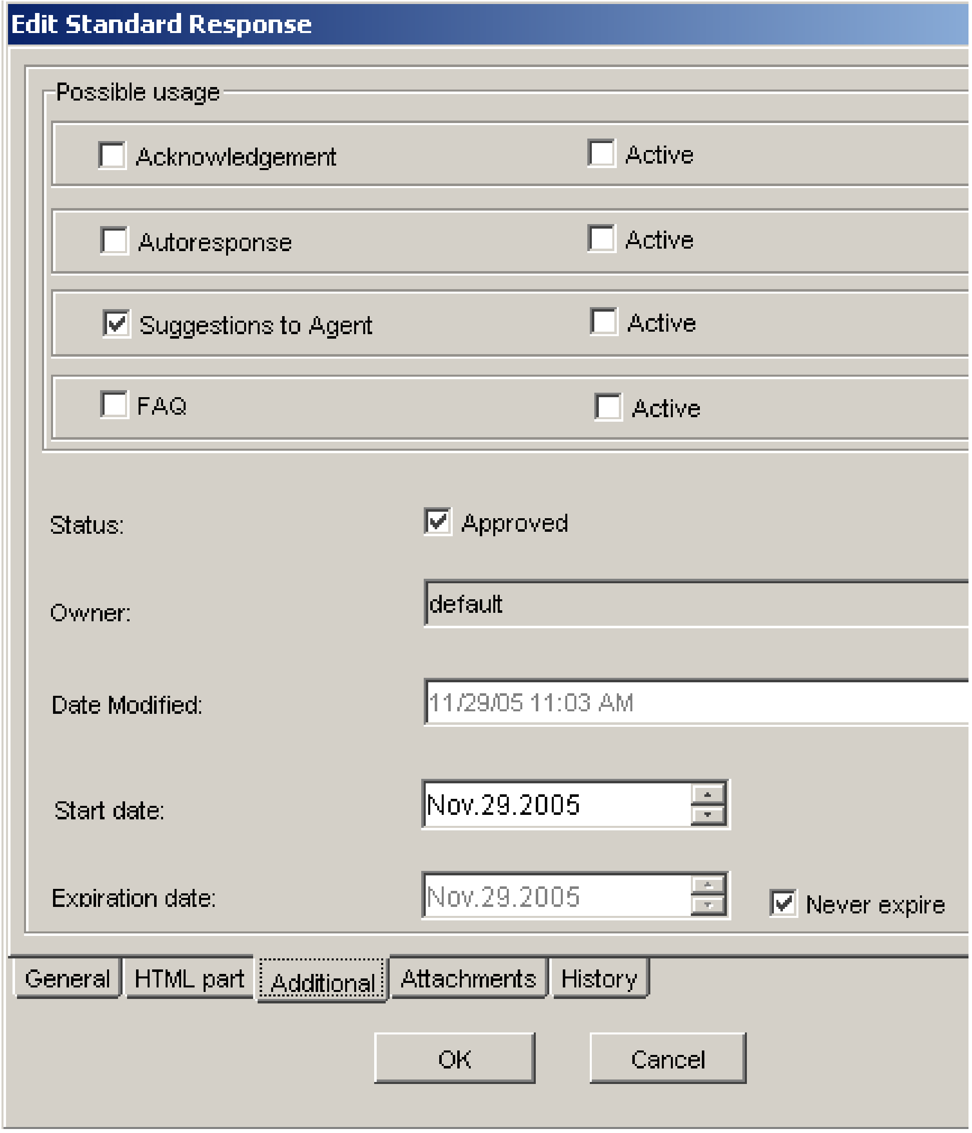
Next Steps

- Go on to the `Additional` tab (next section).

Procedure:
Filling out the Additional tab

Start of procedure

1. You must specify the possible uses of the standard response by using the check boxes on the `Additional` tab, shown in Figure 12 on page 36.



Edit Standard Response

Possible usage

□ Acknowledgement □ Active

□ Autoresponse □ Active

☑ Suggestions to Agent □ Active

□ FAQ □ Active

Status: ☑ Approved

Owner: default

Date Modified: 11/29/05 11:03 AM

Start date: Nov.29.2005

Expiration date: Nov.29.2005 ☑ Never expire

Figure 12: Standard Response, Additional Tab

The possible uses are:

a. `Acknowledgment`—The standard response may be sent to acknowledge receipt of an incoming interaction.

b. `Autoresponse`—The standard response may be used as an automatic response to an incoming interaction.

c. Suggestions to Agent—The standard response may be offered to agents as suggested wording to use in their own replies to interactions.

d. FAQ—The standard response may supply the answer for an item in an FAQ object. This use type applies only with Genesys Content Analyzer.

2. For each usage type, you must also specify this standard response as Active or not. Each category may have multiple standard responses of each usage type, but only one standard response of a given usage type can be Active. For more information on this point see "The Meaning of Active" below.

3. Specify the status using the Approved check box. Only Approved standard responses can appear in Routing objects.

4. Use the other check boxes and fields on this tab to specify the owner, date modified, start date, and expiration date if any.

   If a standard response's expiration date has been reached, it has the following effects:

   - The standard response is not shown in IRD, so it cannot be used in a new or modified strategy.
   - If this standard response was saved in a strategy before the expiration date was reached, E-mail Server does not send the standard response, but returns an error message.

End of procedure

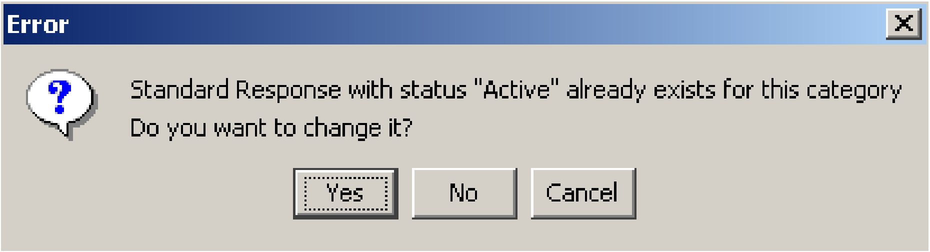
Next Steps

- Read further information about the meaning of *Active,* on this page.
- If you want to add an attachment to the standard response, go on to the Attachments tab (page 38)
- If you want to maintain multiple versions of the standard response, go on the History tab (page 39).
- Learn how to search for standard responses (page 41).
- Create field codes to use in your standard responses, as described in "Field Codes" on page 45.

The Meaning of Active

- Purpose of Active—There are times when the system must select, without immediate user input, a standard response of a given usage type for a given category. If there are multiple standard responses of a single usage type for the category in question, the system selects the one that is designated as Active.

- Changing the Active standard response—If you attempt to select Active for a Standard Response (either a new one or an existing one), and there is already an Active Standard Response with that usage type for that category, Knowledge Manager offers to take the previously-Active Standard Response out of Active status, displaying the message shown in Figure 13.

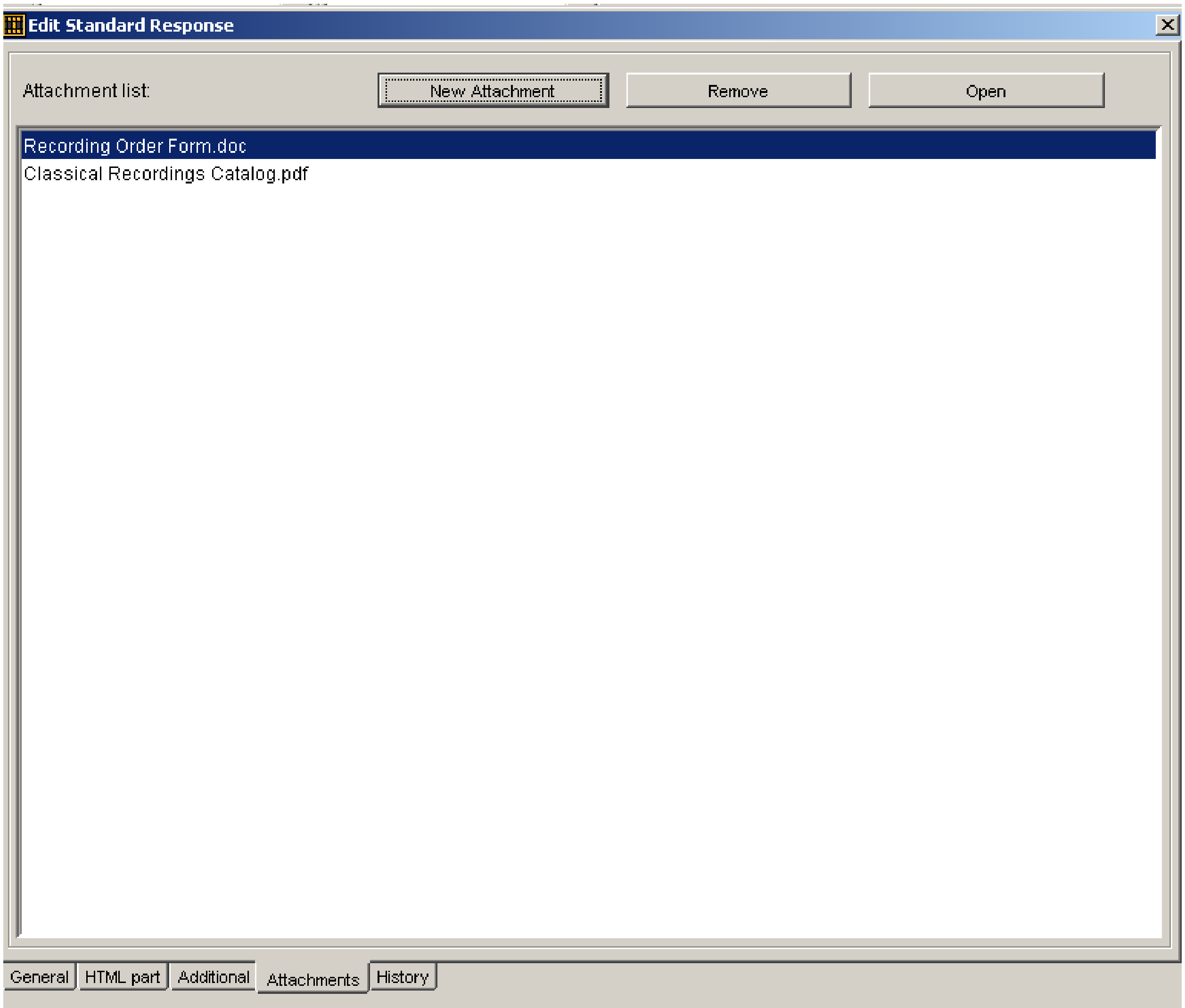


Standard Response with status "Active" already exists for this category
Do you want to change it?

Figure 13: Changing the Active Standard Response

The Attachments Tab

The Attachments tab displays a list of attachments to the standard response, as shown in Figure 14.



Recording Order Form.doc
Classical Recordings Catalog.pdf

Figure 14: The Attachments Tab

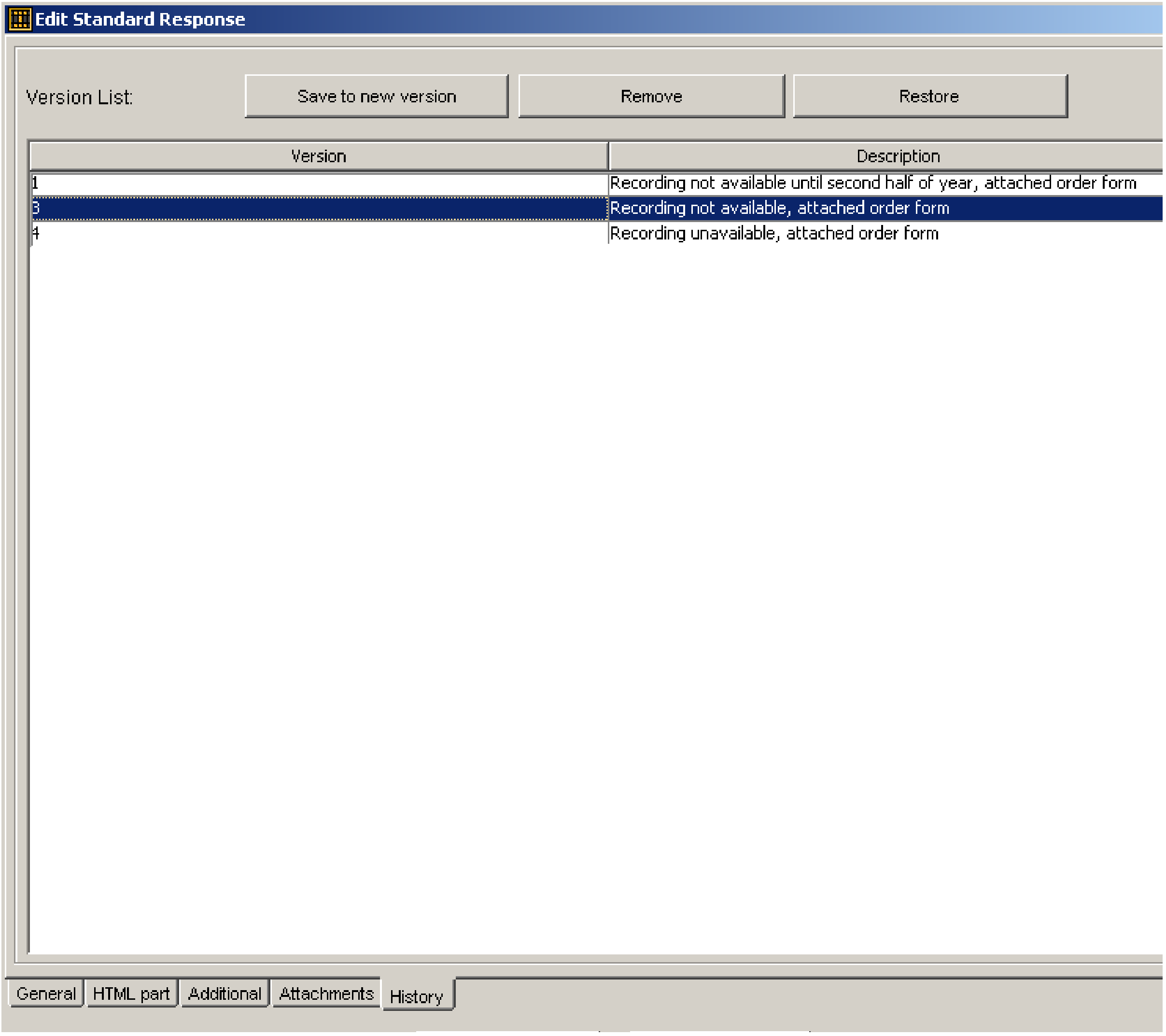
On this tab you can do the following:

- To add an attachment, click `New Attachment.` Browse to the file that you want to attach, then select it.
- To remove an attachment, select it, then click `Remove.`
- To view an attachment, select it, then click `Open` (you cannot double-click on the attachment name to view it).

Note: This opens the attachment for viewing only. You cannot save any changes that you make to the attachment.

The History Tab

Use the `History` tab, shown in Figure 15, for version control.



Version	Description
1	Recording not available until second half of year, attached order form
3	Recording not available, attached order form
4	Recording unavailable, attached order form

Figure 15: The History Tab

On this tab you can do the following:

- To save multiple versions of the standard response, click `Save to new version`.
- To choose the version that appears on the Standard Response pane, select the version and click `Restore`.

Restoring a version restores only the Text and Description parts of the standard response. For example:

1. Version 1 of a standard response has an expiration date of December 10, 2010.
2. Create a Version 2 with `Never expire` selected (no expiration date).
3. Restore Version 1.

4. The restored version has `Never expire` selected. The former expiration date of December 10, 2010 is not restored.

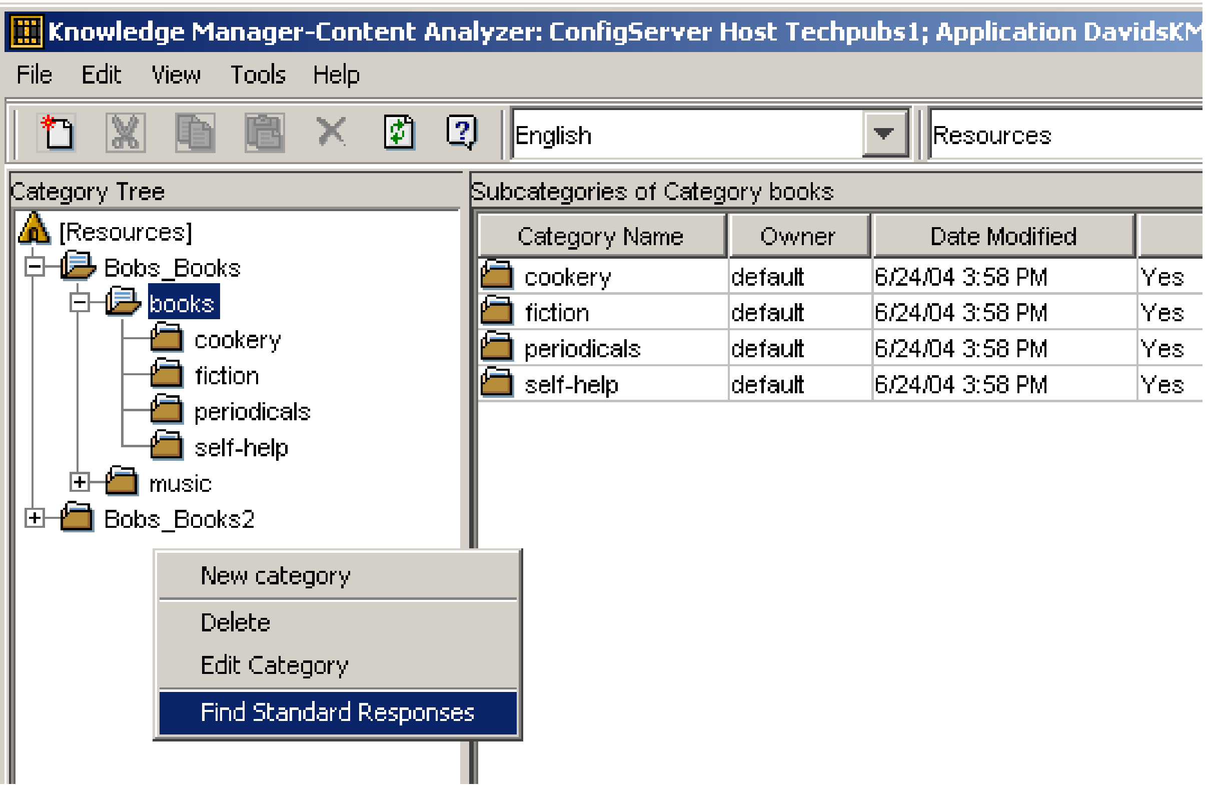
Searching for Standard Responses

You can search for the standard responses that are associated with a selected category and its subcategories.

Procedure: Searching for a standard response

Start of procedure

1. Select a category on the Category Tree pane.
2. Right-click anywhere in the Category Tree pane and select `Find Standard Responses` from the shortcut menu, as shown in Figure 16.



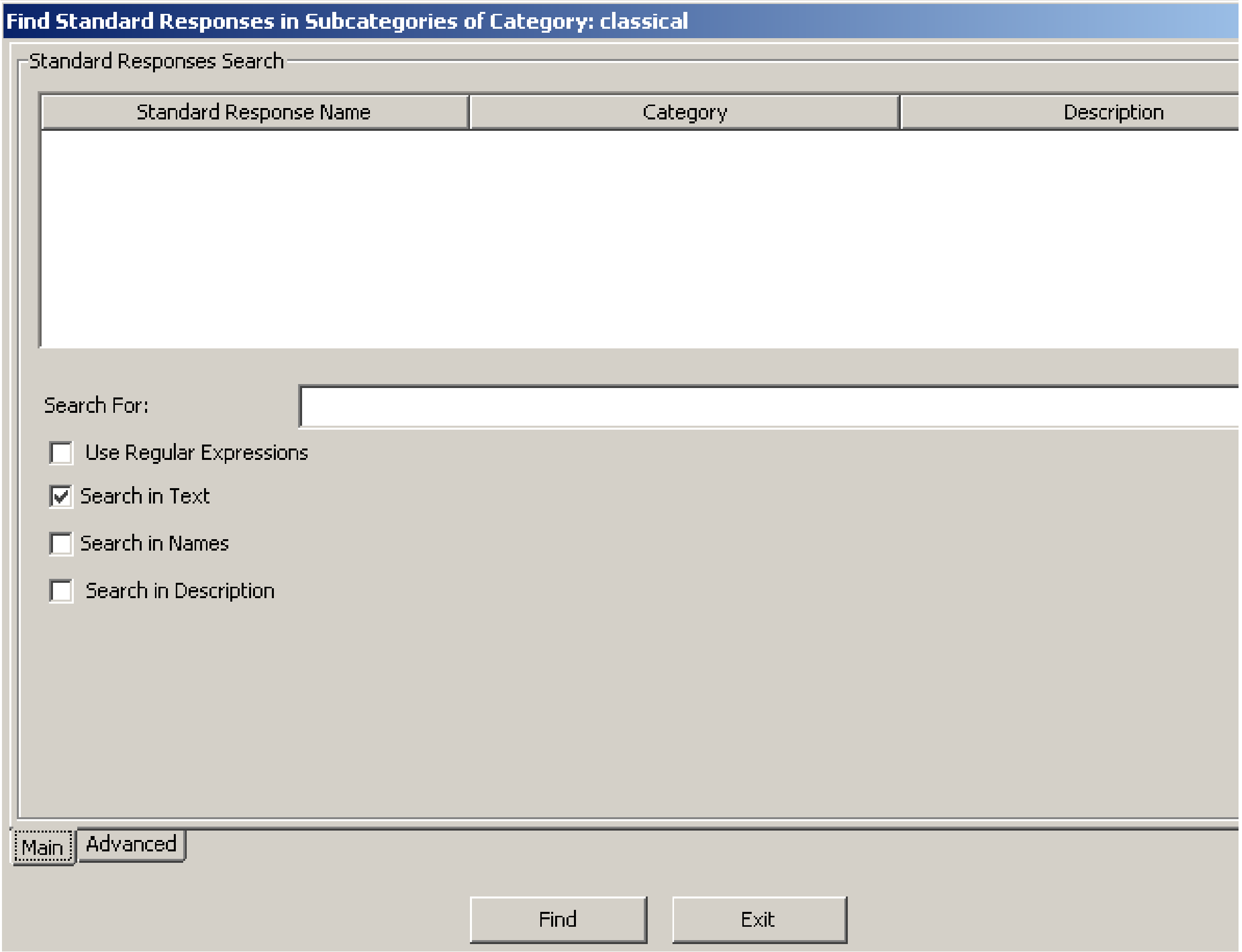
Subcategories of Category books

Category Name	Owner	Date Modified	
cookery	default	6/24/04 3:58 PM	Yes
fiction	default	6/24/04 3:58 PM	Yes
periodicals	default	6/24/04 3:58 PM	Yes
self-help	default	6/24/04 3:58 PM	Yes

Figure 16: Find Standard Responses

The search function searches for standard responses associated with the selected category and all of its subcategories.

3. The `Find Standard Responses in Subcategories of Category: <name>` dialog box appears, as shown in Figure 17


Find Standard Responses in Subcategories of Category: classical
Standard Responses Search
Standard Response Name
Category
Description
Search For:
Main
Advanced
Find
Exit

☐ Use Regular Expressions

☑ Search in Text

☐ Search in Names

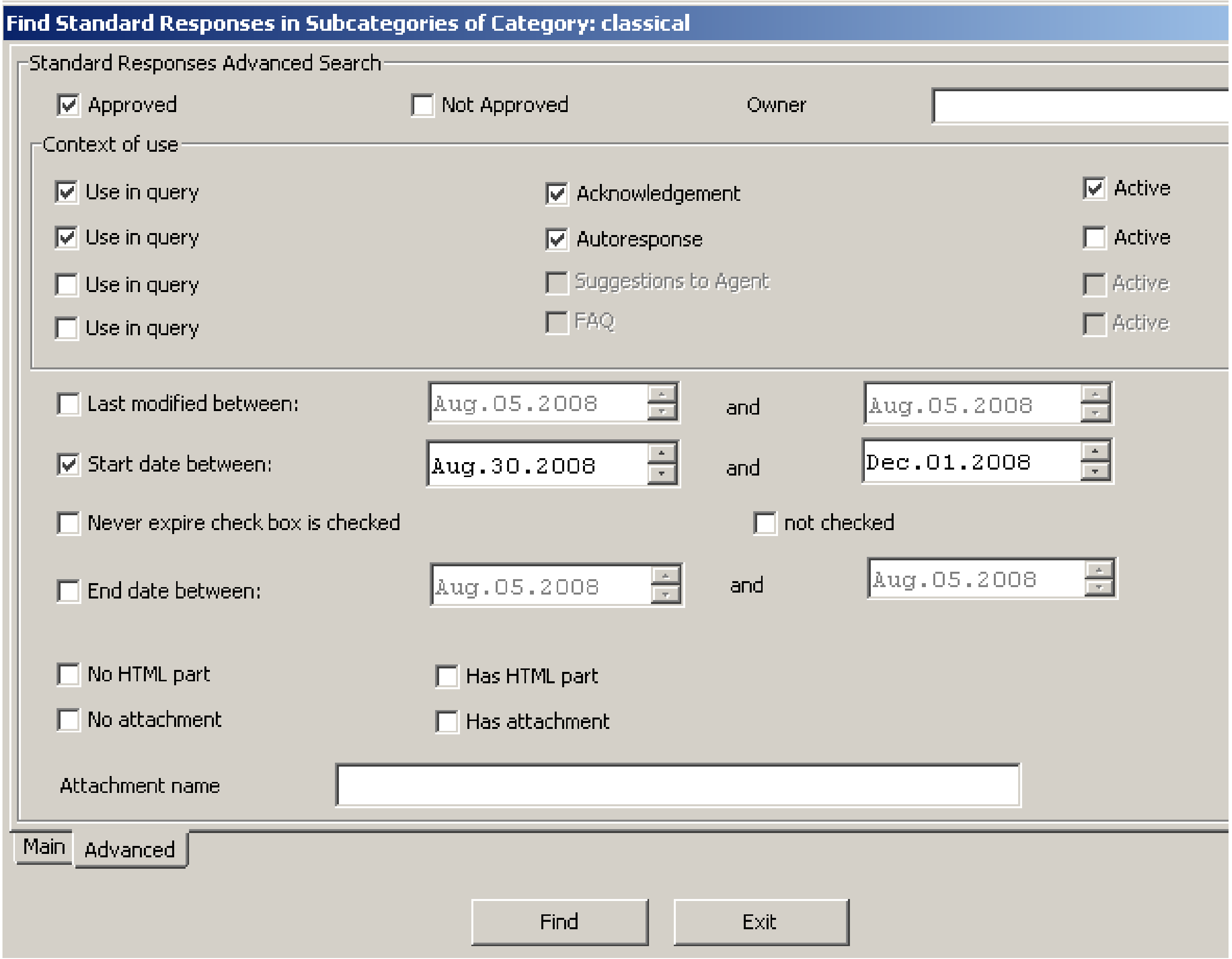
☐ Search in Description

Figure 17: Find Standard Responses Dialog, Main Tab

4. Enter a string to search for in the `Search For` field (if there is nothing in this field, the system reports `Empty search pattern`). Select `Use Regular Expressions` if you want the search to treat the string as a regular expression. For more information see "Regular Expressions" on page 74.
5. Select check boxes to search in the text of the standard response, its name, its description, or any combination. The `Search in Text` box is selected by default (if none of these check boxes is selected, the system reports `No Standard Responses have been found`).

Note: When the `Search in Text` box is selected, Knowledge Manager searches both plain text and HTML versions.

6. Go to the `Advanced` tab, shown in Figure 18, to select further attributes, in any combination.



Find Standard Responses in Subcategories of Category: classical

Standard Responses Advanced Search

☑ Approved ☐ Not Approved Owner

Context of use

☑ Use in query	☑ Acknowledgement	☑ Active
☑ Use in query	☑ Autoresponse	☐ Active
☐ Use in query	☐ Suggestions to Agent	☐ Active
☐ Use in query	☐ FAQ	☐ Active

☐ Last modified between:	Aug.05.2008	and	Aug.05.2008
☑ Start date between:	Aug.30.2008	and	Dec.01.2008
☐ Never expire check box is checked		☐ not checked	
☐ End date between:	Aug.05.2008	and	Aug.05.2008

☐ No HTML part ☐ Has HTML part

☐ No attachment ☐ Has attachment

Attachment name

Figure 18: Find Standard Responses Dialog, Advanced Tab

7. Click `Find`. The dialog box displays the name, category, and description of all standard responses found. Click `Exit` to close the dialog box.

End of procedure

Standard Responses for SMS Gateways

Standard Responses can be used to carry the body of a Short Message Service (SMS) message that E-mail Server sends using an SMS gateway. To accomplish this you must use a routing strategy that includes a CreateSMS object and specifies certain attached data. For details on this strategy configuration see the “Multimedia Objects” section of the “Interaction Routing Designer Objects” chapter of the *Universal Routing 8.1 Reference Manual.*

The form of the special standard response differs according to the requirements of the gateway that you are using. This section provides examples of standard responses that can be used with three available gateways.

Clickatell

For the Clickatell gateway, create a standard response with the following as its body:

```
api_id:1234
user:Name
password:Secret
from: <$AttachedData("OrigSMSNumber")$>
to: <$AttachedData("DestSMSNumber")$>
text: <$AttachedData("SMSText")$>
```

Where:

- `1234` should be replaced with the api_id that you received upon registering for the service.
- `Name` should be replaced with the user name that you created when registering for the service.
- `Secret` should be replaced with the password that you created when registering for the service.
- The user data `"OrigSMSNumber"` contains the number of the originating SMS device.
- The user data `"DestSMSNumber"` contains the number of the recipient SMS device.
- The user data `"SMSText"` contains the text of the SMS to send (limited to 160 characters).

You can add other available parameters to the body. For information about what parameters are available for this gateway, do as follows:

1. Go to `http://www.clickatell.com/`.
2. Select `Developers`, then `SMTP`.

SMS Gateway for Mdaemon and sms2email.com

For the SMS Gateway for Mdaemon or the sms2email.com gateway, create a standard response with the following as its body:

```
<$AttachedData("SMSText")$>
```

Where the user data `"SMSText"` contains the text of the SMS to send (limited to 160 characters).

For more detailed information on Mdaemon:

1. Go to `http://www.achab.com/`.
2. Click `SMS Gateway for MDaemon`, then `Features -Outbound SMS`.

For more detailed information on sms2email.com:

1. Go to `http://www.sms2email.com/`.
2. Click `Developer Info`, then `Email to SMS Gateway`, then `how-to guide`.

Field Codes

Although field codes are used mostly in standard responses, they are the most complex and powerful aspect of standard responses, so they are described in this separate section.

Overview

The main use of field codes is to particularize standard responses, in a manner similar to the Mail Merge feature in word processors.

For example, you can use the field code `<$Contact.FirstName$>` in a response beginning `Dear <$Contact.FirstName$>`, which you send to dozens of recipients. In each message, `<$Contact.FirstName$>` is replaced by the first name of the addressee of the message (the contact) as listed in the Universal Contact Server database.

More generally, a *field code* is a formula that you insert into an outgoing text object, such as an e-mail that E-mail Server generates when triggered to do so by a routing strategy object.

The most common type of such text object is a standard response (triggered by an Autoresponse or Acknowledgement object), but you can also insert field codes into other types, such as chat transcripts, SMS messages, and forwarded or redirected e-mails. In some cases the only place you can insert a field code is in the Subject line using the `Format` tab in a strategy object.

The following is a complete list of the strategy objects that can use field codes either in a standard response or in the Subject line:

- Acknowledgement
- Autoresponse
- ChatTranscript
- Create EmailOut
- Create Notification
- Create SMS
- Forward
- RenderMessageContent

The following is a complete list of the strategy objects that can use field codes only in the Subject line:

- Redirect
- Reply from External Resource

- Send

When a text object containing such a formula is processed, the following happens:

1. The formula performs an operation, which produces a result.
2. The result replaces the field code in the text object.

This process of performing an operation and substituting its result is called *rendering*.

Note: Field codes can be used in outgoing text objects only.

Using UCS Data in Standard Responses: System Variables

In the example just given, `Contact.FirstName` retrieves a piece of data about the interaction. The ability to access interaction data is perhaps the most frequent use of field codes. Although field code formulas can be very complicated, many simply retrieve a single piece of data, such as a contact's name.

You access Universal Contact Server data using predefined variables, called *system variables.* These variables access three predefined objects. Each object has a name and a set of properties. In the example, `Contact` is an object and `FirstName` is one of its properties. The system variable `Contact.FirstName` retrieves the value of the `FirstName` property of the `Contact` object. In similar fashion, there is a system variable for each object+property pair. The objects and properties that you can use in field code formulas are described in the following sections.

Interaction

This object represents the particular interaction being worked on, such as an inbound e-mail. These are its properties:

- `Id`
- `DateCreated`
- `Subject`
- `ToAddress`
- `FromAddress`
- `AttachedData`
- `TimeZone`

Contact

This object represents the contact associated with the interaction being worked on. These are its properties:

- `Id`

- Title
- FirstName
- LastName
- FullName
- PrimaryPhoneNumber
- PrimaryEmailAddress

Agent

This object represents the agent working on the interaction. These are its properties:

- FirstName
- LastName
- FullName
- Signature

Note: Automated responses use the default agent. Create the default agent as a Person object just like any other in Configuration Manager. Then select this Person on the Automated Reply Agent screen of the E-mail Server configuration wizard (or set this Person as the value of the autobot-agent-login-name option in the E-Mail Processing section of the E-mail Server Java application). Since this is the Person who the automated response appears to be from, you may want to name it after your company or institution.

Custom Variables

In addition to the system variables, you can use Knowledge Manager to create custom variables. Custom variables have the following properties:

- Their values are assigned by strategy objects.
- Therefore, standard responses that use field codes containing custom variables must have the usage type Autoresponse or Acknowledgment.

For an example of the use of a custom variable in a standard response, see “Using a custom variable” on page 56. For a complete description of the Routing objects that can use custom variables, see the *Universal Routing 8.1 Reference Manual.*

Note: The names of custom variables must begin with an alphabetic character or underscore, and the remainder of the name must consist only of alphanumeric characters or underscores. This differs from the requirements for the names of other Knowledge Manager objects, which may also contain hyphen and space. For example, `5-usercode` is not an acceptable name for a custom variable, but it is acceptable as the name of a screening rule or category.

Using Your Own Data in Standard Responses

It is possible to incorporate data that you keep external to Universal Contact Server into your standard responses (including automated responses). This data could include case numbers, account information, and so on. Remember that attached data always consists of key-value pairs.

Incorporating external data into standard responses is a two-step process:

1. Retrieve the external information and add it to the interaction as attached data. One place to do this is in a routing strategy (see *Interaction Routing Designer Help*).
2. Now that you have attached the data to the interaction, you can use the `AttachedData` property of the Interaction object to access the data and incorporate it into your standard response. The `AttachedData` property requires one argument, which is the key name. The result of the following formula is the value associated with the `OrderStatus` attached-data key:

   ```
   <$Interaction.AttachedData("OrderStatus")$>
   ```

Field Codes in Knowledge Manager

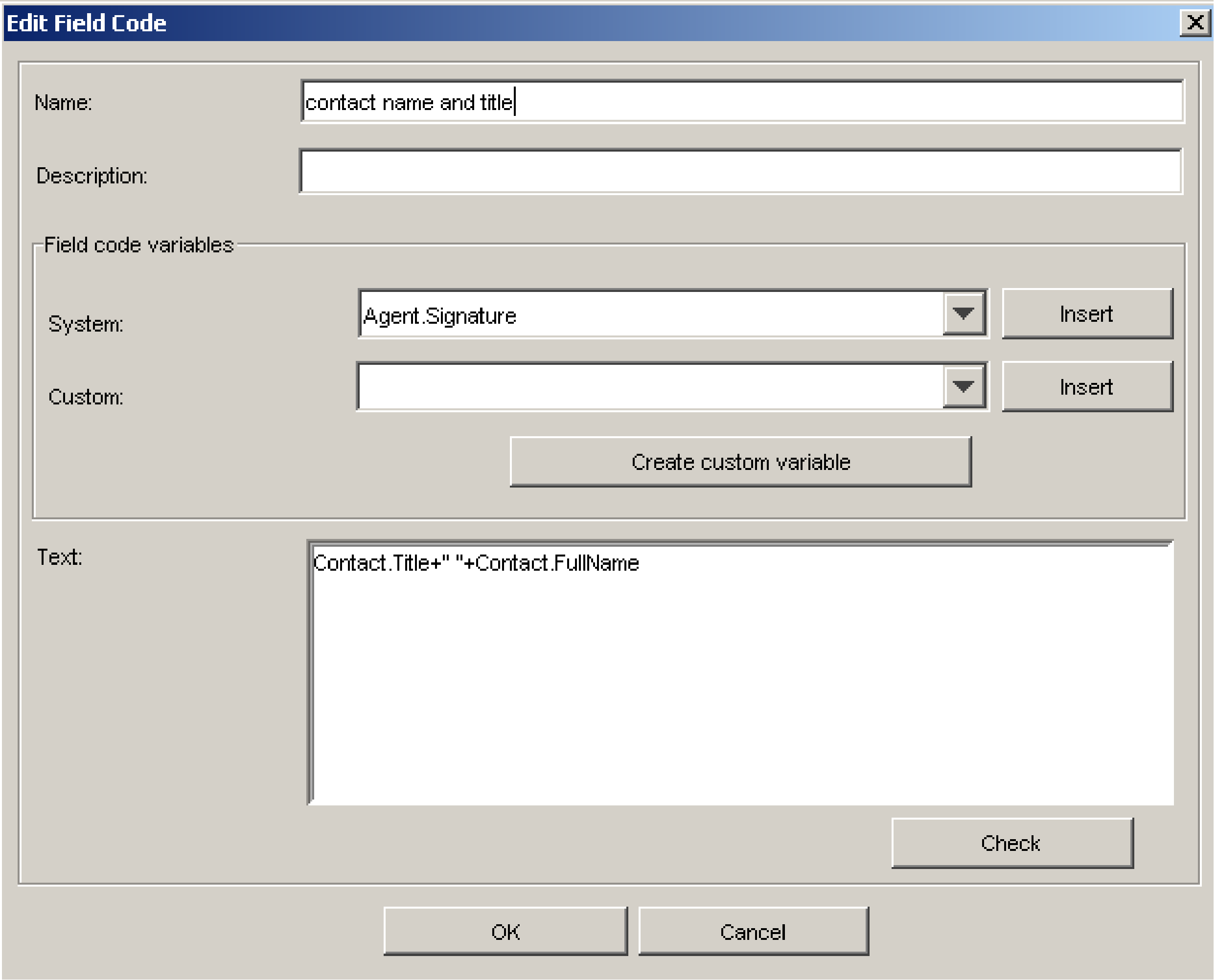
Knowledge Manager separates the task of creating field codes from the task of creating standard responses. This allows you to create complex field codes that include multiple objects, formulas, and constants (see “Complex Field Code” on page 58 for an example). You can then use these complex field codes in multiple standard responses.

Note: Standard Responses that are intended for use in FAQ objects should not contain field codes. FAQ objects contain no means of rendering field codes. See also “FAQ Objects” on page 150.

Procedure: Creating field codes

Start of procedure

1. On the Field Codes tab of Knowledge Manager, right-click and select New Field Code. The New Field Code dialog box (also called Edit Field Code) opens, as shown in Figure 19.



Name: contact name and title

Description:

System: Agent.Signature

Custom:

Text: Contact.Title+" "+Contact.FullName

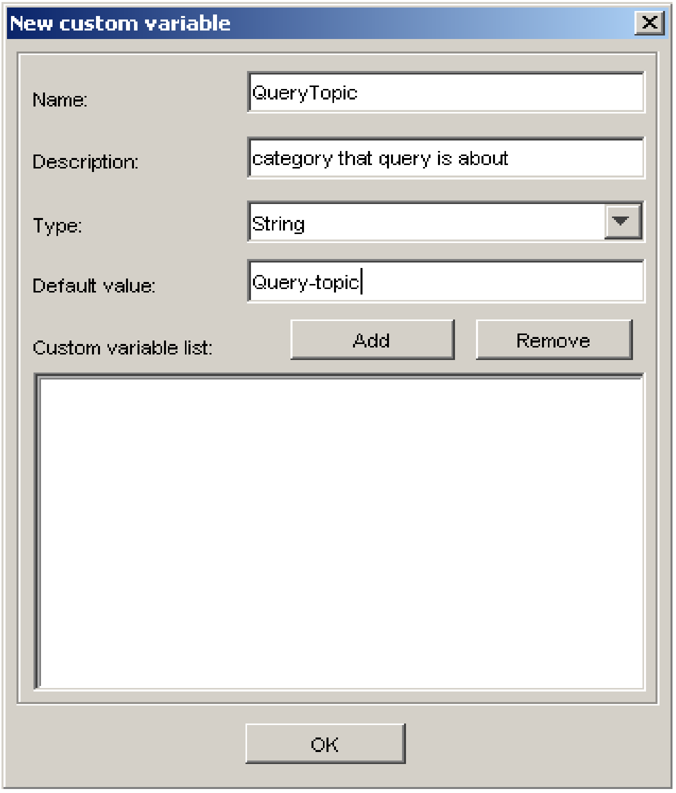
Figure 19: Edit Field Code

The Text field shows the field code as you create and edit it.

2. Enter a name and description for the field code. Name is the only required field in this dialog box.
3. The Field code variables section includes two fields:
   a. System. Click the down arrow to display a list of all system variables.

b. `Custom`. Click the down arrow to display a list of all custom variables that you have created.

To create a new custom variable, click `Create custom variable`.The `New custom variable` dialog box opens, as shown in Figure 20.



Name: QueryTopic

Description: category that query is about

Type: String

Default value: Query-topic

Figure 20: New Custom Variable

- Enter a name (required) and description (optional). Observe the restrictions (see page 48) on custom variable names.
- Select a type (`String` or `Integer`).
- Enter a default value (required). This value is rendered when you click `Check Field Codes` in the Standard Response Editor (see Step 4 in “Creating Standard Responses” on page 28).
- Click `Add`.

You can also use this dialog box to edit and delete existing custom variables. Click `OK` to return to the `New Field Code` dialog box.

4. Select system and custom variables from the drop-down lists, then click `Insert` to insert them into the `Text` field.
5. Enter any other desired text in the `Text` field. This text must conform to the rules described in “Field Code Syntax” on page 54.

6. Click Check to verify that the field code is well-formed (that is, that it has no typographical errors, missing parentheses, and so on).

End of procedure

Next Steps

- Use field codes in a standard response (next section).

Procedure: Inserting field codes into a standard response

Start of procedure

1. On the Categories tab, do one of the following:
   a. Double-click a standard response. The Edit Standard Response dialog box opens.
   b. Right-click and select New. The New Standard Response dialog box opens.

   These two dialog boxes are identical apart from their titles. Figure 21 shows New Standard Response.

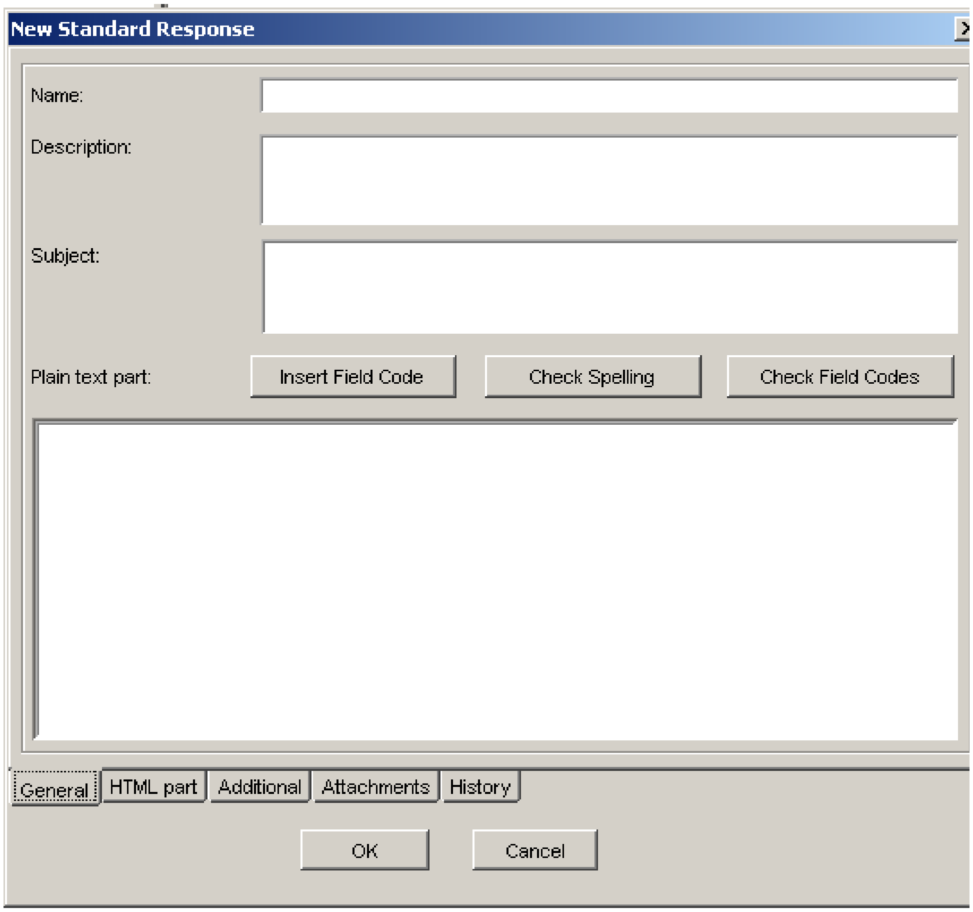
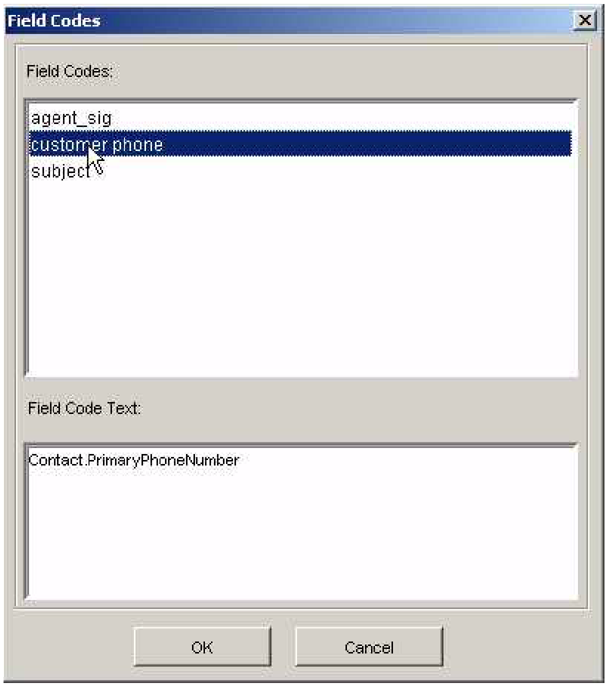


Figure 21: New Standard Response

2. Click `Insert Field Code` to display a list of all the field codes that you have created. The list appears in the dialog box shown in Figure 22 on page 53.



agent_sig
customer phone
subject

Field Code Text:

Contact.PrimaryPhoneNumber

Figure 22: List of Field Codes

3. Select a field code and click `OK` to insert it, together with its required delimiters <$ $>, into a standard response. This returns you to the `Edit Standard Response` (or `New Standard Response`) dialog box.
4. Click `Check Field Codes` to see the standard response with the field codes rendered. The values rendered come from a collection of generic properties that exists for this purpose.

End of procedure

Next Steps

- Examine two examples of the use of field codes (page 56)

Field Code Anatomy

In addition to system variables such as `Contact.FirstName`, field codes may contain formulas. This section provides an outline of formula usage. For details on many of these topics, see Chapter 3 of the *eServices 8.1 Reference Manual*.

You must always delimit field codes by using `<$ · $>`. If you type a field code directly into the body of a standard response, then you must enter the delimiters yourself. If you select from the list of field codes in Knowledge Manager, then the delimiters are added automatically.

The text that appears inside the delimiters is a formula. Field code formulas are very similar to formulas in other applications, such as Microsoft Excel.

A *formula* is a sequence of one or more operands (such as numbers and text strings), separated by operators (such as `+` and `-`).

For example, in the following formula, `2` and `3` are operands and `+` is an operator:

```
<$2 + 3$>
```

Operands can be values that do not change (constants), or values that vary based on the context. In the previous formula, all the operands are constants, so the formula always evaluates to `5`. The next formula, on the other hand, evaluates to a different value for each agent who uses it:

```
<$Agent.Signature$>
```

Field Code Syntax

Here is a summary of field code syntax:

- As stated previously, a field code must be delimited by `<$ ... $>`.
- Alphabetic strings, whether constants in formulas or elsewhere in a field code, must be enclosed in double quotes.
- Numeric constants require no special treatment.
- You must use special characters for some purposes. For example, for your field code to render with a line break, you cannot simply type a carriage return. Instead, you must insert the expression `\n`. A list of these special characters is in Table 3 “Escape Sequences” in Chapter 3 of the *eServices 8.1 Reference Manual.*

Operator Precedence

If you use more than one operator in a formula, the order in which they are evaluated depends on their relative *precedence* (higher precedence operators are evaluated first). For example, multiplication (`*`) has a higher precedence than addition (`+`), so that the formula below evaluates to `14`, not `20`:

```
<$2 + 3 * 4$>
```

You can use parentheses to override the default precedence. The formula below evaluates to `20`:

```
<$(2 + 3) * 4$>
```

For a complete list of operators and their relative precedence, see “Operator Precedence” in Chapter 3 of the *eServices 8.1 Reference Manual.*

Data Types

Operands of several different types may appear in formulas:

- Number
- String (text)
- Date/time

- Boolean (true/false)
- Object (Contact, Interaction, and Agent)

Each data type behaves differently in formulas, and the operators have different meanings when you use them with different data types. For example, the `+` operator means "add" when used with numbers, but "concatenate" (paste together) when used with strings. This formula evaluates to *Uncle Sam Wants You*:

```
<$"Uncle Sam " + "Wants You"$>
```

In addition, some operators cannot be used with some data types at all. For example, you cannot use the multiplication (`*`) operator on two strings.

All formulas, regardless of their final data type, are converted to strings before being merged into your standard response. This conversion follows a set of default rules that depend on the data type. For example, the default rules for numbers round them off to integers. This formula causes `2` to be inserted into your standard response, even though the real result is `2.25:`

```
<$9 / 4$>
```

You can use the `Text` function (see below) or format operator (:) to override the default formatting. Either of the following formulas inserts `2.25` into your standard response:

```
<$Text(9 / 4, "#.##")$>
<$(9 / 4):"#.##"$>
```

For a detailed list of data types and how you can use them, see "Data Types" in Chapter 3 of *eServices 8.1 Reference Manual.*

Functions

When composing formulas, you can use many built-in functions. *Functions* are predefined formulas that perform calculations using values, called *arguments,* which you supply. To use a function, write its name, followed by an opening parenthesis, the arguments for the function separated by commas, and a closing parenthesis.

Function arguments may be of any data type, although individual functions may place restrictions on their arguments. Function arguments may be constants or formulas. The `Length` function, for example, takes a single string argument and returns its length in characters. This formula evaluates to `13:`

```
<$Length("Hello, world!")$>
```

As another example, the `Date` function takes individual date components (year, month, day, and so on), and constructs a date/time value. The formula below evaluates to `1965-11-23 09:03:10:`

```
<$Date(1965, 11, 23, 9, 3, 10)$>
```

Functions may act as arguments to other functions. The `WeekdayName` function takes a single date/time argument and returns the day of the week as a string. The formula below evaluates to `Tuesday:`

```
<$WeekdayName(Date(1965, 11, 23, 9, 3, 10))$>
```

This formula evaluates to `7`:

```
<$Length(WeekdayName(Date(1965, 11, 23, 9, 3, 10))$>)
```

For detailed descriptions of all available functions, see Chapter 3 of *eServices 8.1 Reference Manual.*

Using Objects

All object/property pairs are also available in the `Variables` drop-down menu in the Knowledge Manager `Field Code Editor`.

Object properties can be of any data type. `Agent.FullName`, for example, is a string, but `Interaction.DateCreated` is a date/time.

The data type of an object property can even be another object. For example, `Contact.EmailAddresses` yields another object called a `ContactEmailAddressList`. In cases such as this, you can access the properties of the resulting object by entering a period (.), followed by the property name, just as before. For example, the formula below evaluates to the number of e-mail addresses assigned to the contact:

```
<$Contact.EmailAddresses.Count$>
```

Some object properties require arguments just as functions do. For these properties, write the arguments, enclosed in parentheses after the property name, just as before. For example, the ContactEmailAddressList object has a property named `Exists`, which you can use to test whether a particular e-mail address is assigned to a contact. The data type of this property is Boolean (true/false), and it takes one argument, the e-mail address to test. For example:

```
<$Contact.EmailAddresses.Exists("samd@acme.com")$>
```

For detailed descriptions of all objects and their properties, see “Objects” in Chapter 3 of *eServices 8.1 Reference Manual.*

Examples

This section presents examples of the use of field codes.

Procedure:
Using a custom variable

Purpose: This is a simple example of the use of a custom variable in a standard response.

Prerequisites

This example assumes a category tree that includes categories called `Cookbooks`, `Mysteries`, and `Reference`.

Start of procedure

1. In Knowledge Manager:
   a. Create a custom variable called `QueryTopic` (see "Creating field codes" on page 49).
   b. Create a field code called `Query_Topic` that consists of the variable `QueryTopic`.
   c. Create a standard response of type `Autoresponse` called `AUTO` that includes the sentence `Thank you for your inquiry about <$ Query_Topic $>`.
   d. Create the following screening rules:
      - `Cookbook: RegExFind("cook") || RegExFind("recipe") || RegExFind("food") || RegExFind("cuisine")`
      - `Mystery: RegExFind("murder") || RegExFind("crime") || RegExFind("case of the") || RegExFind("detective")`
      - `Reference: RegExFind("dictionar") || RegExFind("encyclopedia") || RegExFind("almanac")`
2. In Interaction Routing Designer, create a strategy that applies these screening rules one after the other, assigning a different value to the custom variable for each screening rule:
   a. Create a variable called `var_screen`.
   b. Create a strategy. Start the strategy with a `Screen` object. On the `General` tab of the `Screen` object, select the `Cookbook` rule.
   c. On the `Result` tab, click `Assign values of the key-value pairs`. Then under `Output values` select `var_screen` for `Variable` and enter `ScreenRuleMatch` for `Key from output`.
   d. Connect the `Screen` object to a `Generic Segmentation` object. Create two segments: `var_screen = true` and `var_screen = false`.
   e. Connect an `Autoresponse` object to the top green port (the one corresponding to `true`) of the Segmentation object. In the Autoresponse, select the `Select standard response` radio button and select `AUTO` in the associated drop-down list.
   f. Still in this `Autoresponse`, go to the `General` tab and in the `Field Codes` area (bottom of the tab) click the `New` icon, enter `QueryTopic` under `Key`, and enter `cookbooks` under `Value`.

      This will generate an e-mail that includes the sentence *Thank you for your inquiry about cookbooks.*
   g. Return to the `Generic Segmentation` and connect a new `Screen` object to its second green port (the one corresponding to `false`).
   h. On the `General` tab of the new `Screen` object, select the `Mystery` rule. On the `Result` tab, click `Assign values of the key-value pairs`. Then under `Output values` select `var_screen` for `Variable` and enter `ScreenRuleMatch` for `Key from output`.
   i. Proceed as in Steps d–f: Connect this `Screen` object to a new `Generic Segmentation`, again with segments for `var_screen = true` and `var_screen = false`.

j. As in Step e, connect a new Autoresponse object to the green port for true, select the AUTO standard response, and enter QueryTopic under Key. This time enter mysteries under Value.

k. Return to the second Generic Segmentation's green port for false and repeat Steps g–j, creating a third Screen object and Generic Segmentation. In the Screen object, select the Reference rule; in the Segmentation object, set the custom variable to reference.

End of procedure

Figure 23 shows the strategy as described. The single standard response AUTO generates three e-mails, each with a different word filling the blank in *Thank you for your inquiry about ___.*

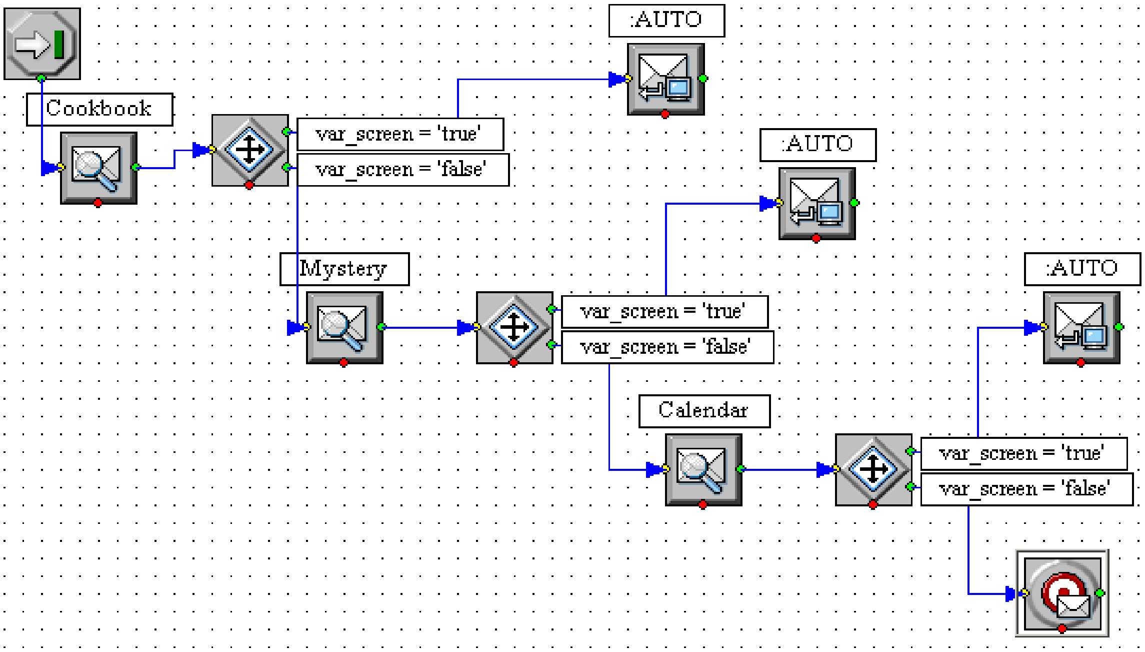


Figure 23: Strategy Using Custom Variable

Next Steps

- Examine an example of a complex field code (page 58).
- Go on to create and manage screening rules (page 59).

Complex Field Code

```
<$ If (Time() - Interaction.DateCreated > 14, "Please accept our
apologies for not having replied sooner. ", "") $>
```

This field code inserts a tardiness apology if more than 14 days have elapsed since the interaction first entered the system. It uses the function If, which has these properties:

- Its syntax is If (Boolean, TrueResult, FalseResult)

- If `Boolean` evaluates to `True`, it returns the second argument.
- If `Boolean` evaluates to `False`, it returns the third argument.

In this example the three arguments of `If` are as follows:

1. `Time() - Interaction.DateCreated > 14` A formula that returns `True` if the difference between the date created and the current system time is more than 14 days. (The result of a mathematical operation on dates is given in days.)
2. `"Please accept our apologies for not having replied sooner. "` A text string apologizing for tardiness, inserted if the formula evaluates to `True`.
3. The null string: if the reply is not late (the formula evaluates to `False`), nothing is inserted in it.

Screening Rules

Screening rules scan an interaction and try to match either a destination address, a regular expression, or both. Screening is performed by Classification Server when it is triggered by a `Screen` object in a routing strategy. A screening rule can optionally be associated with a category.

Note: Screening, like classification (Chapter 2, “Genesys Knowledge Management: Content Analyzer,” on page 89) can operate on any interaction that has text somehow associated with it, whether as the body of the interaction (e-mail, chat), or otherwise (as user data, for example). In practice, it is expected that most interactions that are screened or classified will be e-mail messages; therefore this Guide uses the terms *e-mail* and *message* interchangeably to refer to these interactions. In fact whatever is said here about e-mail applies to any interaction that has associated text.

Screening Rule Editor

The `Screening Rule Editor` enables you to compose and test screening rules.

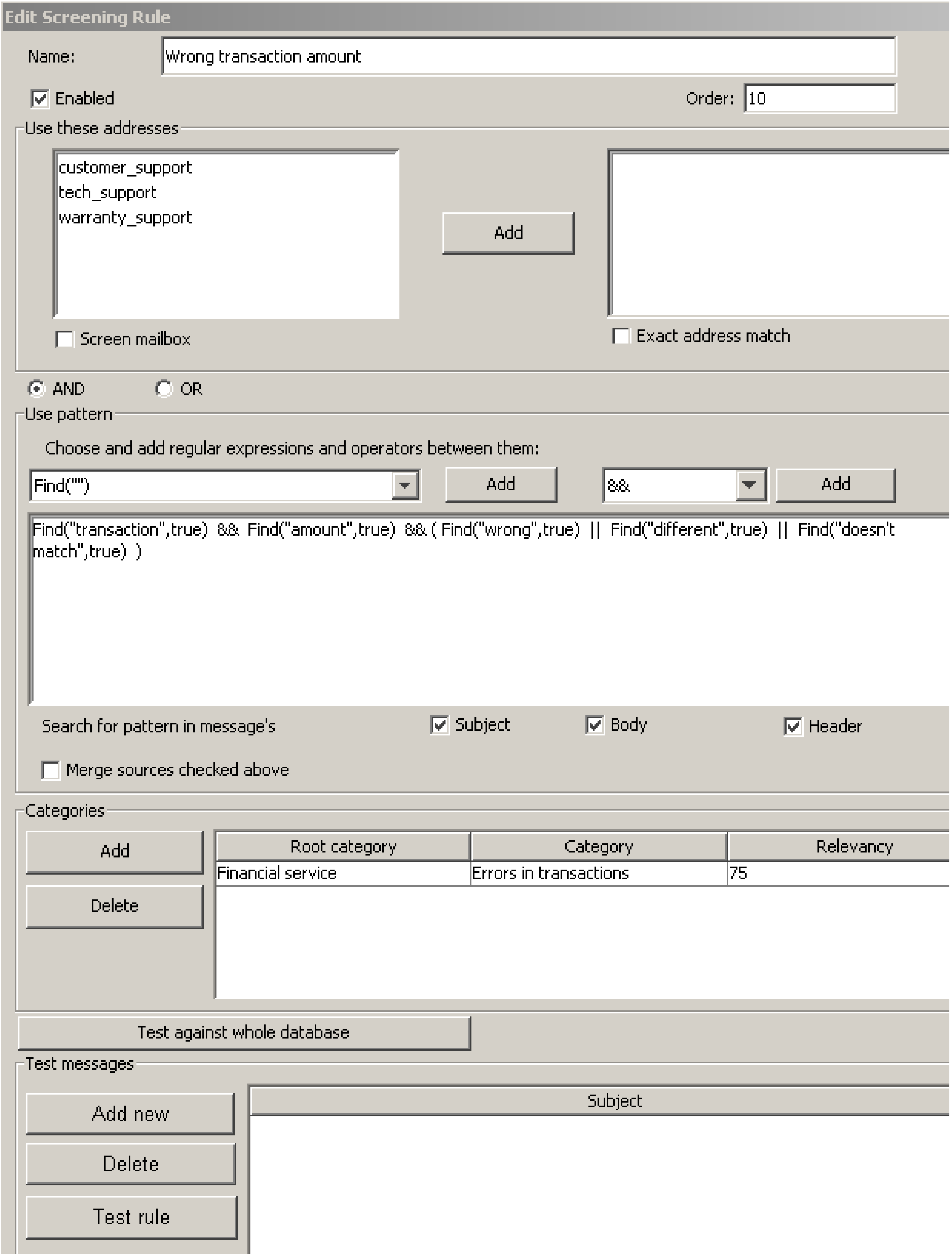
Procedure:
Creating a rule

Purpose: To create or edit a screening rule.

Start of procedure

1. Go to the Screening tab.
2. Do one of the following:
   - Select New from the File menu.
   - Right-click anywhere on the Screening tab and select New.
   - Select an existing screening rule and double-click or select Edit from the File menu.

The Screening Rule Editor opens, as shown in Figure 24 on page 61. If you are creating a new screening rule, the title of the dialog box is New Screening Rule; if you are editing an existing rule, its title is Edit Screening Rule.



Edit Screening Rule

Name: Wrong transaction amount

☑ Enabled Order: 10

Use these addresses

customer_support
tech_support
warranty_support

□ Screen mailbox □ Exact address match

● AND ○ OR

Use pattern

Choose and add regular expressions and operators between them:

Find("") &&

```
Find("transaction",true)  &&  Find("amount",true)  && ( Find("wrong",true)  ||  Find("different",true)  ||  Find("doesn't
match",true)  )
```

Search for pattern in message's ☑ Subject ☑ Body ☑ Header

□ Merge sources checked above

Categories

Root category	Category	Relevancy
Financial service	Errors in transactions	75

Test messages

Subject

Figure 24: Screening Rule Editor

3. Create or edit a name for the rule, observing the limitations on Knowledge Manager names described in the Note on page 26.

4. Use the `Enabled` check box to enable and disable the rule. The rule must be enabled in order to be available when you add a `Screen` object to a routing strategy. However, once a strategy includes a `Screen` object that uses a particular screening rule, disabling the rule does not prevent the strategy from using the rule.

5. Use the `Order` box to specify the order in which you want this rule to apply with respect to other screening rules. This ordering applies only in a Multiscreen routing object in which `All rules` is selected.

6. In the `Use these addresses` area, select an address from the left-hand window, then click `Add` to copy it to the right-hand window.
   - With the `Exact address match` box cleared, the rule looks for messages having that address as a substring of their destination address. For example, `xyz@domainname.com` matches `abc.xyz@domainname.com` and `xyz@domainname.com`.
   - With the `Exact address match` box selected, the screening rule looks for messages having that exact address as a destination. For example, `xyz@domainname.com` matches `xyz@domainname.com` but not `abc.xyz@domainname.com`. This match is not case sensitive.

Notes: The e-mail addresses listed in the left-hand window of `Use these addresses` are defined in this configuration database object: `Business Attributes > EmailAccounts > Attribute Values > Annex > general > address`.

Instead of moving addresses from the left-hand window to the right-hand, you can directly type an address in the right-hand window.

You may leave the right-hand window empty, in which case the rule ignores the address in matching.

7. Still in the `Use these addresses` area, select `Screen mailbox` to make the rule match the POP box from which the e-mail entered the eServices system, rather than the `To` field of the e-mail itself. The difference is that each e-mail enters the system from exactly one mailbox, while the `To` field can contain multiple addresses.

Note: For this feature to work as expected, the E-mail Server `enable-same-mail-from-mailboxes` option must be set to `true`. With this setting, E-mail Server creates a separate interaction for each address in the `To` field (that is, for each mailbox that it pulls the e-mail from when it creates the interaction).

8. Select `AND` or `OR` radio buttons:

- AND—Match the addresses selected in Use these addresses *and* match the pattern defined in Use pattern (see Step 9).
- OR—Match the addresses selected in Use these addresses *or* match the pattern defined in Use pattern (see Step 9).

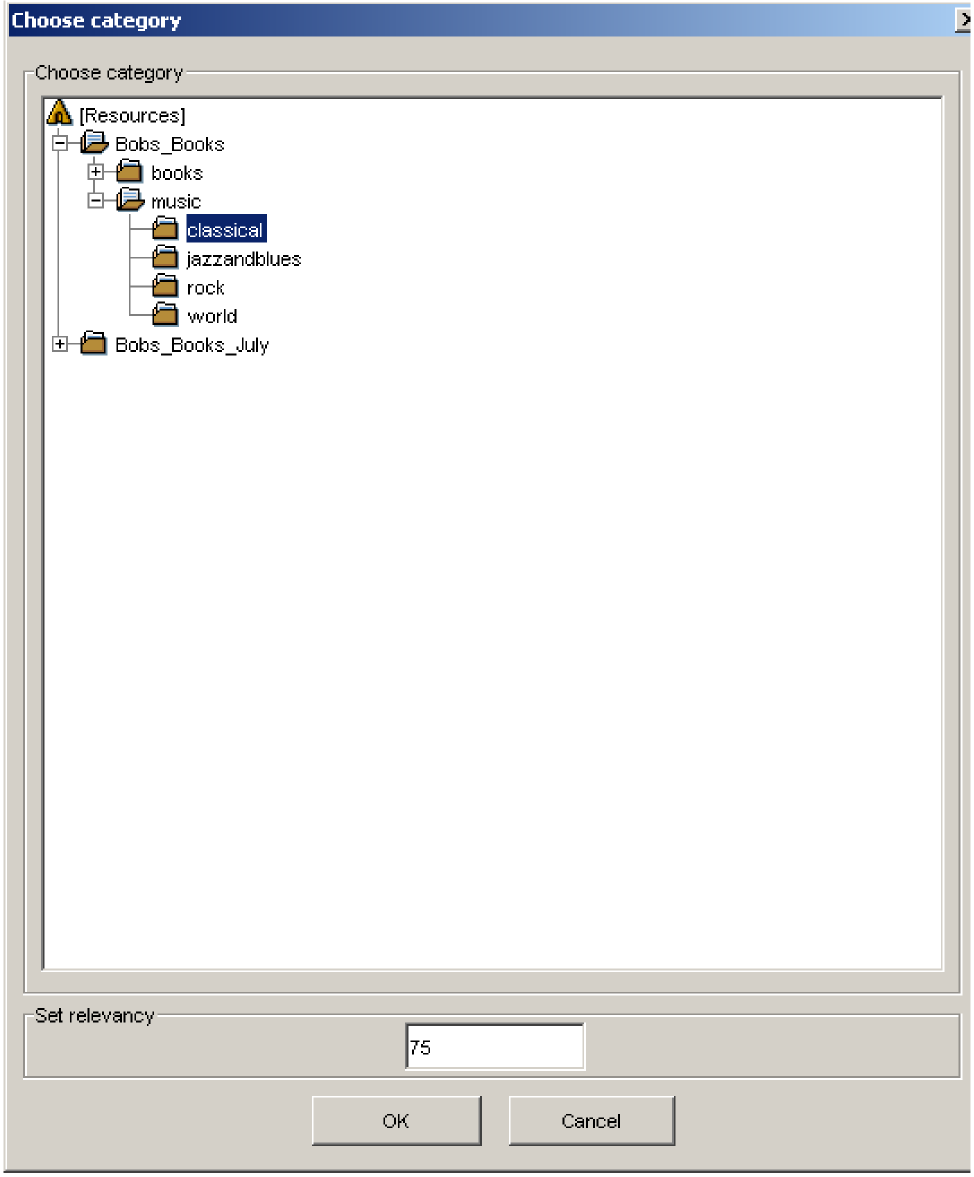
9. In the Use pattern area, compose the rule, using the drop-down lists for functions and operators. See the next section "Syntax and Semantics" for an explanation of how to construct rules.

10. Use the check boxes to have the screening rule apply to the message body, subject, header, or any combination. You must select at least one.

With multiple check boxes selected, there are two ways that a screening rule can behave. Use the Merge sources checked above checkbox as one way to control this; see "Subject, Body, and Header" on page 71 for an explanation.

11. In the Categories area, associate the screening rule with a category:

a. Click Add. The Choose category dialog box appears, as shown in Figure 25.



Set relevancy 75

Figure 25: Choose Category

When an interaction matches this screening rule, the results are similar to classifying it using Genesys Content Analyzer: it receives a category name and a confidence level. The confidence level indicates that the system is x percent confident that this interaction belongs in this category. With Content Analyzer, the system assigns the category and confidence level. But with a screening rule, it is up to you, the user, to decide what category the interaction belongs to and how confident you are of that categorization.

Do so as follows:

b. Select the category you want to associate the screening rule with.

c. Set the relevancy in the `Set relevancy` box.

End of procedure

Next Steps

- Test the rules, as described in the next section.

Testing a Rule

There are two ways to test a screening rule: on the UCS database, and on specially-created text.

Procedure: Testing screening rules on the UCS database

Purpose: To test a screening rule on all interactions in the UCS database that relate to the current tenant.

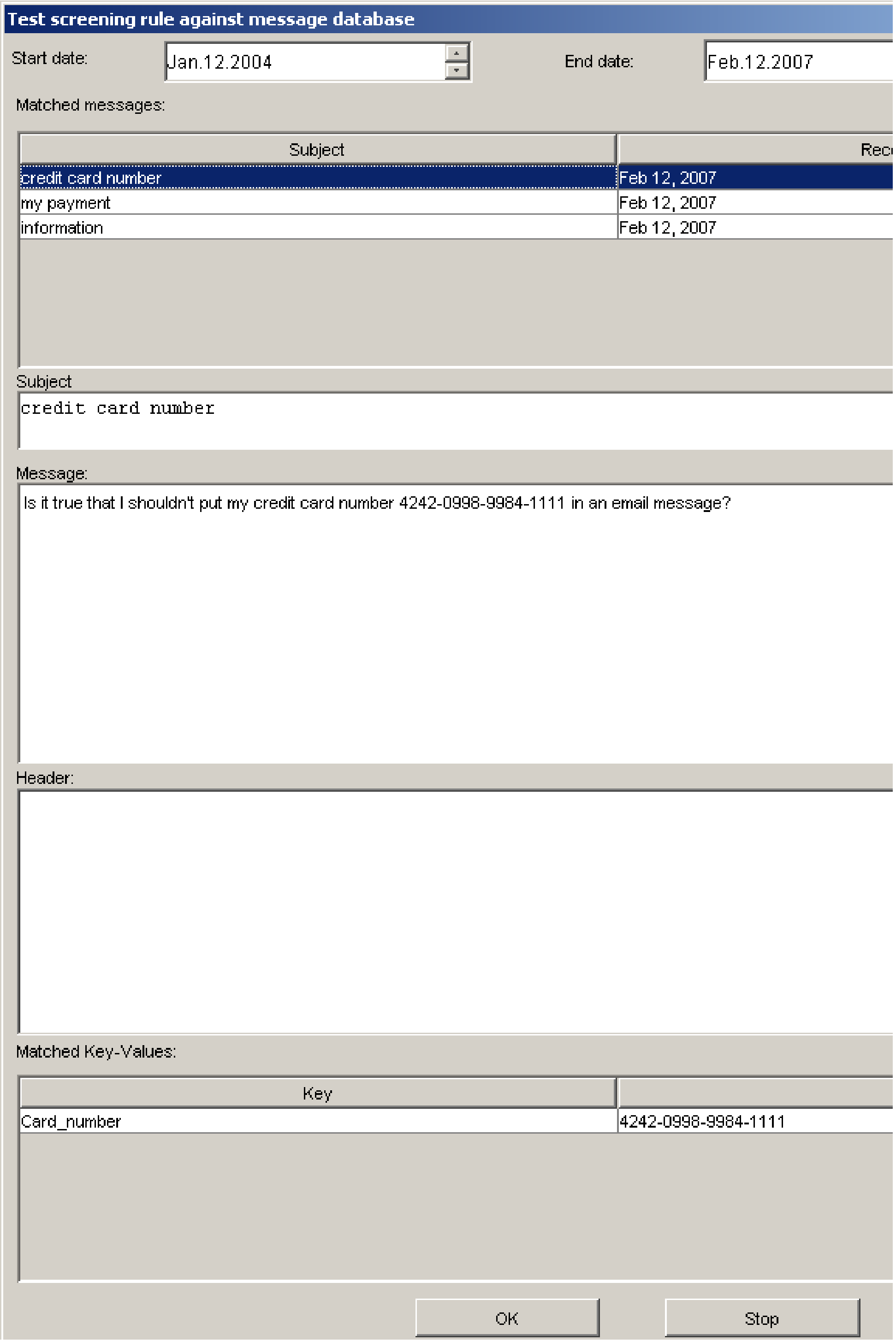
Start of procedure

1. Click `Test against whole database` near the bottom of the `Screening Rule Editor`. The `Test screening rule against message database` dialog box appears, as shown in Figure 26.
2. Adjust the contents of the `Start date` and `End date` boxes to cover the desired timespan, then click `Find`.

   If there are a great many interactions and the testing process is taking too long, you can click `Stop`.

The results of the test display as follows:

- The `Matched messages` area lists all interactions that match the rule.
- The `Subject`, `Message`, and `Header` boxes show the text of the subject, body, and header of the interaction that is selected in the `Matched messages` area.
- The `Matched Key-Values` area shows all keys in the interaction's user data whose values match the rule. In the example in Figure 26, the screening rule created a key-value pair to hold the matched string, in this case a series of sixteen digits that could be a credit card number.



Test screening rule against message database

Start date: Jan.12.2004 End date: Feb.12.2007

Matched messages:

Subject	Rec
credit card number	Feb 12, 2007
my payment	Feb 12, 2007
information	Feb 12, 2007

Subject

```
credit card number
```

Message:

Is it true that I shouldn't put my credit card number 4242-0998-9984-1111 in an email message?

Header:

Matched Key-Values:

Key	
Card_number	4242-0998-9984-1111

Figure 26: Test Screening Rule Against Message Database Dialog Box

End of procedure

Next Steps

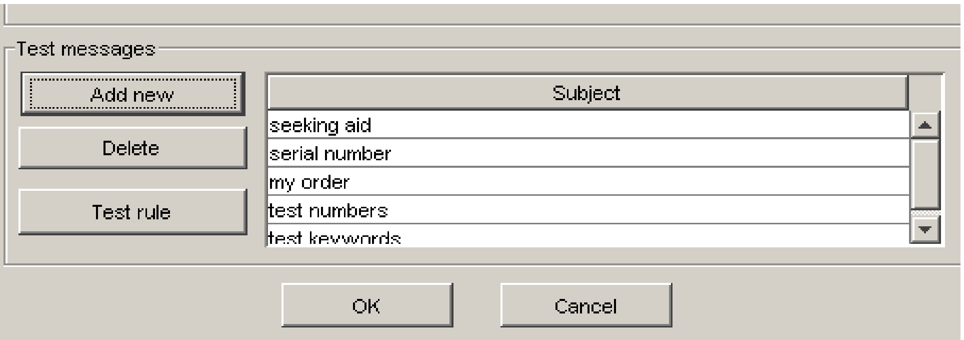
- Test a rule on text that you create.

Procedure: Testing screening rules on specially-created text

Purpose: To test a screening rule on text that you create for that specific purpose.

Summary

At the bottom of the `Screening Rule Editor` is the `Test messages` area, shown in Figure 27. Use this area to create messages that you can test screening rules on.



Subject
seeking aid
serial number
my order
test numbers
test keywords

Figure 27: Test Messages Area

Start of procedure

1. Click `Add new` to open the `Test message` dialog box, in which you can enter the address, subject, and body of a test message. See Figure 28.

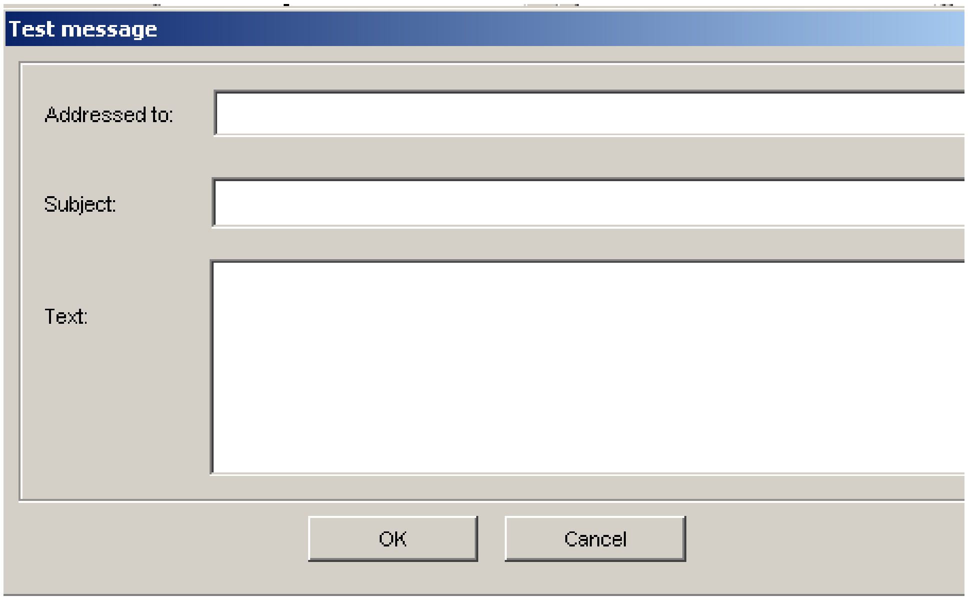


Figure 28: Test Message Dialog Box

   Click `OK` to return to the `Screening Rule Editor`.

2. Manage existing test messages, as follows:
   - Double-click an existing test message to edit it.
   - Select a test message and click `Delete` to delete it.
3. To test a rule:
   a. Select the rule on the `Screening` tab, then open the `Screening Rule Editor` as described in Step 2 on page 60.
   b. Select a test message from the list in the `Test messages` area.
   c. Click `Test rule` to apply the rule to the selected message.
   d. A message window appears, reporting whether the selected message matches the rule.

End of procedure

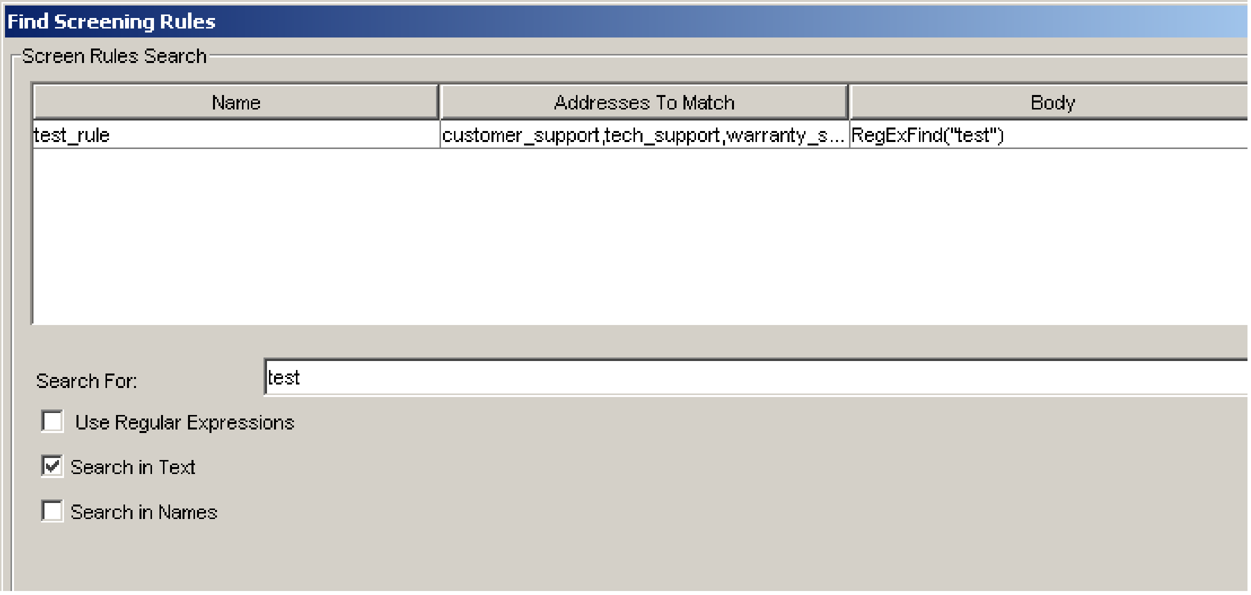
Next Steps

- Learn how to search for screening rules (next section).

Procedure: Finding a rule

Start of procedure

1. Right-click anywhere on the `Screening` tab, then select `Find Screening Rules`. A dialog box with this name displays, as shown in Figure 29.



Name	Addresses To Match	Body
test_rule	customer_support,tech_support,warranty_s...	RegExFind("test")

Search For: test

□ Use Regular Expressions

☑ Search in Text

□ Search in Names

Figure 29: Find Screening Rule

2. Enter a string to search for in the `Search For` field.
3. Select `Use Regular Expressions` if you want the search to treat the string as a regular expression.
4. Select check boxes to search in the screening rule's text, name, or both.
5. Click `Find`. The dialog box displays the name, addresses to match, and body of all screening rules found.

End of procedure

Next Steps

- Read the rest of this “Screening Rules” section to learn more about how screening rules work.
- Go on to learn how to import and export Knowledge Manager objects (page 79).

What Screening Rules Check

Screening rules check the following parts of an interaction, depending on what you select in the Screening Rule Editor, and on the settings in the IRD screening objects:

- The subject, if you select that check box.
- The body, if you select that check box.
- The header, if you select that check box.

  See also “Subject, Body, and Header” on page 71 on how screening rules behave if two or more of the preceding are selected.
- The destination address, if you have put anything in the right-hand box in the `Use these addresses` area of the `Edit Screening Rule` dialog box.
- The value of any key in the user data, if both of the following are true:
  - In the Multiscreen or Classify strategy object, you select a key in the `User data key if specified` drop-down list under `Get screened data from.`
  - In the Screening Rule Editor, you select the `Body` check box in the `Use pattern` area of (see Step 10 on page 63).

  User data is first associated with the interaction by the media server when it creates that interaction. As an example, E-mail Server associates the following user data with the interaction:
  - FirstName (from Contact information)
  - LastName (from Contact information)
  - Mailbox (value of the `address` option in the `pop-client` section of the E-mail Server Application object)
  - To (MIME header field)
  - Subject (truncated to 512 characters)
  - FromAddress (personal part of From header field)
  - FromPersonal (e-mail address part of From header field)
  - All Header fields (except Received, Return-Path, X-MIMETrack, Subject, Sender, From, To, Cc, Bcc) prefixed by `Header_`
  - All parent attached data (originally created by E-mail Server) which can be inherited; that is, all parent attached data:
    - not starting with `Header_`
    - not starting with _ (underscore)
    - not equal to `GEM_Failure`
    - not equal to `GEM_FailureMsg`
    - not equal to `GEM_FailureArgsl`

  User data may then be added or modified by a routing strategy.

Subject, Body, and Header

If you select more than one of the Subject, Body, and Header areas, a screening rule can behave in the following two ways:

The default behavior is for the rule to apply to each area in turn; for example, with `Subject` and `Body` selected, the rule applies first to the Subject, then to the Body.

The alternative behavior is for all selected items to first be concatenated so that the rule applies to all at once. There are two ways to achieve this alternative behavior.

- To enforce it for all screening rules, set the `subject-body-header` option for both Knowledge Manager and Classification Server to `true`.
- To enforce it for a particular screening rule:
  - **a.** Leave `subject-body-header` set to `false`.
  - **b.** Open the rule in the Screening Rule Editor.
  - **c.** Select the `Merge sources checked above` check box.

Note: Setting `subject-body-header` to `true` overrides any selection of the `Merge sources checked above` check box for a particular rule.

Syntax and Semantics

This section describes the structure and meaning of screening rules.

Functions and Arguments

Functions

Screening rules can use three basic functions:

- `Find("<text>"),` where `<text>` is a text string. It returns the result `true` if the interaction contains the exact string between quotes, ignoring case.
- `RegExFind("<regular expression>"),` where `<regular expression>` is a regular expression (see "Regular Expressions" on page 74). It returns the result `true` if the interaction contains any string that matches the regular expression between quotes.
- `RegExMatch("<regExp>"),` where `<regular expression>` is a regular expression. It returns the result `true` only if the entire content of the interaction matches the regular expression between quotes.

Note: `RegExFind` and `RegExMatch` are the same except that `RegExFind` looks for a match anywhere in the body of the interaction, whereas `RegExMatch` demands that the entire body of the interaction match the regular expression.

Arguments

All functions have one required argument, which must appear between double quotation marks, as represented above (`<text>` or `<regular expression>`). This required argument can be followed by one or two optional arguments, depending on the function. The full form of each function, including all arguments, is as follows:

```
Find("<text>", <IgnoreCase>)
RegExFind("<regular expression>",<"key">,<IgnoreCase>)
RegExMatch("<regular expression>",<IgnoreCase)
```

IgnoreCase

The IgnoreCase argument must be a Boolean value (*true* or *false*). All three functions ignore case in searches unless you include the IgnoreCase argument with a value of `false`.

For example:

- `Find("pacific")` finds *Pacific* and *pacific*.
- `Find("Pacific",false)` finds *Pacific* but not *pacific*.

You can also substitute `true` for `false`—for example, `Find("Pacific",true)`—which means that case is ignored. So `Find("Pacific",true)` is the same as `Find("Pacific")`.

Key

The `key` argument must be a string. If this argument is present, the system creates a key-value pair with the following characteristics:

- The key name is the string specified by the `key` argument, prefixed by `ScrKey_`.
- The value is the material that the screening rule matches.

The system then adds this key-value pair to the interaction's attached data.

For example, `RegExFind("[A-Z]\d\d\d","ID_code",false)`:

1. Finds strings consisting of a capital letter followed by three digits (see "Regular Expressions" on page 74).
2. Attaches to the interaction a key-value pair called `ScrKey_ID_code` whose value is A123, X005, M999, or whatever the function found in this interaction to match the regular expression.

Operators

Operators are of two types:

- Binary operators join two functions.
- Unary operators operate on a single function.

`&&` is the binary operator *and*. For example,

```
Find("interest rate") && Find("APR",false)
```

matches a message only if it includes both *interest rate* and *APR*.

|| is the binary operator *or*. For example,

```
Find("station wagon") || Find("convertible")
```

matches any message that includes either *station wagon* or *convertible* (or *Station Wagon* or *station Wagon* or *Convertible*).

`!` is the unary operator *not.* For example,

```
!Find("windows")
```

matches any message that does not include the word *windows.* You can combine `!` with a binary operator. For example,

```
Find("bird") && !Find("goose")
```

matches any message that includes *bird* but does not include *goose.*

Operator Precedence

`p && q || r` is parsed as `(p && q) || r`. For example, consider:

```
Find("debt") && Find("income") || Find("profit")
```

To paraphrase, this screening rule is basically “find X or find Y,” where X is *debt* and *income,* and Y is *profit.* It matches both *debt exceeds income* and *profits are fantastic*. You can modify the default precedence by the explicit use of parentheses; for example:

```
Find("debt") && (Find("income") || Find("profit"))
```

This screening rule is basically “find X and find Y,” where X is *debt* and Y is either *income* or *profit.* It matches both *debt exceeds income* and *debts impact profit.*

Pattern Builder

Knowledge Manager includes a pattern builder that offers the choice of each function type in all possible forms, with and without optional arguments, for a total of eight, as shown in Figure 30.

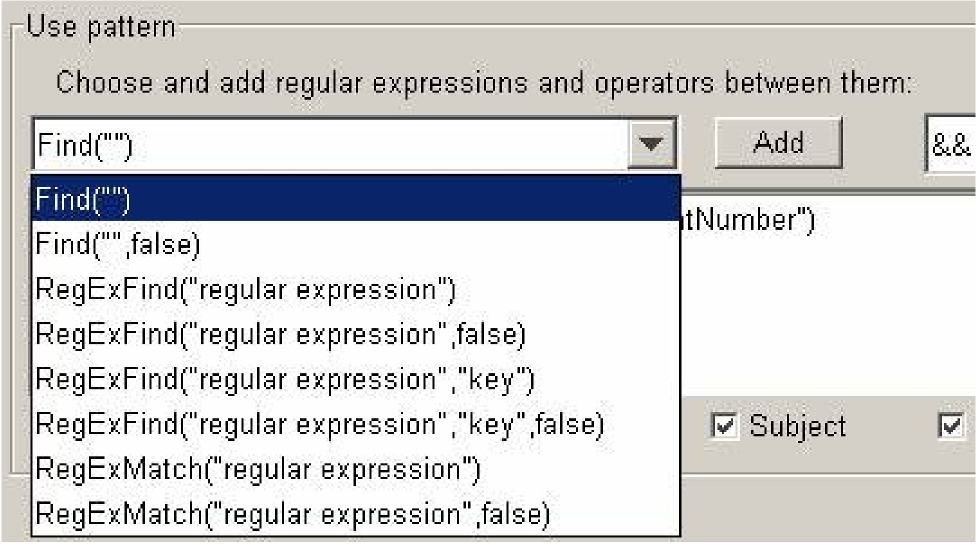

Use pattern
Choose and add regular expressions and operators between them:
Find("")
Add
&&
Find("")
Find("",false)
RegExFind("regular expression")
RegExFind("regular expression",false)
RegExFind("regular expression","key")
RegExFind("regular expression","key",false)
RegExMatch("regular expression")
RegExMatch("regular expression",false)
tNumber")
☑ Subject
☑

Figure 30: Regular Expressions in Knowledge Manager

After you select a form and click `Add`, you must put text between the quotation marks. More specifically, you must:

- For `Find`, put text between the empty quotation marks.
- For `RegExFind` and `RegExMatch`, substitute your desired text for `regular expression` and/or `key`.

Regular Expressions

A regular expression stands for, not one particular character string, but a class of character strings. For example, suppose that you want to find all interactions with U.S. Zip codes in them. U.S. Zip codes are five-digit numbers, so you could in theory write about 9,000 screening rules (`Find("00000")`, `Find("00001")`, `Find("00002")`, and so on). Fortunately, you can use the special symbol `\d`, which stands for any digit, to write a screening rule using a regular expression: `RegExFind("\d\d\d\d\d")`. This screening rule matches any sequence of five digits.

There are often several different ways of writing the same regular expression. For instance, two items separated by a hyphen and enclosed in square brackets denotes a range of which the two items are endpoints. So `[a-d]` matches a, b, c, or d, and `[5-8]` matches any digit between 5 and 8; hence `\d` is the same as `[0-9]`.

Note: In general usage, apart from Genesys eServices, regular expressions are case sensitive. However, in Knowledge Manager regular expressions are not case sensitive unless you add `,false` as described above.

Table 4 lists some of the most commonly-used elements of regular expressions:

Table 4: Elements of Regular Expressions

Symbol	Meaning	Example
.	Any character, including space	`b.t` matches *bat, bet, bit,* and *but.*
\d	Any digit	`\d\d` matches any pair of digits from 00 to 99.
\s	Space	`\d\s\d` matches 1 0, 5 9, and so on.
*	Zero or more instances of the preceding expression	`o*f` matches *oof, of,* and *f.* `me.*d` matches *med, mead,* and *meed.*
+	One or more instances of the preceding expression	`bre+d` matches *bred, breed* and *breeed.*
?	Zero or one instances of the preceding expression	`c?rude` matches *rude* and *crude.*
{x}	X instances of the preceding expression	`st.{2}k` matches *steak, stork,* and *stink.*
^	Any character except the following	`s[^e]t` matches *sat, sit,* and *sot,* but not *set.*
[ ]	Any characters or ranges within the brackets	Any characters: `b[aeiou]at` matches *boat* but not *brat.* Any range(s): `[0-9]th` matches *5th, 6th, 7th* `[a-z]` matches any lowercase letter; `[A-Z]` matches any uppercase letter.
\	Turns off the special meaning of the following symbol	`\*` matches the character * (asterisk); `\.` matches the character . (period or full stop).
|	Or	`[b|p]ig` matches *big* and *pig.* Do not be confused: | means *or* in regular expressions, but || means *or* as one of the Operators used in screening rule formulas.

Here are some other points to keep in mind:

- Space is just another character. The regular expression `savings account` contains a space, and so it does not match the string *savingsaccount.*
- Word boundaries are not considered. The regular expression `read` matches not only *read,* but also *reader, ready, spread, bread,* and so on.
- Use parentheses to group parts of regular expressions together. For example, `RegExFind("(\d{3}\.){2}")` puts `\d{3}\.` in parentheses so that the number-of-instances item `{2}` applies to the all of `d{3}\.`, not just to `\.` This expression matches any group of three digits plus period plus any three digits plus period (for example, 198.351.). Further examples are provided in "Examples" on page 76.
- Regular expressions make use of many more special characters and operators than those listed in Table 4. Much documentation on regular expressions is available on the Web. Because Genesys Knowledge Management uses Java classes for regular expressions, it is best to consult documents describing the particular version of regular expressions used in Java.

Examples

This section provides examples of screening rules.

Credit Card Number

To find text that includes a typical credit card number, you need to match a sequence of four groups of four digits, each group separated by `-` (hyphen):

```
\d\d\d\d\-\d\d\d\d\-\d\d\d\d\-\d\d\d\d
```

Note: This regular expression also works without the `\` (backslash) before the hyphens. However, it is better practice to write `\-` for the character hyphen, because the hyphen also has a special use in range expressions like `[a-z]`.

Or if you want to allow for the possibility that some people will omit the hyphens, use `?` to make the hyphen optional:

```
\d\d\d\d\-?\d\d\d\d\-?\d\d\d\d\-?\d\d\d\d
```

You could also use the repetition notation to shorten each `\d\d\d\d` to `\d{4}`.

North American Phone Number

North American phone numbers consists of ten digits, grouped into two groups of three and one of four. There are a number of ways for the groups to be separated:

203-555-1234

(203) 555-1234

(203)555-1234

203 555-1234

203.555.1234

The following regular expression matches all of the above:

```
(\d\d\d|\(\d\d\d\))[\s\.\-]?\s*\d\d\d[\-\.]\d\d\d\d
```

Table 5 analyzes this regular expression.

Table 5: Phone Number Regular Expression

Symbols	Meaning	Remarks
\d\d\d	Three digits	
\d\d\d|\(\d\d\d\)	Three digits, or three digits enclosed in parentheses	\ turns off the special meaning of the character (
[\s\.\-]?	Space or period or hyphen or zero	Any one of the items enclosed in square brackets, either once or not at all
\s*	Zero or more spaces	
\d\d\d	Three digits	
[\-\.]	Hyphen or period	Note again the need to use \
\d\d\d\d	Four digits	

Telltale Words

To screen for interactions from dissatisfied customers, you might try a regular expression like the following:

```
(not\s([a-z]+\s)*(pleased | satisfied)) | unhappy | complain
```

The first part of this expression matches *not* followed by zero or more words followed by *pleased* or *satisfied*; for example, *not very pleased, not satisfied, not at all satisfied* (but it also matches strings like *can not believe how pleased I am*). The rest matches the single words *unhappy* and *complain.*

Screening for Sentiment and Actionability

You can use Genesys Knowledge Management to analyze interactions that have been brought into the system by Genesys Social Messaging Management. Genesys supplies a sample that analyzes these interactions for:

- The *sentiment* expressed—Positive, negative, or neutral.

- *Actionability*—Whether the interaction calls for attention from an agent.

To use this sample, import the file `SentimentAndActionabilityScreeningRules.kme,` which is located in the `<KnowledgeManagerHome>\SentimentModel` directory. Importing is described in the next section.

This file provides examples of screening rules for detecting sentiment and actionability, plus category trees containing categories that are assigned to interactions that match the rules.

If you have Genesys Content Analyzer, you can use it to analyze sentiment and actionability, as described in "Analyzing Sentiment and Actionability with Content Analyzer" on page 148.

For more information on Genesys Social Messaging Management, see the *eServices Social Media Solution Guide,* available on the Genesys Documentation Wiki at `http://developerzone.genesyslab.com/wiki/index.php?title=Category:EServices_Social_Media_Solution_Guide.`

Importing and Exporting

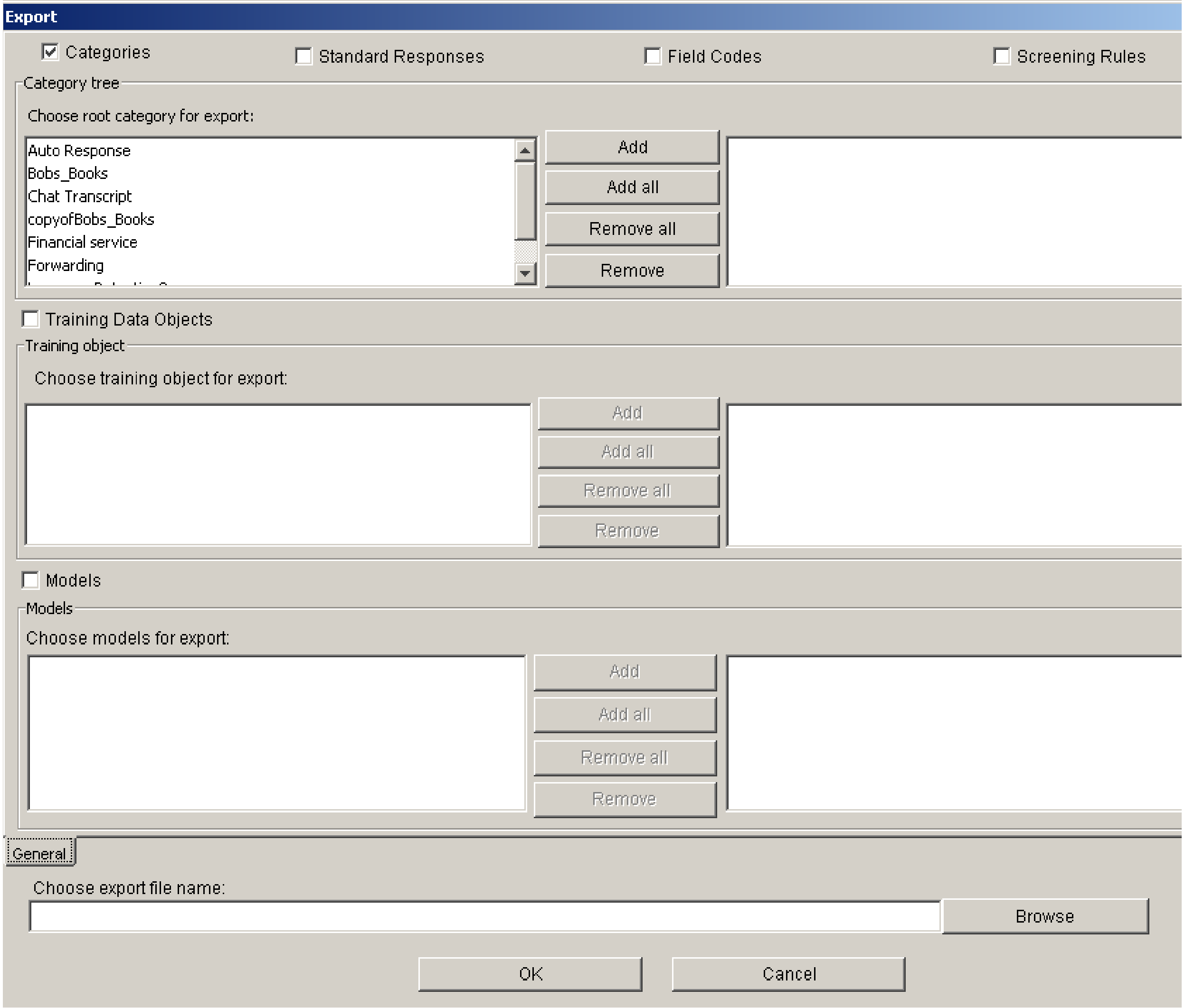
You can import and export categories, standard responses, field codes, screening rules, training objects, and models (training objects and models are restricted to Genesys Content Analyzer; see Chapter 2, "Genesys Knowledge Management: Content Analyzer," on page 89 for more information). Use exported files as backups or to transfer objects between environments.

Note: At this time you cannot import or export FAQ objects.

Procedure:
Exporting Knowledge Manager objects

Start of procedure

1. On any tab, select `Export`. The `Export` dialog box opens, as shown in Figure 31.



Export

☑ Categories ☐ Standard Responses ☐ Field Codes ☐ Screening Rules

Category tree

Choose root category for export:

Auto Response
Bobs_Books
Chat Transcript
copyofBobs_Books
Financial service
Forwarding

☐ Training Data Objects

Training object

Choose training object for export:

☐ Models

Models

Choose models for export:

Choose export file name:

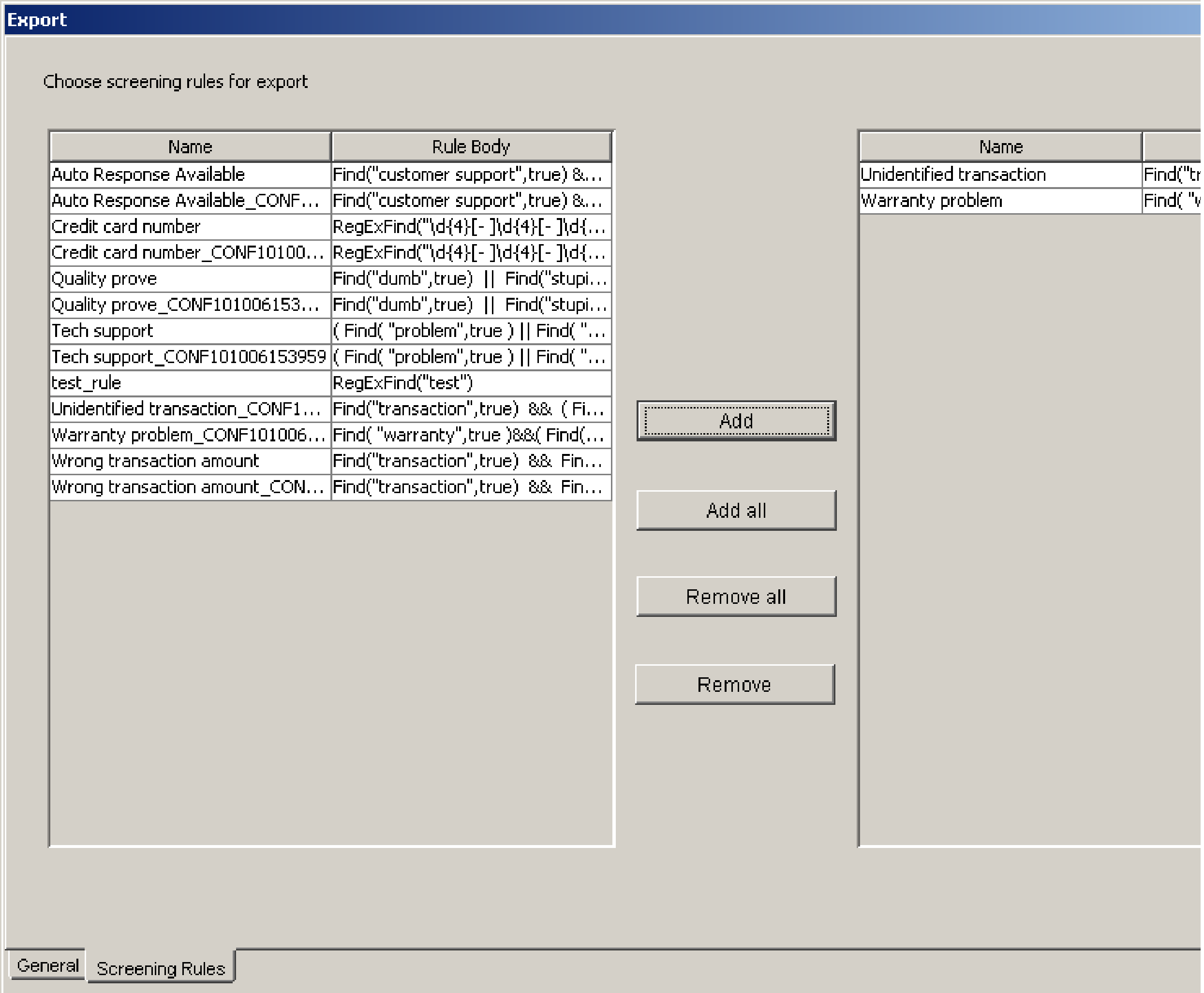
Figure 31: Export Dialog Box

2. Select the object(s) to export, using the check boxes at the top of the dialog box.
   - Categories—Exporting categories is a prerequisite for exporting standard responses, training objects, or models.
   - Standard Responses—Knowledge Manager exports the standard responses belonging to the category tree that you selected.
   - Field Codes—No selection is possible. You must export all or none.

- Screening Rules—If you select this checkbox, a Screening Rules tab appears towards the bottom of the dialog box. You must make further selections on this tab, as described in Step 4

3. Make further selections for category trees, training objects, and models.
   - Each type of object has two fields in the Export dialog box: one on the left listing available objects, another on the right listing the objects selected for export. Use the Add/Add all/Remove all/Remove buttons to move objects between the two fields.
   - The category structures that are selected (that appear in the right-hand category field) determine which training objects and models appear in the Training object and Models areas.

4. Make further selections for screening rules: On the `Screening Rules` tab, shown in Figure 32, move rules from the list of all rules (left-hand field) to the list of selected rules (right-hand field).



Choose screening rules for export

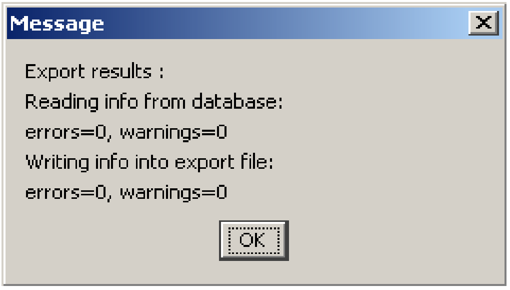
Name	Rule Body
Auto Response Available	Find("customer support",true) &...
Auto Response Available_CONF...	Find("customer support",true) &...
Credit card number	RegExFind("\d{4}[-]\d{4}[-]\d{...
Credit card number_CONF10100...	RegExFind("\d{4}[-]\d{4}[-]\d{...
Quality prove	Find("dumb",true) || Find("stupi...
Quality prove_CONF101006153...	Find("dumb",true) || Find("stupi...
Tech support	(Find("problem",true) || Find("...
Tech support_CONF101006153959	(Find("problem",true) || Find("...
test_rule	RegExFind("test")
Unidentified transaction_CONF1...	Find("transaction",true) && (Fi...
Warranty problem_CONF101006...	Find("warranty",true)&&(Find(...
Wrong transaction amount	Find("transaction",true) && Fin...
Wrong transaction amount_CON...	Find("transaction",true) && Fin...

Name	
Unidentified transaction	Find("tr
Warranty problem	Find("v

Figure 32: Export: Screening Rules Tab

Note: When selecting objects to export, keep in mind that importing will import all objects in the exported file. You cannot select which objects to import.

5. Enter a name for the exported file and click `OK`. The file is created in the directory containing the Knowledge Manager executable.
6. A message appears reporting on any errors or warnings encountered (see Figure 33).



Message

```
Export results :
Reading info from database:
errors=0, warnings=0
Writing info into export file:
errors=0, warnings=0
```

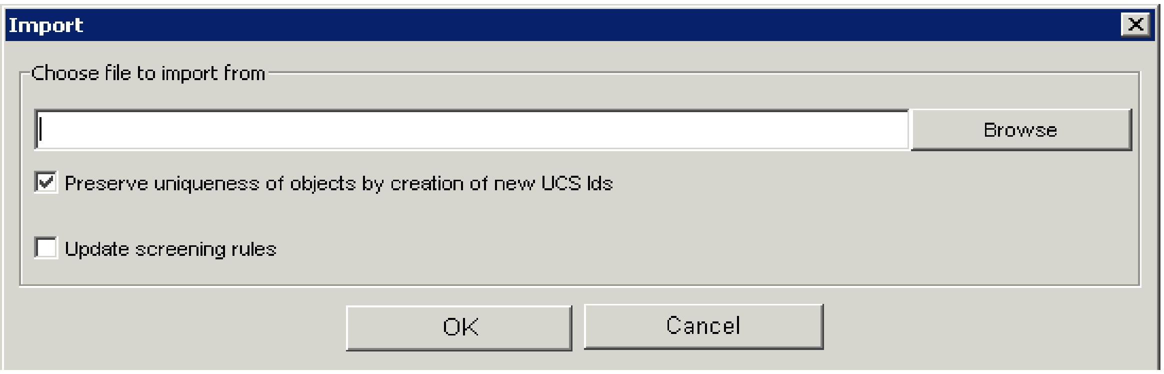
Figure 33: Export Results Message

End of procedure

Procedure:
Importing Knowledge Manager objects

Start of procedure

1. On any tab, select `Import`. The `Import` dialog box opens, as in Figure 34.



Import

Choose file to import from

☑ Preserve uniqueness of objects by creation of new UCS Ids

□ Update screening rules

Figure 34: Import Dialog Box

2. Browse to the file that you want to import, or enter its path and name.

   The check box `Preserve uniqueness of objects by creation of new UCS Ids` controls whether the imported records receive new database IDs. Genesys strongly recommends that you leave this check box selected; otherwise, the imported records keep their old IDs and there is a risk of creating uniqueness conflicts. It is only safe to keep old IDs when you are importing into an empty database. One reason to keep the old IDs would be to preserve compatibility with non–Knowledge Manager components (such as a routing strategy) that need to refer to them.

Note: If the `Preserve uniqueness` check box is cleared and an imported object comes in with an ID that is identical with an existing object's ID, the import process cancels.

The check box `Update screening rules` controls whether imported screening rules overwrite existing screening rules with the same name. If this check box is not selected, screening rules are treated like all other Knowledge Management objects, as described in the preceding paragraph. If the check box is selected, screening rules are treated differently from all other objects: If the names match, the imported screening rule replaces the existing one.

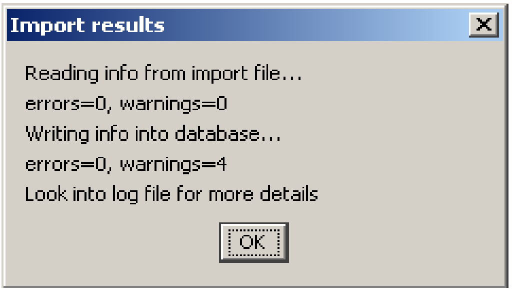
Note: If the imported rule's name does not match any existing rule but its database ID happens to match that of an existing rule, then Knowledge Manager creates a new ID for the imported rule.

3. If a root category being imported has the same name as an existing category, Knowledge Manager asks you to change the name of the category being imported.

   If other objects have the same name as existing objects, Knowledge Manager appends `_<hms>` to the name of the imported object. `<hms>` is a timestamp where `h` is the hour (using a 12-hour clock), `m` is the minutes, and `s` is the seconds. Each unit may be one or two digits; there is no padding. For example, if at 4:25:07 PM you import a screening rule called `Sales,` and there is also an existing rule called `Sales,` the new name of the imported rule is `Sales_4257`.

   Note: This adds between four and seven characters to the name of the object. You should be especially careful of this if any imported object's name is more than 58 characters long: the added characters may produce a new name that violates the 64-character limit on names of Knowledge Management objects (see also page 26). Importing may fail on objects with names that are too long.

4. Click `OK`. All objects in the selected file are imported.
5. A message appears reporting on any errors or warnings encountered (see Figure 35).



Reading info from import file...
errors=0, warnings=0
Writing info into database...
errors=0, warnings=4
Look into log file for more details

Figure 35: Import Results Message

End of procedure

Typical Response Times

This section describes some typical response times for Knowledge Manager tasks. These figures are for a machine with two Pentium 4 processors and 1 GB of RAM.

- Loading and refreshing a category tree takes 4–6 seconds per 2000 categories (each category having one standard response).
- Copying and pasting a category tree takes approximately 100 seconds per 1000 categories.
- Deleting a category tree takes approximately 100 seconds per 1000 categories.

Role-Based Access

Starting with eServices 8.0.1, you can define different levels of access to Knowledge Manager objects. For example, you might create a role called Supervisor which is allowed to create new standard responses but not to delete or approve them.

You define Roles in Genesys Administrator.

Note: Other aspects of eServices configuration can be done either in Configuration Manager or Genesys Administrator, but Genesys highly recommends using only Genesys Administrator for configuring Roles and Tasks.

Logically, a Role is a mapping between `Person` or `Access Group` objects on the one hand, and permitted tasks on the other. The `Person` and `Access Group` objects are created by you to suit your needs, while the tasks are predefined by eServices. Most tasks are defined as the possible application of three actions (create, modify, and delete) to five KM objects (category, standard response, field code, screening rule, and training object); there are several more tasks such as Approve Standard Response. The complete list is given in Table 6.

Procedure: Define an Access Role

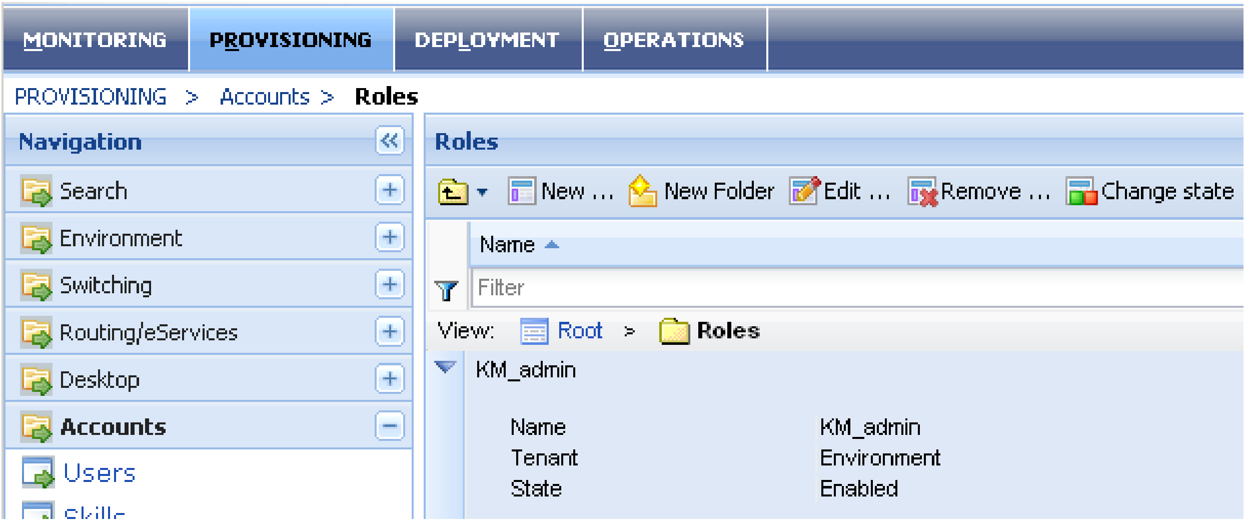
Prerequisites

- The Knowledge Manager `disable-rbac` option has its default value `false`.
- Genesys Administrator is running.
- You have imported the Knowledge Manager metadata template into Knowledge Manager Application object.

Note: For information on using Genesys Administrator, see the Genesys Administrator Help.

Start of procedure

1. In Genesys Administrator, navigate to `PROVISIONING > Accounts > Roles`, as shown in Figure 36. Be sure that the correct tenant shows in the drop-down list.

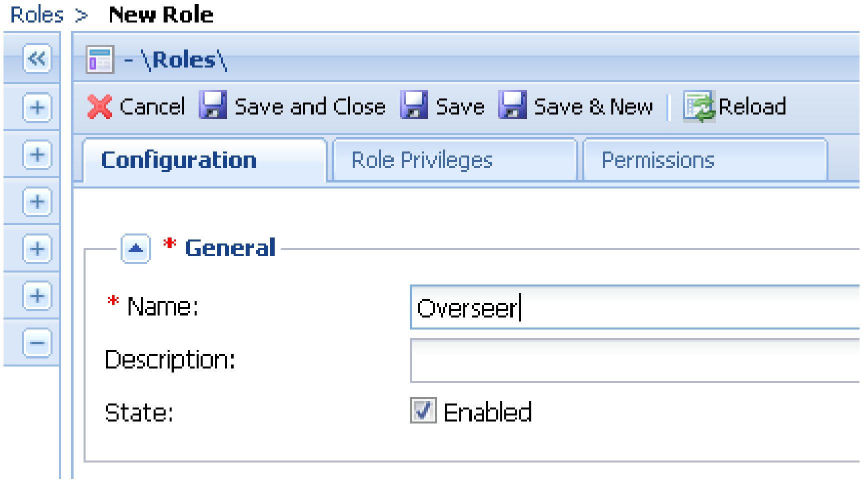


Name	KM_admin
Tenant	Environment
State	Enabled

Figure 36: Roles in Genesys Administrator

2. Click New. On the Configuration tab:

   a. In the General section, enter a name, as shown in Figure 37.



* Name: Overseer

Description:

State: ☑ Enabled

Figure 37: Configuration: General

   b. In the Members section, click Add for either Users or Access Groups, as shown in Figure 38.

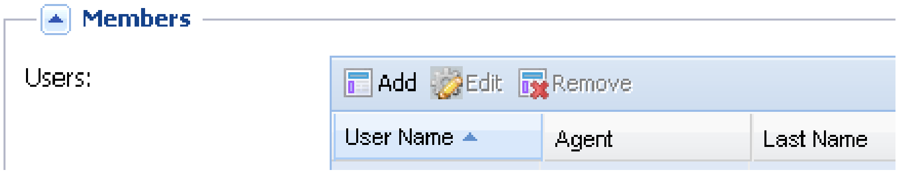
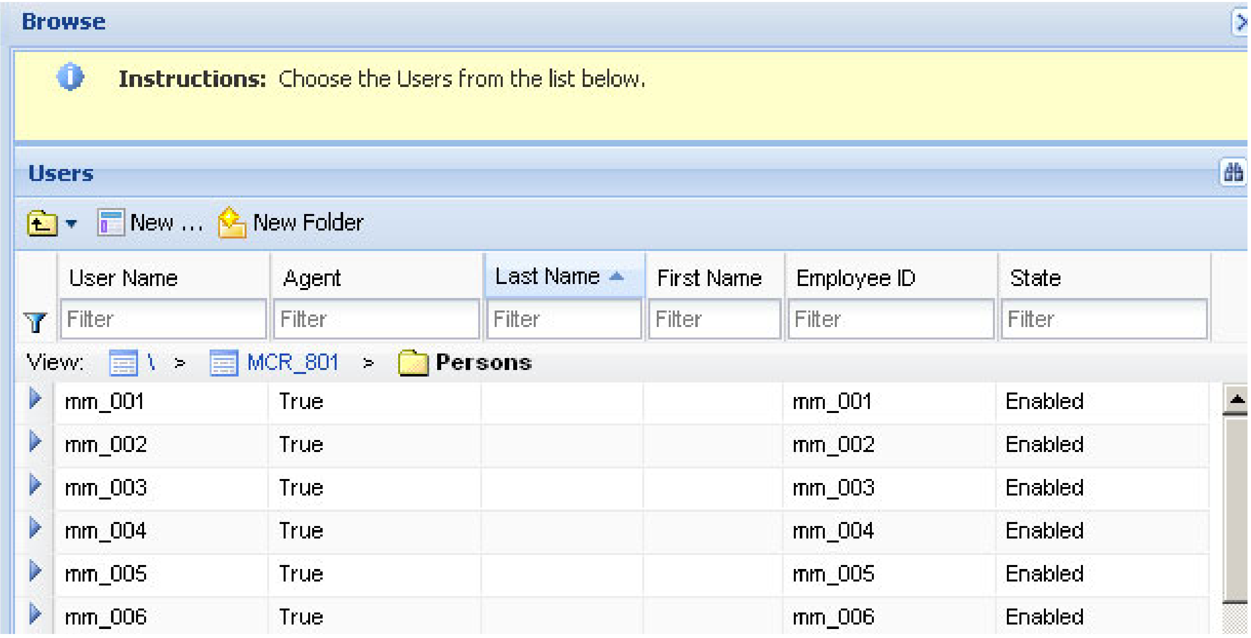


Figure 38: Configuration: Members

   c. In the resulting Browse window, select Users or Access Groups to add to the list of members. Figure 39 shows the Browse window for Users.



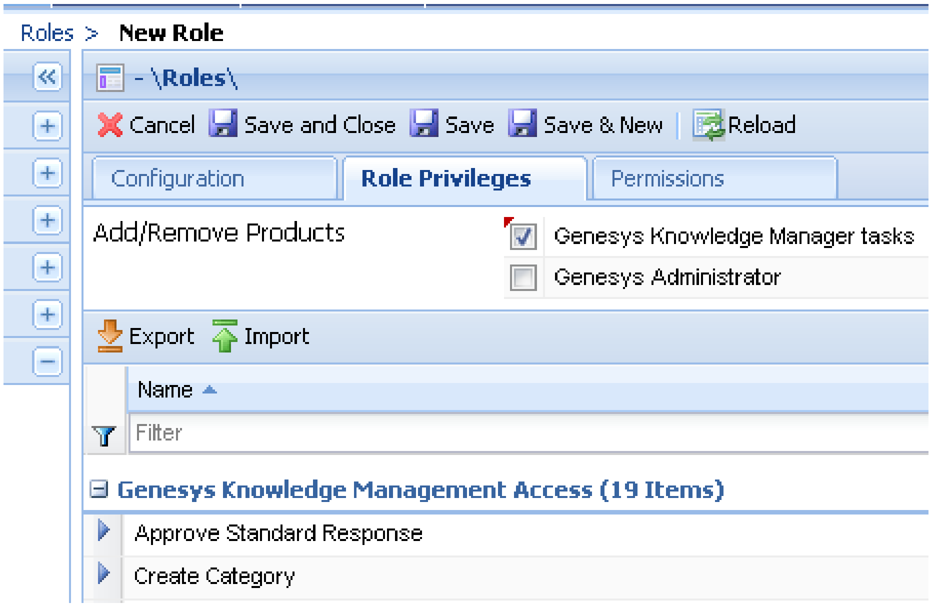
Instructions: Choose the Users from the list below.

View: \ > MCR_801 > **Persons**

User Name	Agent	Last Name	First Name	Employee ID	State
mm_001	True			mm_001	Enabled
mm_002	True			mm_002	Enabled
mm_003	True			mm_003	Enabled
mm_004	True			mm_004	Enabled
mm_005	True			mm_005	Enabled
mm_006	True			mm_006	Enabled

Figure 39: Browse for Users

3. On the `Role Privileges` tab, select the `Genesys Knowledge Manager tasks` box, as shown in Figure 40. If this box does not appear, you must import the metadata template into your Knowledge Manager Application.



Add/Remove Products

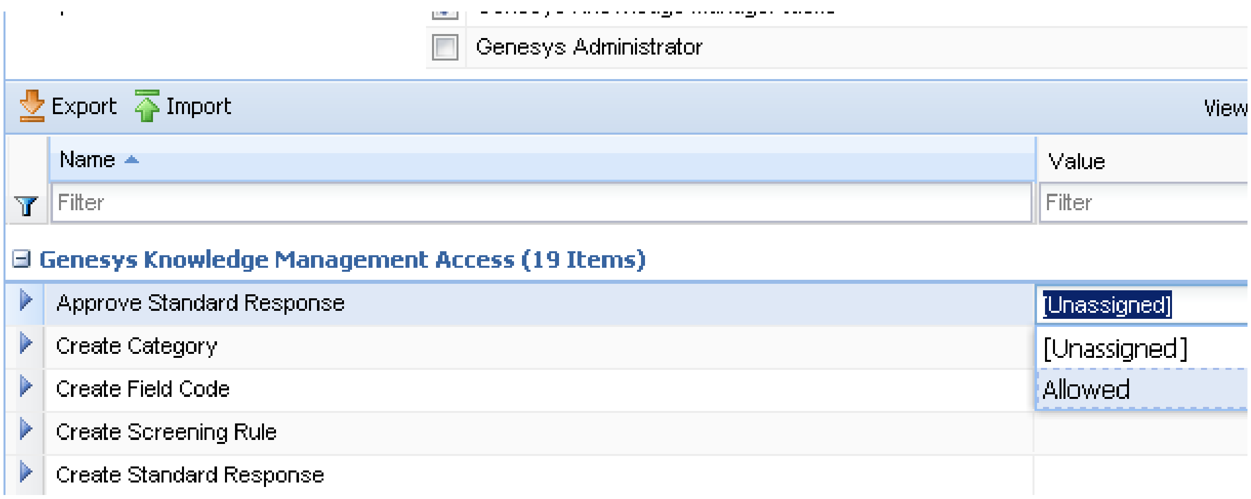
☑ Genesys Knowledge Manager tasks
☐ Genesys Administrator

Genesys Knowledge Management Access (19 Items)

- Approve Standard Response
- Create Category

Figure 40: Knowledge Management Tasks

4. A list of tasks appears. To enable this new role to perform one of them, select the task and change its value to `Allowed`, as shown in Figure 41.



Genesys Knowledge Management Access (19 Items)

Name	Value
Approve Standard Response	[Unassigned]
Create Category	[Unassigned] Allowed
Create Field Code	
Create Screening Rule	
Create Standard Response	

Figure 41: Assigning a Task

> **Note:** The value `[Unassigned]` means that this Role cannot perform this task. But if multiple Roles are assigned to the same User or Access Group, this setting is overridden if another Role sets this privilege as `Allowed`.

5. On the `Permissions` tab, add Users or Access Groups, as described in Genesys Administrator Help.

End of procedure

Table 6 lists all of the tasks provided for Knowledge Manager and provides an example of allocating task permissions among four different Roles called Manager, Administrator, Supervisor, and Agent. This is only an example; you can define any number of roles, with any names.

Table 6: Example Set of Roles

Task	Manager	Administrator	Supervisor	Agent
Create Standard Response	X	X		
View Standard Response	X	X	X	X
Modify Standard Response	X	X	X	
Delete Standard Response	X	X		
Approve Standard Response	X			
Create Category	X	X		
Modify Categories	X	X	X	

Table 6: Example Set of Roles (Continued)

Task	Manager	Administrator	Supervisor	Agent
Delete Categories	X	X		
Create Screening Rule	X	X		
Modify Screening Rule	X	X	X	
Delete Screening Rule	X	X		
Create Field Code	X	X		
Modify Field Code	X	X	X	
Delete Field Code	X	X		
Create Training Object	X	X		
Modify Training Object	X	X	X	
Delete Training Object	X	X		
Import Knowledge Management Objects	X			
Export Knowledge Management Objects	X			

Genesys

Chapter

2 Genesys Knowledge Management: Content Analyzer

This chapter describes using the Genesys Content Analyzer option for Knowledge Manager. It covers these topics:

Overview

Genesys Content Analyzer is an option to Genesys E-mail, requiring an additional license. It adds natural language processing technology to Genesys Knowledge Management.

See the “Overview” section in Chapter 1, “Genesys Knowledge Management: Basics,” on page 19 for an outline of the structure of the Knowledge Manager user interface. Many of the terms and concepts relevant to Genesys Content Analyzer are also defined in the glossary, which begins on page 353.

Models Genesys Content Analyzer applies a classification model—a statistical representation of a category tree—to an incoming interaction and produces a list of the categories that the interaction is most likely to belong to. Each likely category is assigned a percentage rating indicating the probability that the interaction belongs to this category.

Note: Classification, like screening (see “Screening Rules” on page 59) can operate on any interaction that has text somehow associated with it, whether as the body of the interaction (e-mail, chat), or otherwise (as user data, for example). In practice, it is expected that most interactions that are screened or classified will be e-mail messages; therefore this Guide uses the terms *e-mail* and *message* interchangeably to refer to these interactions. In fact whatever is said here about e-mail applies to any interaction that has associated text.

Training Objects

The process of creating a model is called *training.* Training operates on a *training object,* which is a category tree plus a set of text objects, with each text object assigned to one category in the tree.

Import and Export

You can import and export training objects and models. This is also a means of copying training objects. See “Importing and Exporting” on page 78 of the previous chapter.

Components

Genesys Content Analyzer does not have components as such. Rather, it adds functionality to the components of Genesys Knowledge Management:

- It activates Training Server, which has no function in the basic Genesys Knowledge Management but is required for training models.
- It enables Classification Server to categorize incoming interactions using models.
- It enables Knowledge Manager to control the creation of training objects, classification models, and FAQ objects.

Training

Training consists of the following steps, described in this section:

1. Create a training object.
2. Schedule and run the training.

All of these actions are performed by Training Server according to settings you provide using Knowledge Manager, which also displays the results.

Creating a Training Object

A training object combines a category tree and a set of text objects, with each text object assigned to one category in the tree (categorized). The text objects are typically e-mails, but you may choose to have the set of text objects also include the standard responses associated with the category tree.

Training scans the text objects and forms a statistical model of the words and phrases that tend to occur in each category. This is why you may want to have the training scan standard responses: they are very likely to include many words and phrases diagnostic of their category. This is also the reason that more text objects is better: it increases the sample size, which increases the accuracy of the model.

There are five possible sources of categorized text objects:

- E-mails that agents have assigned to categories, as described in this section
- Text objects (in the form of e-mails) that you create in Knowledge Manager, as described in "Creating new e-mails manually" on page 97
- Standard responses, as described in "Adding standard responses or other training objects" on page 99
- Other training objects, as described in "Adding standard responses or other training objects" on page 99
- Uncategorized e-mails that you categorize using the TO Data Analyzer, as described in "Adding Uncategorized E-Mails Using the TO Data Analyzer" on page 101

Once you have a category tree or trees (see "Creating a Category Tree" on page 25), you can create a training object.

Procedure:
Creating a new training object

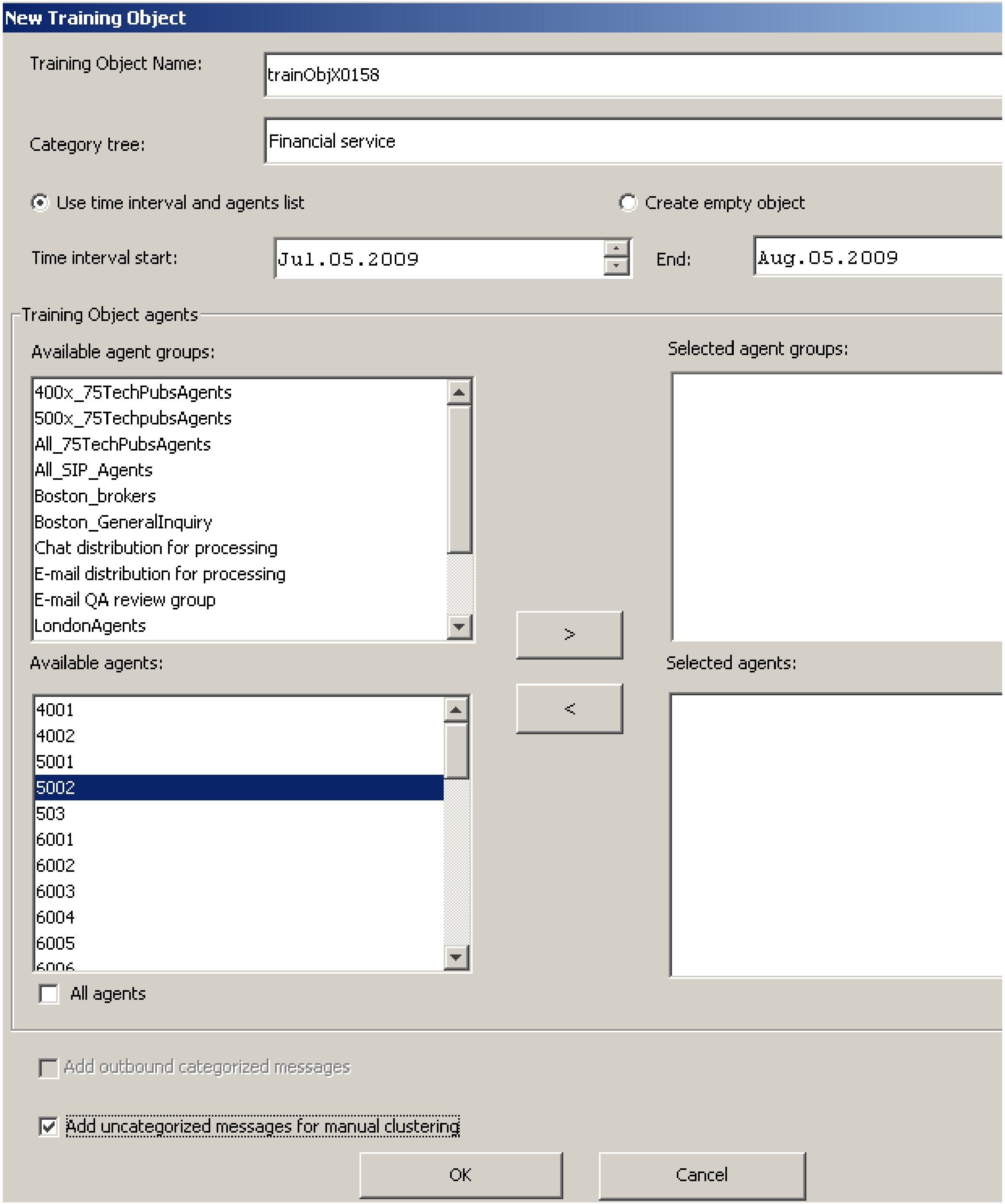
Purpose: To create an object that can be used to produce a classification model.

Prerequisites

- A category tree

Start of procedure

1. In Knowledge Manager, select a tenant and language. This determines which category trees are available. See "Notes on Language" on page 121 for some considerations relevant to choosing a language.
2. On the Training tab, select `File > New`, click the `New` icon, or right-click anywhere and select `New Training Object` on the shortcut menu. The `New Training Object` dialog box appears, as shown in Figure 42.



New Training Object

Training Object Name: trainObjX0158

Category tree: Financial service

● Use time interval and agents list ○ Create empty object

Time interval start: Jul.05.2009 End: Aug.05.2009

Training Object agents

Available agent groups:

400x_75TechPubsAgents
500x_75TechpubsAgents
All_75TechPubsAgents
All_SIP_Agents
Boston_brokers
Boston_GeneralInquiry
Chat distribution for processing
E-mail distribution for processing
E-mail QA review group
LondonAgents

Selected agent groups:

Available agents:

4001
4002
5001
5002
503
6001
6002
6003
6004
6005
6006

Selected agents:

□ All agents

□ Add outbound categorized messages

☑ Add uncategorized messages for manual clustering

Figure 42: New Training Object Dialog Box

3. Enter a name for the object (see page 26 in Chapter 2 for restrictions on the names of Knowledge Manager objects) and select a category structure (category tree).

4. You now have two alternatives, depending on whether you already have a collection of e-mails that have been categorized:

   a. If you do not already have a collection of categorized e-mails, select the `Create empty object` radio button.

   b. If you already have categorized e-mails, the categories were assigned to each e-mail by the agent who handled the interaction. In adding categorized e-mails to a training object, you can simply add all e-mails categorized by any agent at any time, or you can make the following adjustments:

      - Use only the e-mails that were categorized during a certain time period. You may want to exclude earlier times when agents were still learning to use the category system. To do this, set dates in the `Time interval start` and `End` fields.

Note: In defining the time interval, be careful to avoid these common errors: (1) identical start and end dates, resulting in zero e-mails in the training object; (2) too large a time span, which can result in omission of the categorization of the latest e-mails in the set; (3) time span of only a few days, which may lead to omission of the results for the last date.

      - Use only the e-mails that were categorized by selected agents or groups (some agents or groups may be especially reliable in their choice of categories). To do this, use the > and < buttons to move agent names from the `Available agent groups` window or `Available agents` window to the `Selected agent groups` window or the `Selected agents` window.

Notes: Agent names appear in alphabetical order.

For agent groups to appear in the `Available agent groups` box, they must be configured in the Tenant that you are creating this training object in.

To move individual agents, you must first clear the `All agents` check box.

There is a limitation on the number of SQL request parameters when creating a new training object. This becomes relevant when you are dealing with a large number of agents: for example, you will need a correspondingly large number of instances of the parameter `Interaction.OwnerId = xxxxxx`. The limitation varies with the RDBMS you are using, but it is approximately 2,100.

5. The `Add outbound categorized messages` check box is enabled only if Knowledge Manager is operating with UCS 7.5. See further explanation below.

6. Select the `Add uncategorized messages for manual clustering` check box if you want to add uncategorized e-mails to the root category. You can then use the TO Data Analyzer to categorize them, as described in “Adding Uncategorized E-Mails Using the TO Data Analyzer” on page 101.

   Note: The uncategorized e-mails added by this step are from the time interval and/or agents that you designated in the previous step. If you made no designations, the uncategorized e-mails are from your entire UCS database.

End of procedure

Next Steps

- If you are operating Knowledge Manager with UCS 7.5, review “Add Outbound Messages” immediately below.
- To add more text objects, continue with “Adding More Text Objects to the Training Object” on page 97 and “Adding Uncategorized E-Mails Using the TO Data Analyzer” on page 101.
- To proceed with training, continue with “Schedule Training” on page 114.

Add Outbound Messages

The `Add outbound categorized messages` check box, shown in Figure 42 on page 92, behaves differently depending on the version of UCS that Knowledge Manager is operating with.

- If Knowledge Manager is operating with UCS 7.6 or later, the checkbox is disabled. The new training object includes both of the following (as delineated by any time interval and agent list that you set):
  - Categorized inbound e-mails
  - Uncategorized inbound e-mails whose parent is a categorized outbound e-mail.
- If Knowledge Manager is operating with UCS 7.5, the check box is enabled, and has the following effects:
  - Not selected (the default): The training object includes only categorized messages that are inbound and are first in their thread.
  - Selected: The training object includes all categorized messages, both inbound and outbound.

Selecting this check box may be helpful in a situation in which agents do not categorize inbound e-mails, but do categorize the outbound e-mails that they generate in reply. In such a situation you may have an unacceptably small number of categorized e-mails unless you add categorized outbound e-mails. However, note that selecting this check box adds all outbound categorized messages, and those outbound messages that are replies may have content that is not very relevant to the category of the original e-mail.

Copying a Training Object

There are two ways to copy a training object: as an exact copy, or with selected text filtered out.

Procedure:
Making an exact copy of a training object

Summary

To make an exact copy of a training object, first export the training object, then import it.

Start of procedure

1. In the Export dialog box:
   a. Select the category tree.
   b. Select the training object.
2. When importing, select new database IDs (see "Importing Knowledge Manager objects" on page 81). Knowledge Manager asks you to modify the name of the category tree being imported.

End of procedure

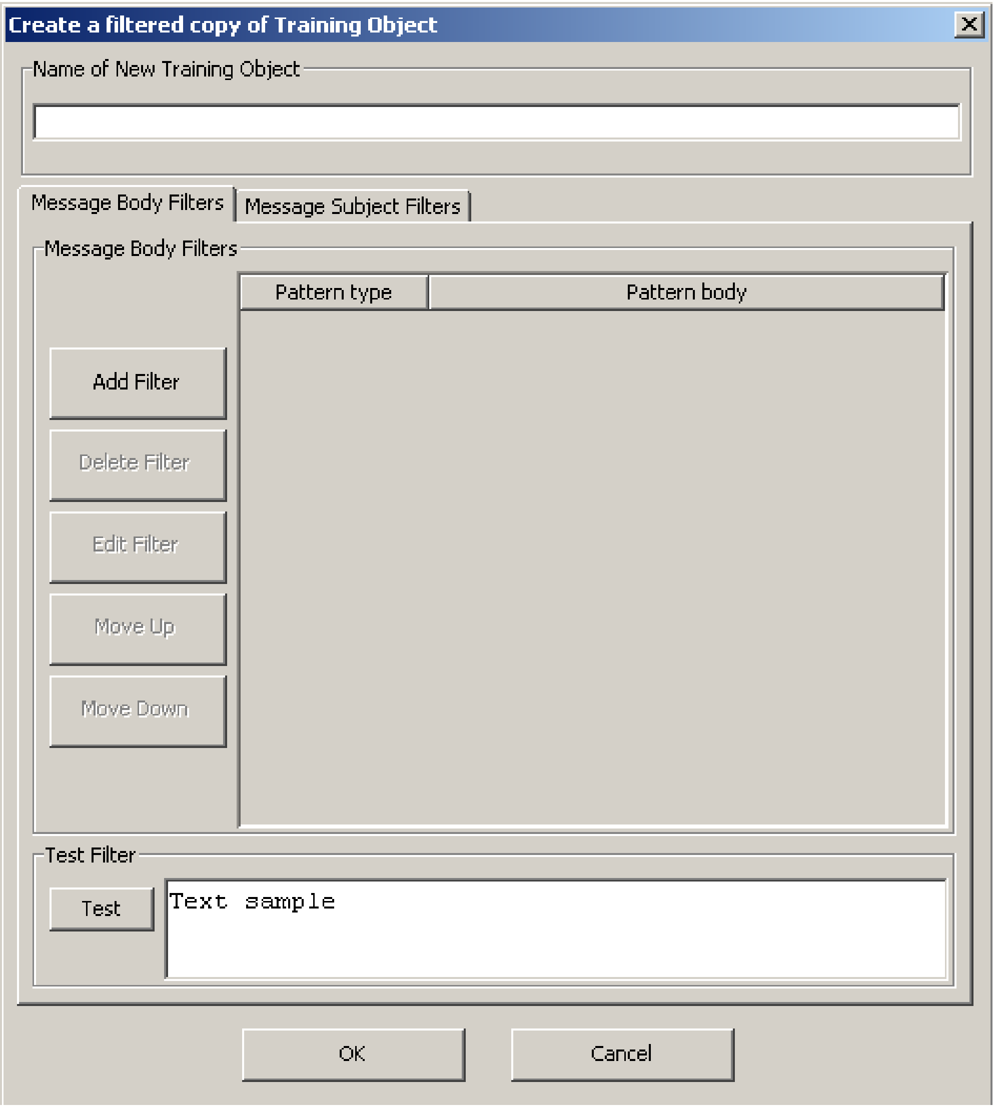
Procedure:
Making a filtered copy of a training object

Summary

To make a copy that filters out text that you specify, proceed as follows.

Start of procedure

1. On the left-hand pane of the `Training` tab, select the root `Training Objects` node.
2. On the right-hand pane, highlight and right-click the training object that you want to make a filtered copy of.
3. Select `Create filtered copy of Training Object` from the resulting drop-down list.



Text sample

Figure 43: Create a Filtered Copy of Training Object

4. On the resulting `Create filtered copy of Training Object` dialog box, shown in Figure 43, proceed as follows:
   - **a.** Enter a name for the new filtered copy of the training object.
   - **b.** Create filters, and test them if you wish.

     This works identically with the `Text Preprocessing` tab of the `Model Training Schedule` dialog box, which is described on page 117, except that here you create separate filters for the body and subject of messages, using the `Message Body Filters` and `Message Subject Filters` tabs.
   - **c.** Click OK to save the new filtered copy of the training object.

End of procedure

Adding More Text Objects to the Training Object

Regardless of whether you have previously-categorized e-mails that you included in your training object, you can add more text objects to it in the following ways:

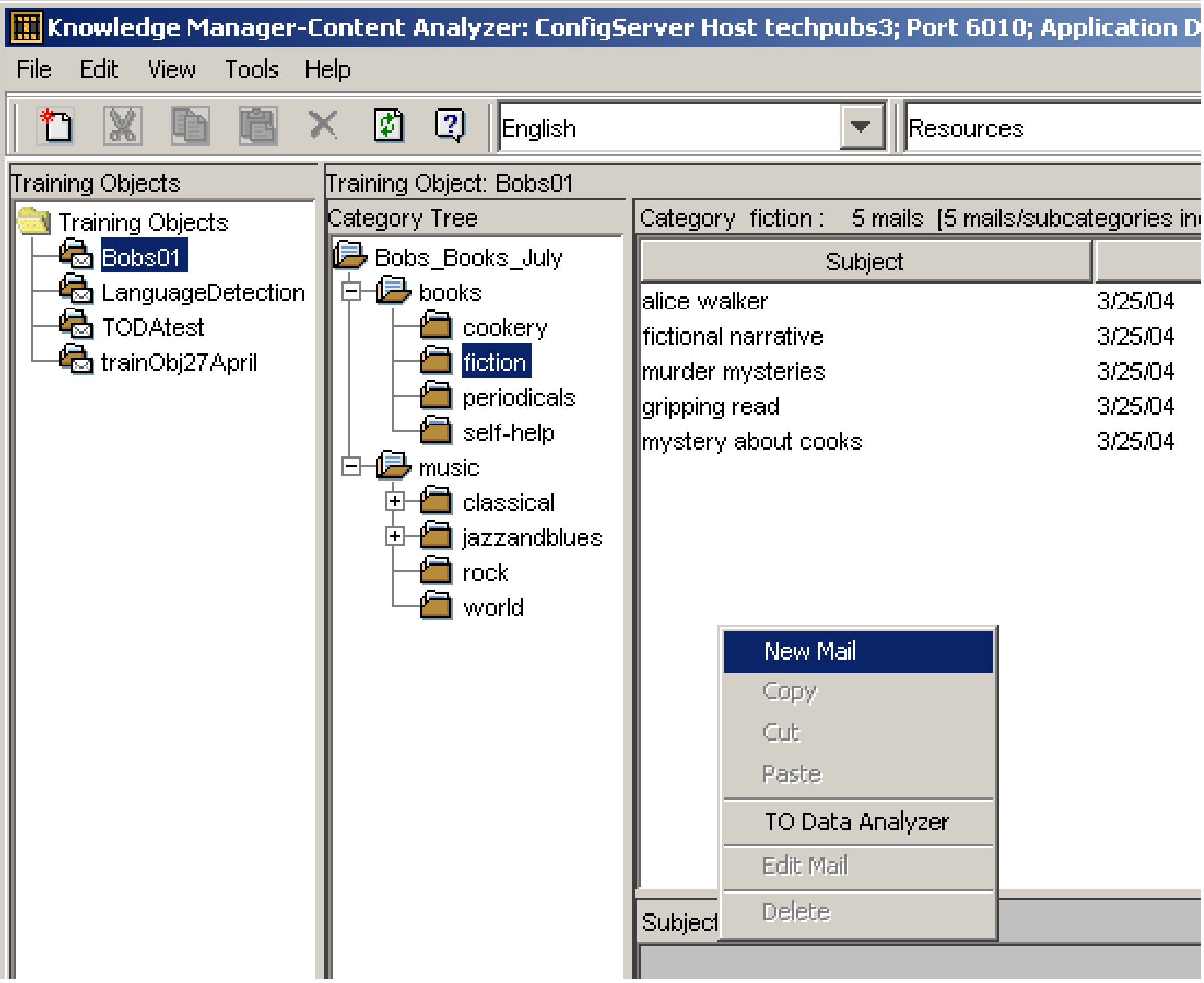
- Create new e-mails manually, as described on page 97.
- Add standard responses or other training objects, as described on page 99.
- Add uncategorized e-mails using the TO Data Analyzer utility, as described on page 101.

Procedure: Creating new e-mails manually

Purpose: To increase the accuracy of a model produced by a training object by adding new manually-created e-mails to the training object.

Start of procedure

1. Close the `New Training Object` dialog box. On the left-hand pane of the `Training` tab, select the training object. This produces a three-pane view, shown in Figure 44.



Subject	
alice walker	3/25/04
fictional narrative	3/25/04
murder mysteries	3/25/04
gripping read	3/25/04
mystery about cooks	3/25/04

Figure 44: Add E-mail to a Training Object

2. On the center pane, select the category that you want to add an e-mail to. Figure 44 shows a user about to add a new e-mail to the `fiction` category.

3. On the right-hand pane, right-click and select `New Mail` from the shortcut menu. The `Mail Editor` opens, as shown in Figure 45.

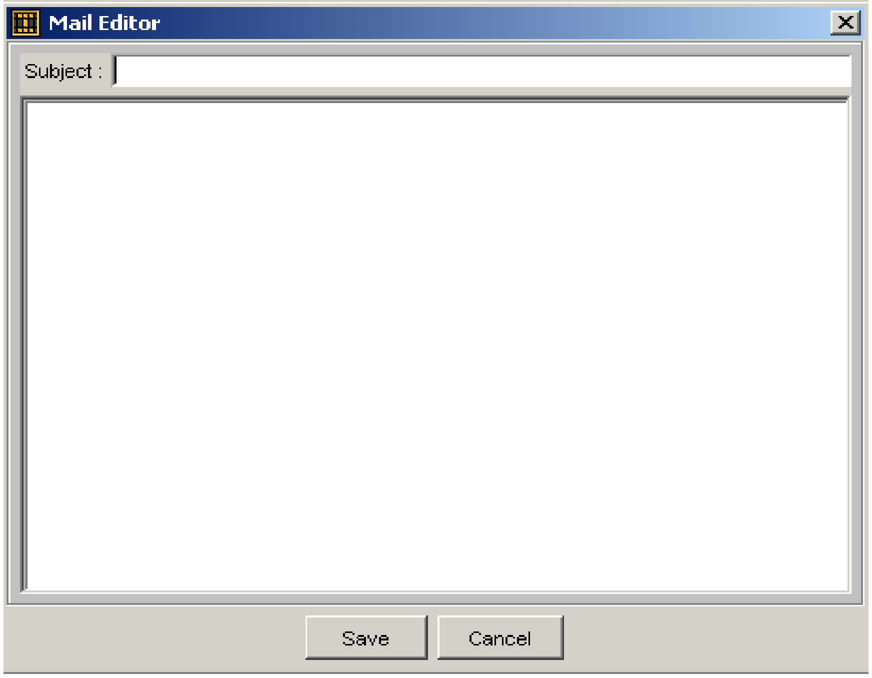


Figure 45: Mail Editor

4. Enter text into the `Subject` and `Body` fields, then click `Save`.

   E-mails that you add in this way are stored as being handled by the agent called `default`.

> **Note:** You can also edit existing e-mails by selecting an e-mail in the right-hand pane, right-clicking, and selecting `Edit Mail`.

End of procedure

Procedure:
Adding standard responses or other training objects

Purpose: To increase the accuracy of a model produced by a training object by adding either standard responses or other training objects to the training object.

Start of procedure

1. To add standard responses, go to the left-hand pane of the `Training` tab and select the root `Training Objects` node. This produces a two-pane view, as shown in Figure 46 on page 100.

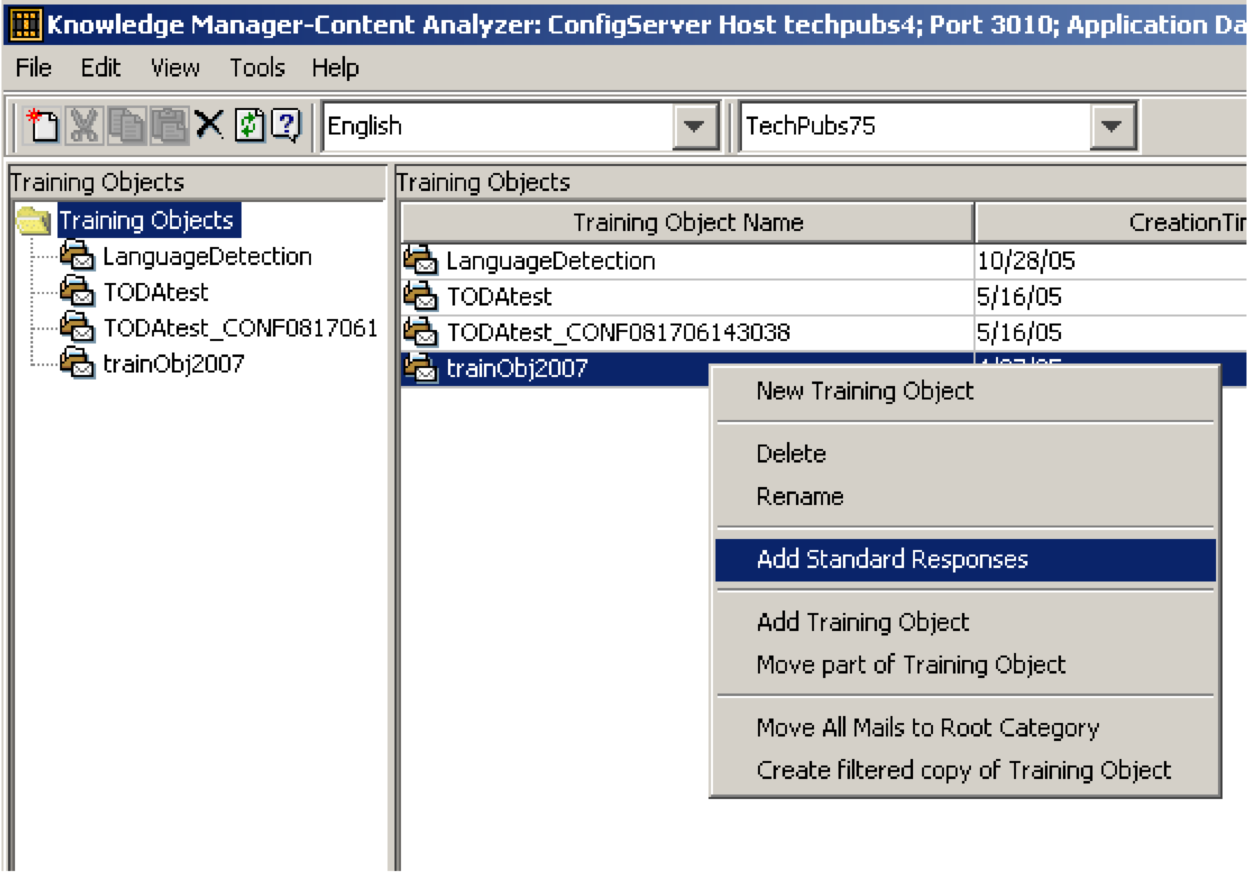

Knowledge Manager-Content Analyzer: ConfigServer Host techpubs4; Port 3010; Application Da
File Edit View Tools Help
English
TechPubs75
Training Objects
Training Objects
LanguageDetection
TODAtest
TODAtest_CONF0817061
trainObj2007
Training Objects
Training Object Name
CreationTir
LanguageDetection 10/28/05
TODAtest 5/16/05
TODAtest_CONF081706143038 5/16/05
trainObj2007
New Training Object
Delete
Rename
Add Standard Responses
Add Training Object
Move part of Training Object
Move All Mails to Root Category
Create filtered copy of Training Object

Figure 46: Add Standard Responses to a Training Object

2. Right-click a training object on the right-hand pane, then choose `Add Standard Responses` from the shortcut menu. Figure 46 shows a user about to add standard responses to the `trainObj2007` training object.
3. To add a training object, repeat Steps 1 and 2, but choose `Add Training Object` from the shortcut menu.

Note: You can add a training object to another training object only if both training objects use the same category tree.

End of procedure

Adding Uncategorized E-Mails Using the TO Data Analyzer

If you have a sizeable database of uncategorized e-mails, the TO (Training Object) Data Analyzer can help you to:

- Group the uncategorized e-mails and build a category tree for them.
- Assign the uncategorized e-mails to existing categories.

In the most basic terms, the TO Data Analyzer does the following:

1. It shows you all uncategorized e-mails one at a time, in an order determined by criteria that you set (if you set no criteria, it shows them in the order in which they were created).
2. As each e-mail displays, you choose whether to include it in a holding area.
3. You then assign the e-mails in the holding area to a category. This can be an existing category or a new one that you create.

The TO Data Analyzer has the tabs listed in Table 7.

Table 7: TO Data Analyzer Tabs

Tab	What it Does	Location of Description
Main	Displays uncategorized e-mails one at a time for you to categorize	"Adding e-mails to a category on the Main tab" on this page
Search Criteria	Sets the criterion determining the order that uncategorized e-mails are displayed in	"Search Criteria Tab" on page 109
Indexing	Displays information on cooccurrence of words in uncategorized e-mails	"Indexing Tab" on page 110

Procedure: Adding e-mails to a category on the Main tab

Purpose: To use the TO Data Analyzer to search for uncategorized e-mails that are a likely fit for a selected category.

Prerequisites

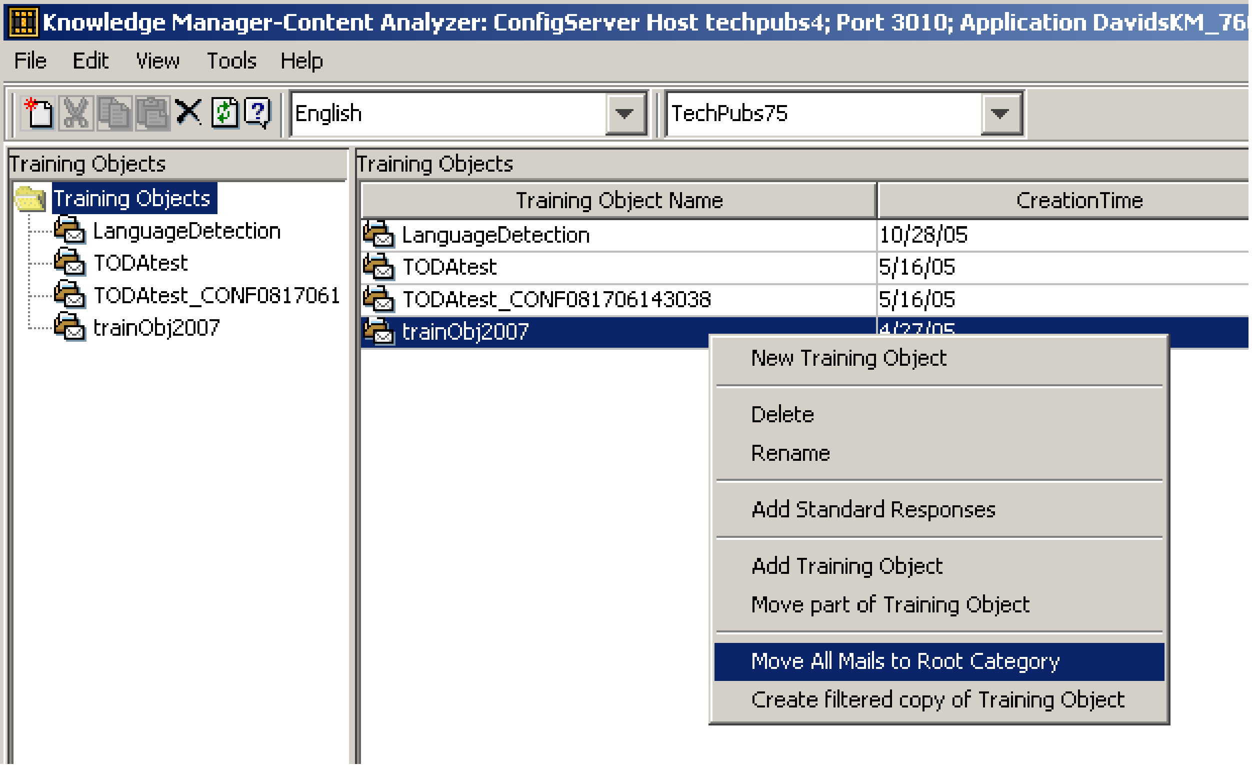
For uncategorized e-mails to be available to the TO Data Analyzer, they must be associated with the root category of the category tree in the training object. You perform this association in either of the following ways:

- When you create the training object, select the `Add uncategorized messages for manual clustering` check box as described in Step 6 of "Creating a Training Object" on page 94.

- In an existing training object, you can move all e-mails to the root category (before you do this it is advisable to make a backup copy of the training object; see “Copying a Training Object” on page 95):

  a. On the left-hand pane, select the root `Training Objects` node to give the two-pane view.

  b. On the right-hand pane, right-click a training object and select `Move All Mails to Root Category,` as shown in Figure 47.

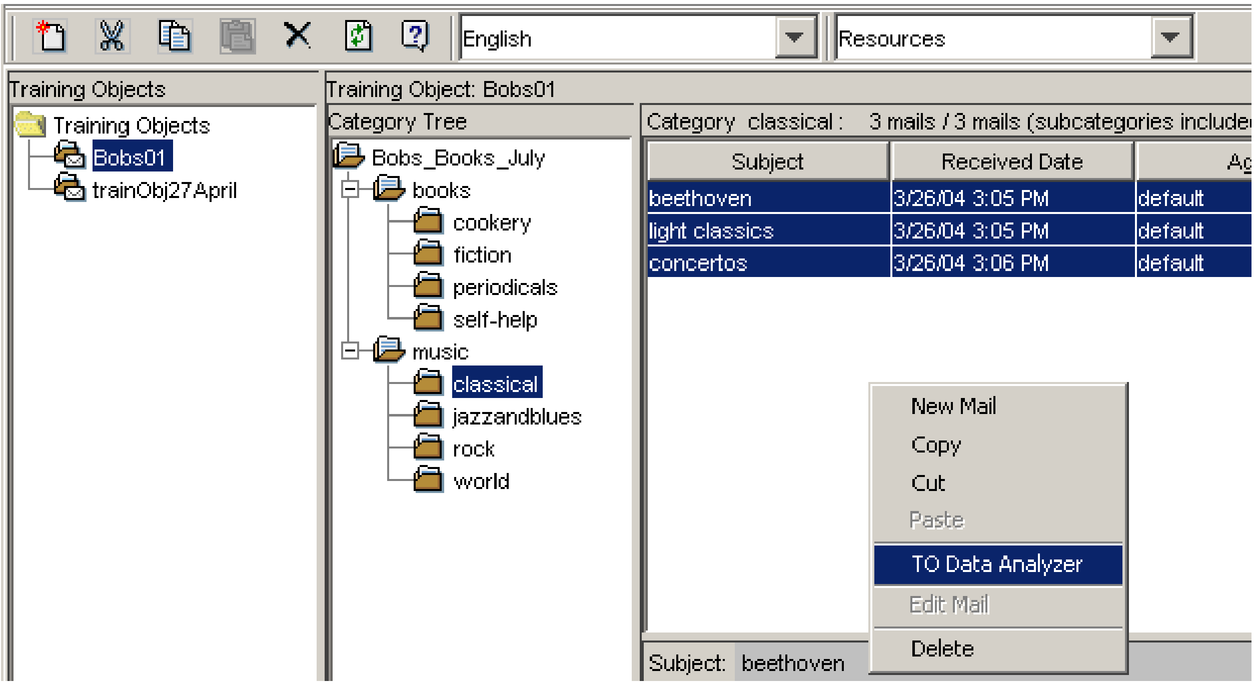


Training Object Name	CreationTime
LanguageDetection	10/28/05
TODAtest	5/16/05
TODAtest_CONF081706143038	5/16/05
trainObj2007	4/27/05

Figure 47: Move All E-Mails to the Root Category

Start of procedure

1. On the `Training` tab, select a training object.
2. On the middle pane, select a category that you want to add e-mails to.
3. On the rightmost pane, select the e-mails already assigned to this category.
4. Still on the rightmost pane, right-click and select `TO Data Analyzer,` as shown in Figure 48.



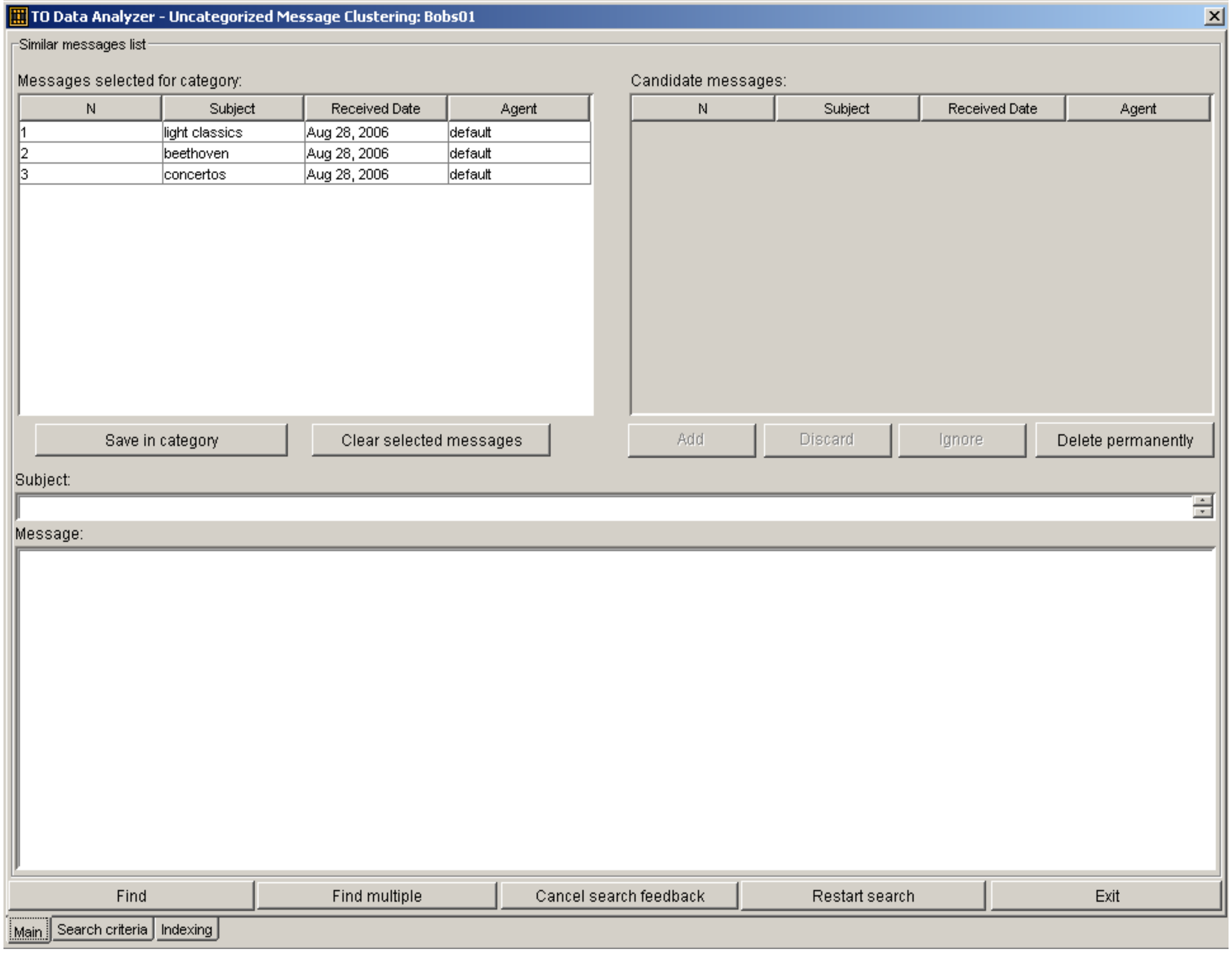
Category classical : 3 mails / 3 mails (subcategories include

Subject	Received Date	Ag
beethoven	3/26/04 3:05 PM	default
light classics	3/26/04 3:05 PM	default
concertos	3/26/04 3:06 PM	default

Subject: beethoven

Figure 48: Start the Training Object Data Analyzer

5. The `Uncategorized Message Clustering` dialog box appears with its `Main` tab showing, as shown in Figure 49.



TO Data Analyzer - Uncategorized Message Clustering: Bobs01

Similar messages list

Messages selected for category:

N	Subject	Received Date	Agent
1	light classics	Aug 28, 2006	default
2	beethoven	Aug 28, 2006	default
3	concertos	Aug 28, 2006	default

Candidate messages:

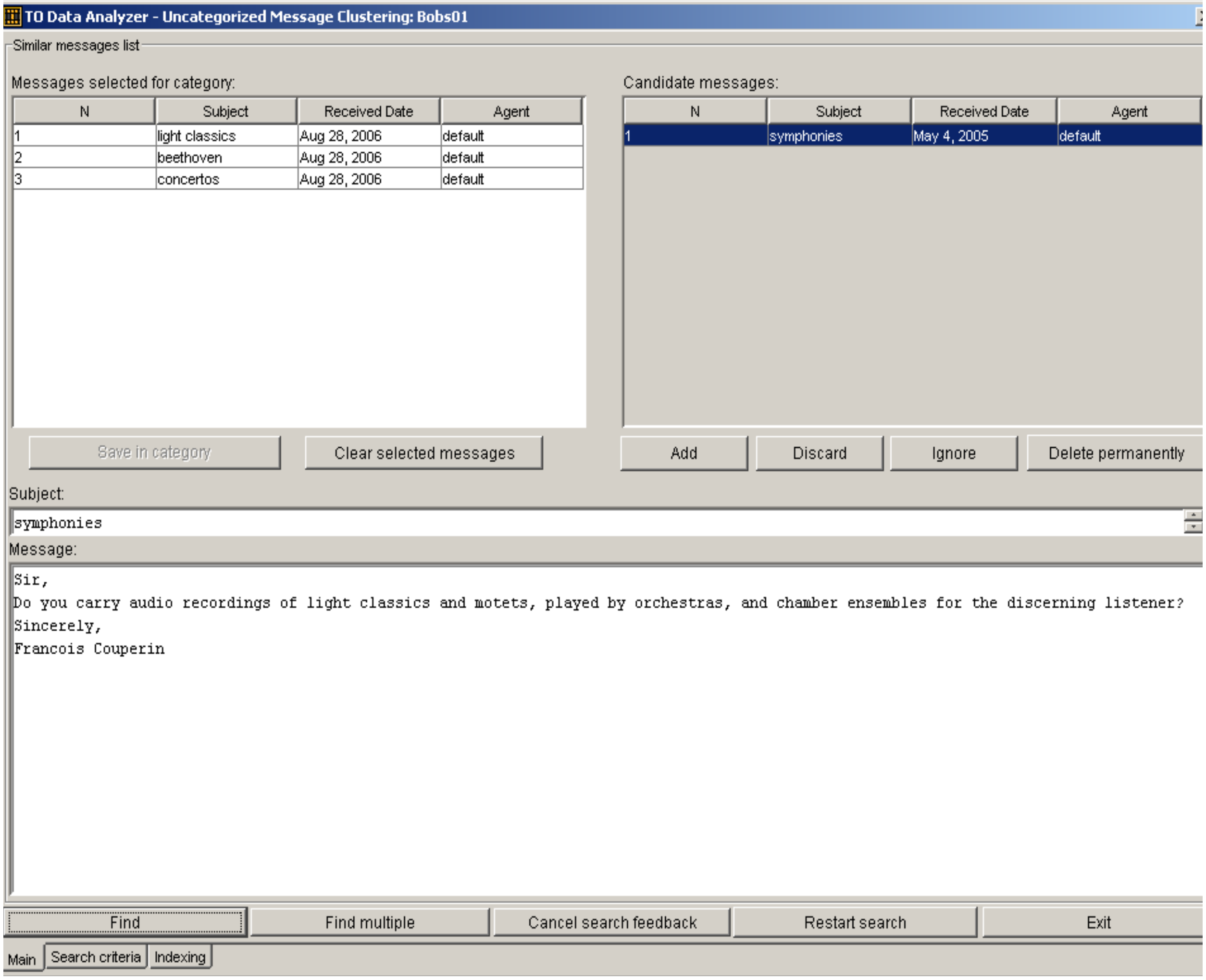
N	Subject	Received Date	Agent

Figure 49: Uncategorized Message Clustering

Note that the e-mails that you selected in Step 3 appear in the `Similar messages list`. These e-mails are now part of the criteria that TO Data Analyzer uses to choose e-mails to display.

6. Click `Find`. The system displays a candidate e-mail as shown in Figure 50:

- The `Candidate messages` box shows information about the e-mail. `N` is number, relevant only for the `Find multiple` command, described below.
- The `Subject` box shows the subject of the e-mail.
- The `Message` box shows the e-mail body.



Messages selected for category:

N	Subject	Received Date	Agent
1	light classics	Aug 28, 2006	default
2	beethoven	Aug 28, 2006	default
3	concertos	Aug 28, 2006	default

Candidate messages:

N	Subject	Received Date	Agent
1	symphonies	May 4, 2005	default

Subject:

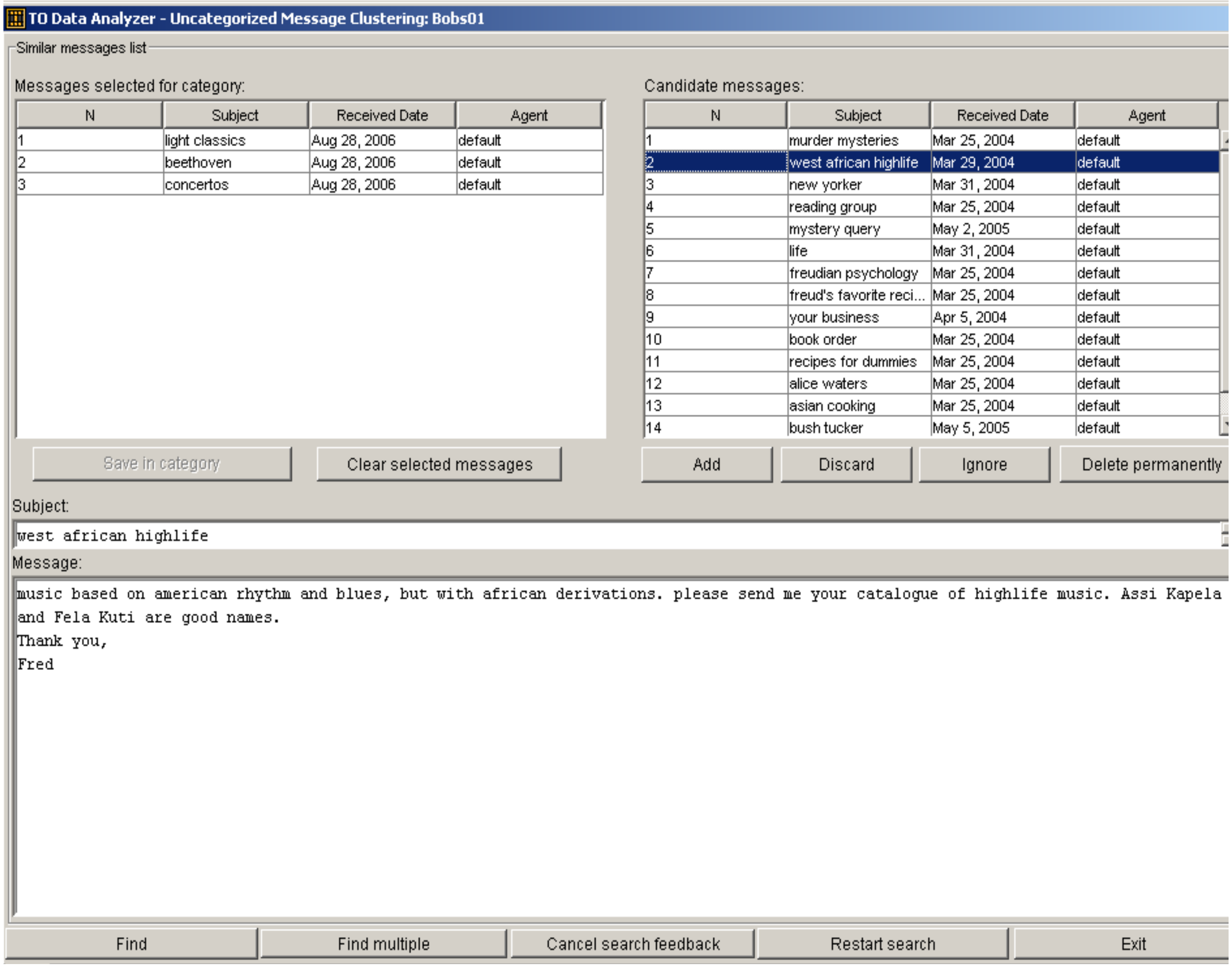
```
symphonies
```

Message:

```
Sir,
Do you carry audio recordings of light classics and motets, played by orchestras, and chamber ensembles for the discerning listener?
Sincerely,
Francois Couperin
```

Figure 50: Message Clustering: First Candidate

As an alternative to `Find`, click `Find multiple`. The system displays the 15 e-mails that are the best candidates for the category, as shown in Figure 51.



Messages selected for category:

N	Subject	Received Date	Agent
1	light classics	Aug 28, 2006	default
2	beethoven	Aug 28, 2006	default
3	concertos	Aug 28, 2006	default

Candidate messages:

N	Subject	Received Date	Agent
1	murder mysteries	Mar 25, 2004	default
2	west african highlife	Mar 29, 2004	default
3	new yorker	Mar 31, 2004	default
4	reading group	Mar 25, 2004	default
5	mystery query	May 2, 2005	default
6	life	Mar 31, 2004	default
7	freudian psychology	Mar 25, 2004	default
8	freud's favorite reci...	Mar 25, 2004	default
9	your business	Apr 5, 2004	default
10	book order	Mar 25, 2004	default
11	recipes for dummies	Mar 25, 2004	default
12	alice waters	Mar 25, 2004	default
13	asian cooking	Mar 25, 2004	default
14	bush tucker	May 5, 2005	default

Subject:

```
west african highlife
```

Message:

```
music based on american rhythm and blues, but with african derivations. please send me your catalogue of highlife music. Assi Kapela
and Fela Kuti are good names.
Thank you,
Fred
```

Figure 51: Message Clustering: Multiple Candidates

7. Use the buttons at the bottom of the dialog to select one of the following actions for this candidate (or candidates, if you clicked `Find multiple` and then used SHIFT-click or CTL-click to select multiple candidates):
   - `Add`—Add to the selected category and adjust the search accordingly (see "Search Criteria Tab" on page 109 for more explanation).
   - `Discard`—Omit from the selected category and adjust the search accordingly.
   - `Ignore`—Omit from the selected category but do not adjust the search.
   - `Delete permanently`—Remove from the training object permanently. This e-mail will not appear again.

Note: `Delete permanently` is useful for deleting garbage e-mails, that is, e-mails that you do not want to use anywhere in the training object. With `Discard` and `Ignore`, in contrast, the e-mail is no longer considered in the current search for the selected category, but it remains in the training object.

8. Click `Find` again, and the next candidate e-mail displays in the `Subject` and `Message` boxes. Decide whether to add, discard, or ignore this e-mail.
9. Continue in the same way, repeating Step 8. Knowledge Manager continues to display e-mails in the order determined by the criteria that you have set (see the next section, "Search Criteria Tab", for details on setting criteria). When there are no more uncategorized e-mails, `{ Mail not found }` displays in the `Message` box.
10. When you are done adding e-mails to the `Similar messages list`, click one of the following:
    - `Save in category`—This adds the e-mails to a category, as explained in the next step.
    - `Clear selected messages`—This clears all e-mails from the `Similar messages list`. You can start again by clicking `Find`.
11. If you click `Save in category`, the `Choose category` dialog box appears, as shown in Figure 52.

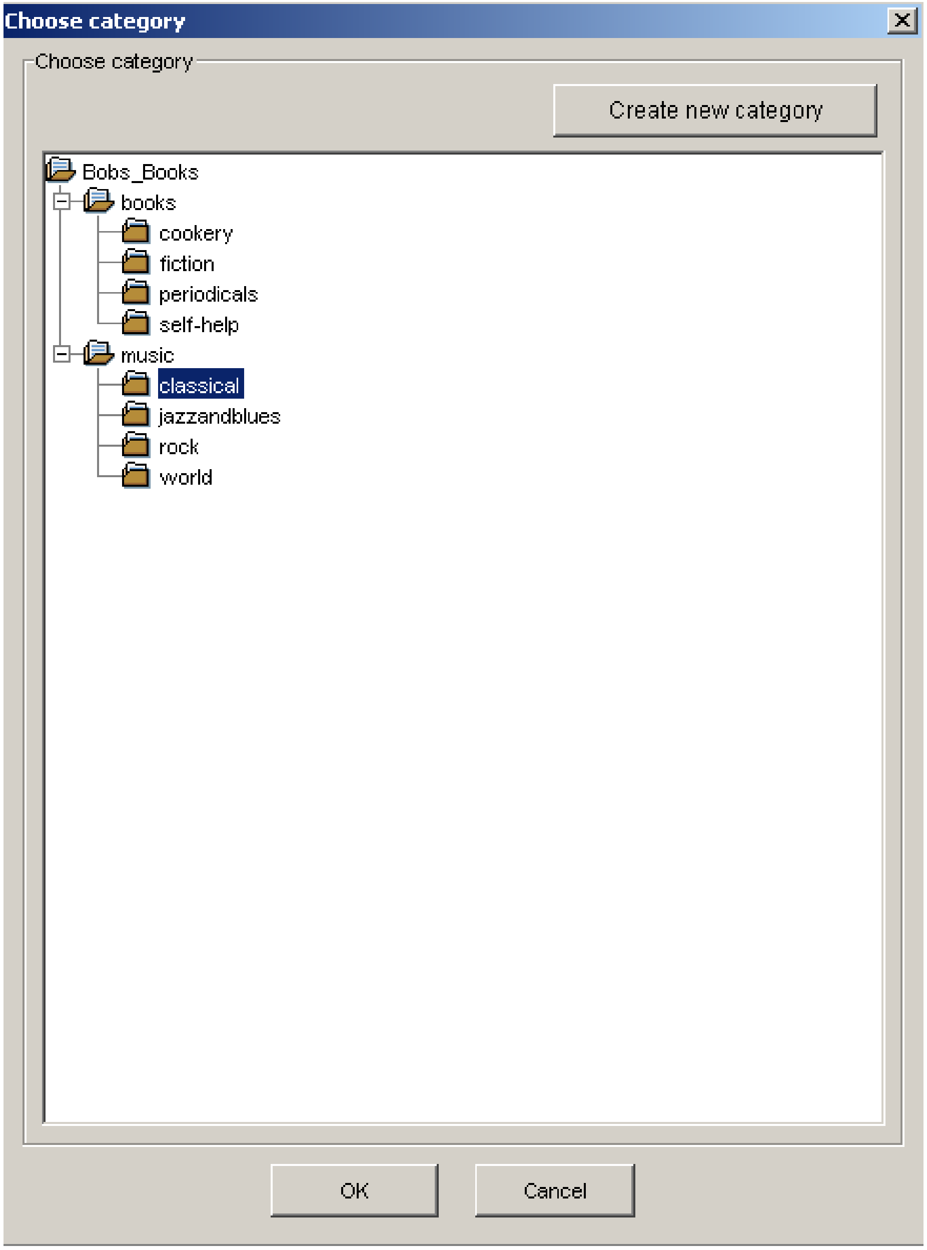


Figure 52: Choose Category Dialog Box

The `Choose category` dialog box displays the category tree that you are working with. If you selected a category in Step 2, that category is selected in this dialog box, but you can select another category (but not the root category). You can also click `Create new category` to add a new category which becomes a subcategory of the category selected in the `Choose category` dialog box. Click `OK` to add the e-mails in the `Similar messages list` to the selected category.

End of procedure

Next Steps

- Optionally, use the Search Criteria tab to specify the order that Knowledge Manager shows you the uncategorized e-mails (this page).
- Study the example of using the TO Data Analyzer on page 113.

Search Criteria Tab

When you click Find, Knowledge Manager shows you all uncategorized e-mails one at a time, in an order determined by criteria that you set using TO Data Analyzer's Search Criteria tab.

If Knowledge Manager finds that an e-mail is a good fit for the criteria, this section says that it *gives priority* to that e-mail.

Note: Knowledge Manager gives priority to e-mails that meet the criteria best. Those that are a bad fit for the criteria are not excluded; they are simply put later in the list.

If you set no criteria, Knowledge Manager displays the e-mails in the order in which they were created.

Similar Messages List

Knowledge Manager gives priority to e-mails that resemble the e-mails in the Similar messages list. E-mails move to the Similar messages list in two ways:

- You select them before right-clicking TO Data Analyzer, as in the procedure described in the previous section.
- You click Add when they display in the Uncategorized message clustering dialog box.

The previous section instructs you to select all of the e-mails in the category that you are interested in. Of course, if you think that some e-mails in the category are not a good fit, you should not bring them to the Similar messages list.

Text Length

Use the Min and Max boxes to set limits on the size (number of characters) of e-mails. One use of this is to exclude very long e-mails, which would take you too long to read in the Message box.

Include and Exclude Text

Enter text in these boxes to adjust the way that Knowledge Manager assigns priority. E-mails that include text that matches the Include text box receive higher priority. E-mails that include text that matches the Exclude text box receive lower priority. These boxes are especially useful when you are starting out with nothing in the `Similar messages list`.

What you enter in these boxes is literal text, not regular expressions.

Note: E-mails that include text that matches the `Exclude text` box are not, in fact, excluded. They are simply moved towards the end of the list.

Refining or Resetting the Search

Each time you click `Find`, Knowledge Manager presents the e-mail that best fits the criteria that you have set. It continues to do this until it has presented all of the uncategorized e-mails. Knowledge Manager keeps track of which e-mails it has presented. If you alter the criteria, it then presents the best fit *from among those it has not yet presented.* If you alter the criteria and want Knowledge Manager to start scanning the e-mails from the beginning again, you must click `Restart Search` before clicking `Find`.

It is important to understand that you alter the search criteria each time you click `Add` or `Discard` (this is in addition to the possibility of you changing the contents of the `Include text`, `Exclude text`, and `Text length` boxes). When you click `Add` or `Discard`, you confirm or reject Knowledge Manager's guess as to which e-mail best fits the criteria, and Knowledge Manager uses your confirmation or rejection to adjust the criteria.

If, after going through a number of candidate e-mails, you decide that you are on the wrong track, you can click `Restart Search`, and Knowledge Manager restarts its search from the beginning, using only the criteria supplied by the contents of the `Uncategorized message clustering` dialog box (similar messages, text length, include/exclude text), and discarding all of your preceding Add/Discard input. However, any e-mails that you have added to the `Similar messages list` remain there after you click `Restart Search`.

Indexing Tab

This tab, shown in Figure 53, displays information on cooccurrence patterns of words in uncategorized e-mails.

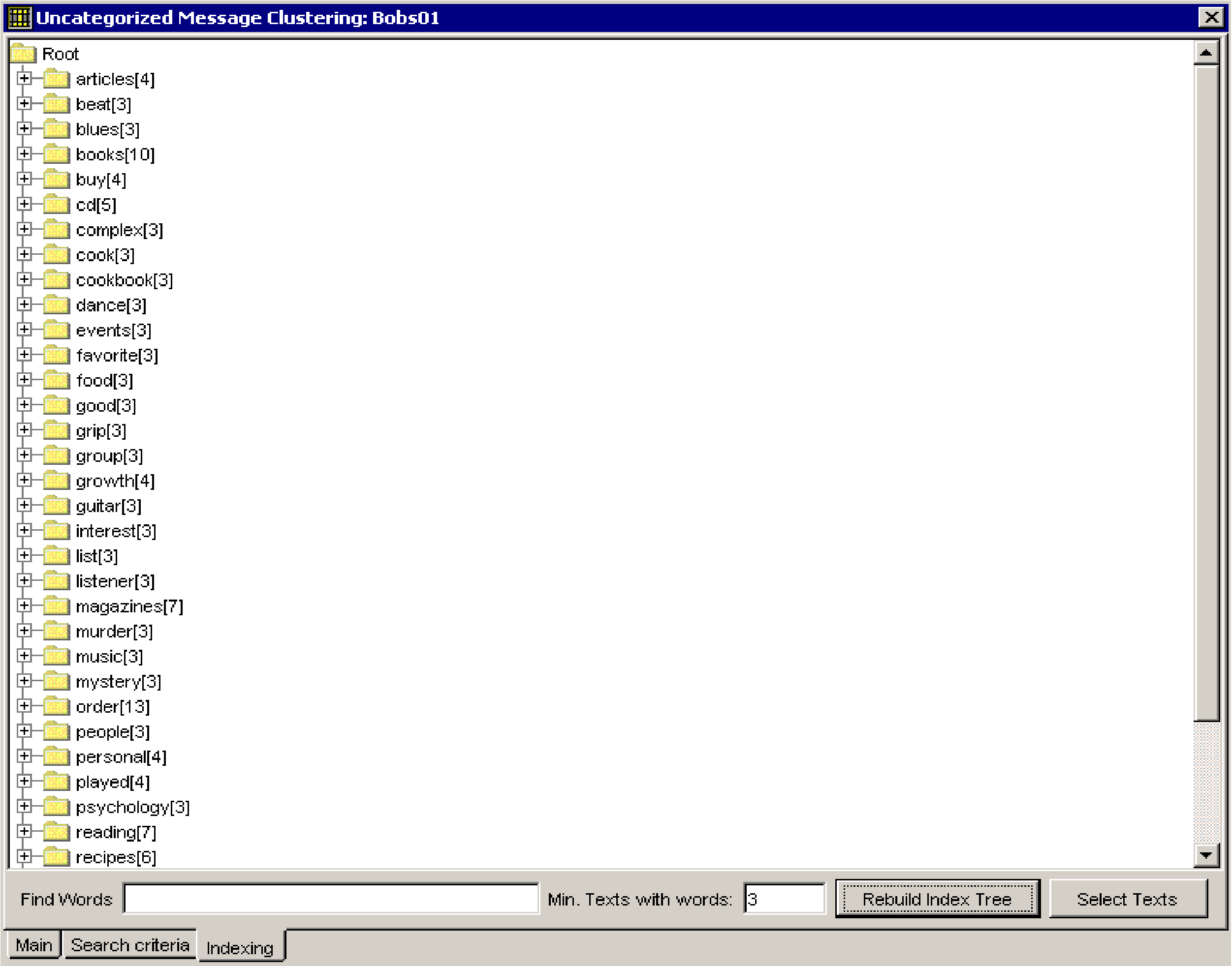
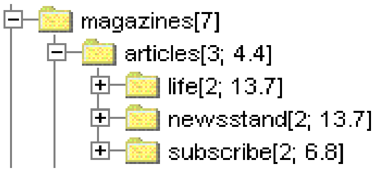


Figure 53: Indexing Tab

The tab displays, in tree form, a list of the words that occur in all uncategorized e-mails (except stop words; see page 123).

The index tree consists of folder icons, each labeled with a word, with the number of occurrences (number of e-mails it occurs in) in square brackets. These words can be called *head words.*

Each head word folder expands to a list of the words (also folders) that cooccur with the head word—that is, that occur together with the head word in one or more e-mails. Each cooccurring word is followed by square brackets containing two numbers: the number of e-mails this word occurs in, and a ratio. This ratio is the rate of occurrence with this head word divided by rate of occurrence in whole corpus. Figure 54 provides an example.



```
magazines[7]
  articles[3; 4.4]
    life[2; 13.7]
    newsstand[2; 13.7]
    subscribe[2; 6.8]
```

Figure 54: Indexing Tab Example

Among the information displayed in this example is the following:

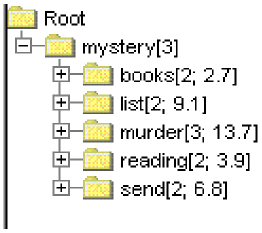
- *magazines* occurs in seven uncategorized e-mails
- *articles* occurs in three of those seven e-mails, which is 4.4 times as often as it occurs in the entire corpus of uncategorized e-mails.
- Of the three e-mails containing *magazines* and *articles,* two also contain *newsstand.* This is 13.7 times as often as *newsstand* occurs in the entire corpus.

This indicates that the words *articles* and *newsstand* are highly likely to occur together, which means e-mails that contain both words are good candidates for grouping together in a category. If you select *newsstand,* then click `Select texts,` the display switches to the `Main` tab, showing that all e-mails that contain *magazines, articles,* and *newsstand* have been put in the `Candidate messages list.`

At the bottom of the tab are the following:

- Two boxes for filtering the words that are displayed:
  - `Find Words`—Restrict the words displayed. More on this below.
  - `Min. Texts with words`—The word must occur in at least this number of e-mails to be displayed in the list.
- Two buttons that initiate actions:
  - `Rebuild Index Tree`—Rebuild the tree to apply the filters that you set in the `Find Words` and `Min. Texts with words` boxes.
  - `Select Texts`—Select a word in the index tree, then click this button to put all e-mails containing this word in the `Candidate messages` list.

Use the `Find words` box to restrict the words displayed. Enter a single word to display only that word and the words that occur with it. Enter multiple words to specify which cooccurring words to start the list with. Figure 55 shows the result of entering *mystery* in the `Find words` box, then clicking `Rebuild Index Tree.`



```
Root
  mystery[3]
    books[2; 2.7]
    list[2; 9.1]
    murder[3; 13.7]
    reading[2; 3.9]
    send[2; 6.8]
```

Figure 55: Find Words = "mystery"

Figure 56 shows the result of entering *mystery reading* in the `Find words` box: the index tree shows only the head word *mystery* and the cooccurring word *reading*.

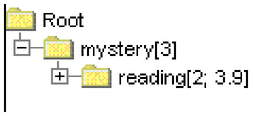


Figure 56: Find Words = "mystery reading"

Procedure: Example of use

Summary

This section presents an example of using the TO Data Analyzer to add a category and build subcategories for it.

Prerequisites

The example makes the following assumptions:

- Part of your business deals with DSL service.
- Your category tree does not represent this DSL service sector.
- You have a collection of uncategorized e-mails, some of which deal with DSL service.

Start of procedure

1. On the `Categories` tab, add a `DSL service` category.
2. Create a training object using this category tree.
3. On the `Training` tab, select the `DSL service` category and open the TO Data Analyzer.
4. In the `Include Text` box, enter `DSL`.
5. Click `Find` repeatedly, browsing through the uncategorized e-mails and looking for common themes.
6. As you do this, you find a number of e-mails inquiring about the status of a DSL service order. You decide they should go in a subcategory that you will call "DSL Shipping Status."
7. Add one of these e-mails to the `Similar messages list`.
8. Add the word `shipping` to the `Include Text` box to refine the criteria.
9. Click `Restart Search`, then `Find`. This starts the search from the beginning with the revised criteria.
10. Continue, clicking `Find`, then clicking `Add` for e-mails that deal with DSL shipping status and `Discard` for others.

11. When you have enough e-mails on the Similar messages list (between seven and 30), click Save in Category.
12. In the Choose category dialog box, select the DSL service category, then click Create new category.
13. In the New category dialog box, enter DSL Shipping Status in the Category name box, then click OK.
14. Back in the Choose category dialog box, click OK to save the new category and its associated e-mails.
15. Click Restart Search and clear the Include Text box of everything except DSL.
16. Start again from Step 5, looking for another common theme that you can use as a subcategory.

End of procedure

Schedule Training

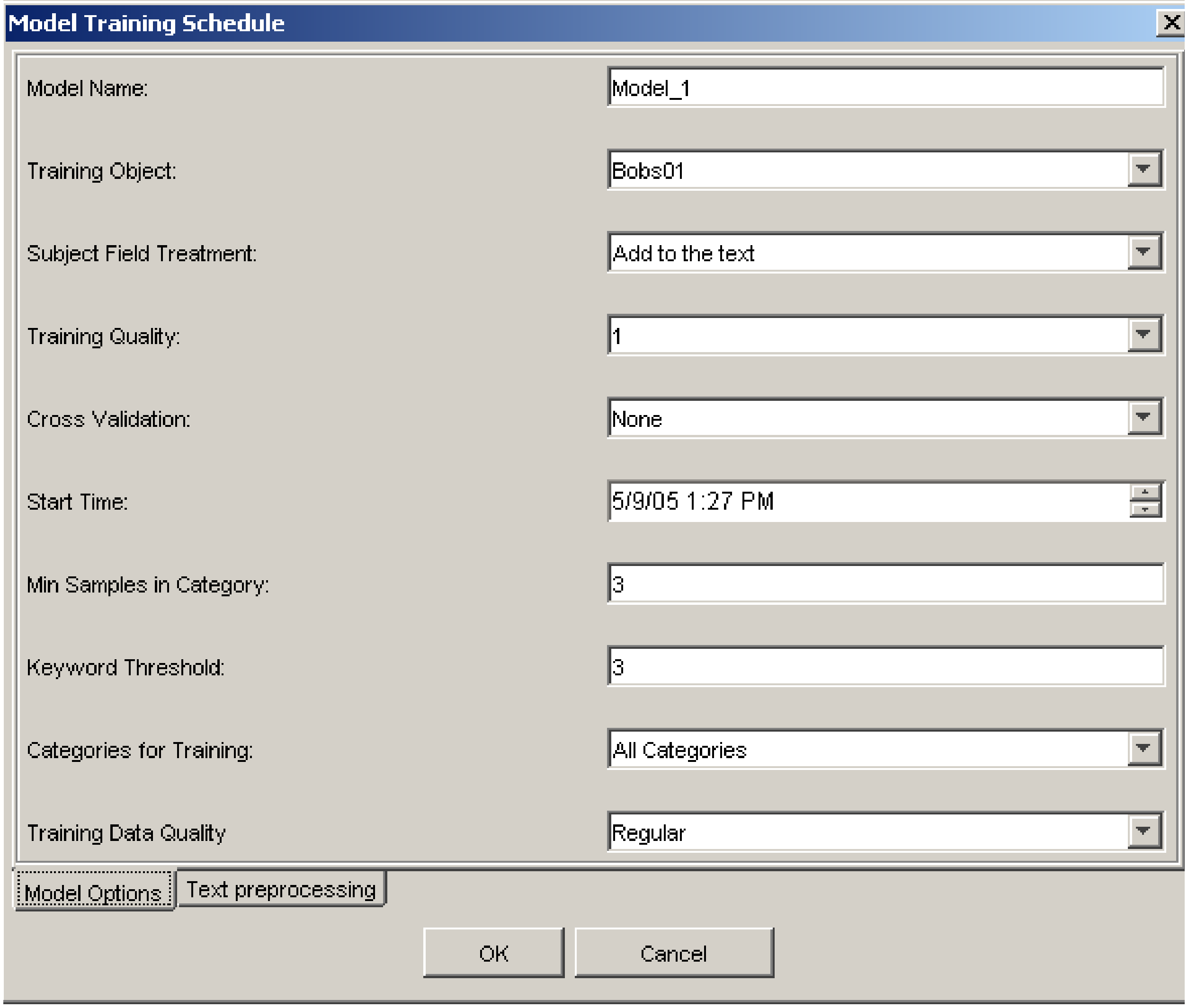
When you have a training object with enough e-mails, you are ready to schedule training.

There are the following two options:

- You can schedule training that uses an existing scheduled job as a template. This is a convenient way to change the time that an existing job is scheduled to run. To do this:
  a. On the Training Schedule tab, right-click the existing job that you want to use as a template.
  b. From the context menu, select New Training Job (Use this Job as Template).
- You can create a new training job from scratch. To do this, on the Tools menu of the Training tab, select Schedule New Model Training.

Both options bring up the Model Training Schedule dialog box. With option 1, the fields of the dialog box are populated with values copied from the existing job that was used as a template. With option 2, the fields are populated with default values, as shown in Figure 57 on page 115.

This dialog box, which has two tabs, opens on the Model Options tab.



Model Name:	Model_1
Training Object:	Bobs01
Subject Field Treatment:	Add to the text
Training Quality:	1
Cross Validation:	None
Start Time:	5/9/05 1:27 PM
Min Samples in Category:	3
Keyword Threshold:	3
Categories for Training:	All Categories
Training Data Quality	Regular

Figure 57: Model Training Schedule: Model Options Tab

Procedure:
Scheduling training using the Model Options tab

Purpose: To specify how and when a training object will be processed to produce a model.

Prerequisites

- A training object containing a sufficient number of e-mails or other text objects. See “When to Train” on page 145 for suggestions about judging whether there are enough text objects.

Start of procedure

1. Model Name—Enter a name for the model that will result from the scheduled training. See page 26 in Chapter 2 for restrictions on the names of Knowledge Manager objects.
2. Training Object—Select a training object.
3. Subject Field Treatment—Select from the following treatments of the Subject field of e-mails:
   - Ignore—Training does not consider the content of the Subject field
   - Add to the text—Training considers the content of the Subject field.
   - Add with double weight—Training gives the content of the Subject field twice as much importance as the content of the e-mail body.
4. Training Quality—Draft is the lowest quality, 6 is the highest. Note the following:
   - Training time increases as you move from Draft quality to level 3 quality. But once the quality goes above 3, there is not much difference in training time.
   - Genesys recommends that you use Draft quality only when you want to obtain a preliminary reading of the model's quality estimation. For production, use quality 2–6.
5. Cross Validation—Select either no cross-validation, or cross-validation that splits the data into 3, 5, or 10 sets. See "Cross-Validation" on page 125 for explanation.

   If you select cross-validation, training produces an accuracy rating for the model along with the model itself. This has the advantage of not requiring an extra testing step, but it increases the training time.
6. Start Time—Enter a start time or select a unit (day, month, hour, minute) and change its value using the up and down arrows. Because training can use a large proportion of system resources, you will probably want to schedule it for nonpeak hours.

   Note: Be sure to set a time later than the present moment.
7. Min Samples in Category—Enter the minimum number of text objects that a category must have in order to be included in training. Categories with no or few text objects make poor subjects for training.
8. Keyword Threshold—Enter the minimum number of text objects that a keyword must occur in for that keyword to be considered in training.

   A relatively high value for this setting can reduce training time, but it can also reduce quality. What counts as a high or low value for this setting depends on the total size of the training object. For example, if a training object has 5 to 10 text objects per category, a high keyword threshold might be 2 or 3. If a training object has 30 to 50 text objects per category, a high keyword threshold might be 20.

9. Categories for Training—Select All Categories or Terminal Categories Only. A *terminal category* is one that contains no subcategories. It may be that a category tree uses nonterminal categories only or mostly for organizing the terminal categories. In this case few or no text objects are associated with the nonterminal categories, and there is little to be gained by including the nonterminal categories in training.
10. Training Data Quality—Select Regular unless you know that the training object contains many wrongly categorized text objects. If it does, select Unreliable to set the categorization algorithm to run in an altered mode that gives better results with this type of data.

End of procedure

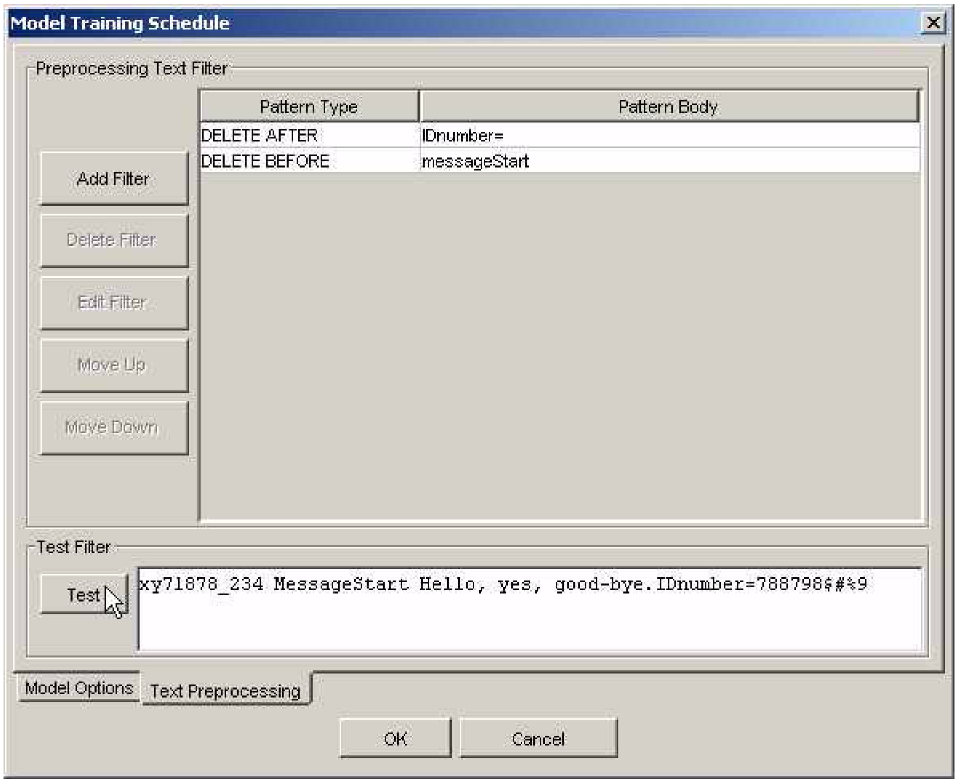
Next Steps

- Optionally, remove superfluous or misleading text from the training object (next section).
- Once the model is trained, test it ("Testing Models" on page 125).

Text Preprocessing Tab

The Text Preprocessing tab, shown in Figure 58 on page 118, enables you to remove extraneous text from the text objects of a training object. From this tab, you can create filters (patterns) that search for text and perform various deletion operations. This can be helpful when the e-mails that you want to use for training contain significant amounts of text that has both of these characteristics:

- It is predictable enough in content to be identifiable by a regular expression.
- It is irrelevant or misleading for classification purposes.



Pattern Type	Pattern Body
DELETE AFTER	IDnumber=
DELETE BEFORE	messageStart

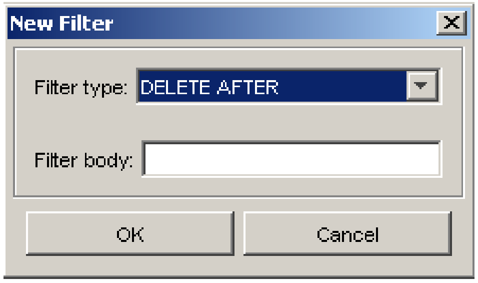
```
xy71878_234 MessageStart Hello, yes, good-bye.IDnumber=788798$#%9
```

Figure 58: Model Training Schedule: Text Preprocessing Tab

Procedure: Creating and testing filters

Start of procedure

1. Click `Add filter`. The `New Filter` dialog box appears, as shown in Figure 59.



Filter type: DELETE AFTER

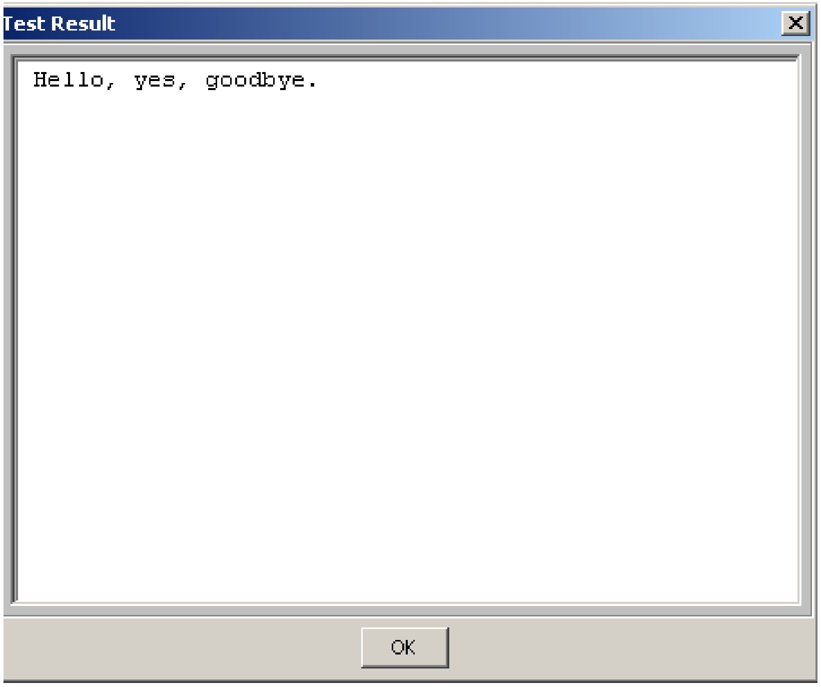
Filter body:

Figure 59: New Filter Dialog Box

2. Choose a type from the `Filter type` drop-down list. The filter type specifies the action to take; for example, delete all text up to and including the matched text. See “Filter Types” on this page for descriptions. Filter type is called `Pattern Type` on the main `Text Preprocessing` tab.

3. Enter text in the `Filter body` box. The filter body contains the text to match, as either a literal string or a regular expression (see "Regular Expressions" on page 74). Filter body is called `Pattern Body` on the main `Text Preprocessing` tab.
4. Click `OK`.

   Figure 58 on page 118 shows an example using two filters. The first deletes the text *IDnumber=* and anything following it. The second deletes the text *messageStart* and anything preceding it.
5. Continue by testing the filter: enter sample text in the window in the `Test Filter` area.
6. Click `Test.` A new window displays the result of applying all filters, in order. Figure 60 shows the result of the test on the text shown in Figure 59.



```
Hello, yes, goodbye.
```

Figure 60: FIlter Test Result

End of procedure

Filter Types

The following is a list of the available filter types:

- `DELETE AFTER`—Search for a match to the pattern body, then delete all text after and including the matching text.
- `DELETE BEFORE`—Search for a match to the pattern body, then delete all text before and including the matching text.
- `DELETE ALL IF FIND`—Search for a match to the pattern body, then delete the entire e-mail that includes the matching text.

- DELETE ALL IF NOT FIND—Search for a match to the pattern body, then delete the entire e-mail if it does not include the matching text.
- DELETE PATTERN—Search for a match to the pattern body, then delete only the text that matches the pattern.

Examples

Table 8 displays simple examples of text-preprocessing filters.

Table 8: Examples of Preprocessing Filters

Pattern Type	Pattern Body	Input Text	Test Result
DELETE AFTER	finch	one two finch three four	one two
DELETE BEFORE	finch	one two finch three four	three four
	[Mm]essage_?[Ss]tart	x897 message_Start one two three	one two three
DELETE ALL IF FIND	finch	one two finch three four	[a]
		one two three four	one two three four
	internal\d\d	one two three internal36 four	
DELETE ALL IF NOT FIND	finch	one two finch three four	one two finch three four
	finch	one two three four	
DELETE PATTERN	f.*ch\s	one two finch three four	one two three four
		one two fach three four	one two three four

[a] If you test this filter, the resulting window contains the message TEXT HAS BEEN DELETED. In actual use of DELETE ALL IF FIND or DELETE ALL IF NOT FIND, the entire text object is deleted from the training object.

Tables 9 and 10 present a more complex example using all five filter types. Table 9 lists the filters used in the example.

Table 9: Preprocessing Filters Example

Filter Number	Pattern Type	Pattern Body
1	DELETE BEFORE	MessageStart
2	DELETE AFTER	IDnumber=
3	DELETE ALL IF FIND	internal\d\d

Table 9: Preprocessing Filters Example (Continued)

Filter Number	Pattern Type	Pattern Body
4	DELETE ALL IF NOT FIND	nihil_obstat
5	DELETE PATTERN	company

Table 10 shows an example of input text and the results of applying the filters from Table 9 to it.

Table 10: Results of Testing the Example

Input Text	Test Result
x88_2 MessageStart nihil_obstat: Hello, companyyes, good-bye.IDnumber=7989	nihil_obstat: Hello, yes, good-bye.

The results in Table 10 come about as follows:

1. Filter 1 deletes the text *x88_2 MessageStart.*
2. Filter 2 deletes the text *IDnumber=7989*.
3. Filter 3 does nothing (it finds a match for nihil_obstat).
4. Filter 4 does nothing (it fails to find a match for internal/d/d).
5. Filter 5 deletes *company.*

Notes on Language

Selecting a Language

The first step in creating a training object is to select a tenant and language. The language that you choose has special relevance in the following two cases:

- If you want to build a model that classifies according to language, you must use a tree whose language is specified as `unknown`. First you must add this language attribute in `Configuration Manager > Business Attributes > Languages`.
- Selecting English activates a lexical analyzer (see "Lexical Analyzer" on this page) that is specific to English. If you are operating in a language other than English, you should not select English because the English lexical analyzer will hinder the training.

Selecting any language other than English activates a default lexical analyzer. You can also create a lexical analyzer that is specific to any language you choose, as described in "Lexical Analyzer" on this page.

Lexical Analyzer

The function of a lexical analyzer is to convert input text (such as the text of an e-mail) to an array of words or stems.

> A stem is a basic item that is shared by a family of words; for example, *see, saw, seen,* and *seeing* all have the same stem. As the example shows, the stem cannot always be found by simply removing endings from related words (consider also *go, went, gone*).

Content Analyzer includes the following:

- A lexical analyzer for English that converts words to stems, deletes digits and special characters, and segments the result into an array.
- A default lexical analyzer that is simpler than the English analyzer. The default analyzer considers any sequences of alphabetic characters (a–z, A–Z) to be words. It considers all other characters to be word separators.
- A sample lexical analyzer that you can modify to apply to a language of your choice.
- A lexical analyzer for Japanese, available with the Content Analyzer – Japanese option (see page 123).

The customized Lexical Analyzer is a class that implements the LexicalAnalyzer and Serializable interfaces. The LexicalAnalyzer interface includes two methods:

```
public interface LexicalAnalyzer {
  public String getLanguage();
  public String[] convert(String text);
}
```

- `public String getLanguage()` returns the name of the language that this lexical analyzer applies to.
- `public String[] convert(String text)` converts text to words or stems.

You can add one or more lexical analyzers for languages of your choice. To do this, you must prepare a Java class that implements the LexicalAnalyzer interface with the two methods just described.

The lexical analyzer example is located in `<KnowledgeManagerHome>\LexicalAnalyzerExample`. `<KnowledgeManagerHome>` is normally `C:\Program Files\GCTI\eServices 8.1.2\Knowledge Manager`. The source code is in `LexAnalyzerTest.java`. To adapt it to a language of your choice, use the following procedure.

Procedure: Adapting the lexical analyzer

Start of procedure

1. Select a name for the language that your analyzer will apply to.
2. Adapt the `LexAnalyzerTest` class to the target language, changing the name of the class and substituting the language name that you selected for the name used in the example (`English09`). For the purposes of this description, suppose you rename the class `MyLexAnalyzer`.
3. Compile the `MyLexAnalyzer` class using the following command:

   ```
   javac -classpath "gcengine.jar" MyLexAnalyzer.java
   ```

   The `gcengine.jar` file is located in the `LexicalAnalyzerExample` directory.
4. Copy the resulting `MyLexAnalyzer.class` file to the home directories of Knowledge Manager, Training Server, and Classification Server.

End of procedure

Stop Words

Stop words are words that are so common that there is little to be gained in searching for them or listing their occurrences. Examples for English are *the, a, an, of, to, is,* and so on. The system does not consider stop words when performing classification, and stop words do not appear on the `Indexing` tab of the TO Data Analyzer (page 110).

The installation packages for Genesys Content Analyzer install a file of stop words for English, called `English.stop,` in the home directories of Knowledge Manager and Training Server. This is a simple text file containing a list of words separated by carriage return. You can create other files of stop words for other languages.

Note: The stop word file must be in the UTF-8 format (prior to release 7.6, stop word files required the ANSI format).

Content Analyzer – Japanese

Genesys Content Analyzer – Japanese is a lexical analyzer for Japanese, available as an extra option. To use it, contact your Genesys representative to purchase a license, then proceed as follows:

- Locate the `license.dat` file and copy it to `<KnowledgeManagerHome>\LexicalAnalyzerGLA\lang`. Overwrite the dummy `license.dat` file that is already there.

- Add a language called Japanese_GLA to Configuration Manager > Business Attributes > Languages.

Large Training Objects

If your training object is very large (over 50,000 e-mails), training may consume considerable memory and time. To reduce this consumption without impacting quality, follow these recommendations when you schedule training (see "Scheduling training using the Model Options tab" on page 115):

- Set Cross Validation to None.
- Set Keyword Threshold above 25.
- Set Min Samples in Category above 25.
- Set Training Quality below 4. A level of 3 or 4 is adequate for production use.

You should also allocate memory as follows:

- Ensure that the host machine of Training Server has at least 4 GB of RAM for Solaris, or 2 GB of RAM for Windows.
- In the .sh or ProcessParameters.ini file, change the parameter -Xmx800m as follows:
  - On Windows, change to -Xmx1400m. This is enough for a training object of about 40,000 e-mails, the maximum recommended size on this platform.
  - On Solaris, change to -Xmx3000m. This is enough for a training object of about 100,000 e-mails, the maximum recommended size on this platform.

  For large training objects, these recommendations supersede those in the "Knowledge Manager" section on page 191 of Chapter 5, "Ongoing Administration and Other Topics".

A successful test has been done with the following parameters:

Host: Solaris, Enterprise 450 Model 4300 with 4000 MB RAM

Training object: 100,000 e-mails in 1,000 categories

Cross Validation: None

Keyword Threshold: 25

Min Samples in Category: 25

Training Quality: 3

The expected computational time is between 12 and 18 hours.

Note that the model produced has no quality ratings because you set Cross Validation to None. Genesys strongly recommends against using cross-validation on such large training objects. To obtain quality ratings for the model, build an additional small training object and test the model on it (see "Testing a Model on a Training Object" on page 126).

Testing Models

There are four methods of testing a model. The following two methods test the model's accuracy and produce ratings of it (see "Reading and Understanding the Ratings" on page 134):

- If you select cross-validation (this page) when you schedule training, Training Server produces accuracy ratings along with the model.
- You can test the model on a training object (page 126).

The following two methods show what category the model assigns to selected text objects, but does not test the accuracy of that categorization:

- You can test the model on text that you compose (page 129).
- You can apply the model to the uncategorized messages of a training object.

Cross-Validation

In cross-validation, Training Server follows these steps:

1. It builds one model using all of the data.
2. It divides the data into *x* partitions, where *x* = 3, 5, or 10.
3. It builds a number of partial models: as many as there are partitions, each one using a different combination of *x*-1 partitions.

   For example, if the data is divided into the three partitions A, B, and C, Training Server builds model X using partitions A and B, model Y using partitions A and C, and model Z using partitions B and C.
4. It tests each of these partial models against the partition that it omitted when it was built.

   In the example, it tests model X against partition C, model Y against partition B, and model Z against partition A.
5. It aggregates the results of all these tests and presents them as the rating of the entire model.

These ideas underly the concept of cross-validation:

- The best way to test a model is to apply it to data that was not used in building the model.
- A model built using most of the data is usefully similar to the model built using all of the data, so the results of testing (for example) all possible 90-percent models are a good indication of the quality of the 100-percent model.

Because cross-validation adds to the time required to build a model, you may not want to select cross-validation for very large training objects or for objects for which you selected training quality level `6`.

Testing a Model on a Training Object

You can test a model on a training object. This process applies the model to the texts in the training object and compares the resulting classification with the classification in the training object itself. The training object must use the same category tree as the model you are testing on it.

Note: It is not advisable to test a model on the training object that generated it: the results will be unrealistically favorable.

Setting Up Training Objects for Testing

You may want to create a new training object just for the purpose of testing. Use the same category tree but different text objects. There are two ways to do this:

- Starting with a collection of categorized e-mails, create two new training objects. See "Creating two new training objects" on this page.
- Starting with an existing training object, create a second training object using randomly-chosen text objects from the first. In more detail:

  a. Move five percent of the text objects, randomly selected, from one training object to another.

  b. Train a model on the first training object.

  c. Test the model on the second training object.

  This is similar to using cross-validation with two partitions. "Extracting random text objects" on page 127 describes this process in detail.

Procedure: Creating two new training objects

Prerequisites

- A large collection of categorized e-mails from a relatively long period of time.

Start of procedure

1. Divide your collection into two parts, either according to date received or according to the agent that handled the interaction.

   The two parts may be mutually exclusive or not. For example, one part could be interactions from January through June of last year. and the other could be from July through December of last year. Or one part could be all interactions from last year, and the other part could be interactions from November and December of last year.

2. Create two training objects, one using each part of the collection.
3. Build a model on one training object.
4. Test the model on the other training object, as described in "Testing a model on a training object" on page 128.

End of procedure

Procedure: Extracting random text objects

Prerequisites

- This example assumes that you have a large training object called `T01`.

Start of procedure

1. Make a copy of `T01`, calling it `CopyofT01`.
2. Create an new empty training object, called `T02`, using the same category tree as `T01`.
3. On the `Training` tab, left hand pane, select the root `Training Objects` node to give the two-pane view.
4. On the right-hand pane, right-click `T02`. Select `Move part of Training Object`, as shown in Figure 61.

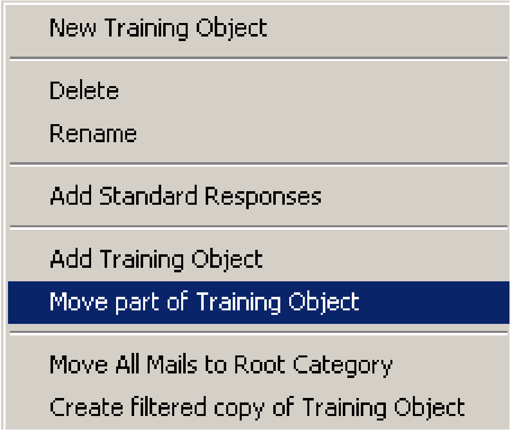
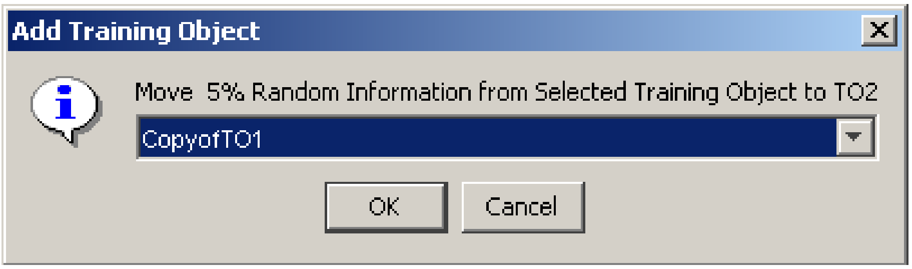

New Training Object
Delete
Rename
Add Standard Responses
Add Training Object
Move part of Training Object
Move All Mails to Root Category
Create filtered copy of Training Object

Figure 61: Move Part of Training Object

5. In the resulting `Add Training Object` dialog box, select `CopyofT01`, as shown in Figure 62.



Add Training Object

Move 5% Random Information from Selected Training Object to TO2

CopyofTO1

Figure 62: Add Training Object Dialog Box

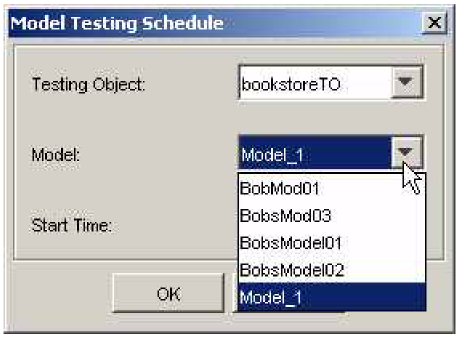
6. Knowledge Manager randomly selects five percent of the text objects in `CopyofTO1`, copies them to `TO2`, and deletes them from `CopyofTO1`.
7. Train a model on `CopyofTO1`, then test it on `TO2`, as described in the next section.

End of procedure

Procedure:
Testing a model on a training object

Start of procedure

1. On the `Training` tab, select `Tools > Schedule Model Testing`. The `Model Testing Schedule` dialog box opens, as shown in Figure 63.



Model Testing Schedule

Testing Object: bookstoreTO

Model: Model_1

Start Time:

Figure 63: Model Testing Schedule

2. Select a testing object—that is, select a training object to use.
3. Select a model to test.
4. Enter a start time.
5. Click `OK`.

If the results are good and if your two training objects include some non-overlapping items, you can merge the two objects, by adding one to the other:

6. On the two-pane view of the Training tab (see page 100), select one training object on the right-hand pane.
7. Right-click and select Add Training Object from the shortcut menu.
8. In the dialog box that opens, select the other training object from the drop-down list, then click OK.

End of procedure

Testing a Model on Composed Text

Purpose: To test a model by seeing how it classifies a text object that you write for the purpose.

Start of procedure

1. On the Models tab, go to the left-hand pane and select the root Models node.
2. On the right-hand pane, select the model that you want to test, then right-click and select Test from the shortcut menu. A dialog box appears, titled Model: <modelname>, as shown in Figure 64.

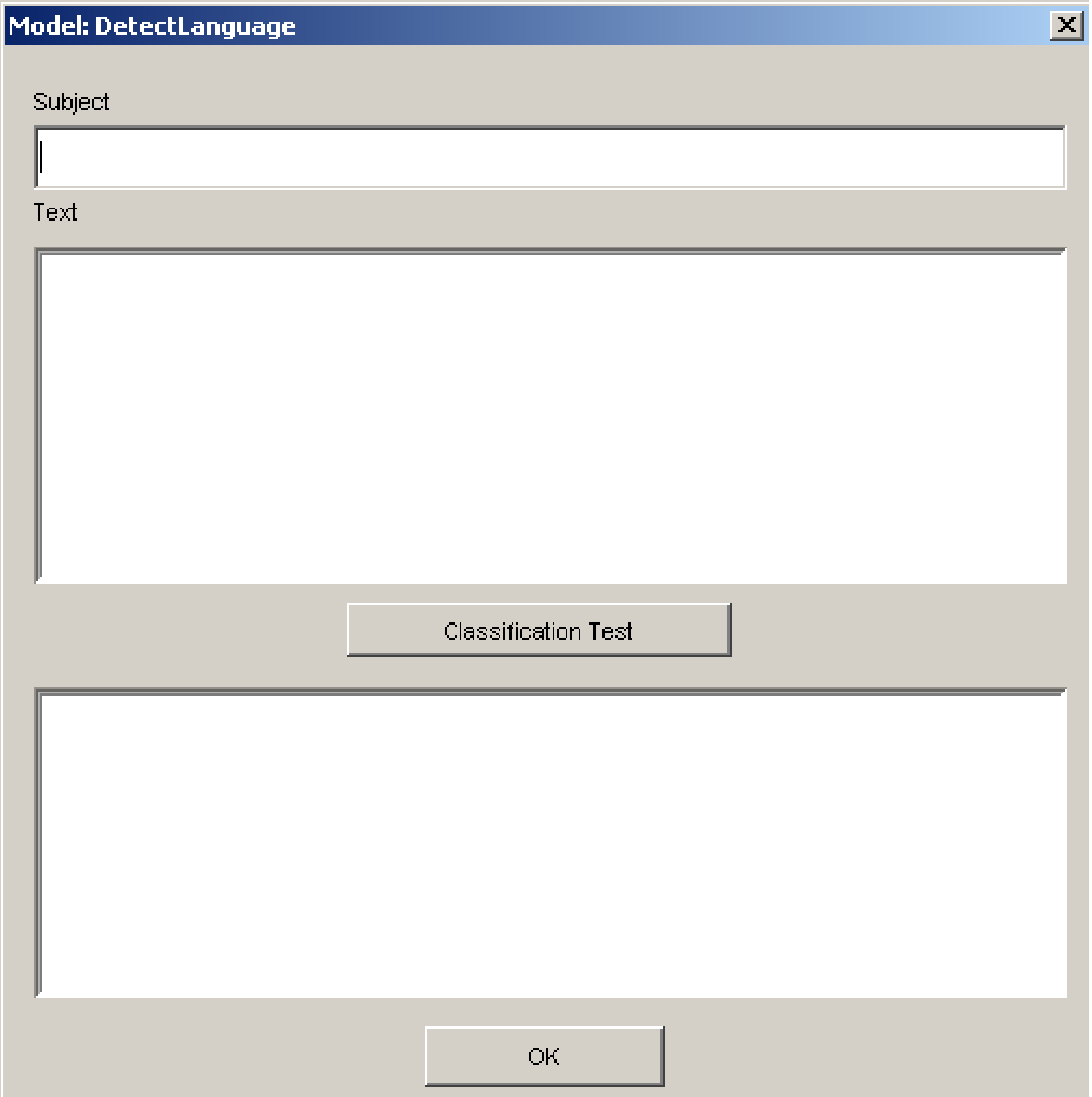
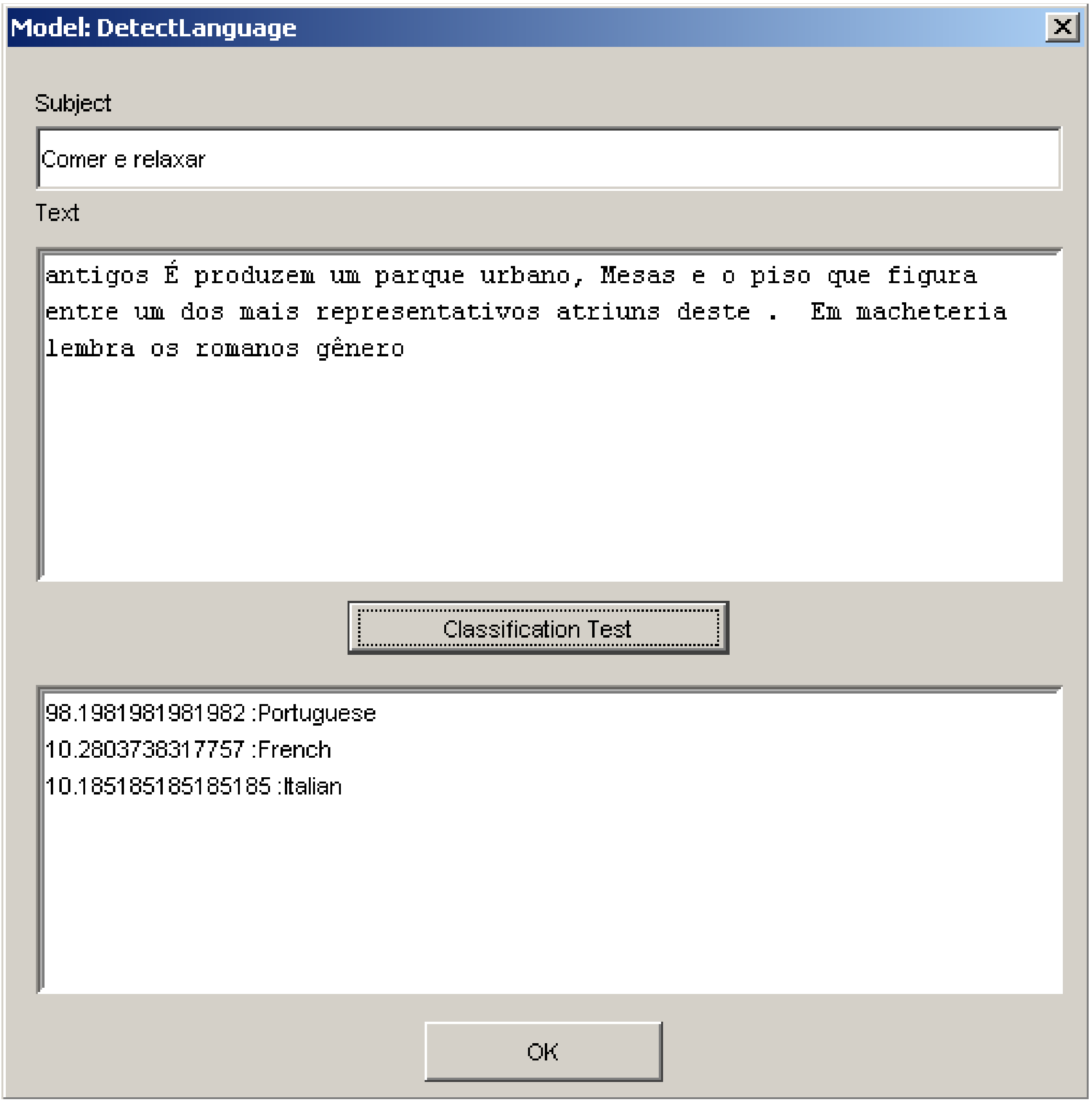


Figure 64: Model Test Dialog Box

3. Enter text in either or both of the `Subject` and `Text` boxes, then click `Classification Test`.
4. Results display in the lowest box. Figure 65 shows the results of a test on the `DetectLanguage` model that is supplied with Genesys Content Analyzer.



Subject

Comer e relaxar

Text

antigos É produzem um parque urbano, Mesas e o piso que figura entre um dos mais representativos atriuns deste . Em macheteria lembra os romanos gênero

```
98.1981981981982 :Portuguese
10.2803738317757 :French
10.185185185185185 :Italian
```

Figure 65: Model Test Results

The results are in the form of a list of categories, each category preceded by the rating of the confidence with which the system assigns the test text to that category.

End of procedure

Testing a Model on Uncategorized Messages

Prerequisites

- There must be uncategorized messages in the training object’s root category. You accomplish this by doing either of the following:

- Assign uncategorized messages to the root category when creating the training object (Step 6 on page 94).
- Move all text objects to the root category after creating the training object (see the text preceding Figure 47 on page 102).

Start of procedure

1. On the `Models` tab, go to the left-hand pane and select the root `Models` node.
2. On the right-hand pane, right-click the model that you want to test, then select `Test Uncategorized messages from TO`, as shown in Figure 66.

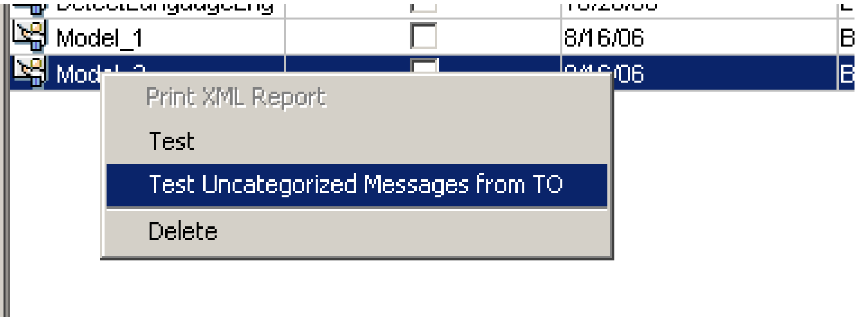


Figure 66: Test Uncategorized Messages

3. In the resulting window, select a training object from the drop-down list, as shown in Figure 67. Be sure to select a training object that contains a good number of uncategorized messages.

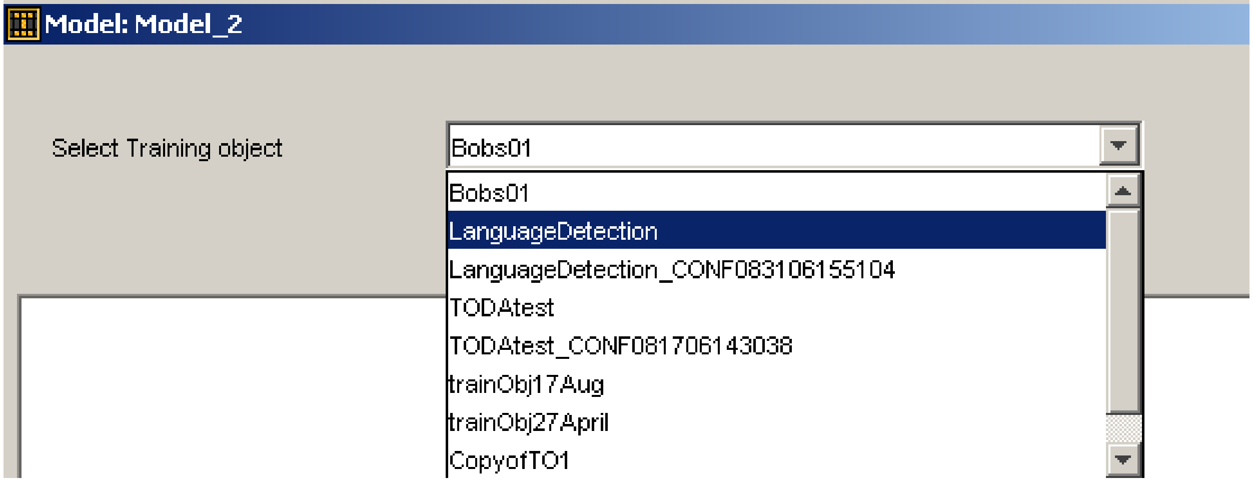
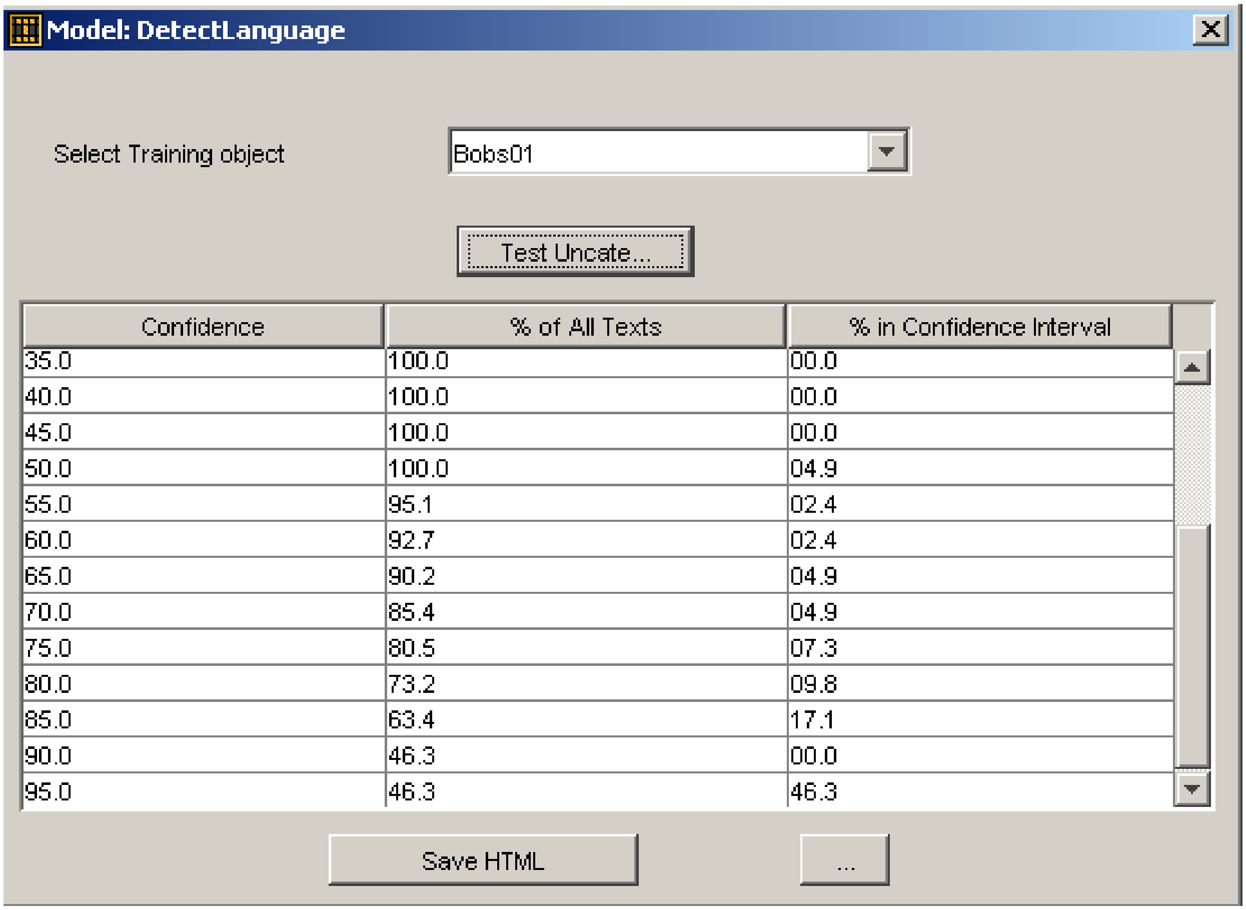


Figure 67: Select a Training Object

4. Click `Test Uncategorized texts`. The results are displayed as in Figure 68.



Select Training object: Bobs01

Confidence	% of All Texts	% in Confidence Interval
35.0	100.0	00.0
40.0	100.0	00.0
45.0	100.0	00.0
50.0	100.0	04.9
55.0	95.1	02.4
60.0	92.7	02.4
65.0	90.2	04.9
70.0	85.4	04.9
75.0	80.5	07.3
80.0	73.2	09.8
85.0	63.4	17.1
90.0	46.3	00.0
95.0	46.3	46.3

Figure 68: Results of Test Uncategorized Messages

The results show the following for each confidence level:

- `% of All Texts`: The percentage of texts that were classified above this level of confidence.
- `% in Confidence Interval`: The percentage of texts that were classified with a level of confidence between this level and the next higher level on the scale.

These results tell you how well the model does, according to its own internal metric, at assigning new texts to some category or other. They do not evaluate the accuracy of these category assignments.

To save the results as an HTML file, click `Save HTML`, provide a name for the file, then click `Save`.

End of procedure

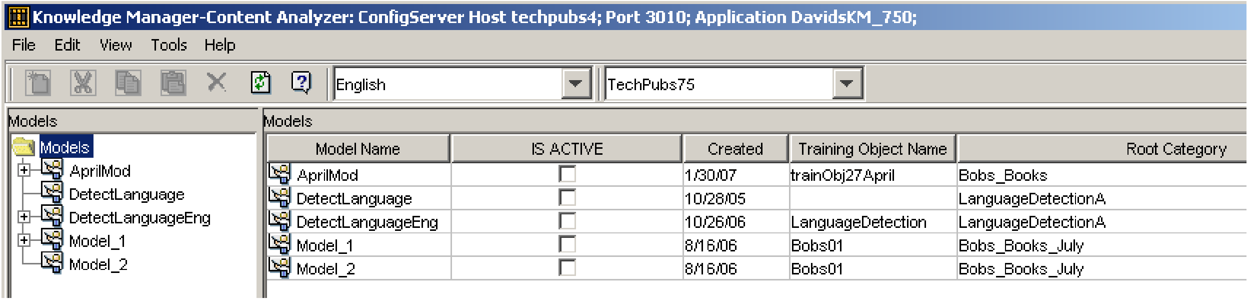
Using and Rating Models

You can have many models, but you can use only one at a time for classification. You designate the model to use in classification by setting it as `Active` on the `Models` tab.

Reading and Understanding the Ratings

The `Models` tab displays a browserlike tree structure on its left-hand pane. You can use the structure as follows:

- Select the root `Models` node to display a list of summary information about all models, as shown in Figure 69. The `IS ACTIVE` column consists of check boxes; select one check box to select the model that will be active in classification.

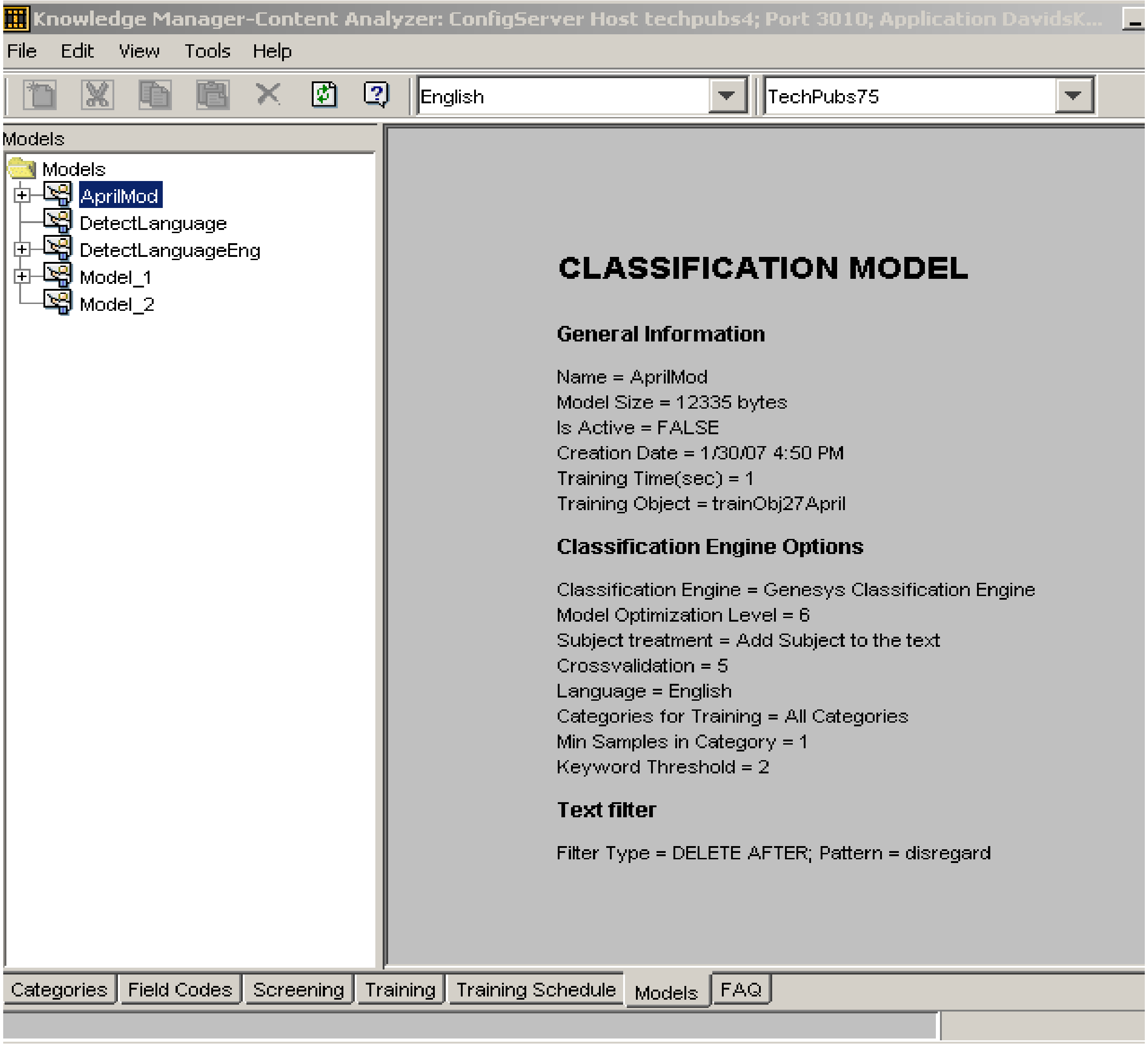


Knowledge Manager-Content Analyzer: ConfigServer Host techpubs4; Port 3010; Application DavidsKM_750;

Model Name	IS ACTIVE	Created	Training Object Name	Root Category
AprilMod	□	1/30/07	trainObj27April	Bobs_Books
DetectLanguage	□	10/28/05		LanguageDetectionA
DetectLanguageEng	□	10/26/06	LanguageDetection	LanguageDetectionA
Model_1	□	8/16/06	Bobs01	Bobs_Books_July
Model_2	□	8/16/06	Bobs01	Bobs_Books_July

Figure 69: Models Tab: Root Node Selected

- Select a model node to display detailed information about the model, as shown in Figure 70 on page 135.



CLASSIFICATION MODEL

General Information

Name = AprilMod
Model Size = 12335 bytes
Is Active = FALSE
Creation Date = 1/30/07 4:50 PM
Training Time(sec) = 1
Training Object = trainObj27April

Classification Engine Options

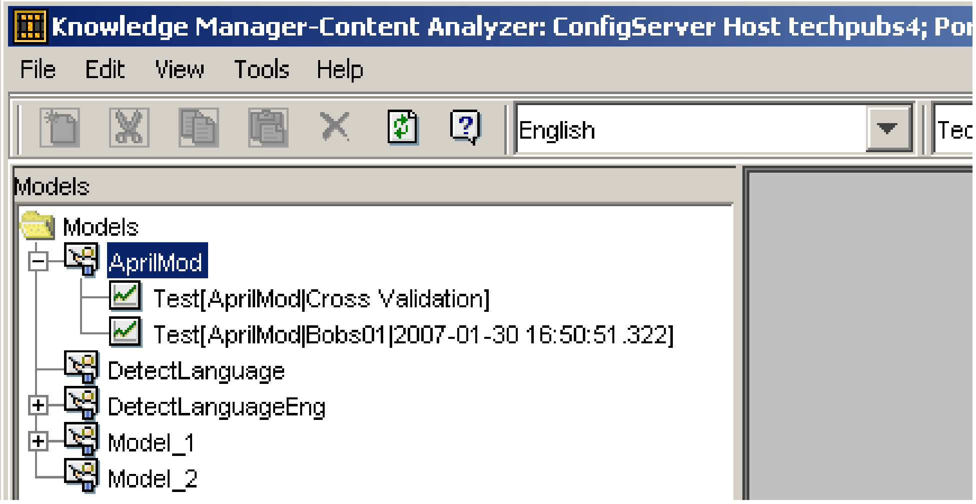
Classification Engine = Genesys Classification Engine
Model Optimization Level = 6
Subject treatment = Add Subject to the text
Crossvalidation = 5
Language = English
Categories for Training = All Categories
Min Samples in Category = 1
Keyword Threshold = 2

Text filter

Filter Type = DELETE AFTER; Pattern = disregard

Figure 70: Models Tab: Model Node Selected

- If a model has been tested ("Testing Models" on page 125), its node can expand to display ratings nodes, as shown in Figure 71.


Knowledge Manager-Content Analyzer: ConfigServer Host techpubs4; Por
File
Edit
View
Tools
Help
English
Tec
Models

Models
- AprilMod
  - Test[AprilMod|Cross Validation]
  - Test[AprilMod|Bobs01|2007-01-30 16:50:51.322]
- DetectLanguage
- DetectLanguageEng
- Model_1
- Model_2

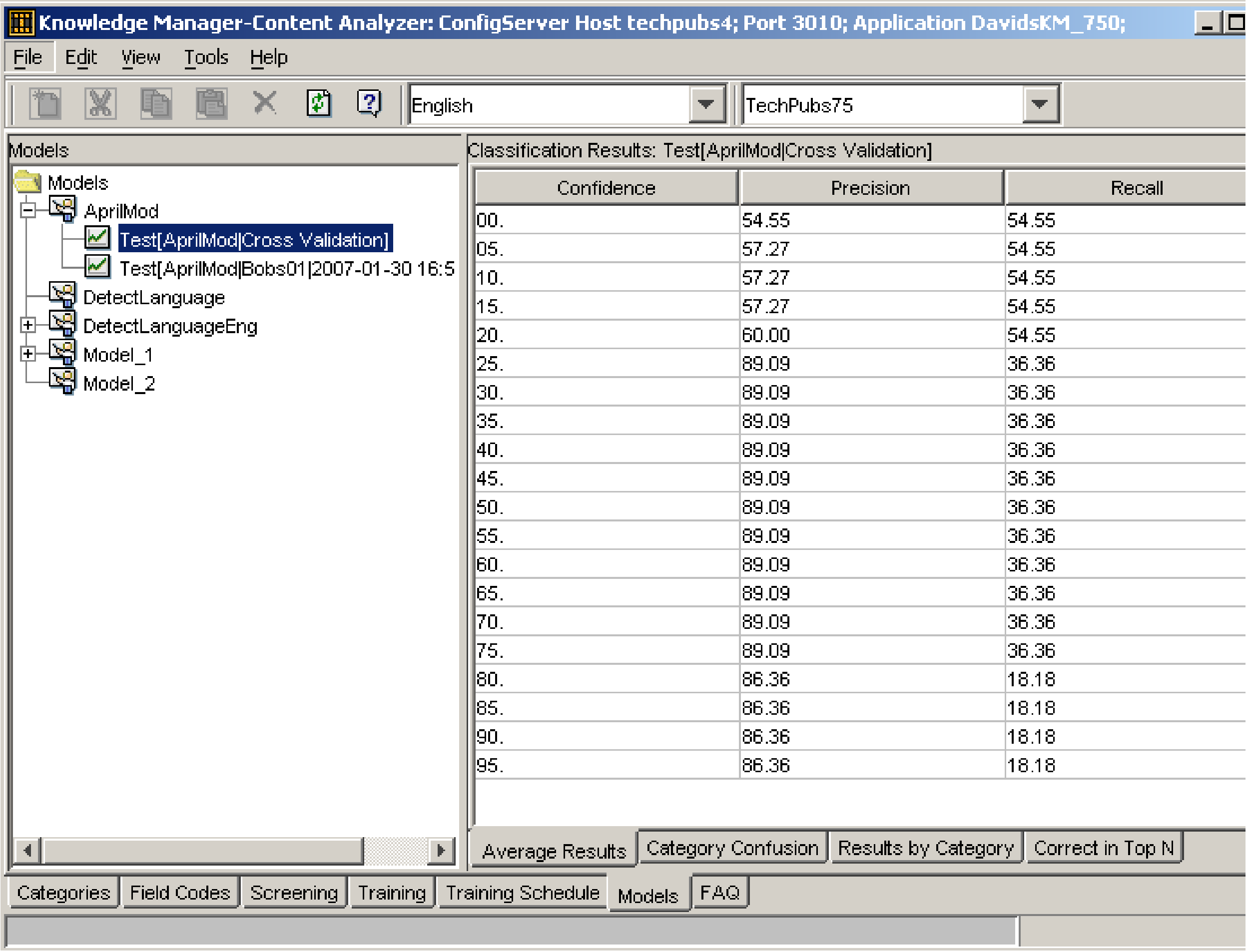
Figure 71: Models Tab: Ratings Nodes Displayed

The ratings node label has the form `Model <modelname>[<ratingsource>],` where <ratingsource> is either `'CrossValidation` or `on<testObjectName> Testing Object/Time=<date time>` For example, in Figure 71, the Model `New04` has two sets of ratings. One is from cross-validation applied during generation of the model, and the other is from testing the model on the training object `Bobs01`.

- Select a model’s ratings node to display its ratings on four subtabs, as described in the following sections.

Average Results Subtab

This subtab, shown in Figure 72 on page 137, rates how well the model classifies, averaged across all categories.



Classification Results: Test[AprilMod|Cross Validation]

Confidence	Precision	Recall
00.	54.55	54.55
05.	57.27	54.55
10.	57.27	54.55
15.	57.27	54.55
20.	60.00	54.55
25.	89.09	36.36
30.	89.09	36.36
35.	89.09	36.36
40.	89.09	36.36
45.	89.09	36.36
50.	89.09	36.36
55.	89.09	36.36
60.	89.09	36.36
65.	89.09	36.36
70.	89.09	36.36
75.	89.09	36.36
80.	86.36	18.18
85.	86.36	18.18
90.	86.36	18.18
95.	86.36	18.18

Figure 72: Models Tab: Average Results Subtab

The `Confidence` rating corresponds to the attribute of the same name that you set in the IRD objects `Classify` and `Classification switch`.

To understand `Precision` and `Recall`, consider several possible ways of looking at the performance of a model. If your model attempts to assign a certain number of items to a particular category X, you can make the following counts:

a = the number of items the model correctly assigns to X

b = the number of items the model incorrectly assigns to X

c = the number of items the model incorrectly rejects from X (that is, items that the model should assign to X but does not)

From these quantities, you can calculate the following performance measures:

- Precision = $a/(a + b)$
- Recall = $a/(a + c)$

Generally, for increasing precision you pay the price of decreasing recall. That is, the model assigns an item to a category only when it is very sure that the item belongs. But by insisting on being very sure, it runs the risk of rejecting items that really do belong in the category.

Here is another way to look at it. Suppose that at a confidence rating of 50, precision = `70` and recall = `80`. Then, for text T and category C, the statements in Table 11 hold.

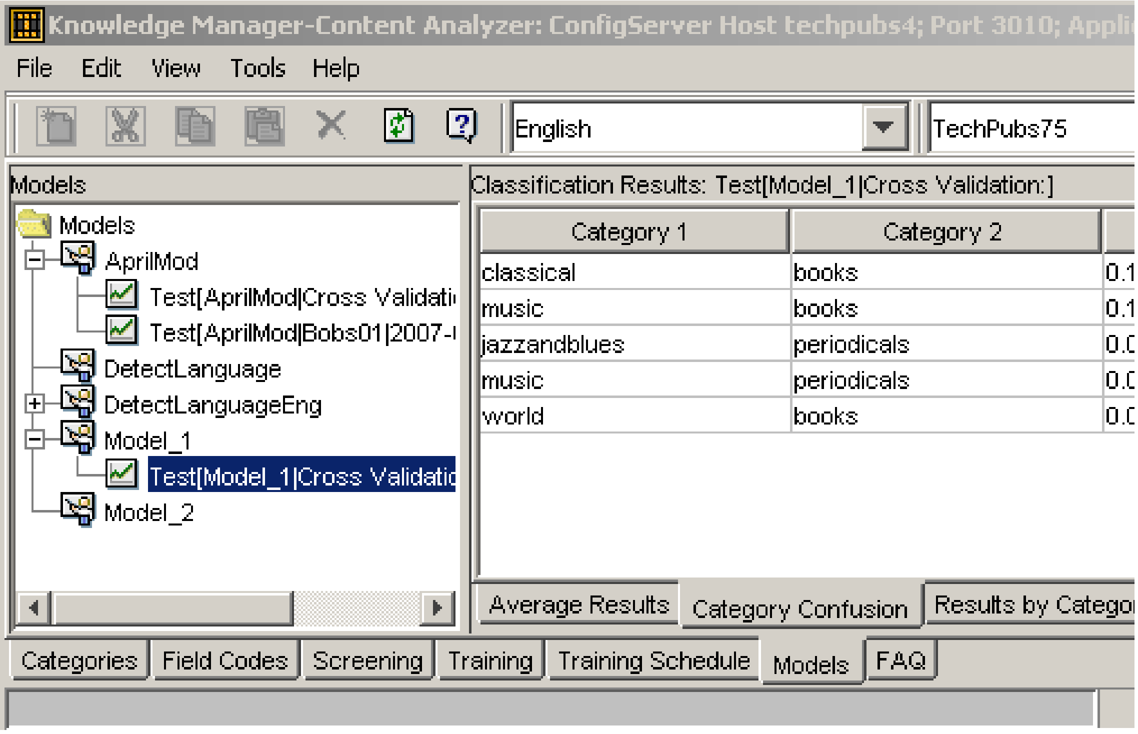
Table 11: Example of Precision and Recall

If you know that	then you can infer that	this percent of the time
T belongs to C,	The model will classify T as C, with confidence of over 50%,	80%
The model classifies T as C, with confidence of over 50%,	T does belong to C,	70%

Use the `Average Results` ratings as an assessment of the overall quality of the model. See also “Applying the Ratings” on page 146.

Category Confusion Subtab

This subtab, shown in Figure 73, lists up to 10 pairs of categories that the model is likely to confuse.



Classification Results: Test[Model_1|Cross Validation:]

Category 1	Category 2	
classical	books	0.1
music	books	0.1
jazzandblues	periodicals	0.0
music	periodicals	0.0
world	books	0.0

Figure 73: Models Tab: Category Confusion Subtab

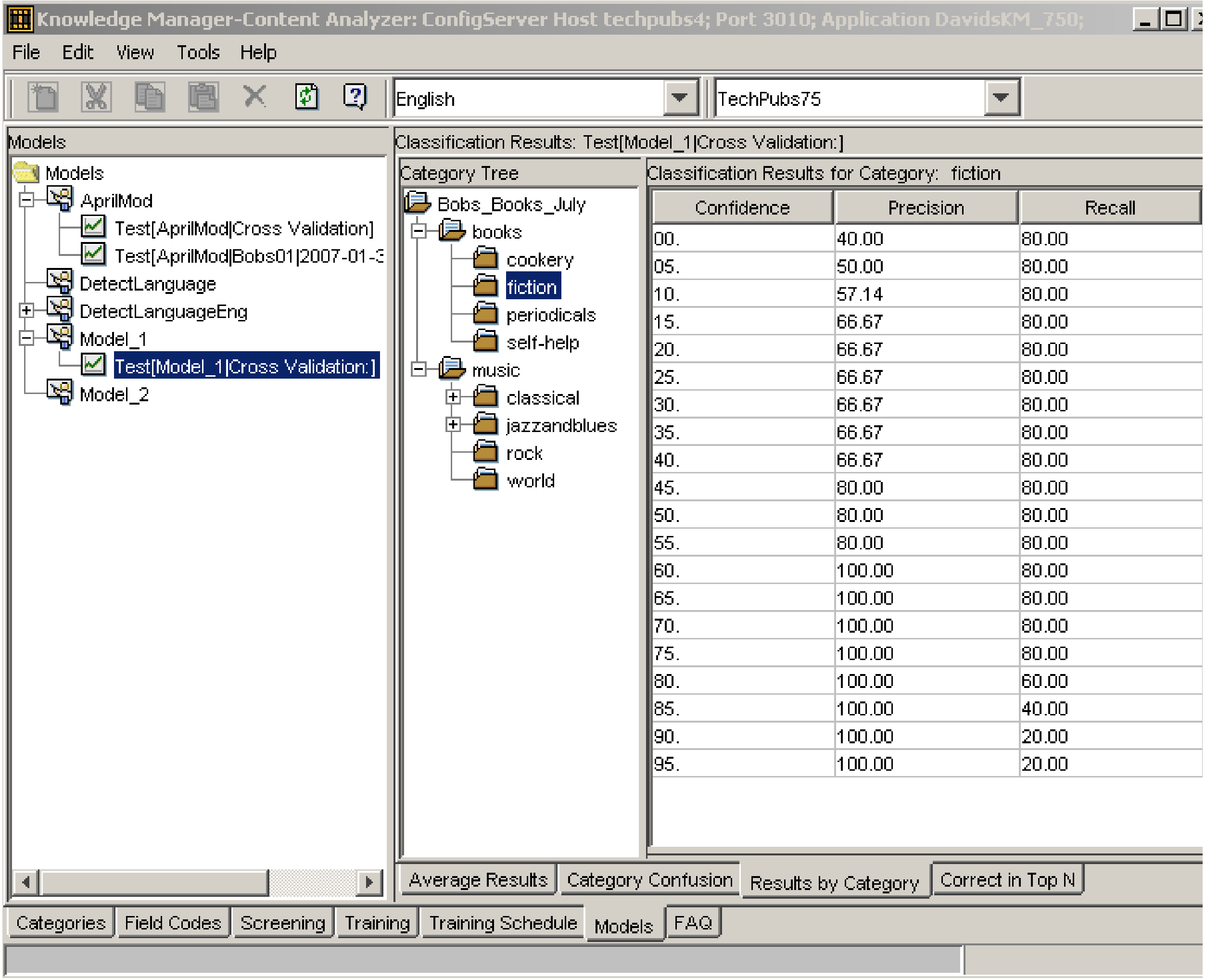
The `Confusion` column gives the probability (between 0 and 1) of confusion between the categories in the `Category 1` and `Category 2` columns. A rating of 0.5 would mean total confusion: the model cannot distinguish A from B. A rating of 1.0 would mean that the model always calls A B and always calls B A—a complete reversal.

If a pair of categories has a rating of over 0.2 and both categories have more than three or four members, you should consider modifying them. You can modify them in either of two directions:

- Merge them (decide that they are so similar they amount to a single category).
- Further differentiate them by adding more highly contrasting e-mails to them in the training object.

Results by Category Subtab

This subtab, shown in Figure 74, displays the same ratings as the two preceding subtabs, but for a single category. A central pane displays the category tree; select a category to display `Confidence`, `Precision`, `Recall`, and `Confusion` on the right-hand pane.



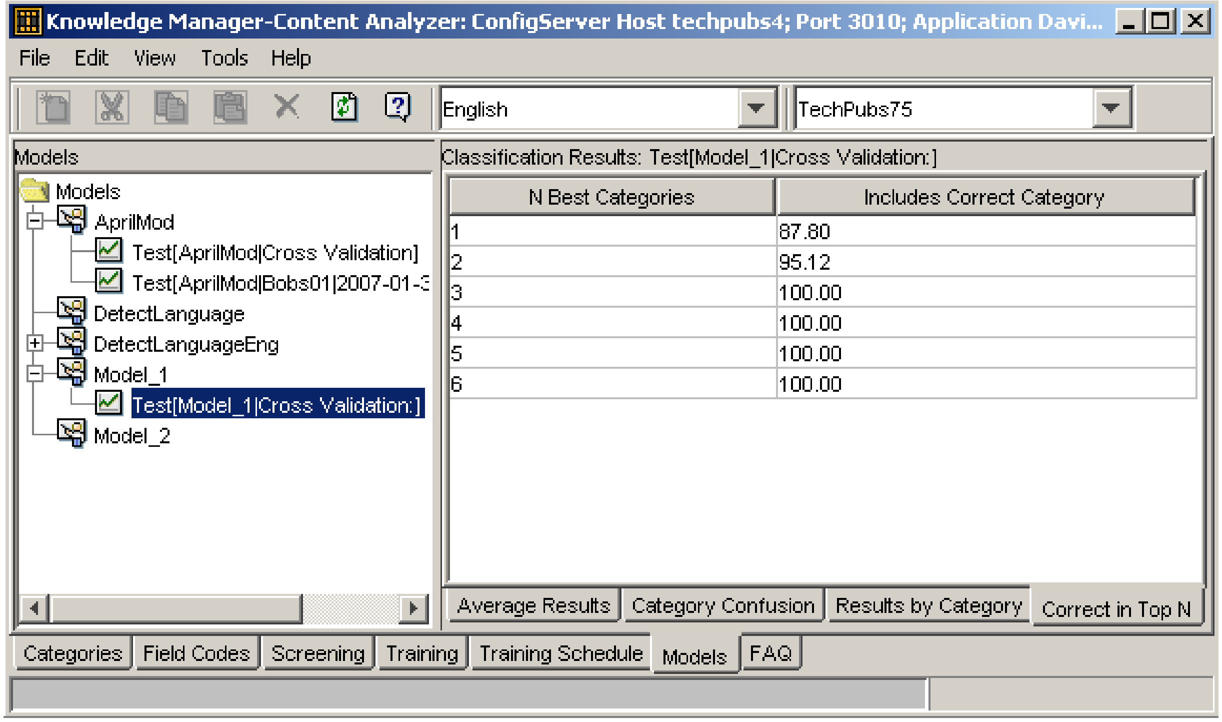
Confidence	Precision	Recall
00.	40.00	80.00
05.	50.00	80.00
10.	57.14	80.00
15.	66.67	80.00
20.	66.67	80.00
25.	66.67	80.00
30.	66.67	80.00
35.	66.67	80.00
40.	66.67	80.00
45.	80.00	80.00
50.	80.00	80.00
55.	80.00	80.00
60.	100.00	80.00
65.	100.00	80.00
70.	100.00	80.00
75.	100.00	80.00
80.	100.00	60.00
85.	100.00	40.00
90.	100.00	20.00
95.	100.00	20.00

Figure 74: Models Tab: Results by Category Subtab

Use the `Results by Category` ratings to help you set the confidence level in IRD objects, as described in “Applying the Ratings” on page 146.

Correct in Top N Subtab

When a model classifies a text object, it returns a list of categories and the probability that the object belongs to them. Ranking the returned categories with the highest probability first, how likely is it that the correct category appears within the top two, the top three, and so on? Ratings of this likelihood are displayed on the `Correct in Top N` subtab, shown in Figure 75.



English

TechPubs75

Classification Results: Test[Model_1|Cross Validation:]

N Best Categories	Includes Correct Category
1	87.80
2	95.12
3	100.00
4	100.00
5	100.00
6	100.00

Figure 75: Models Tab: Correct in Top N Subtab

Look at the first row for a good indication of the overall quality of the model. Its general meaning is that a single classification attempt using this model will be correct for this percentage of categories.

The other rows can help you further assess model quality. A very accurate model may have a 95 per cent probability that the correct category appears in the top two or three. For a less accurate model, you may have to go down to the top five or six to achieve 95 per cent coverage.

You can also use this rating to advise agents how many categories to look at when choosing a standard response. If there is a 95 per cent probability that the right category is in the top three, you can advise agents to examine only the top three categories.

See also “Applying the Ratings” on page 146.

Reporting on the Ratings

You can produce a report on the ratings of a model, either by directly printing it or by generating an HTML file. To obtain a report, first select a model’s ratings node to display the ratings. Then do one of the following.

- Print directly:
  a. On the right-hand pane, right-click and select `Print`.
  b. Proceed through Page setup and Print dialog boxes.

This report contains the figures from the Average Results and Correct in Top N tabs.

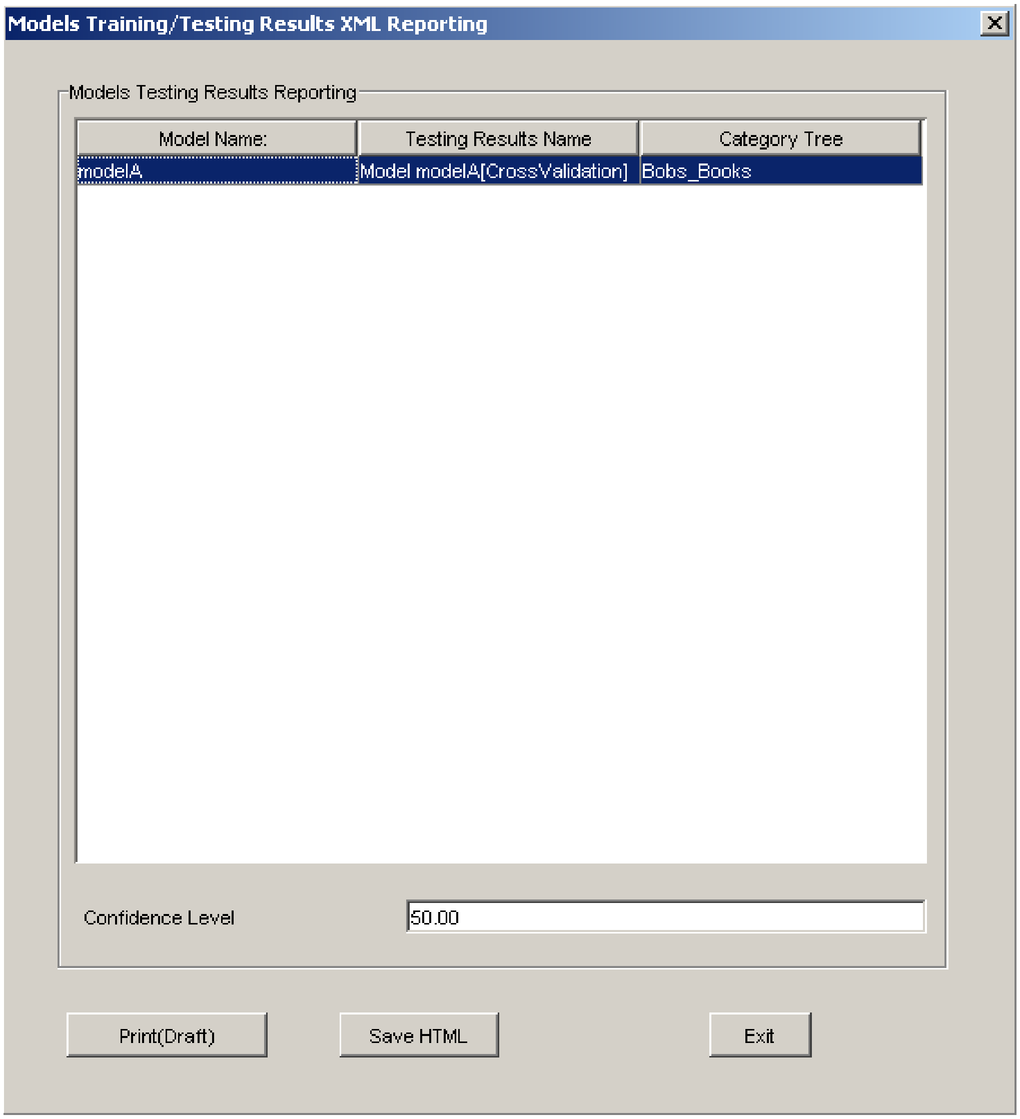
- Use the following procedure to produce an HTML file.

Procedure: Producing an HTML report on ratings

Start of procedure

1. On the left-hand pane, right-click and select Print XML Report.

   The Models Training/Testing Results XML Reporting dialog appears, as shown in Figure 76.



Models Training/Testing Results XML Reporting
Models Testing Results Reporting

Model Name:	Testing Results Name	Category Tree
modelA	Model modelA[CrossValidation]	Bobs_Books

Confidence Level 50.00

Print(Draft)
Save HTML
Exit

Figure 76: Models Training/Testing Results XML Reporting Dialog Box

2. Select the model(s) that you want to report on.
3. Set a confidence level. This determines the way that results by category are displayed in the report; see item 5 below for details.
4. Do one of the following:
    - Click `Print(Draft)`. This produces a printout of the HTML file.
    - Click `Save HTML`, then choose a filename and location for the report.

End of procedure

The resulting report has the following structure:

1. Introductory material, including definitions of precision, recall, and an additional measure called F1, which is a kind of averaging (more precisely, the harmonic mean) of precision and recall.
2. Model name and information, as it appears when you select the model on the `Models` subtab.
3. Microaverage Table, which reproduces the statistics from the `Average Results` subtab.
4. Correct category In Top N Categories, which reproduces the statistics from the `Correct in Top N` subtab.
5. Results for Categories, which reproduces the statistics from the `Results by Category` subtab. It does this by listing the following for each category:
    - Name
    - Precision at the confidence level that you set when producing the report.
    - Recall at the confidence level that you set when producing the report.
    - F1 averaging for the precision and recall in the preceding two items.
    - The top two categories likely to be confused with this category, with their confusion ratings.

Improving the Results

If the results of testing a model are unsatisfactory, there are several things you can do to try to produce an improved model.

- Add more data to the training object
- Analyze your category tree. Are some categories never or seldom used? Are a few categories so general that they absorb most e-mails, leaving little for other categories? See "Design and Use Considerations" for more on this topic.

Design and Use Considerations

Getting started with Genesys Content Analyzer requires four basic steps. This section provides information and advice on these steps, as follows:

1. Create a category tree and standard responses. See "Design."
2. Create a training object using the tree. Add text objects (e-mails and other objects) to the training object. See "Design."
3. Train a new model using the training object. See "When to Train" on page 145.
4. Test the model and use the resulting ratings. See "Applying the Ratings" on page 146.

Design

In designing your category trees and standard responses, remember that they will have two very different groups of "users"—that is, agents and training.

- Training uses the categories plus categorized e-mails to generate models.
- Agents use categories in two quite different ways:
  - They use the categories to find standard responses.
  - They give feedback on the category/standard response system, essentially indicating, "Yes (no), the standard response of this category is (is not) a good match with this e-mail," affirming that this e-mail should/should not be tagged with this category. This tagging becomes one of the attributes of the interaction as it is stored in the Universal Contact Server database.

Note: Agents can use standard responses without giving feedback, but if they do not give feedback you cannot collect enough categorized e-mails to be useful for training. You then have to create e-mail manually from Knowledge Manager's `Training` tab.

Given the importance of high-quality feedback, you may want to designate a special group of agents for this purpose: define the categorizing of interactions as one of their main duties. Remember: the more categorized e-mails you have and the more accurate the categorization is, the more likely the system is to produce accurate models

In designing your category trees and standard responses, keep in mind the following:

- Do not create too many categories. Many categories allows for many standard responses, and if there are large numbers of standard responses agents are likely to use some responses very little or not at all. This creates the following chain of causation:

  a. There are very few e-mails tagged with a particular category.

  b. The system cannot train for that category.

  c. The system cannot suggest that response.

  d. That category and its response continue to be used very little.

  In short, excess categories are likely to not be used.
- Try to make categories sufficiently distinct. If two or more standard responses apply to very similar situations, training has difficulty producing a model that can tell them apart.
- Avoid categories/responses that are too general, like “Not enough information.” Agents will use only one or two such general responses and ignore any others, with two undesirable results:
  - Training has a hard time producing a good model because the e-mails it uses have a huge variety of content.
  - The system is unable to include the unused categories/responses in training, because there are very few e-mails tagged with those categories in the database.

When to Train

When does your training object have enough categorized text objects to make training worthwhile? Here are some possible situations and comments on them.

Uniformly Low Feedback

In this situation, all categories have a small amount of feedback (less than about 12 text objects per category). This object is not fully ready for training. You can still try training a model, but you should be aware that the results probably will not be very good.

Unbalanced Feedback: Mostly Low

In this situation, all categories except a small group have a small amount of feedback (less than about 12 text objects per category). The small group (one to five categories) may have several hundred or even thousands of feedback objects per category. You can train a model, but the resulting model will mostly return the categories from the small group. A situation of this type may have these causes:

- It may be an accurate reflection of the situation. For example, your company may sell 25 products but just three of them may account for 90 per cent of its business.

- It may reflect shortcomings in the system:
  - Agents may not use standard responses properly.
  - The standard responses and/or the category tree may be poorly designed.

To determine which of these causes obtains, inspect your category tree, standard responses, and agents' use of them. If the situation arises because of shortcomings in the system, consider doing the following:

- Bring some balance into the training object by deleting some of the text objects associated with categories of the high-feedback group.
- Modify the low-feedback categories.

Unbalanced Feedback: Mostly High

In this situation, some categories have a small amount of feedback (less than about 12 text objects per category), but a significant number (over 50) of categories have a large amount of feedback (over 30–50 text objects or more per category). This is a rather common situation. You can train a model and it will work acceptably on the high-feedback categories. But consider modifying the low-feedback categories.

Uniformly High Feedback

In this situation, almost all categories have significant feedback (over 50 text objects per category). This is the best situation. It means that agents are frequently using almost all standard responses. You can train the model and it should perform well on all categories.

Applying the Ratings

Ratings of models, described in "Testing Models" on page 125, have several possible uses.

Assessing Overall Quality

To assess the overall quality of a model, look at the ratings on the `Average Results` and `Correct in Top N` subtabs (see page 136 and page 140).

Identifying Confusing Categories

To identify categories that may be too similar and/or have insufficient feedback, use the ratings on the `Category Confusion` subtab (see page 138). Consider modifying any pair of categories that has a confusion rating over 0.2.

Using in Routing

To decide where to set the `Confidence` level in an IRD `Classify` object (see "eServices Objects" in the "Interaction Routing Designer" chapter of *Universal Routing 8.1 Reference Manual*), use the ratings on the `Average Results` and `Results by Category` subtabs (see page 136 and page 139). The way that you use these ratings differs according to the task that you are performing at this step of the strategy:

- If you are using the classification to choose an autoresponse or acknowledgment: Set a relatively high `Confidence` value, one at which `Precision` is 85 or higher and `Recall` is 5 or higher.
- If you are using the classification to choose standard responses as suggestions to the agent: Set a `Confidence` value of 1. Even a very low `Precision` rate (for example, 15) is safe because an agent will make the final decision on whether to use the standard response. Also the lower the `Confidence` value, the more categories are returned, and:
  - The higher the probability that the correct category is among them.
  - The more categories the agent can provide feedback on.
- If you are using the classification to determine where the interaction goes in the next step of the strategy: Set the `Confidence` value at the point where `Precision` is approximately equal to `Recall`.

Language Detection Model

As part of Content Analyzer, Genesys provides a model that classifies e-mails as English, French, German, Italian, Portuguese, Russian, or Spanish.

To import this model and its training object, use the following procedure.

Procedure: Importing a language classification model

Start of procedure

1. Select `unknown` as the language. If there is no such language you must create one in Configuration Manager (see "Notes on Language" on page 121).
2. Select the `Import` command (see also "Importing Knowledge Manager objects" on page 81)
3. Click `Browse` and navigate to `<KnowledgeManagerHome>\LanguageModel\lang.kme`. `<KnowledgeManagerHome>` is normally `C:\Program Files\GCTI\eServices 8.1.0\Knowledge Manager`.

4. Click `OK`.

End of procedure

The training object consists of seven categories, one for each language. Each category contains a number of text objects in its language. You can add more text objects to these categories as well (see "Adding More Text Objects to the Training Object" on page 97). This could be especially valuable if you have a collection of text objects (such as e-mails) whose subject matter relates to your business. After you add text objects, you must train a new model to take advantage of the added data.

You can also add other languages to the model, as follows:

Procedure:
Adding more languages to the model

Start of procedure

1. On the `Categories` tab (still with `unknown` selected as the language), add a category for the new language to the `LanguageDetection` category tree, as described in "Creating a Category Tree" on page 25.
2. On the `Training` tab, select the `LanguageDetection` training object, then select the new language category in the training object.
3. Add text objects in the language, as described in "Adding More Text Objects to the Training Object" on page 97.
4. Train a new model that includes the new language, as described in "Schedule Training" on page 114.

End of procedure

You can do this for any language supported by E-mail Server (E-mail Server supports all languages that are supported by the version of JRE that is supplied with Genesys eServices). However, Genesys has not tested any language other than those listed above.

Analyzing Sentiment and Actionability with Content Analyzer

You can use Genesys Content Analyzer to analyze the sentiment and actionability of interactions that have been brought into the system by Genesys Social Messaging Management. Genesys supplies samples that demonstrate these capabilities.

Sentiment

The following procedure describes deploying the sentiment sample.

Procedure: Deploying the sentiment analysis sample

Start of procedure

1. In Configuration Manager or Genesys Administrator, create a language called English_Sentiment.
2. With Knowledge Manager set to that language, import the file `EnglishSentiment.kme`, which is located in the `<KnowledgeManagerHome>\SentimentModel` directory.

End of procedure

This provides:

- A model `SentimentSampleModel` for analyzing sentiment.
- The training object `Sentiment` that created that model.
- A category tree `SentimentDetection` that contains the categories to assign to interactions as a result of the analysis.

Actionability

To use the actionability sample, import the file `Actionability.kme`, which is located in the `<KnowledgeManagerHome>\ActionabilityModel` directory.

This provides:

- A model `Actionability` for analyzing actionability.
- The training object `Actionability` that created that model.
- A category tree `Actionability` that contains the categories to assign to interactions as a result of the analysis.

Next Steps

You can use the sample training objects to produce new models, improving the quality by making adjustments such as:

- Altering the settings such as those for quality level (page 116) and cross-validation (page 125).
- Using the Mail Editor (page 98) to edit the content of the messages in the training object.
- Using the Mail Editor to add more sample messages to the training object.

Genesys also provides sample screening rules for detecting sentiment and actionability, as described in "Screening for Sentiment and Actionability" on page 77.

For more information on Genesys Social Messaging Management, see the *eServices Social Media Solution Guide,* available on the Genesys Documentation Wiki at `http://developerzone.genesyslab.com/wiki/index.php?title=Category:EServices_Social_Media_Solution_Guide`.

FAQ Objects

Taking a category tree and its associated standard responses as input, Knowledge Manager can produce an FAQ object. From this object Knowledge Manager can produce a .jar file, which can in turn be used to:

- Build a web application that accepts written requests and, using content analysis, returns a set of standard responses.
- Present the contents (or a selection from the contents) of the standard response library as answers to frequently-asked questions.

An FAQ object combines a category tree, a training object based on the tree, and, optionally, a model built from the training object. The model is required in order to build a web application.

FAQ objects allow you to include in your web application a means of gathering user feedback about the correctness of a returned standard response. The application then uses this feedback to update the confidence rating of that particular standard response. This functionality is exemplified in the FAQ sample in the Simple Samples that are installed along with Web API Server. For a description of this sample and its source code, see the *eServices 8.1 Web API Client Developer's Guide.*

Sample FAQ .jar File

The sample FAQ that is supplied with Knowledge Manager demonstrates the way that an FAQ object can present a question/answer list. This sample is also included in the Web API samples (see the *eServices 8.1 Web API Client Developer's Guide).*

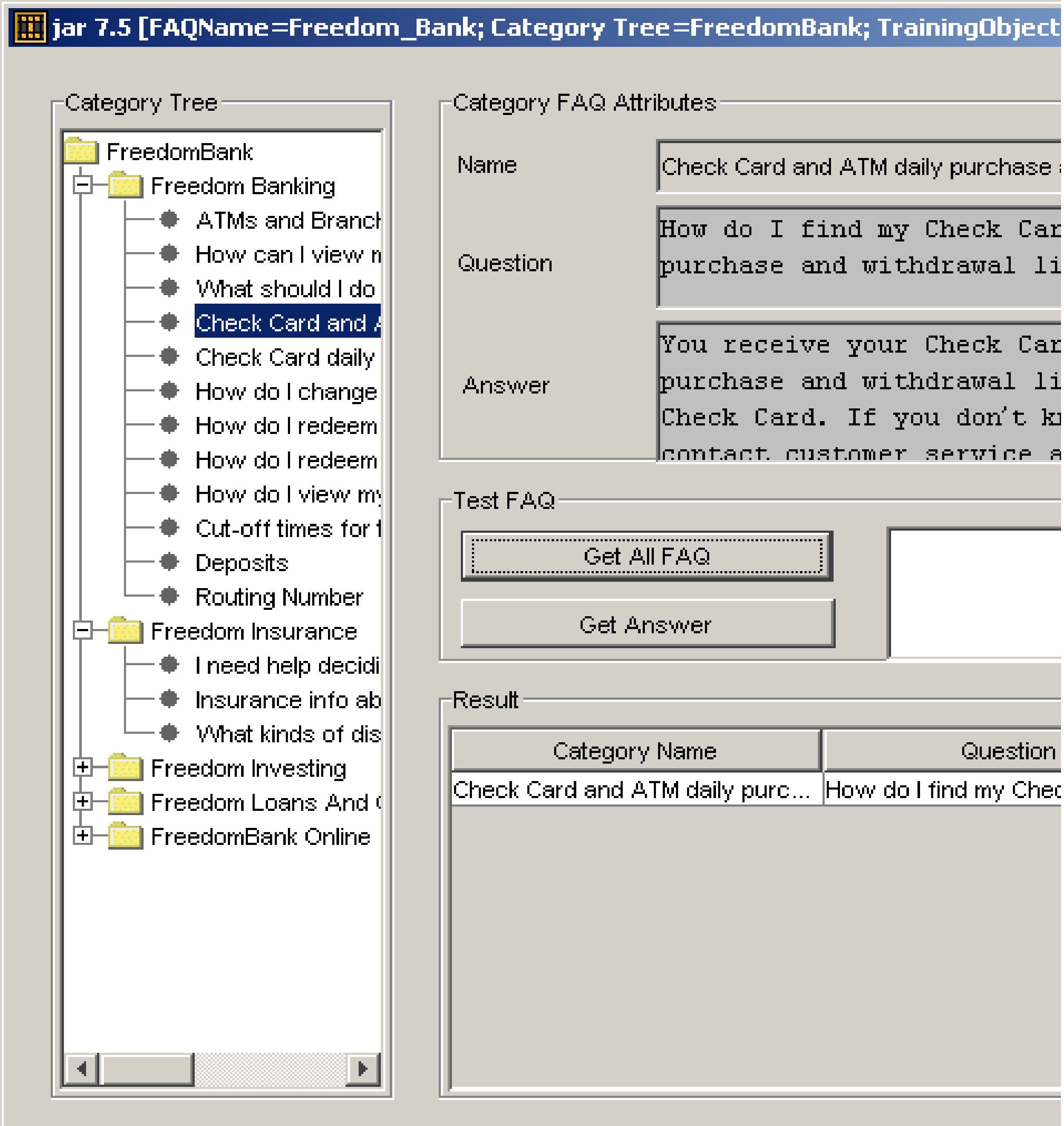
The filename of the sample is `faq_example.jar`. The installation places it in a directory called `FAQExample` in the Knowledge Manager home directory (normally `C:\Program Files\GCTI\eServices 8.1.0\Knowledge Manager`). To use this sample:

1. Use a text editor to open its batch file `unit_test.bat`.
2. Edit the line

   ```
   set JAVAHOME=D:\jdk1.4\bin\java
   ```

so that it points to the location of Java in your environment.

3. Use the batch file to launch the sample. You should see the window shown in Figure 77.



Name: Check Card and ATM daily purchase a

Question: How do I find my Check Car
purchase and withdrawal li

Answer: You receive your Check Car
purchase and withdrawal li
Check Card. If you don't kn
contact customer service a

Category Name	Question
Check Card and ATM daily purc...	How do I find my Chec

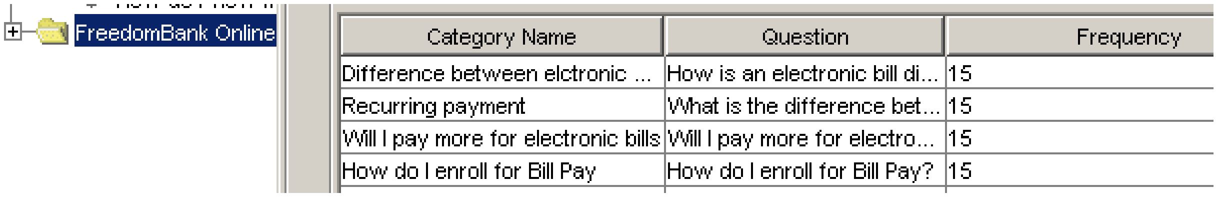
Figure 77: FAQ Sample

This FAQ object works as follows:

- Click a category on the left-hand pane to see its name and associated question and answer displayed in the boxes at upper right.

The main function of the nonterminal categories (represented as folders) is to organize the terminal categories (represented as grey disks), which are the main locus of questions and answers. Mostly the nonterminal categories do not have answers, and their `Category Question` box displays either a duplicate of the category name or a short description, rather than a question.

- Click `Get All FAQ` to display, in the `Result` area, a list of all categories contained in the selected category (and its subcategories), along with the question and frequency for each. An example is shown in Figure 78.



Category Name	Question	Frequency
Difference between elctronic ...	How is an electronic bill di...	15
Recurring payment	What is the difference bet...	15
Will I pay more for electronic bills	Will I pay more for electro...	15
How do I enroll for Bill Pay	How do I enroll for Bill Pay?	15

Figure 78: Get All FAQ for a Selected Category

The `Result` area lists all subcategories of the category that is selected on the left-hand pane, providing the following information:

- Category name
- The question associated with the category
- The frequency rating—that is, the number of text objects that are associated with this category in the training object that is part of the FAQ

- To test the FAQ object, enter a sample question in the `Testing Question` box, then click `Get Answer`. The `Result` area displays the answers that the system provides for the sample question at the selected level of the tree. In addition to the category name, question, and frequency, the system also displays a confidence rating. Confidence takes the model that is part of the FAQ object and tells you how good that model is at assigning new questions to this category (see "Reading and Understanding the Ratings" on page 134).

More About FAQ Objects

To be used in an FAQ object, a category must have all of the following attributes:

- Answer: It must have a standard response of the `FAQ` usage type. This attribute is optional for nonterminal categories.
- Question: It must have an associated question, to which the standard response can serve as the answer.
- Selection: It must be selected for inclusion in the FAQ object.

  You must select the categories to include because you cannot assume that all categories in the tree are suitable for use in an FAQ list. For example, your tree might include a category that exists only to provide the standard response *Your account is overdrawn.* Or you might want to use a single category tree to produce multiple FAQ objects, with some categories selected in one FAQ object but not in others.

Requirements are slightly different for terminal categories (those without subcategories) and nonterminal categories (those with subcategories), as shown in Table 12.

Table 12: Required Attributes for FAQ Categories

Attribute	Nonterminal Category	Terminal Category
Answer	Optional	Required
Question	Required	Required
Selected	Required	Required

You can create answers and questions either on the `Categories` tab or the `FAQ` tab. But selection can be done only on the `FAQ` tab, after you generate the FAQ object.

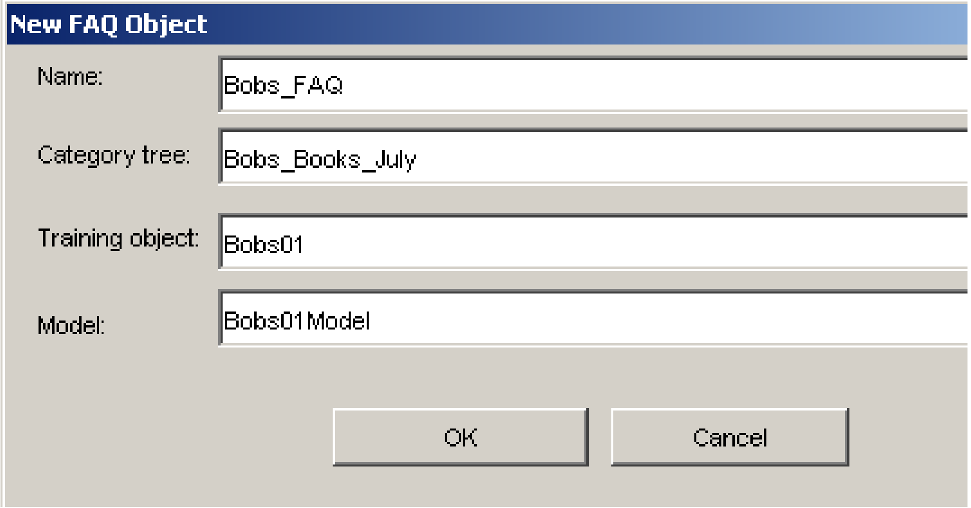
Creating an FAQ Object

Use the following procedure.

Procedure: Creating a new FAQ object

Start of procedure

1. On the `FAQ` tab, do one of the following:
   a. Select `File > New`.
   b. On the left-hand pane, right-click and select `New FAQ Object`.



Name: Bobs_FAQ

Category tree: Bobs_Books_July

Training object: Bobs01

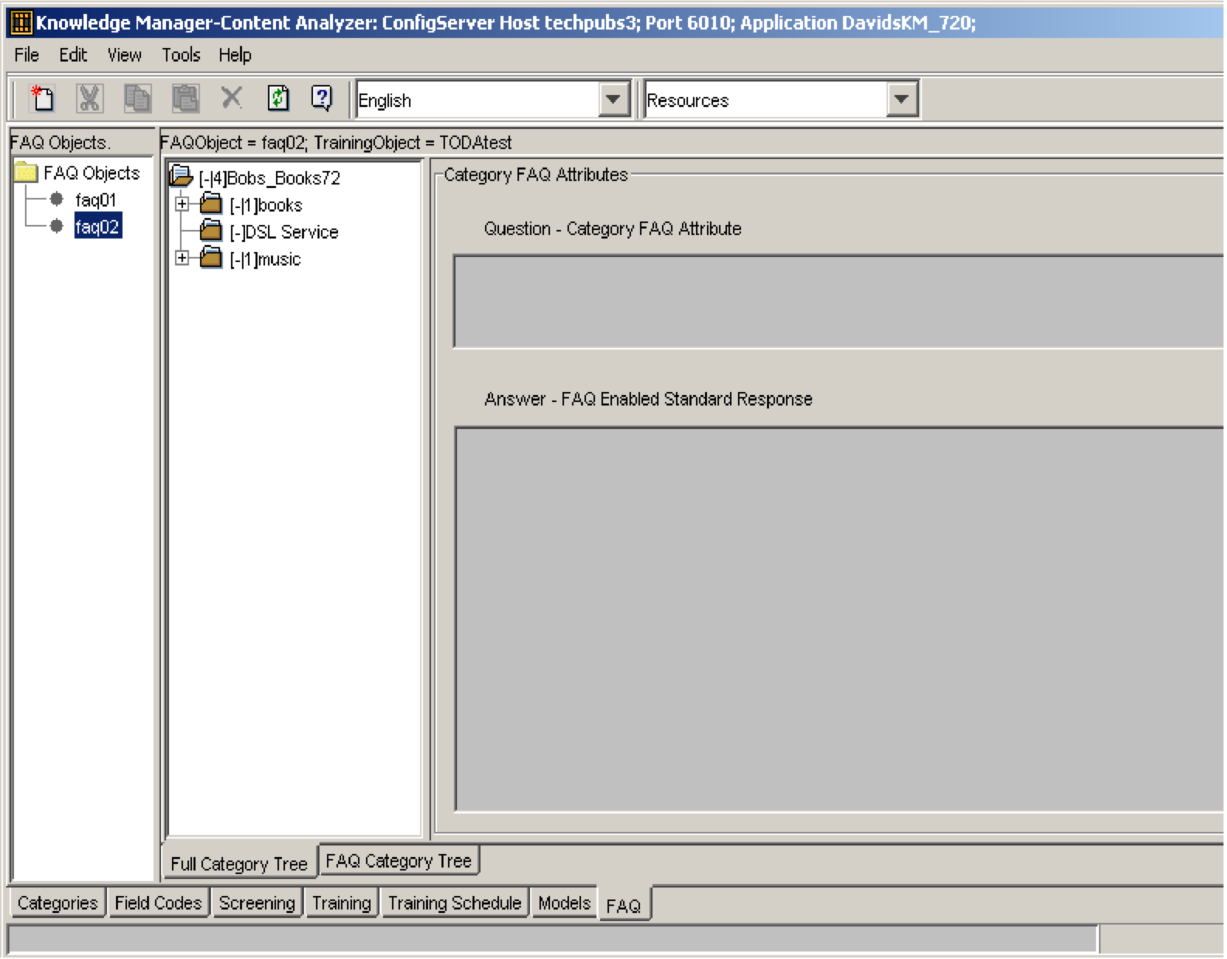
Model: Bobs01Model

Figure 79: New FAQ Object Dialog Box

2. The `New FAQ Object` dialog box appears, as shown in Figure 79. On it:
   a. Enter a name.
   b. Select a category tree, training object, and model.

   The category tree and training object are required. You can create an FAQ object without a model, but you will not be able to use it in conjunction with content analysis.
3. Click `OK`.

The `FAQ` tab then appears as in Figure 80.


Knowledge Manager-Content Analyzer: ConfigServer Host techpubs3; Port 6010; Application DavidsKM_720;

FAQObject = faq02; TrainingObject = TODAtest

Figure 80: FAQ Object

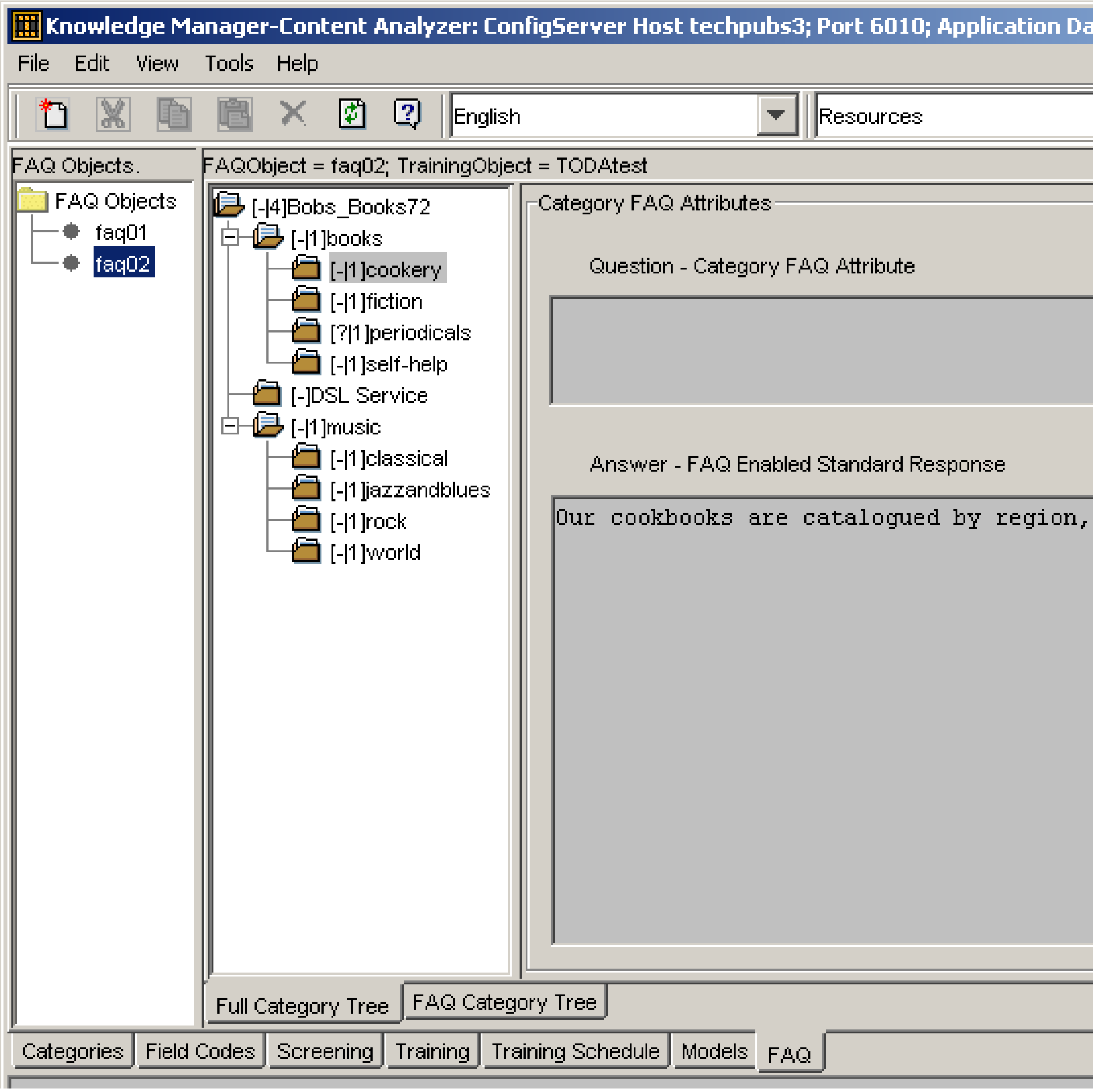
End of procedure

The `FAQ` tab contains the following panes:

- Left—Displays a list of all FAQ objects
- Center, with two subtabs:
  - Full Category Tree—Displays the entire category tree that serves as the source of the FAQ object that is selected in the left pane
  - FAQ Category Tree—Displays the selected FAQ object itself
- Right—These panes vary, depending on which of the center subtabs is selected.

Full Category Tree Subtab: Configuring the Category Tree

Expanding the category tree on the center pane produces the details shown in Figure 81.



Knowledge Manager-Content Analyzer: ConfigServer Host techpubs3; Port 6010; Application Da

English

Resources

FAQObject = faq02; TrainingObject = TODAtest

Question - Category FAQ Attribute

Answer - FAQ Enabled Standard Response

Our cookbooks are catalogued by region,

Figure 81: Center Pane of FAQ Object Tab

The following information displays:

- The strip above the center and right-hand panes displays the names of the FAQ object, the training object, and the model if any.
- The right-hand panes display the question and answer (standard response) associated with the category that is selected on the center pane.

The display on the center pane requires more explanation.

On the center pane, each category name appears in the form `[<statuscode>|<number>]<categoryname>`

<number> is the number of text objects (e-mails and other objects) associated with this category in the training object.

The status code combines the following information:

- Whether the category has the attributes that are required for it to be selected for use in the FAQ object.
- Whether the category is selected for use in the FAQ object.

Table 13 lists the possible statuses:

Table 13: Category Status in FAQ Objects

Status	Meaning
-	Category lacks the required attributes; therefore it cannot be selected.
?	Category has the required attributes but is not selected
+	Category has the required attributes and is selected

Answer

In the context of an FAQ object, an answer is a standard response that has the FAQ usage type selected and is specified as `Active`. You select this usage type on the `Additional` tab of the `New Standard Response` or `Edit Standard Response` dialog boxes. There are two ways to access these dialog boxes:

- On the `Categories` tab, as described in “Creating Standard Responses” on page 28.
- On the `FAQ` tab:
    - To edit an existing standard response, right-click a category and select `Edit Response`.
    - To create a new standard response, right- click a category and select `Set Response`.

Question

Create these questions on the `FAQ attribute` tab of the `New category` or `Category` dialog boxes. As with the usage type of a standard response, there are two ways to access these dialog boxes:

- On the `Categories` tab, as described in “Creating a Category Tree” on page 25.
- On the `FAQ` tab, right-click a category and select `Set/Edit Question`.

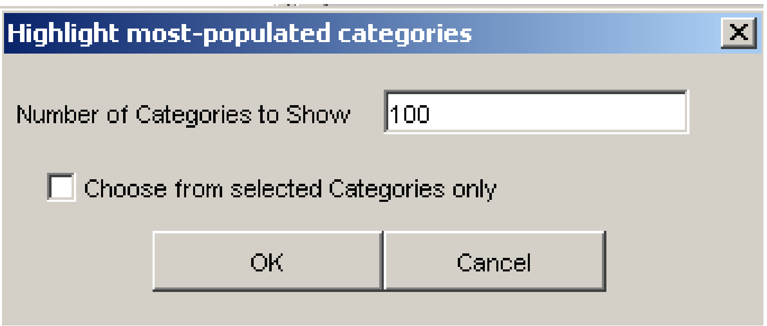
Selection

To select a category for inclusion in the FAQ object, right-click it and select:

- `Select` to select only this category.
- `Select With Children` to select this category and all subcategories under it.

Most-Used FAQs

Right-clicking anywhere on the `Full Category Tree` subtab and selecting `Highlight Top Categories` opens the `Highlight most-populated categories` dialog box, as shown in Figure 82.



Highlight most-populated categories

Number of Categories to Show: 100

□ Choose from selected Categories only

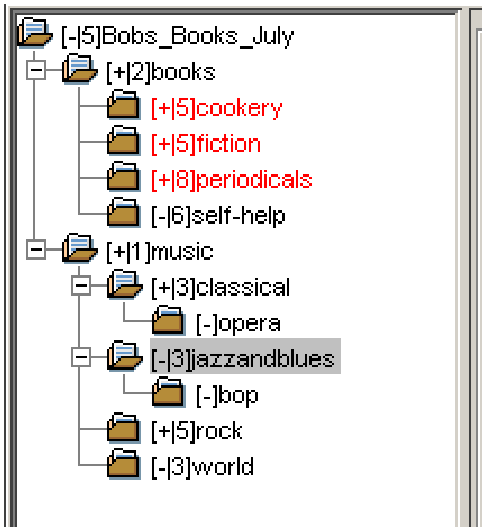
Figure 82: Highlight Most-Populated Categories Dialog Box

When you click `OK`, the dialog box closes, and the center pane highlights, in red, the top *X* categories, where:

- *X* is the number in the `Number of Categories to Show` box.
- *top* means having the largest number of associated text objects in the training object.

Select the `Choose from selected Categories only` check box if you want the calculation to consider only the categories that you have selected for inclusion in the FAQ object (the ones with `+` status).

Figure 83 shows an example.



```
[-|5]Bobs_Books_July
  [+|2]books
    [+|5]cookery
    [+|5]fiction
    [+|8]periodicals
    [-|6]self-help
  [+|1]music
    [+|3]classical
      [-]opera
    [-|3]jazzandblues
      [-]bop
    [+|5]rock
    [-|3]world
```

Figure 83: Three Most-Used FAQs

Figure 83 is the result of the following settings:

- `Number of Categories to Show` is set to 3.
- `Choose from selected Categories only` is selected.

If the `Choose from selected Categories only` check box were not selected, the highlighted categories would be `periodicals`, `self-help`, and `Bobs_Books_July`.

Note: If multiple categories are in a tie for inclusion in the top X, the categories that come first (highest) in the listing are the ones highlighted. In Figure 83 there are three categories with five text objects, but the top three has room for only two of them. So `cookery` and `fiction` are highlighted, but not `rock`, which is further down the list.

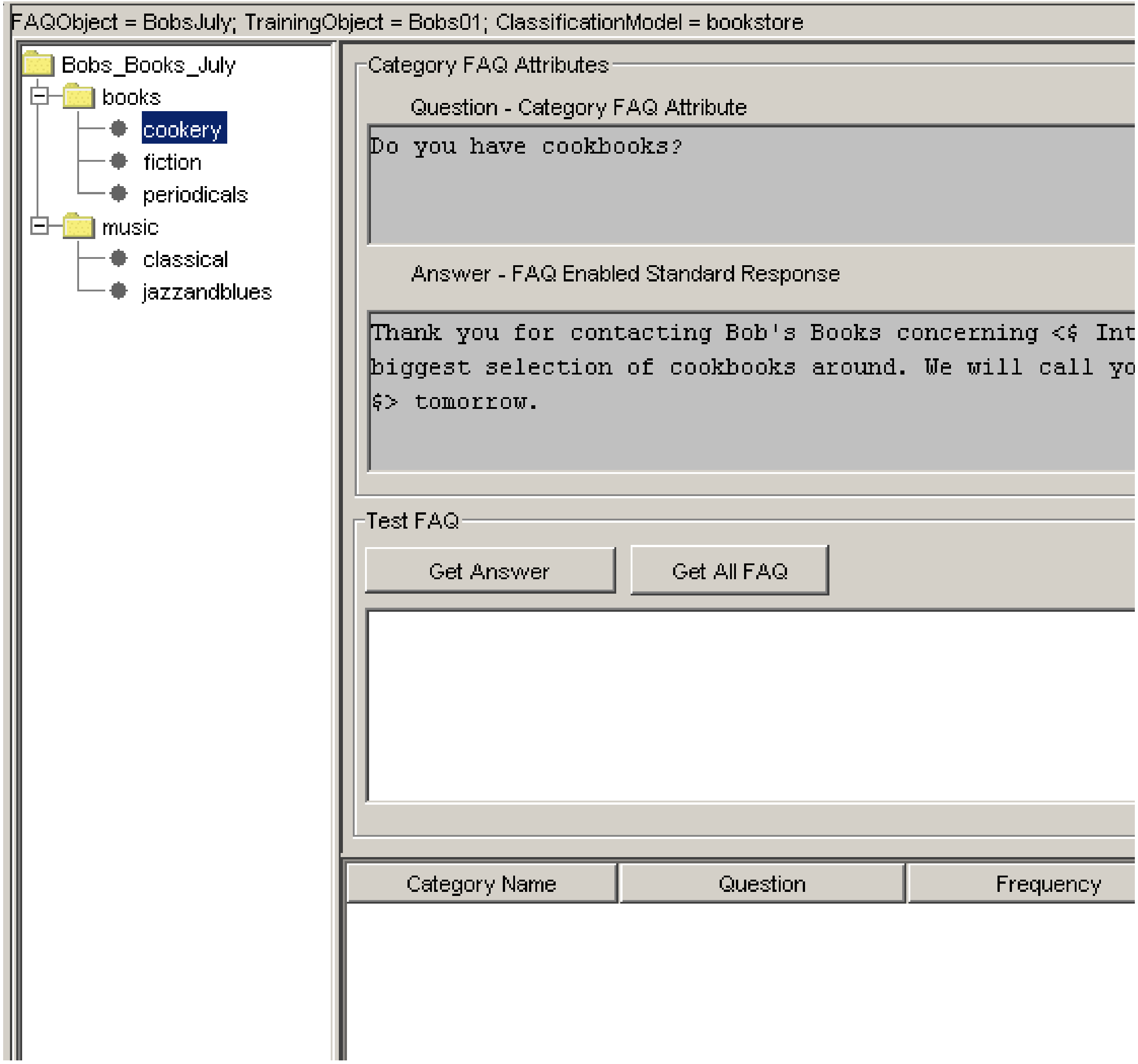
Shortcut Menu

The complete list of commands that appear when you right-click a category is as follows:

- `Select`—Select the category.
- `Select With Children`—Select the category and all subcategories under it.
- `Unselect`—Unselect the category.
- `Unselect With Children`—Unselect the category and all subcategories under it.
- `Expand Tree`—Self-explanatory.
- `Collapse Tree`—Self-explanatory.
- `Highlight Top Categories`—Open the `Highlight most-populated categories` dialog box.
- `Set/Edit Question`—Create or edit a question for the category.
- `Set Response`—Create a new standard response for the category. For the response to appear in the FAQ object, you must go to the `Additional` tab and select `Active` FAQ usage.
- `Edit Response`—Edit an existing standard response for the category.

FAQ Category Tree Subtab: Viewing and Testing

After you have finished configuring the category tree, you can display the result on the `FAQ Category Tree` subtab, as shown in Figure 84.


FAQObject = BobsJuly; TrainingObject = Bobs01; ClassificationModel = bookstore

Do you have cookbooks?

Thank you for contacting Bob's Books concerning <$ Int
biggest selection of cookbooks around. We will call yo
$> tomorrow.

Figure 84: FAQ Category Tree Tab

The FAQ Category tree is a preview of the FAQ object using a format similar to the sample described previously (see "Sample FAQ .jar File" on page 150).

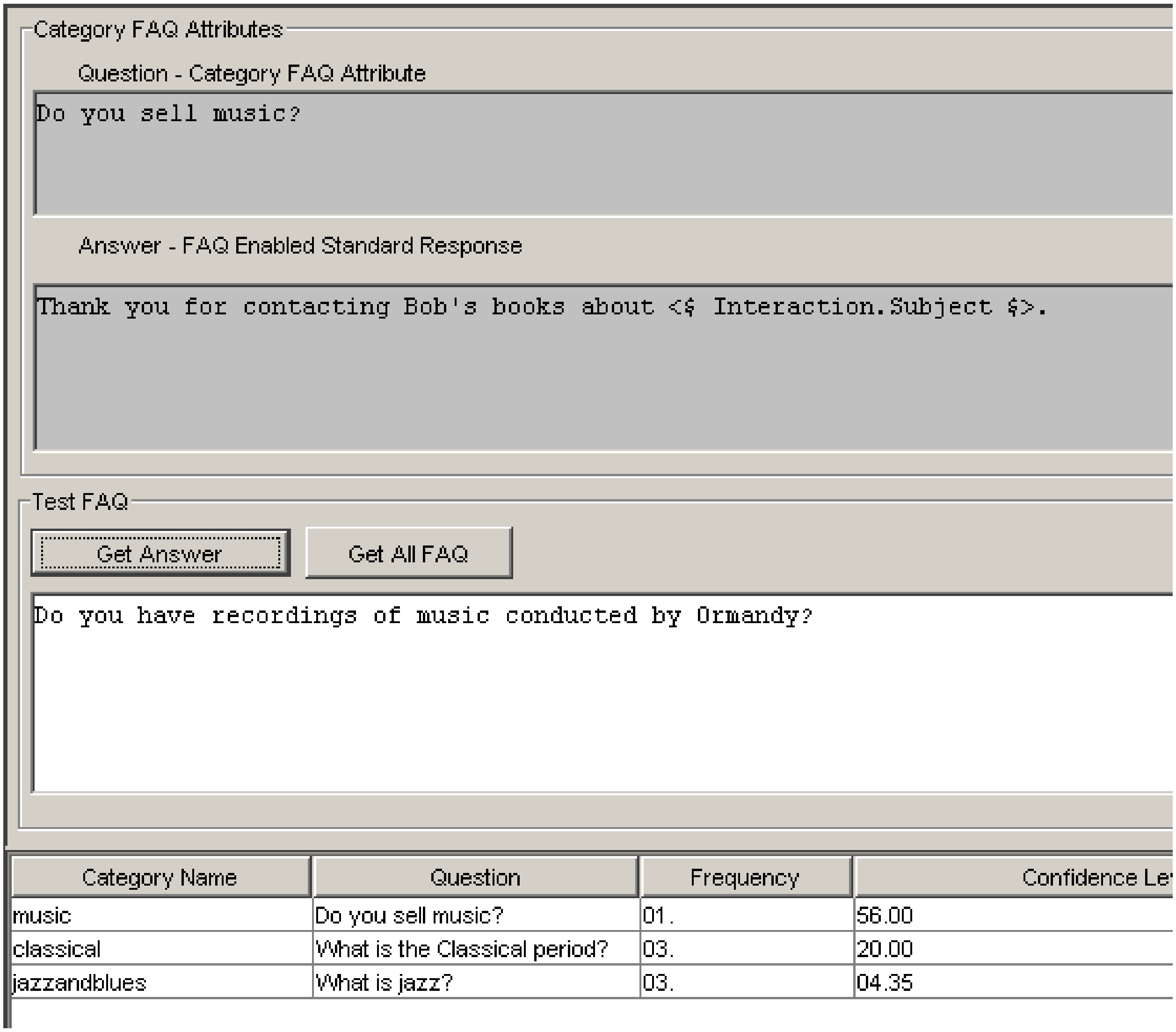
Selecting a category on the `FAQ Category Tree` subtab produces the following results:

- The `Category FAQ Attributes` area displays the question and answer for the category that is selected on the center pane.
- In the `Test FAQ` area, click `Get All FAQ` to display a list of all categories contained in the selected category (and its subcategories), along with the question and frequency for each

Notes: Text in the text box has no effect on this action.

Categories that have questions but no answers do not appear on this list.

- In the `Test FAQ` area, enter a sample question in the text box, then click `Get Answer` to display the answers that the system provides for the sample question at the selected level of the tree. (If the FAQ object does not include a model, this button is dimmed.) An example is shown in Figure 85.



Question - Category FAQ Attribute

Do you sell music?

Answer - FAQ Enabled Standard Response

Thank you for contacting Bob's books about <$ Interaction.Subject $>.

Do you have recordings of music conducted by Ormandy?

Category Name	Question	Frequency	Confidence Le
music	Do you sell music?	01 .	56.00
classical	What is the Classical period?	03.	20.00
jazzandblues	What is jazz?	03.	04.35

Figure 85: Test FAQ: Get Answer

Figure 85 shows the result of the question *Do have recordings of dance music conducted by Ormandy?* with the `Music` category selected (see Figure 84 for the category tree in question).

The bottom pane lists all answers supplied by the system, showing the category name, question, frequency (meaning the number of text objects associated with the category in the training object, also shown as <number> on the `Full Category Tree` tab display), and confidence rating (see “Reading and Understanding the Ratings” on page 134).

Generating an FAQ.jar File

Use the following procedure.

Procedure:
Generating and testing an FAQ.jar file

Start of procedure

1. Select an FAQ object on the left-hand pane, then do one of the following:
   - Select `Tools > Build FAQ JAR file`.
   - Right-click on the left-hand pane and select `Build FAQ JAR file`.

The `Build FAQ Jar file` dialog box appears, as shown in Figure 86.

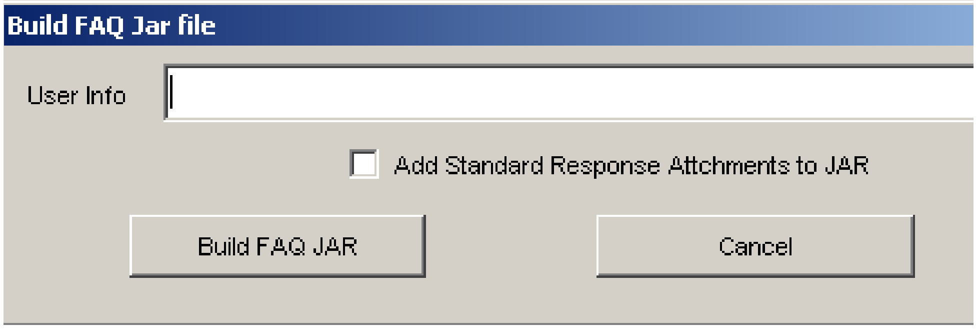


Figure 86: Build FAQ Jar File Dialog Box

2. In the `User Info` text box, enter the text that you want to appear in the title bar of the window that displays the FAQ object when you test it (see the procedure that immediately follows).
3. Select `Add Standard Response Attachments to JAR` if the answers in your FAQ object have attachments that you want to include in the FAQ.jar file.
4. Click `Build FAQ JAR`.
5. The `JAR File Chooser` dialog box appears. Enter a file name.
6. To test the resulting `FAQ.jar` file, make a copy of the `unit_test.bat` file that accompanies the FAQ example.
7. Edit it so that it targets the .jar file that you built instead of `faq_example.jar`.

Running the .bat file displays your FAQ object in the same way as the FAQ example (see “Sample FAQ .jar File” on page 150). Notice that the title bar of the FAQ object window displays the text that you entered in the `User Info` text box of the `Build FAQ Jar file` dialog box.

End of procedure

Next Steps

- You can now use the `FAQ.jar` file to create web applications.

Typical Response Times

This section describes some typical response times for Genesys Content Analyzer. For other functions of Knowledge Manager see “Typical Response Times” on page 83 of Chapter 2.

Unless otherwise stated, these figures are for a machine running Windows 2000 with two Pentium 4 processors and 1 GB of RAM.

- Deleting a training object takes approximately 4 seconds per 1,000 e-mails.
- Copying e-mails from one training object to another takes approximately 8 seconds per 1,000 e-mails.
- Creating a model (training time) naturally varies with the number of categories, number of e-mails, selected training quality, and selected cross-validation. As one example, for a training object containing 76 categories and 73,000 mails, with training quality set to level 1 and no cross-validation, training time is approximately 29 minutes. This is on a host running Windows 2000 with one 600 MHz processor and 1 GB of RAM.
- Cross-validation may increase training time significantly. Table 14 shows, for selected cross-validation levels, the factors of increase of cross-validation over no cross-validation.

Table 14: Increase of Training Time with Cross-Validation

Cross-Validation Level	Factor
3	1.9–2.9
5	4.0–4.7
10	8.0–9.0

For example, training a model at cross-validation level 3 takes between 1.9 and 2.9 times as long as the same model with no cross-validation.

- Classification performance depends on the size and nature (level of training quality) of the model. Table 15 shows some examples, all of which use a test object that contains 3,726 text objects.

Table 15: Classification Performance

Model	Host Machine	Classification Rate
• 72,734 text objects • Size = 285 KB • Quality = 1 • Cross validation with split to three sets	• Two Pentium 3 processors • 512 MB RAM • Windows operating system	31 objects classified per second
• 72,734 text objects • Size = 309 KB • Quality = 3 • Cross validation with split to 10 sets	• Two Pentium 3 processors • 1 GB RAM • Windows operating system	28 objects classified per second
• 72,734 text objects • Size = 309 KB • Quality = 3 • Cross-validation with split to 10 sets	• Four 350 MHz processors • 4 GB RAM • Solaris operating system	15 objects classified per second

- An FAQ object can process 30–50 classification requests per second on the model that contains 500–1,000 categories.

Chapter

3 Multi-Tenancy

This chapter describes multi-tenant configuration of eServices. It covers these topics:

Overview

Multi-tenancy is the capability of maintaining a pool of resources and controlling access to it by more than one tenant or business instance. For general information about multi-tenancy, see the *Framework 7 Configuration Manager Help* and the *Framework 7 Getting Started Guide.*

Configuration

The following eServices applications can be shared across tenants: Chat Server, Interaction Server, SMS Server, Social Messaging Server, Universal Contact Server (UCS) and Web API Server. For all other eServices applications you must deploy one instance per tenant.

In a multi-tenant environment, each eServices application must have one Tenant specified on its `Tenants` tab, with the following exceptions:

- UCS, Chat Server, Classification Server, SMS Server, and Interaction Server can have more than one Tenant specified.
- Web API Server can have more than one Tenant specified if the server itself is an object of load balancing.

This means that clients of Web API Server must include a `tenant` parameter in their requests, even in single-tenant environments, and even if the client itself is single-tenant.

Notes: Changes to the tenant specification of SMS Server and Social Messaging Server do not take place dynamically—you must restart these servers for changes to take effect.

The same applies to Interaction Server prior to release 8.1.2. However, in Interaction Server 8.1.2 and later, changes to the tenant specification take effect immediately.

Interaction Server

Since release 7.2, Interaction Server has had two possible Application types, `Interaction Server` and `T-Server`. With type `Interaction Server` it does not require an associated multimedia switch. Clients of Interaction Server should not expect it to specify any switch.

Note: Support of multi-tenancy requires the use of the `Interaction Server` application type.

For backward compatibility with release 7.1, clients are nevertheless able to associate Interaction Server with a multimedia switch, as follows: If a Tenant that is specified by Interaction Server contains a multimedia switch, clients associate Interaction Server with this switch. This works only if the Tenant contains exactly one multimedia switch.

See also “Interaction Server” on page 187 for other consequences of this difference in application type.

Integrated Capture Points

Capture Points support multi-tenancy by mapping each particular interaction to a tenant based on configured attributes. Refer to Chapter 9 on page 265 for more information about Capture Points functionality in Interaction Server.

Limitations

Observe the following limitations:

- Deploy at most one multimedia switch per tenant.
- Deploy at most one Interaction Server per tenant.
- Deploy at most one UCS per tenant.
- Deploy at most one UCS database per tenant. Databases can be shared between tenants. Figure 87 shows possible ways of arranging multiple tenants, Universal Contact Servers, and databases.

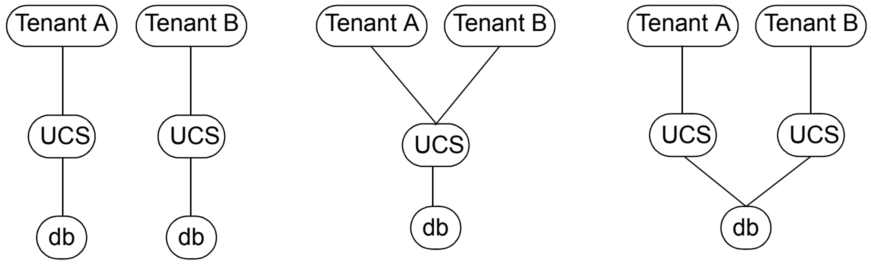


Figure 87: Multi-Tenant Configurations with UCS and the UCS Database

Chapter

4 Load Balancing and Backup Configuration

This chapter describes load balancing. It covers these topics:

Overview

Load balancing provides greater scalability and availability of service by providing multiple instances of certain eServices servers. The redundancy provided by load balancing helps prevent loss of data.

Load balancing can take place within a tenant or across tenants. There are three types of load balancing:

- Web API Server can balance among multiple instances of the following servers within a single tenant or across tenants:
  - Chat Server
  - E-mail Server
  - Interaction Server
  - Stat Server
  - UCS

  Web API Server does this using the Load-Balancing API. It keeps track of available server instances through Solution Control Server (SCS).

- Web API Server can balance among multiple instances of its own type (multiple Web API Servers).
- Interaction Server can balance, within a single tenant only, among multiple instances of Classification Server and E-mail Server. It can also balance among multiple instances of Universal Routing Server (URS) as long as all instances have the same strategy loaded. See “Interaction Server” on page 173 for details.

Web API Server

Web API Server uses SCS to monitor the state of all servers. Load balancing that involves Web API Server includes:

- Load balancing between instances of Web API Server.
- Load balancing between instances of the following servers: Callback Server, Chat Server, E-mail Server, Interaction Server, Stat Server, UCS.

Load-balancing components react to:

- Changes in the configuration; specifically, the connection settings of Web API Server and the status (enabled or disabled) of relevant applications.
- Application states (running or stopped) as reported by SCS.

Load balancing performs the following actions:

- For all instances of servers that are listed on Web API Server’s `Connections` tab:
  - **a.** The instance reports its status as RUNNING to SCS when it starts. The instance is then ready to process interactions.
  - **b.** The load balancer then marks the server’s status as RUNNING.
  - **c.** A web application can now use the service provided by that instance.
- If the server’s status changes to Service Unavailable:
  - It is excluded from the list of available servers and cannot be given in response to a web application’s request.
  - If the server’s status changes again to Started (Service Available), it returns to the list of available servers. In the case of Chat Server, it can continue to handle an online session that was established before the status changed to Service Unavailable.
- If the server shuts down:
  - It is excluded from the list of available servers and cannot be given in response to a web application’s request.
  - All interactions that it was handling are either lost (in the case of Chat Server) or handled by other instances of the same type of server (E-mail Server).
- If the server is disabled in the Configuration Layer:

  - It is likewise excluded from the list of available servers and cannot be given in response to a web application's request.
  - None of the clients that are already working with this instance of the server are affected.
  - As soon as all interactions that the server is handling end, the server can be shut down.
- If the server is enabled, it returns to the list of available servers.

Load-Balancing Configuration for Web API Server

In the example eServices configuration shown in Figure 88 on page 171, there are two instances of Web API Server, two of E-mail Server, and three of Chat Server, with the Chat Server instances divided between two tenants. All application servers are available to both Web API Servers.

Note: To enable cross-Tenant load balancing, you must add the Tenants to the `Tenants` tab of the Web API Server that you want to serve as the point of load balancing.

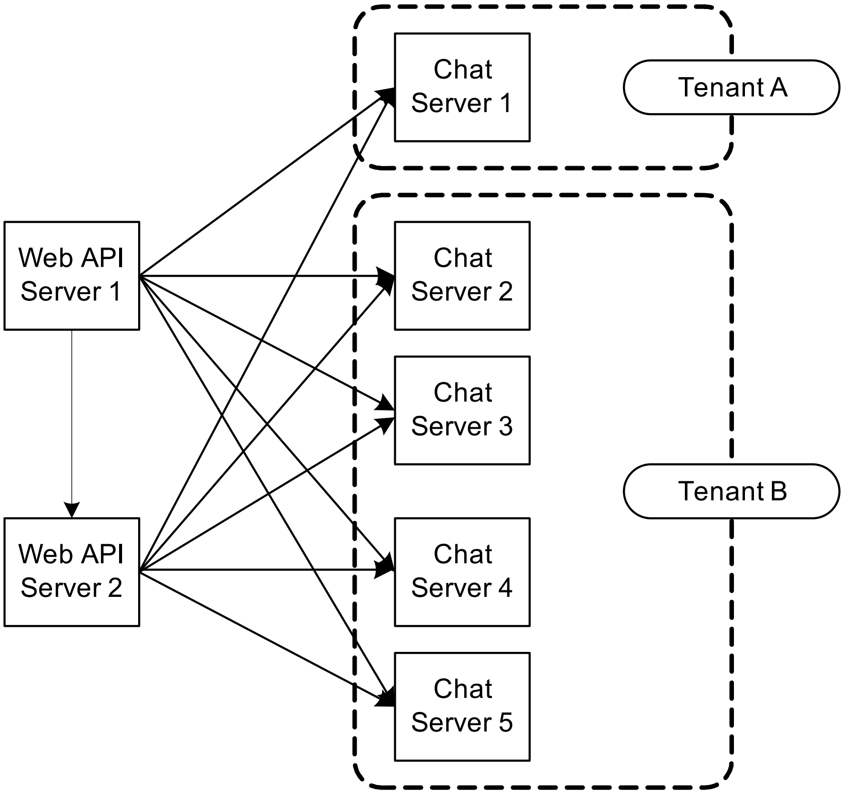


Figure 88: Load Balancing Configuration

To simplify the overall configuration, you can use application objects of type `Application Cluster` to group available servers in the configuration. An

Application Cluster is a configuration object that stores connection specifications. The eServices configuration wizard offers the opportunity to create Application Clusters. If you did not create an Application Cluster while running the wizard, you can add one manually, using the following procedure.

Procedure: Configuring an Application Cluster

Start of procedure

1. In Configuration Manager, locate the template `ApplicationCluster_<version-number>.apd` in the `templates` directory of your product CD and import it.
2. Use the template to create a new Application Cluster.
3. On the `Connections` tab, add connections to the servers that Web API Server will balance.
4. On the `Server` and `Start Info` tabs, enter arbitrary characters in the empty fields. Web API Server ignores this information, but there must be something in these fields for you to be able to save the Application.
5. In your Web API Server Application object or objects, add a connection to the Application Cluster.

End of procedure

Application Clusters are transparent to load-balancing components, and the configuration that uses connections through Application Clusters is equivalent to a configuration that uses direct connections between servers.

The configuration shown in Figure 88 is reconfigured in Figure 89 using an Application Cluster.

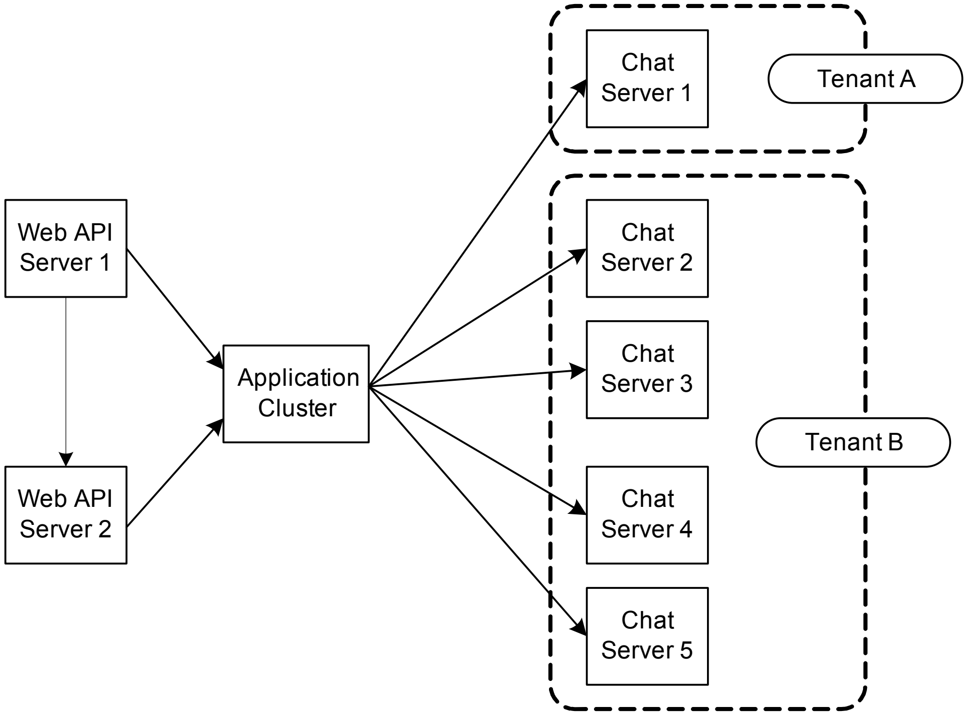


Figure 89: Load Balancing with Application Cluster

Load-Balancing API

To use eServices 8.1 load-balancing capabilities, use the Load-Balancing API in the web application.

The Load-Balancing API provides the following functionality:

- It can select a particular server instance from a set of instances of the specified server type.
- Upon the first request to a server instance, it can create an alias for the selected server instance and store it for future use.
- It can use the alias to obtain connection parameters (host name and port) of the server instance.
- It has access to configuration information.

The eServices 8.1 simple samples demonstrate how to use the eServices Load-Balancing API to develop web applications that use load balancing. For details, see the "About Web API Clients" chapter in the *eServices 8.1 Web API Client Developer's Guide.*

Interaction Server

Interaction Server can balance among multiple Universal Routing Servers and among eServices application servers. It can also connect to its database via multiple Database Access Points (DAPs). It does not use a user-accessible API for load balancing.

Balancing Universal Routing Servers

Interaction Server can balance among multiple instances of URS. This balancing proceeds by strategy: when an interaction reaches a strategy object in a workflow, Interaction Server selects (in round-robin fashion) from among all URS instances that have that strategy loaded.

To enable this type of load balancing, you must:

- Configure a connection from each URS to Interaction Server.
- For all participating URS instances, set the `agent_reservation` option to `true.`
- On the `Annex` tab of Interaction Server, set the `agent_reservation` option to `true` for the Application names of all participating URS instances.

> **Note:** For additional information on the `agent_reservation` option, see the "Configuration Options" chapter in the Universal *Routing 8.1 Reference Manual* and "System Availability and Redundancy" in the *Universal Routing Deployment Guide.*

- Choose each URS when activating each strategy in Interaction Routing Designer (IRD). Do this by shift-clicking all of the desired URS instances in the `Choose Routing Server` window of the Strategy Activation Wizard.

Suppose Interaction Server has two URS instances connected to it: URS 1 has Strategies A and C loaded, and URS 2 has Strategies B and C loaded. Then,

- For interactions that arrive at Strategy A in a workflow, Interaction Server submits them to URS 1.
- For interactions that arrive at Strategy B in a workflow, Interaction Server submits them to URS 2.
- For interactions that arrive at Strategy C in a workflow, Interaction Server balances between URS 1 and URS 2.

If any instance of URS shuts down, Interaction Server detects that this instance is not available. If any interactions were pending in the unavailable URS, Interaction Server resubmits them to an available URS that has the required strategies loaded.

Balancing eServices Application Servers

An *application server* is a server that Interaction Server invokes when triggered to do so by a routing strategy. For example, a Classify object in a strategy triggers Interaction Server to invoke Classification Server. To do this, Interaction Server uses a protocol called *External Services Protocol* or ESP; therefore these servers are also called ESP servers.

The application servers that Interaction Server can balance among are:

- Classification Server.
- E-mail Server.
- SMS Server.
- Social Messaging Server.

Note: E-mail Server, Chat Server, SMS Server, and Social Messaging Server have a dual role, as follows:

- When Interaction Server contacts these servers using ESP (for example, asking E-mail Server to generate an autoresponse), they are application servers (ESP servers) and Interaction Server is their client.
- When these servers contact Interaction Server, using the Interaction Management Protocol, and ask to submit an incoming interaction, they are media servers and clients of Interaction Server.

For more information on these protocols, see the *Genesys Events and Models Reference Manual.*

Balancing Directly

You can load balance by configuring connections from Interaction Server directly to each instance of the application server.

This method encounters a limitation with multiple custom ESP servers that provide different service types. Custom ESP servers generally have the Third Party Server Application type in the Configuration Layer. This means that if, for example, you have several custom servers handling fax interactions and several custom servers handling IM interactions, and you configure them for load balancing by making direct connections, Interaction Server will be unable to distinguish the ones that handle fax from the ones that handle IM, and will therefore send fax requests to the IM servers and vice versa. The solution for this is to use Application Clusters, described in the next section.

Balancing Using Application Clusters

Starting in release 8.0.0, eServices supports the use of Application Clusters for ESP servers. You can configure a separate Application Cluster for each type of ESP server. Then a client such as URS can call on the appropriate Application Cluster by name.

Note: Only one level of Application Clusters is allowed.

Balancing DB Servers

Interaction Server must work with only one database. However, Interaction Server supports multiple DAP connections to the same database through different DB Servers. You can configure this using multiple connections (DAPs) to one database.

Note: If for some reason the configuration of Interaction Server's DAP is erroneously changed to point to a different database, Interaction Server overlooks the error: it sends an error message saying that the configuration has changed, and continues to work with the original database. But if, in this same scenario, a switchover to a backup Interaction Server occurs, the backup Interaction Server has no way of knowing that the DAP configuration has (erroneously) changed, so it connects to the new database and starts sending requests to it.

It must be emphasized that Genesys recommends that you do not change the DAP configuration on the fly.

Capture Points

Capture Points are integrated into Interaction Server in the 8.0.2 release. Therefore, the application host in the Capture Point configuration is not taken into account and the host of Interaction Server is used.

A Capture Point can be configured as a primary/backup pair. In this case, the host of the primary application must be the same as host of the primary Interaction Server and the host of the backup application must be the same as host of the backup Interaction Server.

The primary Interaction Server (by configuration) will search for the primary Capture Point application and use its configuration to start the capture point. The backup Interaction Server (by configuration) will search for the backup Capture Point application and use its configuration to start the Capture Point. If there is no backup Capture Point configured, the backup Interaction Server will use the primary Capture Point application.

Generally, there is no need to configure a backup Capture Point application in Configuration Manager or Genesys Administrator; a backup Interaction Server will start the "backup" Capture Point instances.

Note: The JMS Capture Point is the only integrated capture point supported in Interaction Server 8.0.2. The File Capture Point is new in Interaction Server 8.0.21, the Database Capture Point is added in release 8.1.0, and the Web Service Capture Point is added in release 8.1.2.

SMS Server

SMS Server is scalable if necessary in order to deal with high volumes of inbound SMS messages arriving from an SMS Center.

Genesys suggests the following method of scaling:

- Take the set of telephone numbers from which SMS messages can arrive and divide it into subsets. For example, you can define one subset as numbers in the (650) area code, and a second one as numbers in the (925) and (510) area codes.
- Deploy one SMS Server for each subset.
- Be sure that every served telephone number belongs to exactly one subset.
- Have all of the SMS Servers running simultaneously, to serve all inbound traffic.

Backup Configuration

This section describes the types of backup configuration supported in eServices 8.1.

High-Availability Support

Table 16 lists the types of high-availability support possible for eServices servers.

Table 16: High-Availability Support

Component	Type of Support
Chat Server	Warm standby and load balancing[a]
Classification Server	Warm standby and load balancing[b]
Co-Browsing Server	Load balancing[a]
E-mail Server	Warm standby and load balancing[a,b]
Interaction Server	Warm standby
Interaction Server Proxy	Warm standby
SMS Server	Warm standby
Social Messaging Server	Warm standby
Training Server	Load balancing[c]

Table 16: High-Availability Support

Component	Type of Support
Universal Contact Server	Warm standby
Universal Contact Server Proxy	Warm standby
Web API Server	Load balancing[a]

[a] Supported through load balancing on Web API Server and SMS Server.

[b] Supported through ESP load balancing (see page 174) by Interaction Server.

[c] Supported in that it can process multiple training jobs. However, if an instance of Training Server becomes unavailable while it is processing a job, then a second running instance of Training Server will not pick up the job for processing. Instead, you must restart the first instance.

For general information on warm standby, see the *Framework 8.1 Architecture Help* and the *Framework 8.1 Deployment Guide.*

UCS and Interaction Server Proxies

Large numbers of custom desktop (ESP client) connections to Interaction Server and UCS may give rise to performance issues. To mitigate the issues caused by a high load on the server, Genesys introduced Interaction Server Proxy and UCS Proxy in release 7.6.1. Desktop applications can be configured to connect to these Proxies instead of the main server, significantly reducing the load on the server. For example, it is easier for the server to handle 20,000 clients that operate through ten proxies (only ten connections) than to handle the same 20,000 clients that each connect separately.

For a description of how to deploy these Proxy servers, see the "Manual Deployment-UCS Proxy, Interaction Server Proxy, and SMS Server" chapter of the *eServices 8.1 Deployment Guide.*

Because there are so many variables in deployment (choice of operating system, number of clients, details of architecture, and so on), it is not possible to provide exact guidelines as to when deploying a Proxy server would be advantageous. However it may be stated that you can anticipate performance issues when the number of clients exceeds 10,000.

Figure 90 diagrams a sample deployment using both Interaction Server Proxy and UCS Proxy. Each of the agent desktops in the diagram can represent several thousand agents.

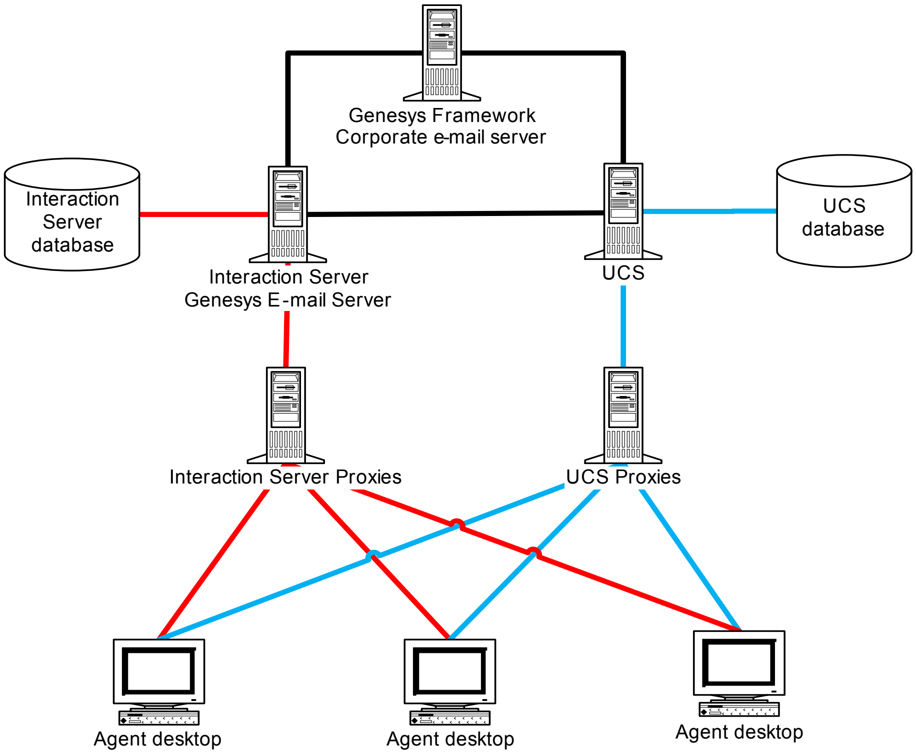


Figure 90: Sample Architecture Using Proxy Servers

Chapter

5

Ongoing Administration and Other Topics

This chapter describes ongoing administration, security, and other topics in these sections:

Administration

This section presents some recommendations for monitoring and adjusting your eServices configuration.

General Recommendations

Parameters to Check

Check that the following parameters do not significantly exceed their average values:

- Memory usage
- CPU load
- Number of handles for eServices-related processes (with Windows operating system)

Loading on Application Servers

Monitor the loading on application servers (Classification Server, E-mail Server). If application servers are being overloaded, do one or both of the following:

- For all routing strategies that process interactions with no agent involvement, adjust the limit on the number of interactions that Interaction Server can submit to Universal Routing Server (URS). You can set this limit for a strategy using the `max-submitted-interactions` option. See "Interaction Server Options" in Chapter 2 of the *eServices 8.1 Reference Manual.*
- Add instances of the required application server on other hosts.

Database Performance

If you are running Microsoft SQL, Genesys recommends as follows.

Microsoft SQL 2000

- In general, patch up to Service Pack 4.
- If Microsoft SQL is running on a machine with over 2 GB of RAM, use Windows' AWE (Address Windowing Extensions) mode. To avoid performance degradation, patch Microsoft SQL according to Microsoft's recommendation "FIX: Not all memory is available when AWE is enabled on a computer that is running a 32-bit version of SQL Server 2000 SP4" (see http://support.microsoft.com/?kbid=899761). This patch brings Microsoft SQL to version 8.00.2040.

Microsoft SQL 2005

- There is an issue that occurs with Microsoft SQL 2005: when the database is very large (on the order of one million interactions), there are periodically exceptions in the `Stat` service, and CPU activity rises to 100%.

  To avoid this issue, configure the UCS DAP as follows:

  a. Create a `settings` section.

  b. In this section, create an option called `prepare` and set its value to `false`.

  This DAP configuration applies to Microsoft SQL 2005 only; configuring the DAP in this way with Microsoft SQL 2000 degrades performance.

See also "Improving Database Performance" on page 212 in this chapter for further suggestions for improving the performance of the Interaction Server database.

UCS

Character Sets

Oracle

The character set `WE8ISO8859P1` does not have any representation of characters in the range 128–159. Because of this, with an Oracle database, attempting to save characters in this range in a column of type `NCHAR` or `NVARCHAR` results in corrupted data. Genesys recommends that you set the Oracle `NLS_CHARACTERSET` parameter to `WE8MSWIN1252` instead of `WE8ISO8859P1`. `WE8MSWIN1252` is a superset of `WE8ISO8859P1`, so there will be no data loss.

For support of nonlatin charsets, use the following parameter settings in Oracle:

```
NLS_CHARACTERSET AL32UTF8
NLS_NCHAR_CHARACTERSET AL16UTF16
```

DB2

DB2 must use the UTF-8 codeset for the UCS database.

Access to Configuration Server

Be sure to run UCS with a user that has write access to the Configuration Server database for all the tenants associated with this UCS (that is, the user specified on the `Security` tab of the UCS Application object).

This means that UCS does not support Configuration Server Proxy version 8.0.2 and earlier, which has only read access to the Configuration Server database. UCS does support Configuration Server Proxy version 8.0.3 and later.

Contact Identification and Creation

If UCS cannot identify a contact, its default behavior is to create a new contact record. For description of this behavior and ways to customize it, see Chapter 6, “Contact Identification and Creation,” on page 227.

Database Tuning for Attachments

UCS uses the `Content` field of the `Document` table to store attachments; also, the `Content` field of the `ixnContent` table stores raw e-mails, including attachment data. If you plan to store large attachments (bigger than 5 MB), you should tune the database according to the recommendations of your database vendor.

For example, increasing the block size of database files for these fields can greatly enhance performance in access and storing of large attachments, at the cost of a slight loss of performance with smaller ones. Also, some databases offer the ability to partition data according to specified criteria. Both tables have a `theSize` column that you can use to do such partitioning. This could enable you to store small attachments in a specific file and large ones in another, for example.

Refer to the tuning guides of your database vendor for more information.

User Other than Schema Owner

To enable a user who is not the schema owner to run UCS with an Oracle database:

1. Open the script `ucs_oracle_create_additional_user.sql,` which is located in the `sql_scripts` directory of UCS's starting directory.
2. Locate the following lines:

   ```
   ucs_user := 'UCS_RUNTIME';
   ucs_db_creator := 'UCS_OWNER';
   ```
3. Replace `UCS_RUNTIME` with the name of the non-owner that you want to be able to run the database.
4. Replace `UCS_OWNER` with the name of the Oracle user that created (is the owner of) the schema.
5. Run the script from an Oracle account that has SYSDBA privileges. This creates the user identified in Step 2 and creates synonyms of all objects so that they are accessible to the newly created user.
6. Adjust the DAP that UCS uses (or create a new DAP if you want to retain the existing one), as follows:
   - On the DB Info tab, set the `User Name` equal to the user identified in Step 2, and set the password equal to the user name. This is how the script creates the user and password. If you want a different password you must modify it in Oracle.
   - On the `Options` tab, `settings` section, create the `db-schema-name` option. For its value, enter an upper-case version of the name of the Oracle user that created the UCS database schema (the user identified in Step 3).
7. After completing these steps on the main database, repeat them for the archive database.

Note: When using UCS with a limited DB user, UCS is not able to check for the existence of table indexes. The log will display a message warning that indexes do not exist, but you can ignore this warning. Using a limited user does not prevent DB from using indexes.

Required Queries

Users of the UCS DB must have permission to run certain queries on the database, for the following purposes:

- List user's tables—required for launching UCS
- Read NLS (national language support) configuration—required for normal operation of UCS
- List user's indexes—required for normal operation of UCS
- Read the configured maximum number of cursors—required for normal operation of UCS, on Oracle only

The following examples use `CONTACTSERV_USER` and `CONTACTSERVARC_USER` as the names of users of the main and archive databases respectively:

Oracle

To list user's tables:

```
SELECT T.TABLE_NAME, T.COLUMN_NAME, DECODE (T.NULLABLE, 'N', 'NO',
'YES') AS IS_NULLABLE FROM ALL_TAB_COLUMNS T WHERE
UPPER(T.OWNER)='CONTACTSERV_USER' ORDER BY T.COLUMN_ID
```

To read NLS configuration:

- In 8.1.0:

  ```
  SELECT * FROM sys.props$ WHERE name LIKE 'NLS%CHARACTERSET%'
  ```

- In 8.1.1 and later:

  ```
  SELECT * from SYS.NLS_DATABASE_PARAMETERS WHERE PARAMETER LIKE
  'NLS%CHARACTERSET%
  ```

To list user's indexes:

```
SELECT I.INDEX_NAME, IND.COLUMN_NAME as COLUMN_NAME, I.UNIQUENESS as
IS_UNIQUE FROM USER_INDEXES I, USER_IND_COLUMNS IND WHERE
I.INDEX_NAME = IND.INDEX_NAME AND I.GENERATED='N' ORDER BY
INDEX_NAME
```

SQLServer

To list user's tables:

```
exec sp_tables  @table_name = null,  @table_type = '''TABLE'''
```

To read NLS configuration:

```
SELECT DATABASEPROPERTYEX ('CONTACTSERV_USER', 'Collation')
```

To List user's indexes:

```
SELECT i.name INDEX_NAME, c.name AS COLUMN_NAME, CASE WHEN (i.status
& 2)<>0 THEN 'true' ELSE 'false' END AS IS_UNIQUE FROM sysindexes i
INNER JOIN sysindexkeys k ON i.id=k.id AND i.indid=k.indid INNER
JOIN syscolumns c ON c.id=i.id AND c.colid=k.colid WHERE
INDEXPROPERTY (i.id , i.name , 'IsAutoStatistics' ) = 0 ORDER BY
index_name
```

DB2

To list user’s tables:

```
SELECT TABNAME AS TABLE_NAME, COLNAME AS COLUMN_NAME, TYPENAME AS
TYPE_NAME, LENGTH AS TYPE_LENGTH FROM syscat.columns WHERE
UPPER(TABSCHEMA)='CONTACTSERV_USER' ORDER BY COLNO
```

To read NLS configuration:

```
SELECT TYPENAME, CODEPAGE FROM syscat.datatypes WHERE TYPENAME LIKE
'%CHAR%' OR TYPENAME LIKE '%LOB%'
```

To List user’s indexes:

```
SELECT INDNAME AS INDEX_NAME, COLNAMES AS COLUMN_NAME, UNIQUERULE AS
IS_UNIQUE FROM syscat.indexes WHERE
UPPER(TABSCHEMA)='CONTACTSERV_USER' ORDER BY INDNAME
```

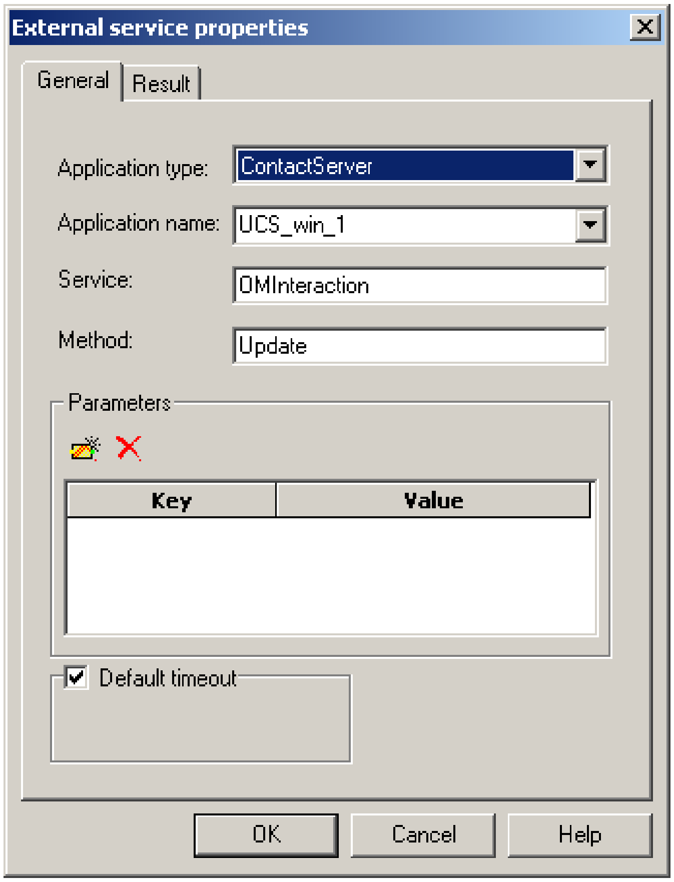
Updating Interaction Data in a Routing Strategy

It is possible for a routing strategy to update certain interaction attributes. To ensure that such an update is also made in the UCS database, use the following procedure.

Procedure: Updating interaction data in the database from the routing strategy

Start of procedure

1. Log in to Interaction Routing Designer and open your strategy for editing.
2. In the strategy, create an `External Service` block to call the `OMInteraction` service, using the `Update` method. See Figure 91 for a sample configuration.



Application type: ContactServer

Application name: UCS_win_1

Service: OMInteraction

Method: Update

Key	Value

☑ Default timeout

Figure 91: ExternalService Object

3. Save and reload the strategy.

End of procedure

Interaction Server

Be aware of the following:

- Use CCPulse to monitor interaction queues (in interaction workflows) for signs of problems with routing strategies. If the number of interactions in a queue increases abnormally, it may be a sign that the strategy that processes interactions from that queue is not loaded in Universal Routing Server.
- Depending on the amount of configuration objects and the volume of the interactions stored in the Interaction Server database, it might take considerable time for Interaction Server to start up and shut down.
- As described in "Interaction Server" on page 166, Interaction Server has two possible Application types in the Configuration Layer. `Interaction Server` is the normal type; the `T-Server` type is also available for backward compatibility. Be aware that an Interaction Server 7.6 or later of

type `T-Server`, upon startup, will make two attempts to connect to Configuration Server. The first attempt will generate `trace`-level alarms (about a missing application of type: Interaction Server) that you should ignore. The second attempt will succeed.

- If you want to use the Dynamic Workflow Management functionality, be sure to run Interaction Server with a user that has write access to the Configuration Server database for all of the tenants associated with this Interaction Server (that is, the user specified on the Security tab of the Interaction Server Application object).

  In this situation Interaction Server does not support Configuration Server Proxy, which has only read access to the Configuration Server database.

"Improving Database Performance" on page 212 in this chapter presents suggestions for improving the performance of the Interaction Server database.

Classification Server

Modifying any of the following may have repercussions elsewhere in the system:

- Categories
- Standard responses
- Field codes
- Screening rules

If you modify any of these objects, it would be prudent to check any compiled strategies that use the following:

- Acknowledgment
- Attach Categories
- Autoresponse
- Chat Transcript
- Classify
- Classify Switch
- CreateEmailOut
- CreateSMS
- Forward
- Multi Screen
- Screen

Perform this check by recompiling the strategies in question. If this is not possible, monitor the Classification Server log for errors related to screening rules and UCS logs for errors related to rendering of standard responses.

SMS Server

SMS Server supports the following SMPP v3.4 operations:

- BIND_TRANSCEIVER

  BIND_TRANSCEIVER_RESP

  The purpose of the SMPP bind operation is to register an instance of an ESME (External Short Messaging Entity) with the SMSC (Short Message Service Center) system and request an SMPP session over this network connection for the submission or delivery of messages.

- UNBIND

  UNBIND_RESP

  The purpose of the SMPP unbind operation is to deregister an instance of an ESME from the SMSC and inform the SMSC that the ESME no longer wishes to use this network connection for the submission or delivery of messages.

- SUBMIT_SM

  SUBMIT_SM_RESP

  This operation is used by an ESME to submit a short message to the SMSC for onward transmission to a specified short message entity (SME).

- DELIVER_SM

  DELIVER_SM_RESP

  DELIVER_SM is issued by the SMSC to send a message to an ESME. Using this command, the SMSC may route a short message to the ESME for delivery.

- ENQUIRE_LINK

  ENQUIRE_LINK_RESP

  This message can be sent by either the ESME or SMSC and is used to provide a confidence check on the communication path between an ESME and an SMSC.

The protocol referred to in this section is described in “Short Message Peer to Peer Protocol Specification v3.4, 12-Oct-1999 Issue 1.2,” which can be downloaded at `http://smsforum.net/`.

Security

Genesys makes the following security recommendations for deploying eServices:

- Put Web API Server in the DMZ.
- Put all other eServices components in the internal network.
- Open ports in the firewall between the DMZ and the internal network to allow Web API Server to connect with other eServices components. Table 17 lists each component and the port to open.

Table 17: Port Types in Firewall

Server	Port
Configuration Server	Default port on `Server Info` tab
Message Server	Default port on `Server Info` tab
Solution Control Server.	Default port on `Server Info` tab
Interaction Server	Default port on `Server Info` tab
Chat Server	Port specified by the `webapi-port` option in the `settings` section. If not specified, default port on `Server Info` tab.
E-mail Server	Port specified by the `webapi-port` option in the `settings` section
Stat Server	Default port on `Server Info` tab
Co-Browsing Server	HTTPS
UCS	Port specified by the `ucsapi` option in the `ports` section

- Open a port in the firewall to allow Solution Control Server to connect to the Local Control Agent (LCA) located on the host of Web API Server.
- Open ports in the firewall to allow SMS Server to connect to the SMSCs specified in the SMS Server's configuration

Limitations

This section describes recommended limitations.

General

If you set up periodic time synchronization on your network, be sure to use a dedicated Network Time Protocol (NTP) server. Without NTP, synchronization can cause errors in E-mail Server and UCS.

Knowledge Manager

Basic Limitations

For Knowledge Manager, observe the following limitations:

- Categories: 3,500 categories
- Standard responses: 50 per category
- Attachments: 20 per standard response, 5 MB per attachment
- Field codes: 1,000
- Screening rules: 1,000
- Training objects: 200,000 e-mails, 20 KB per e-mail, 510 B for each e-mail's subject field

Screen Resolution

For Knowledge Manager to operate correctly, you must set a minimum screen resolution of 1280 x 1020.

Memory Allocation

You can adjust the memory size that Java allocates for Knowledge Manager processes by using the parameter `-Xmx1000m` in the following line in the .bat file:

```
start "Knowledge Manager" /b "%GES_HOME%\jre\bin\javaw" -Xmx1000m
-classpath %CLASSPATH% -Djava.security.manager
-Djava.security.policy=.\java.policy Genesys.iknow.manager.TM_start %*
```

`-Xmx1000m` means that 1,000 MB is allocated for Knowledge Manager; changing this number changes the allocation. The following considerations bear on adjusting this parameter.

- In some cases, Knowledge Manager does not work when you attempt to launch it from a machine that has a remote connection to the host of Knowledge Manager. As a workaround, lower the value of `-Xmx1000m` to `-Xmx512m`. In the unlikely event that this does not work, try a further decrease to `-Xmx256m`.
- You may want to adjust this parameter for better performance with large training objects (see "Large Training Objects" on page 124), or before importing or exporting large files. For DB2 and Oracle, see also the recommendations in "Adjusting Database Configuration" below.

However, if this parameter is too low, it may impose limits on Knowledge Manager lower than those listed in "Basic Limitations" above. If so, you can consider increasing this parameter.

For a similar issue with UCS see "UCS" on page 192.

Adjusting Database Configuration

To prevent problems when using Knowledge Manager to import or export very large files, Genesys has the following recommendations about database configuration.

- For DB2, do as follows:
  - **a.** In the DB2 Control Center, select `System > Instance > Databases`.
  - **b.** Select the database desired.
  - **c.** Right-click the desired database.
  - **d.** In the resulting shortcut menu, select `Configure`.
  - **e.** In the resulting dialog box, select `Logging`.
  - **f.** Increase the number of files and/or file size.
- For Oracle, use Enterprise Manager to increase the number of rollback segments. Refer to Oracle documentation for details.
- For Microsoft SQL, no special configuration is needed.

UCS

By default Java allocates 512 MB of memory for UCS processes. This may not be enough when importing a large archive into Knowledge Manager, in which case an `OutOfMemoryError` exception may occur.

In such a case you can temporarily increase the amount of memory allocated for UCS processes by adjusting the Java option `-Xmx512m` to `-Xmx1000m`, as described in the following sections.

Windows

In the `ContactServerDriver.ini` file:

```
[JavaArgs]
```

```
-Xmx1000M
```

Unix

In the `contactServer.sh` file, change `-Xmx512M` to `-Xmx1000M` in the appropriate section, as in the following:

```
# Sun JVM
$JAVACMD -Xmx1000M -server -XX:+UseConcMarkSweepGC -XX:+UseParNewGC
-XX:+ExplicitGCInvokesConcurrent -Xbootclasspath/p:${JVM_ADDCHARSETS}
-Djava.rmi.dgc.leaseValue=60000 -Djava.library.path="${GMLLIB}"
-Dgenesys.cfglib="${LICENSE}" -Dtkv.multibytes="true"
-Djava.util.logging.config.file=${BASEDIR}/cv/jul.properties
-Dorg.restlet.engine.loggerFacadeClass=org.restlet.ext.slf4j.Slf4jLogge
rFacade -classpath "${COMP_CLASSPATH}"
com.genesyslab.icc.contactserver.ContactServerEngine $*
else
# Ibm JVM
$JAVACMD -Xmx1000M -XX:+UseConcMarkSweepGC -XX:+UseParNewGC
-XX:+ExplicitGCInvokesConcurrent -Xbootclasspath/p:${JVM_ADDCHARSETS}
-Djava.rmi.dgc.leaseValue=60000 -Djava.library.path="${GMLLIB}"
-Dgenesys.cfglib="${LICENSE}" -Dtkv.multibytes="true"
-Djava.util.logging.config.file=${BASEDIR}/cv/jul.properties
-Dorg.restlet.engine.loggerFacadeClass=org.restlet.ext.slf4j.Slf4jLogge
rFacade -classpath "${COMP_CLASSPATH}"
com.genesyslab.icc.contactserver.ContactServerEngine $*
```

For a similar issue with Knowledge Manager, see “Memory Allocation” on page 191.

Chat Server

General Limitations

Table 18 lists recommended limitations for a Genesys Chat solution running on a single host with two Intel Xeon 3.0GHz processors.

Table 18: Chat Server Limitations

Item	Maximum
Message size	4 KB (Genesys Desktop limitations)[a]
Transcript size	54 KB (Genesys Desktop limitation)[a]

Table 18: Chat Server Limitations (Continued)

Item	Maximum
Concurrent sessions (in a realistic simple scenario)	1000 per Chat Server
Messages per second	50 (rare temporary peaks up to 150)
Sessions opened and closed per second	10 (rare temporary peaks up to 30)

a. Chat Server does not have these restrictions.

In Table 18, *optimum* means without significant delay.

Chat Server also has a timeout that you can configure using the `user-register-timeout` option (default value 30 seconds). This is the maximum time between opening a socket and either of the following:

- Receiving a registration over the socket
- Receiving a flex packet over the socket

This timeout prevents keeping unused connections open.

Connection Delay with Antivirus

It may take some time (up to several minutes on some UNIX Platforms) for Chat Server to connect to an unopened port on a Windows host on which an antivirus program is running. For example, if Chat Server is running on Linux and is trying to connect to an inactive UCS instance, it could take up to three minutes for Chat Server to detect that the listening port is not open.

Matching Contact Attributes

When a home user asks to open a chat session, the web interface gets him or her to fill in some identifying information, such as e-mail address, phone number, first name, last name, an so on.

This identifying information becomes a part of the *user data* that is associated with the interaction. The web interface relays this user data to Chat Server, and Chat Server sends it to UCS.

UCS then looks to see if the home user matches any of the people that it has represented as contacts in its database. It does this according to the following algorithm:

Table 19: UCS Search Order for Contact Records

Attribute Name	Search Order
EmailAddress	0
PhoneNumber	1
FirstName	2
LastName	2

UCS is hard-coded to use this algorithm with interactions coming from Genesys media servers, namely e-mail, chat, and callback interactions. For other media the algorithm can be customized.

So if the user data includes an attribute called `EmailAddress`, UCS looks for a contact in its database whose `EmailAddress` attribute has the same value as the user data attribute. (For details on the structure of this part of the UCS database, see the "Contact Package" chapter in *eServices 8.0 Selected Conceptual Data Models for the UCS Database.*) The name of the user data attribute must be exactly `EmailAddress`—if it is `email_address` or anything else, UCS will not try to match its value with the stored value of `EmailAddress`.

If UCS finds no matching contact, it creates a new one using the user data (see Chapter 6, "Contact Identification and Creation," on page 227 for more information).

For either a matching contact or a new one, UCS sends the following, as data about the contact for this interaction, to Chat Server:

- The matched attribute (if not e-mail address, then phone number, and so on).
- The attribute `ContactID`.
- All other attributes of this contact that UCS has stored in its database, except:
- If any user data has an attribute name that matches an attribute name in the UCS `Contacts` table, UCS returns the value of the attribute from the user data, not the value from the `Contacts` table. It does not modify the value in the `Contacts` table.

The last point can cause a problem, as in the following example:

1. Home user Steve Jones wants to open a chat session. In the web interface, he types in his correct e-mail address sjones@here, then erroneously types his first name as Speve.

2. UCS finds a contact record for sjones@here.
3. UCS returns to Chat Server data about an existing contact whose e-mail address is sjones@here and whose first name is Speve. UCS still has the correct first name Steve in its database, but the user data, with the erroneous Speve, preempts the correct data for the purposes of this chat interaction.
4. The system uses the user data to generate the message prompt that marks the home user in the chat display. As a result, the chat session displays something like the following:

   ```
   14:52:20 SpeveJ has joined the session
   14:52:30 SpeveJ > Hi.
   ```
5. The Agent Desktop displays the incorrect first name (in the user data on the lower left pane) and the correct first name (on the Customer Records pane on the right). The agent sees the incorrect first name and opens the chat session by typing, “Hello Speve, how can I help you?”
6. The interaction passes through a strategy that generates an automatic response, which opens, “It was good chatting with you, Speve.”

To avoid this type of problem, be sure that the system (including strategies and desktop) as well as its users refer to the UCS database, rather than user data, for contact attributes. In the example just cited, the agent must be sure to look at the Customer Records (right-hand) pane of the Desktop for the name of the contact. However, it is not possible to avoid the use by the system of user data to generate the message prompt (`SpeveJ` in the example).

It is also advisable to closely monitor the inventory of contact attributes that can become user data.

Multilingual Processing

Genesys Chat can process multiple languages simultaneously, including:

- Chat transcript messages, with no restrictions. The data is transferred in UTF-16.
- Attached data (such as first name, last name, subject, and so on) and ESP messages (submitting messages to chat session from the strategy). The data is transferred in UTF-8. There are the following limitations:
  - Chat Server, Interaction Server, and URS must be deployed on a UNIX platform running under the UTF-8 locale. Windows is not supported.
  - Unless UCS is running on UNIX with the UTF-8 locale, its startup script must be configured to use `-Dfile.encoding=UTF-8`.
  - Desktops—Only Genesys Desktop supports UTF-8, which also must be configured in the startup script. Interaction Workspace does not support UTF-8 at present.

  - Web API Server—To handle multiple languages, Web API Server must also be configured with the UTF-8 encoding. Web page headers must be updated to support UTF-8. Web samples must be modified to use `request.getParameter(<param name>)` rather than `i18nsupport.GetSubmitParametr(request, <param name>)`.
  - Routing strategies cannot contain string constants in multiple encodings, so such data (for example, for sending a message to a chat session) must be obtained from some external source, such as a database, in UTF-8 encoding.

Attached Data

This refers to data that is attached to the interaction during processing, for example by the media server that creates the interaction, or by a routing strategy. Interaction Server places no limit on such data, but its size does have an effect on performance. Genesys recommends that you limit attached data to a maximum of 16 KB per interaction.

In general, attach only data that is needed for routing and/or reporting; do not attach data if you are not sure it will be used.

Interaction Server

You should be aware of the following:

- Interaction Server does not support the following requests:
  - RequestQueryServer
  - RequestQueryLocation
  - RequestDeletePair (when URS sends this request after RequestRouteCall)
- It is not desirable to run Interaction Server in an environment in which servers and clients differ as to the codepages used (by operating systems or databases). In such an environment, characters of non-Latin alphabets may appear as the symbol ? (question mark) in log files and in applications with a user interface, such as Agent Desktop. The functionality of other features of the solution may also be restricted or compromised.
- Making an on-the-fly change to the host or port specification (on the `Server Info` tab) of a backup Interaction Server will cause it to exit.
- When Interaction Server sends a database request right before disconnecting from DB Server, and the request executes after disconnecting, Interaction Server fails to generate events to clients for submitted interactions.

E-mail Server

Attachments

There is no limit on the size of attachments to e-mails. You can use the `maximum-msg-size` option to limit the overall size of incoming messages (that is, the total size of all message parts, including the body and any attachments).

Compatibility with UCS

E-mail Server 8.1.2 can work only with UCS 8.1.1 or later (however UCS 8.1.1 can work with any version of E-mail Server).

Co-Browsing Server

Co-browsing clients support the HTTPS protocol only.

Unicode Character Support

Although UCS supports Unicode character sets, other components of eServices and of the Genesys suite (in particular, Interaction Server and URS) do not. This means that interactions that use a Unicode character set may be corrupted. Specifically, what may be corrupted is any part of the interaction's data that is handled by Interaction Server or URS (or any other component that does not support Unicode). This includes attributes such as Subject, FirstName, and LastName. It does not include the body of the interaction, which is handled by UCS only.

The following scenario provides an example of how this corruption can happen:

1. URS processes an interaction that includes Unicode user data, such as Subject. Because URS does not support Unicode, the Subject and other user data is corrupted.
2. UCS receives `RequestStopProcessing,` either from URS or the agent desktop.
3. UCS saves the interaction's user data (this is done in case the user data has changed during processing), copying certain properties from the user data into the corresponding fields of its `Interaction` table.

   This user data includes the interaction's subject, the value of which is copied into the `Subject` attribute. But the user data was corrupted during processing by URS, so the corrupted data is now stored in the UCS database.
4. If the corrupted subject data is used to compose another e-mail (such as reply or redirect), the subject of the new e-mail is also corrupted.

As a workaround for this scenario, you can modify the applicable strategy so that it deletes the Subject user data before it issues `RequestStopProcessing`.

E-mail Server: Advanced Topics

This section provides information on various topics relevant to E-mail Server.

Handling Unparsable E-Mails

If E-mail Server is unable to parse an incoming e-mail, it creates a new e-mail interaction (a "wrapping message") with the following characteristics:

- The header is the same as the header of the original, unparsable e-mail.
    - If the header of the original e-mail is unparsable, the subject of the new interaction is `Unknown subject`.
    - If the From address of the original e-mail is not valid, the From address of the new interaction is `unknown@<default_domain>`, where <default_domain> is the domain specified by the `default_domain` configuration option of the E-mail Server application.
- The text of the new interaction is `Error encountered during preprocessing of this message + <reason_for_failure> + Original Incoming Email is attached to this Email`.
- The original e-mail is attached to the new e-mail.
- The new e-mail has an attached key-value pair, whose key is `_WrappingMessageReason` and whose value is a text string that describes the reason for creating the wrapping message.

JavaMail Properties

E-mail Server uses the JavaMail API library 1.4. JavaMail can make use of numerous properties, which are documented at the following locations:

- Environment properties: http://java.sun.com/products/javamail/JavaMail-1.4.pdf (Appendix A: Environment Properties)
- JavaMail Session properties: http://java.sun.com/products/javamail/javadocs/overview-summary.html
- JavaMail Session properties for IMAP: http://java.sun.com/products/javamail/javadocs/com/sun/mail/imap/package-summary.html
- JavaMail Session properties for POP3: http://java.sun.com/products/javamail/javadocs/com/sun/mail/pop3/package-summary.html

These properties are treated in different ways in eServices, depending on

- Whether they are set internally by E-mail Server.

- Whether they can be modified by users.

These two parameters define three different categories of property:

- Set internally and not user-modifiable
- Set internally and user-modifiable
- Not set internally and user-modifiable

The next three sections list the properties in each category and describe how to set the ones in user-modifiable categories.

Set Internally, Not User-Modifiable

mail.pop3.class

mail.imap.class

Set Internally, User-Modifiable

mail.debug

mail.pop3.host

mail.pop3.user

mail.pop3.port

mail.pop3.connectiontimeout

mail.pop3.timeout

mail.pop3.socketFactory.class

mail.pop3.socketFactory.fallback

mail.pop3.socketFactory.port

mail.imap.host

mail.imap.user

mail.imap.port

mail.imap.connectiontimeout

mail.imap.timeout

mail.imap.socketFactory.class

mail.imap.socketFactory.fallback

mail.imap.socketFactory.port

You can modify these using existing configuration options, as shown in the following table. In this table, <protocol> is either POP3 or IMAP; for example, `mail.<protocol>.timeout` covers `mail.pop3.timeout` and `mail.imap.timeout`.

Table 20: JavaMail Properties Controlled by Configuration Options

JavaMail Property	Configuration Option
mail.debug	enable-debug
mail.<protocol>.connectiontimeout	connect-timeout
mail.<protocol>.timeout	protocol-timeout
mail.<protocol>.user	mailbox
mail.<protocol>.host	server
mail.<protocol>.port mail.<protocol>.socketFactory.port	port
mail.<protocol>.socketFactory.class mail.<protocol>.socketFactory.fallback	enable-ssl

See the *eServices 8.1 Reference Manual* for complete information on these options.

Not Set Internally, User-Modifiable

Any of the properties not listed in the two preceding sections can be modified by creating options in E-mail Server's `pop-client` section. The option name is the property name. For the value, see the JavaMail documentation listed above.

Note: Do *not* use this method to modify the properties, listed in the preceding section, that are controlled by configuration options.

Here is an example of adding an option to modify a JavaMail property: Some POP3 servers do not properly implement TOP, an optional POP command. This can create conflicts between the results of the TOP and RETR commands, which in turn can prevent E-mail Server from parsing the retrieved e-mail. To prevent these conflicts, you can create an option that invokes JavaMail's mail.pop3.disabletop property. The option name is `mail.pop3.disabletop`, it must be in the `pop-client` section, and its value must be `true`. E-mail Server then does not use TOP to retrieve messages, only RETR.

Delivery Status Notification and Message Disposition Notification

Outbound e-mails can include a request for a return message indicating whether and how the original e-mail was delivered. In Genesys eServices, you do this using the `Send Email` object in a routing strategy, as described in the *Universal Routing 8.1 Reference Manual.* The return message is of one of the following three types:

- If delivery fails: InboundNDR
- If delivery succeeds:
  - InboundReport
  - InboundDisposition

These types are represented as attribute values of the `Interaction Subtype` Business Attribute in Configuration Manager. E-mail Server assigns the return message to the appropriate type, and UCS stores it as a child of the outbound e-mail that contained the request.

The following sections describe each type and its contents.

InboundNDR

If one of the SMTP servers involved in the transport of the original e-mail fails to deliver it, E-mail Server submits the return message to the system with subtype `InboundNDR` (NDR stands for non-delivery report). There are two ways that E-mail Server can detect an inbound e-mail as an NDR:

RFC 3464

For this way, both of the following must be true:

- The e-mail conforms to RFC 3464, which means that it includes information about delivery status.
- That information indicates that delivery failed for at least one recipient.

In this case, E-mail Server submits the e-mail to the system with attached data.

Message Parts

For this way, all of the following must be true:

1. The e-mail contains either a `message/rfc822` part or a `message/rfc822-headers` part.
2. The part referred to in (1) contains a `Message-ID` header.
3. One of the following must be true:
   - The e-mail's From field contains one of the values specified in the E-mail Server `ndr-senders-list` option (the default is `mailer-daemon,postmaster,mmdf`).

- The `Message-ID` header referred to in (2) matches the message ID of some interaction already stored in the UCS database.

In this case, E-mail Server submits the e-mail to the system with no particular attached data. The subtype `InboundNDR` indicates the failure of delivery; the message itself contains no additional information.

Structured Information

When E-mail Server attaches data to the inbound interaction, it is of two kinds. The first kind, structured information, is listed in Table 21.

Table 21: Attached Data: Structured Information

Key	Possible Values	Description
_DSNInfo.RecipientCount	Any integer	Number of recipient addresses covered in this DSN
_DSNInfo.Recipient1.Recipient	Any string	Recipient address
_DSNInfo.Recipient1.Action	delayed delivered expanded failed relayed	Action applied for this recipient
_DSNInfo.Recipient2.Recipient	Same as _DSNInfo.Recipient1.Recipient	
_DSNInfo.Recipient2.Action	Same as _DSNInfo.Recipient1.Action	
_DSNInfo.RecipientN.Recipient	Same as _DSNInfo.Recipient1.Recipient	
_DSNInfo.RecipientN.Action	Same as _DSNInfo.Recipient1.Action	

In RecipientN in the last two rows of Table 21, N is the value of `_DSNInfo.RecipientCount`: the number of recipients covered in this InboundReport.

Note: A non-delivery report may arrive even if the outbound e-mail did not request it. If it does, E-mail Server still submits it to the system with the subtype `InboundNDR`.

Raw Information

The second kind of information that E-mail Server attaches is raw information. That is, all information included in the reply e-mail's header is attached as key-value pairs, with the key name formed by prefixing `_DSNRawInfo` to the field name used in the reply. Some examples are:

```
_DSNRawInfo.Reporting-MTA
_DSNRawInfo.RecipientCount
_DSNRawInfo.Recipient1.Original-Recipient
_DSNRawInfo.Recipient1.Action
_DSNRawInfo.Recipient1.Status
_DSNRawInfo.Recipient1.Remote-MTA
_DSNRawInfo.Recipient2.Final-Recipient
```

For details, see RFC 1894 (`http://www.ietf.org/rfc/rfc1894.txt`).

InboundReport

You request delivery status notification (DSN) by selecting the `Delivery status notification` box in a `Send Email` routing object. The reply to this request receives the subtype `InboundReport`. This reply conforms with RFC 1894, and includes, as attached data:

- The structured information listed in Table 21 on page 203.

  Note that if the `_DSNInfo.RecipientN.Action` key has a value of `failed`, E-mail Server assigns the reply the subtype `InboundNDR`, not `InboundReport`.
- The raw information contained in the keys whose names start with `_DSNRawInfo`, as described in the previous section.

Note: Depending on the implementation of the SMTP servers involved, there are the following possibilities:

- Individual DSN messages can be generated, each one related to one or several recipient addresses.
- A single DSN message can be generated, related to one or several recipient addresses.

InboundDisposition

You request message disposition notification (MDN, also called *read receipt*) by selecting the `Message disposition notification` box in a `Send Email` routing object. The reply to this request receives the subtype `InboundDisposition`. This reply conforms with RFC 3798, and includes, as attached data, the information listed in Table 22.

Table 22: Attached Data in InboundDisposition

Key	Possible Values	Description
_MDNInfo.ActionMode	manual-action automatic-action	Mode of the action applied to the e-mail
_MDNInfo.DispositionType	displayed deleted	What was done with the e-mail

Table 22: Attached Data in InboundDisposition (Continued)

Key	Possible Values	Description
_MDNInfo.SendingMode	MDN-sent-manually MDN-sent-automatically	How the message disposition notification is being sent
_MDNInfo.Recipient	Any string	Recipient address covered by this message disposition notification

This reply is sent as long as all of the following conditions are met:

- It was requested in the outbound e-mail. This is independent of whether a delivery status notification was also requested.
- Delivery succeeded.
- Either of the following:
  - The recipient agreed to send the read receipt.
  - The recipient mailer was configured to automatically send read receipts.

In addition to the structured information listed in Table 22, all information included in the reply e-mail's header is attached as key-value pairs, with the key name formed by prefixing `_MDNRawInfo` to the field name used in the reply. Some examples are:

```
_MDNRawInfo.Disposition
_MDNRawInfo.Final-Recipient
_MDNRawInfo.Original-Message-ID
_MDNRawInfo.Reporting-UA
```

For details, see RFC 3798 (`http://www.ietf.org/rfc/rfc3798.txt`).

Customizing the Format of External Resource E-mails

E-mails from external resources are received in plain text format. To customize the format in which these e-mails are presented, use the following procedures, described in this section:

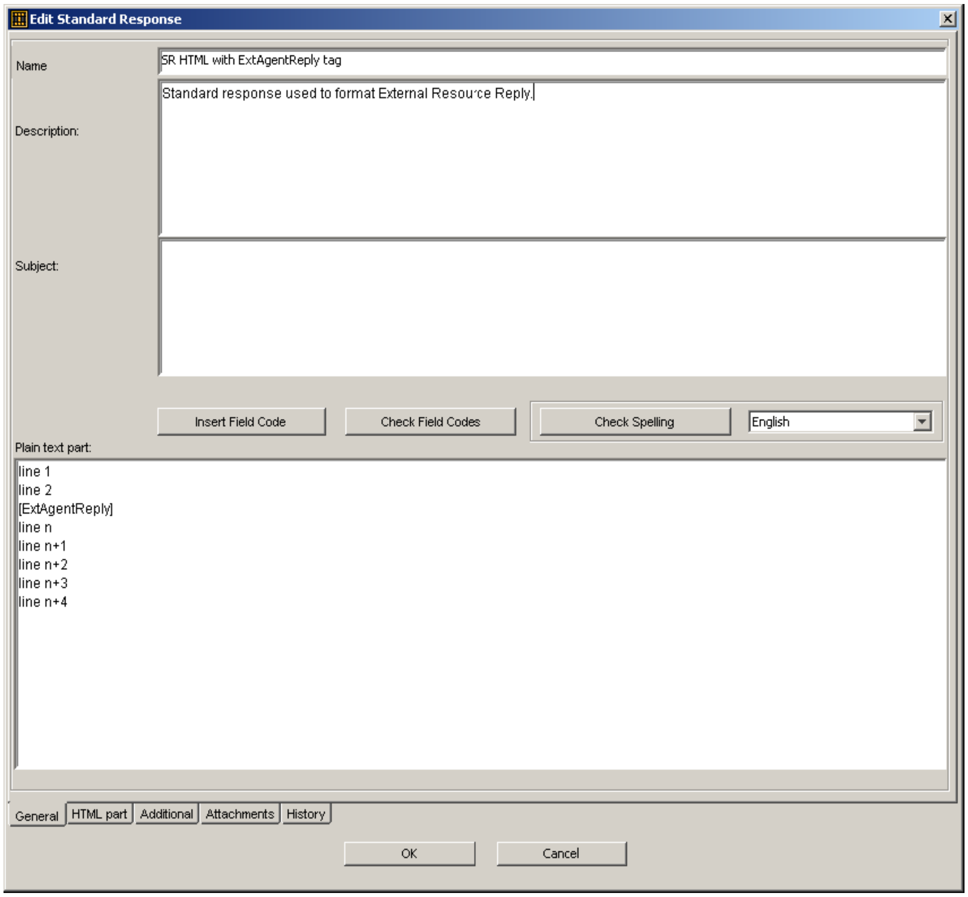
1. "New standard response for responses from external resources" on page 206
2. "Updating the strategy to include the standard response" on page 207
3. "Finding the standard response ID" on page 210

Procedure:
New standard response for responses from external resources

Purpose: To declare a new standard response in Knowledge Manager that will be used to customize e-mails built by the `ResponseFromExtResource` service.

Start of procedure

1. Create a new standard response in Knowledge Manager (as described in "Creating Standard Responses" on page 28) containing the tag [ExtAgentReply].
2. Refer to Figure 92 on page 206 and Figure 93 on page 207 for a sample configuration.



Name: SR HTML with ExtAgentReply tag

Description: Standard response used to format External Resource Reply.

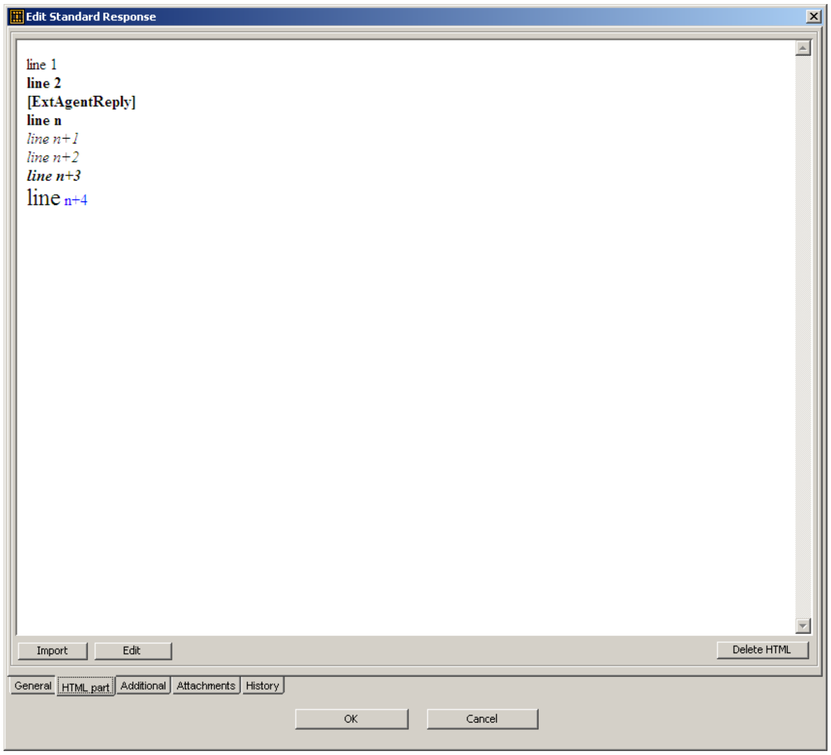
Subject:

English

Plain text part:

```
line 1
line 2
[ExtAgentReply]
line n
line n+1
line n+2
line n+3
line n+4
```

Figure 92: Standard Response for External Reply: General Tab



line 1
line 2
[ExtAgentReply]
line n
line n+1
line n+2
line n+3
line n+4

Figure 93: Standard Response for External Reply: HTML Part Tab

End of procedure

Next Steps

- Locate the Standard Response ID (see Procedure: Finding the standard response ID, on page 210).
- Update the strategy (see Procedure: Updating the strategy to include the standard response, on page 207).

Procedure: Updating the strategy to include the standard response

Purpose: To update the strategy to include the standard response ID. This will provide the strategy with the standard response you wish to use for responses from external resources.

Prerequisites

- A standard response exists for responses from external resources (see Procedure: New standard response for responses from external resources, on page 206).

Start of procedure

1. Login to Interaction Routing Designer and open your strategy for editing.
2. In the strategy, create an `External Service` block to call the E-mail Server's `CreateReplyFromExtResource` service. See Figure 94 for an example.
3. On the `General` tab, enter the following values:

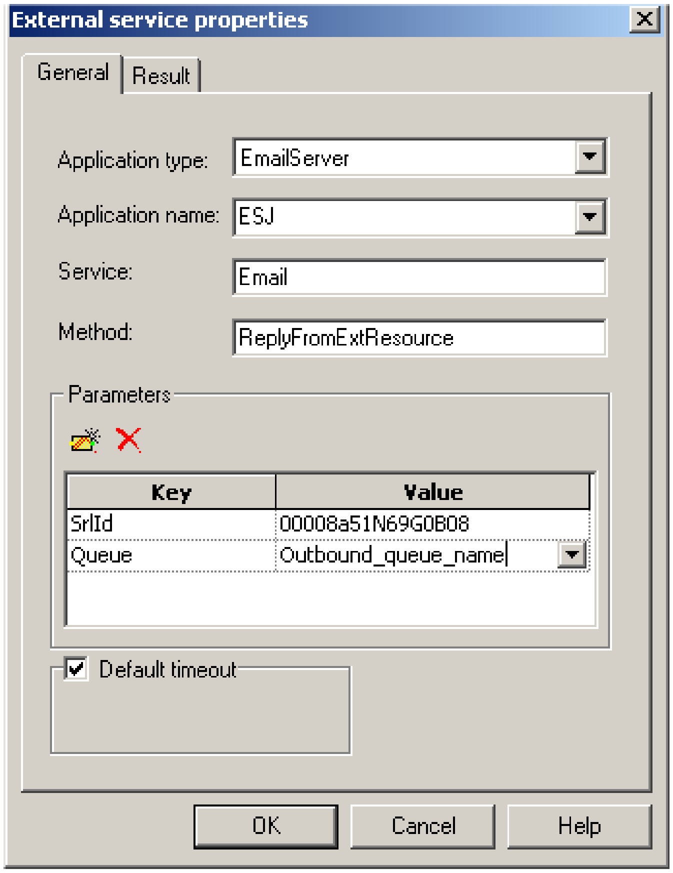
   `Application type: EmailServer`

   `Application name:` <name of your E-mail Server application>

   `Service: Email`

   `Method: ReplyFromExtResource.`
4. In the `Parameters` section add two entries: `SrlId` and `Queue`.
   a. For the `SrlId` parameter, the value is the ID of the standard response you created for external resource responses. To locate the ID, refer to Procedure: Finding the standard response ID, on page 210.

b. For the `Queue` parameter, the value is the name of the queue upon which the strategy to send e-mails is loaded. The name of this queue can be found in Configuration Manager in the `Scripts` folder under the specific Tenant.



Application type: EmailServer

Application name: ESJ

Service: Email

Method: ReplyFromExtResource

Key	Value
SrlId	00008a51N69G0B08
Queue	Outbound_queue_name

☑ Default timeout

Figure 94: External Service Block Properties

Note: Additional optional parameters can be added. Refer to the Reply E-mail From External Resource block description in the *Universal Routing Reference Manual* for more information.

5. Save and reload the modified strategy.

End of procedure

Procedure: Finding the standard response ID

Purpose: To locate the standard response ID. This ID is needed to update the strategy to allow for the customization of e-mails built from the `ReplyFromExtResource` service.

Prerequisites

- A standard response exists for e-mail responses received from external resources (see Procedure: New standard response for responses from external resources, on page 206).

Start of procedure

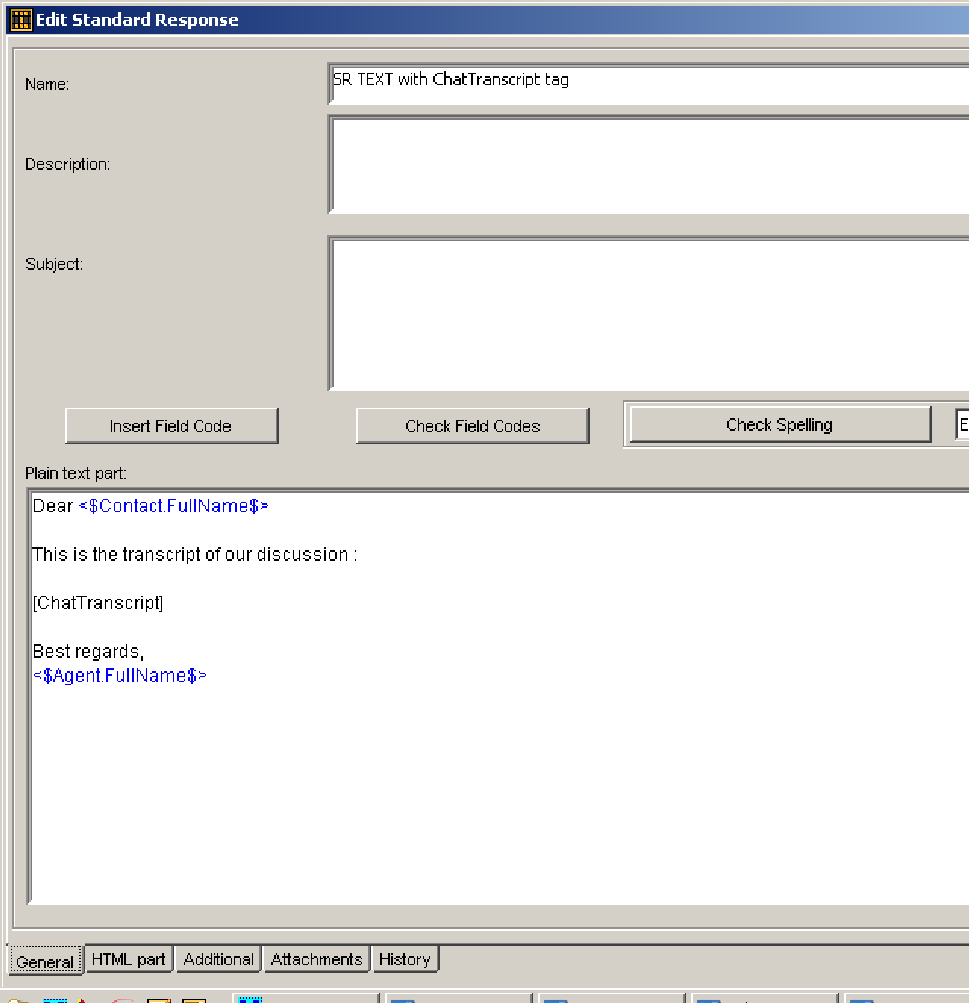
1. In Configuration Manager, locate the standard response created in Procedure: New standard response for responses from external resources, on page 206. Under your tenant, navigate to `Resources -> Business Attributes -> Category Structure-> Attribute Values`.
2. Locate your standard response and open it. The standard response ID appears in the `Name` field but it is not possible to copy it from here.
3. Click the `Annex` tab.
4. In the `General` section double-click the `Id` option.
5. The option value is the standard response ID. Copy the option value.

End of procedure

Other Customizations

Chat Transcripts

Ordinarily when including the transcript of a chat session in an e-mail, the transcript is appended to the end of the e-mail. To insert the transcript somewhere else, place the tag [ChatTranscript] where you want the transcript to appear. You can do this in the plain text or the HTML version or both. Figure 95 shows an example.



SR TEXT with ChatTranscript tag

Dear <$Contact.FullName$>

This is the transcript of our discussion :

[ChatTranscript]

Best regards,
<$Agent.FullName$>

Figure 95: Chat Transcript Tag in Standard Response

List of Forbidden Headers

E-mail Server does not allow certain headers to be added to outbound e-mail. These excluded headers are listed in the file `com/genesyslab/icc/emailserver/ForbiddenHeaders.properties`, contained in `esj.jar`, which is normally located in `\GCTI\eServices 8.1.0\E-mail Server Java\<E-Mail Server Application name>\lib`.

You can modify this list by extracting `ForbiddenHeaders.properties` to the `<E-Mail Server Application name>` directory, then editing the content. E-mail Server will then use this modified file. If there is no such file in the `<E-Mail Server Application name>` directory, E-mail Server uses the one in `esj.jar`.

List of Attached Files

Starting in release 8.1.0, inbound e-mails can include an attached data type `_AttachmentFileNames,` which contains a list of the names of files attached to the inbound e-mail.

Interaction Server: Advanced Topics

Chapter 9, "Capture Points Functionality in Interaction Server," on page 265 also relates to this subject matter.

Improving Database Performance

To optimize the performance of Interaction Server, try the following steps:

1. Design or redesign the Business Process for greater efficiency; for example, by minimizing the number of processing steps. This provides for the most performance gain for custom Business Processes.
2. Analyze and optimize the SELECT statements generated by Interaction Server. Analyze the execution plans for the generated SELECT statements and create appropriate indexes. This is especially important if you have added an custom Business Processes: the standard indexes provided with the default schema do not take account of any custom database fields, specific conditions configured, and other items added by custom Business Processes. This step might provide all the performance gain that you need.
3. Perform a general tuneup on the database.
4. Partition the database. The remainder of this section deals with this.

General Remarks on Partitioning

A partition is a division of a logical database or its constituent elements into independent parts. Database partitioning may be done for reasons of performance, manageability, or availability. This section concentrates on partitioning to improve performance.

By splitting a large table into several smaller tables, queries that need to access only a fraction of the data can run faster because there is less data to scan. Maintenance tasks, such as rebuilding indexes or backing up a table, can also run more quickly. Placing logical parts on physically separate hardware provides a major performance boost since all this hardware can perform operations in parallel.

Interaction Server performs large numbers of queries, updates, inserts, and deletes on its database. While it is relatively easy to achieve optimal performance with updates, inserts, and deletes, queries (SELECTs) are different.

The Interaction Server database consists of a single major table that stores all the interaction data. Every interaction in the system is always assigned to some interaction queue, represented by value of the field `queue` in the Interaction Server table. Business processes may employ dozens or even hundreds of queues.

Queues can vary greatly in the way they are used: some hold many interactions which are rarely processed at all (for example, an archive queue), others hold a small number of interactions with a high processing rate (for example, a queue for interactions that need some preliminary processing).

If these two types of queue are separated into different partitions, then the slower selection rate of the first type will not interfere with the high-speed selections of the second type. So the `queue` field is a natural choice to partition the data on. The remainder of this section describes partitioning by queue.

Planning

Decide for which queues it makes sense to separate data into logical partitions. Start by surveying the queues in your Business Processes and separate them out into three types:

1. Queues that contain high numbers of interactions; for example, post processing backlog or archive queues.
2. Queues that should not contain lots of interactions because all interactions in these queues should be processed immediately. A good example is the first queue in a Business Process which is meant for some preliminary processing (such as performing classification, calculating and attaching some user data, or sending an acknowledgment).
3. Queues that feed strategies that wait for resources (agents) to become available; usually there is a single such distribution queue in a Business Process.

Here is the rationale for separating data into at least three partitions that correspond to these three types of queues:

1. Separating Type 1 queues, those with many "inactive" interactions, ensures that these interactions are not even considered when SELECT statements are executed to pull interactions from Type 2 ("active") queues. Even if there are complex conditions for some views in your Business Process, there is much less data to scan because the majority of the interactions in an archive or post processing backlog are not touched by these scans.
2. Separating Type 2 queues is logical because most of the time these queues should be completely empty. Selecting new interactions out of these queues is trivial since there are not many interactions to select from.
3. Type 3 is the most demanding. While the rate of processing can be high, if there are many agents and handling time is relatively low, interactions may still accumulate in these queues when the peak inbound rate is higher than

the processing rate. This means that SELECT statements are executed frequently against many records. If there are multiple queues of this type, it may be beneficial to assign them to separate partitions.

Hardware Planning

While purely logical separation of data may be of some benefit, placing the partitions on separate hard drives provides the best performance gain.

In planning which drives in your system to dedicate to Interaction Server database partitions, it is advisable set aside one drive for the operating system and one for database log files, and place the Interaction Server database partition on other drives.

The rest of this section presents an example of partitioning using Microsoft SQL.

Creating the Database

To create a database with several file groups that will hold data for different partitions, use an SQL statement similar to the following:

```
CREATE DATABASE [itx_partitioned] ON PRIMARY
(NAME = N'itx802partitioned', FILENAME =
N'D:\MSSQL\DATA\itx802partitioned.ndf',
     SIZE = 44828672KB, MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB),
FILEGROUP [P1]
   (NAME = N'itx802partitioned1', FILENAME =
N'E:\MSSQL\DATA\itx802partitioned1.ndf',
     SIZE = 2048KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB),
FILEGROUP [P2]
   (NAME = N'itx802partitioned2', FILENAME =
N'F:\MSSQL\DATA\itx802partitioned2.ndf',
     SIZE = 2048KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB),
FILEGROUP [P3]
   (NAME = N'itx802partitioned3', FILENAME =
N'G:\MSSQL\DATA\itx802partitioned3.ndf',
     SIZE = 2048KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )
LOG ON
(NAME = N'itx802partitioned_log', FILENAME =
N'H:\MSSQL\DATA\itx802partitioned_log.ldf',
     SIZE = 5095872KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)
GO
```

Or you can create the database and file groups using Microsoft SQL Management Studio.

You can create as many file groups as your resources allow.

Partition Function

The partition function calculates the logical partition number for any specific record based on the record's field value. We only need to consider the value of the 'queue' field since we want to partition data according to queues.

The following is an example of the partition function:

```
CREATE PARTITION FUNCTION [QNamePFN](varchar(64)) AS
   RANGE RIGHT FOR VALUES (N'Archive', N'Distribution', N'Inbound')
GO
```

Note that an SQL server partition function is always a range function. The values for the range function *must* be sorted in ascending order so that you can clearly see which range any particular value falls into.

In the example above, all data that comes earlier in the alphabet than `Archive` is placed in the first partition. All data whose alphabetical order is the same or later than `Archive` but earlier than `Distribution` is placed in the second partition. All data whose alphabetical order is the same or later than `Distribution` but earlier than `Inbound` is placed in the third partition. All other data is placed in the forth partition.

Note that it is the partition scheme that assigns a specific partition to the range; you can actually assign different ranges to the same partition.

Also, this partition function applies to all queues in the Business Process. For example, if there is a queue `Begin` it falls into the second range and will be assigned to the same partition as the `Archive` queue. But if `Begin` is not a Type 1 queue this result may be less than ideal. One way to ensure that every queue is assigned to its intended partition is to list all the queues in the Business Process in alphabetical order in the partitioning function, and then specify the appropriate partition for each queue. If a new queue is added to the Business Process, you can alter the partition function and partition scheme to take account of this new queue.

Partition Scheme

The partitioning scheme uses the partitioning function to define which records (with a particular value of the partitioning function) go to which partition.

```
CREATE PARTITION SCHEME [QNamePScheme]
AS PARTITION [QNamePFN] TO ([PRIMARY], [P1], [P2], [P3])
GO
```

Since our partitioning function is based solely on the value of the 'queue' field, our partitioning scheme tells the database which queue goes to which partition.

Partitioning the Table

To partition the table, simply specify the partitioning scheme for the table:

```
create table interactions
(
   id    varchar(16)   not null,
   ...
) on QNamePScheme(queue)
Go
```

Note that we explicitly specify that the queue field value should be given to the partitioning scheme and subsequently to the partitioning function to decide which partition the record should go to.

Verification

The following SQL statement is an easy way to monitor how many records are stored in each partition for a given partitioned table (the `interactions` table in this example):

```
SELECT
   p.partition_number, fg.name, p.rows
FROM
   sys.partitions p
      INNER JOIN sys.allocation_units au
         ON au.container_id = p.hobt_id
      INNER JOIN sys.filegroups fg
         ON fg.data_space_id = au.data_space_id
WHERE
   p.object_id = OBJECT_ID('interactions')
```

This produces results similar to those shown in Table 23.

Table 23: Result of Verification Statement

Partition_number	Name	Rows
1	PRIMARY	1
2	P1	1
3	P2	1
4	P3	1

Table 23 shows that each partition contains a single record. If you insert a new record and execute the above statement again, it will show which partition the new record has been placed in, verifying your partition function and scheme.

Notes: To compare performance of the partitioned database with an unpartitioned database, you will need to artificially create a certain distribution of interactions between partitions (different queues) and see how fast the same SELECTs are being executed.

When interactions change queues, the records are physically relocated into different partitions (according to the partition scheme).

Converting to and from BLOB

Interaction Server ordinarily stores attached data in the `flexible_properties` field as a BLOB (binary large object).

Converting from BLOB

You can convert attached data to a custom field by running Interaction Server in a special utility mode, in which Interaction Server uses the key-value format of this attached data to convert all such fields to custom fields.

To run Interaction Server in utility mode, launch it from a command line with the following option:

```
-convert-fields [command_or_parameters]
```

where the optional `command_or_parameters` is one of the following:

`reset`—Ensures that the next run in utility mode will start processing from the beginning, rather than picking up where it left off.

`bulk-size=N`—Determines the number of records that are processed before committing the transaction. The default value is 100, valid values are any integer in the range 1–1000.

Here is an example command line:

```
interaction_server -host genesys_host -port 9876 -app IxnSrv05
-convert-fields reset
```

You can also have Interaction Server convert an existing database field into a BLOB, stored in the `flexible_properties` field. To do so, use the following procedure.

Procedure:
Converting a field to a BLOB

Start of procedure

1. Open the corresponding Business Attribute Value in Configuration Manager.
2. In the `translation` section, add an option called `to-delete` and give it the value `yes` (the `translation` section is described in "Custom Properties" on page 256).
3. Run Interaction Server in utility mode, as described previously (page 217).

   Interaction Server, in utility mode, moves the content of all such fields into the `flexible_properties` field and leaves the custom field with an empty value.

> **Note:** When Interaction Server runs in utility mode all of its other features are disabled: it cannot process interactions or open ports for clients.

End of procedure

Event Logger

In release 7.6.1 and later, Interaction Server includes Event Logger, a mechanism for storing reporting event messages in a database. You can configure it to store all reporting events or a selected subset. You can also create multiple instances of it.

Interaction Server generates, to registered reporting engines, messages that provide a detailed picture of the processing of each interaction. The attributes of these messages include much information about the interaction itself, such as its type, time received, associated agents, queues and workbins it was placed it, and so on. For a reference listing of these events and their attributes, see the "Reporting Messages" section of the "Other Protocol Events Used by Interaction Server" chapter of the *Genesys Events and Models Reference Manual.*

All configuration for the logger functionality is done in the Database Access Point (DAP) associated with the logger database.

Procedure: Deploying Event Logger

Start of procedure

1. Create a database to store the reporting data.
2. Locate the correct setup script for your RDMBS and run it on the database you created in Step 1.

   This script is called `eldb_<database_name>.sql`, where <database_name> is either *db2, mssql,* or *oracle* (for example, `eldb_mssql.sql`). To locate the script, go to the `Script` subdirectory of the installation directory of your Interaction Server, then open the subdirectory named after your RDBMS; for example, `\InteractionServer_801\Script\Oracle`.
3. Create a Database Access Point (DAP), filling in the usual mandatory settings on the `General` and `DB Info` tabs.
4. On the DAP's `Options` tab, create a section called `logger-settings`. This is the only mandatory section; its existence tells Interaction Server to use this DAP for storing reporting events.
5. In the `logger-settings` section, add at least one of the options described in "Options List" on page 222 (the section must contain at least one option in order to be valid).
6. Optionally add any of the following section types:
   - `event-filtering`—Contains options filtering out certain classes of event messages
   - `custom-events`—Specifies a custom mapping of the CustomEventId attribute value of `EventCustomReporting` (the option name) to the Event Logger table to store them in (the option values)
   - Custom data sections—Five sections that enable you to map the name of any event onto a custom field in the Logger database

   See "Options List" on page 222 more information.
7. On Interaction Server's `Connections` tab, add a connection to the DAP.

   For multiple instances of the Event Logger, run the creation script multiple times, creating multiple databases. Also create a DAP for each database.

End of procedure

Managing Data

For the `rpt_interaction`, `rpt_agent`, and `rpt_esp` tables, Genesys supplies a set of scripts that deletes events as soon as processing of the interaction stops, the agent logs out, or the external service responds, respectively. For custom reporting events that are stored in the `rpt_custom` table, the event-driven trigger trg_del_cust_delay purges them from the rpt_custom table, with a configurable delay (the default is 10 minutes).

If you want to preserve this data, you can disable the triggers trg_delete_stopped, trg_delete_resp, trg_del_cust_delay, and trg_delete_logout after you run the setup script. For Oracle, additionally, disable the triggers trg_mark_cust_logged, trg_mark_responded, trg_mark_ended_session and trg_mark_stopped_ixn.

You can reenable the triggers any time and resume removing records from the database automatically.

Of course event messages increase rapidly in number as interactions are processed, so you will want to take measures to periodically delete data from the database or move it elsewhere.

Also note that after creating or removing custom fields in a database, some triggers become invalid. If this happens, you must recompile them to be sure they work properly.

Classification of Events

The logger functionality classifies reporting events in two ways:

- By activity type—that is, whether the activity refers to an interaction, an agent, an ESP server (see “ESP” on page 355), or is of a custom type. The database contains tables for each type: interaction activity is stored in rpt_interaction, agent activity is stored in rpt_agent, and ESP server activity is stored in rpt_esp. Custom activity can be stored in rpt_interaction, rpt_agent, or rpt_custom, depending on the configuration in the custom-events section of the Event Logger DAP.
- By endpoint type—that is, whether that interaction is being transmitted to a queue, strategy, agent, or ESP service. You can filter out events according to endpoint type, as described in “event-filtering Section” on page 222. A few events do not have an endpoint type; you cannot filter these events.

Table 24 lists the events and their classifications.

Table 24: Classification of Reporting Events

Event	Activity	Endpoint
EventPropertiesChanged	Interaction	-
EventPartyAdded	Interaction	Agent, Strategy
EventPartyRemoved	Interaction	Agent, Strategy
EventRevoked	Interaction	Agent
EventInteractionSubmited	Interaction	-

Table 24: Classification of Reporting Events (Continued)

Event	Activity	Endpoint
EventProcessingStopped	Interaction	-
EventHeld	Interaction	-
EventResumed	Interaction	-
EventPlacedInQueue	Interaction	Queue
EventPlacedInWorkbin	Interaction	Queue
EventAgentInvited	Interaction	Agent
EventRejected	Interaction	Agent
EventTakenFromQueue	Interaction	Queue
EventTakenFromWorkbin	Interaction	Queue
EventAgentLogin	Agent	Agent State
EventAgentLogout	Agent	Agent State
EventDoNotDisturbOn	Agent	Agent State
EventDoNotDisturbOff	Agent	Agent State
EventMediaAdded	Agent	Agent State
EventMediaRemoved	Agent	Agent State
EventNotReadyForMedia	Agent	Agent State
EventReadyForMedia	Agent	Agent State
EventAgentStateReasonChanged	Agent	Agent State
EventMediaStateReasonChanged	Agent	Agent State
EventExternalServiceRequested	ESP Server	ESP Server
EventExternalServiceResponded	ESP Server	ESP Server
EventCustomReporting	Interaction, Agent, or Custom	-

Options List

This section provides short descriptions of the DAP options that configure the Event Logger's behavior. See the *eServices 8.1 Reference Manual* for full details.

logger-settings Section

`batch-size`—Defines the minimum number of records to store in internal memory before flushing to the database. Valid values are 1–5,000; the default is 500.

`mandatory-logging`—Determines how Interaction Server behaves if the Event Logger database is unavailable. If set to true, Interaction Server stops interaction processing and waits until the database is available. If set to `false`, Interaction Server does not use this logger and continues processing.

`max-queue-size`—Defines the maximum number of records that are kept in memory while waiting to be written to the database. If the number of records exceeds this maximum, the data are discarded from memory and are not written to the database. Valid values are 10,000–100,000; the default is 20,000.

`storing-timeout`—Defines a time interval, in milliseconds, between operations of writing to the database. Valid values are 500–60,000; the default is 1,000.

Note: `storing-timeout` and `batch-size` define limits that trigger writing to the database: writing takes place as soon as one or the other is reached.

`schema-name`—Specifies the name of the schema used to access the database.

custom-events Section

This section contains options that list custom events by their identifiers and specify which table (interaction, agent or custom) stores them.

event-filtering Section

This section contains eight options, six of which are named for one of the endpoint types listed in Table 24 on page 220:

`log-agent-state`

`log-agent-activity`

`log-queue`

`log-strategy`

`log-esp-service`

`event-filter-by-id`

With the value `false`, events associated with the named endpoint type are filtered out. For example, setting `log-queue` to a value of `false` prevents the events `EventPlacedInQueue`, `EventPlacedInWorkbin`, `EventTakenFromQueue`, and `EventTakenFromWorkbin` from being stored.

The remaining two options in this section are:

- log-userdata—With the value false, data from custom fields is filtered out.
- event-filter-by-id—A list of comma-separated event identifiers. Only these events are stored in Event Logger. If this option is not present or contains no event identifiers, event filtering by identifier is not applied.

  These event identifiers are listed in the *Platform SDK 8.1 API References* for Java and .NET. For example, the identifier of EventRejected is 168.

itx-custom-data, esp-custom-data, esp-service-data, agent-custom-data, custom-custom-data Sections

These sections contain options specifying a list of events that are to be stored in custom fields of the event logger database. All five sections work identically, the differences being (a) the events from which the user data is taken and (b) the database table that stores them. These differences are shown in Table 25.

Table 25: Custom Data Sections

Section	Source Event	Logger Database Table
itx-custom-data	UserData and EventContent attributes of interaction-related reporting events	rpt_interaction
esp-custom-data	UserData attribute of EventExternalServiceRequested and EventExternalServiceResponded	rpt_esp
esp-service-data	Envelope3rdServer attribute of EventExternalServiceRequested and EventExternalServiceResponded	rpt_esp
agent-custom-data	EventContent attribute of EventCustomReporting	rpt_agent
custom-custom-data	EventContent attribute of EventCustomReporting	rpt_custom

For an explanation of the Envelope3rdServer attribute, see “EventExternalServiceRequested Attributes” and “Structure of Envelope3dServer Attribute” in the “Reporting Messages” section of the “Other Protocol Events Used by Interaction Server” chapter of the *Genesys Events and Models Reference Manual.*

To use these options, you must first add a field to the appropriate Event Logger database table. Its data type must be the same as that of the mapped user data key.

In these sections, the options have the following characteristics:

- The name is a user data key name (case-sensitive).
- The value is three semicolon-separated strings, which specify the following:
  - **a.** The name of the field that you added to the database table. This value is required.
  - **b.** The data type: string, integer, or timestamp. The default is string, with default length 64. If your data type is other than string, or if it is string and you want to specify a non-default length (next item), this value is required.
  - **c.** Optionally, the length. The default for the string type is 64. There are no default values for integer and timestamp.

For example, if you have a data key called CustomerSegment, you can add a custom field to store this data as follows:

1. Add a field called `customer_segment` to the `rpt_interaction` table.
2. In the `itx-custom-data` section, create an option called `CustomerSegment`.
3. Give it this value: `customer_segment;string;64`.

   Since string and 64 are the default values for type and length respectively, the value of this option could also be simply `customer_segment`.

Message Queues

You can have Event Logger send events to a message queue, such as IBM MQ-Series, or Microsoft Message Queue (MSMQ). This provides a mechanism for reliable reporting events delivery to Interaction Server's reporting clients. Disconnection of the client does not lead to a loss of reporting events. Instead, events are stored in the message queue and delivered to the client (or otherwise read by the client) after it reconnects.

To use this functionality, you must create a DAP object that is specifically for the streaming of reporting events into MSMQ or MQ-Series. Both Interaction Server and the client connect to this DAP. This DAP must have the following sections and options, which partly resemble the sections and options of the Event Logger DAP and are also documented in the *eServices 8.1 Reference Manual:*

- `event-filtering` section. Contains the single option `event-filter-by-id`, whose value is a list of comma-separated event identifiers. Only these events are sent to the message queue; events not listed are not sent. This option is analogous to the option of the same name used in the Event Logger DAP
- `logger-settings` section

- `delivery-protocol.` Possible values are:
  - `mq-series`—For the MQ-Series message queue system
  - `msmq`—For the MSMQ message queue system
  - `dbmq`—Reserved for future use
- `delivery-queue-name`—The name of the queue to send messages to.
- `udata`—A list of comma-separated event identifiers. Only these events are sent to the message queue with their user data included. Events not on the list are sent without user data. If this option is absent or empty, user data is included in all events.

Note: The event identifiers used in `event-filter-by-id and udata` are listed in the *Platform SDK 8.1 API References* for Java and .NET.

Chapter

6 Contact Identification and Creation

This chapter describes how UCS identifies and creates new contacts, in these sections:

Overview

When a new interaction enters the system, UCS performs the following tasks:

1. Contact identification—UCS checks whether this interaction is coming from a known contact: more precisely, whether the contact data included in the new interaction matches an existing contact in the UCS database. UCS does this in response to a request from a media server, the `Identify Contact` Routing strategy object, or the Agent Interaction SDK `CreateInteraction` method.
2. Contact creation—If the contact does not exist in the database, UCS creates a new record to represent it.

This chapter first describes the default procedure that UCS follows for these tasks, then the ways in which you can modify this default.

Contact Identification

To perform contact identification, UCS takes the contact data included in the new interaction and tries to match it with its existing contact records based on certain attributes.

Default Behavior

What UCS Checks

The default list of the attributes that UCS checks is `FirstName`, `LastName`, `Title`, `EmailAddress`, and `PhoneNumber`.

Note: This chapter refers to contact attributes by their system names. The system name is the one that is used in the UCS and Configuration Server databases, and the one that appears in the `Name` box of the attribute's `Properties` window in Configuration Manager. In Configuration Manager there is also a display name which usually differs slightly; for example, system name `PhoneNumber`, display name `Phone Number`.

Order of Checking

The attributes that UCS uses in contact identification have a ranking which tells UCS what priority to give them in searching. The default ranking is:

`EmailAddress`—0

`PhoneNumber`—1

`FirstName`—2

`LastName`—2

`Title`—3

The general procedure that UCS follows in the default case can be diagrammed as in Figure 96, where Attribute N is the highest-ranked attribute available in the interaction's contact data, Attribute N-1 is the next highest, and so on.

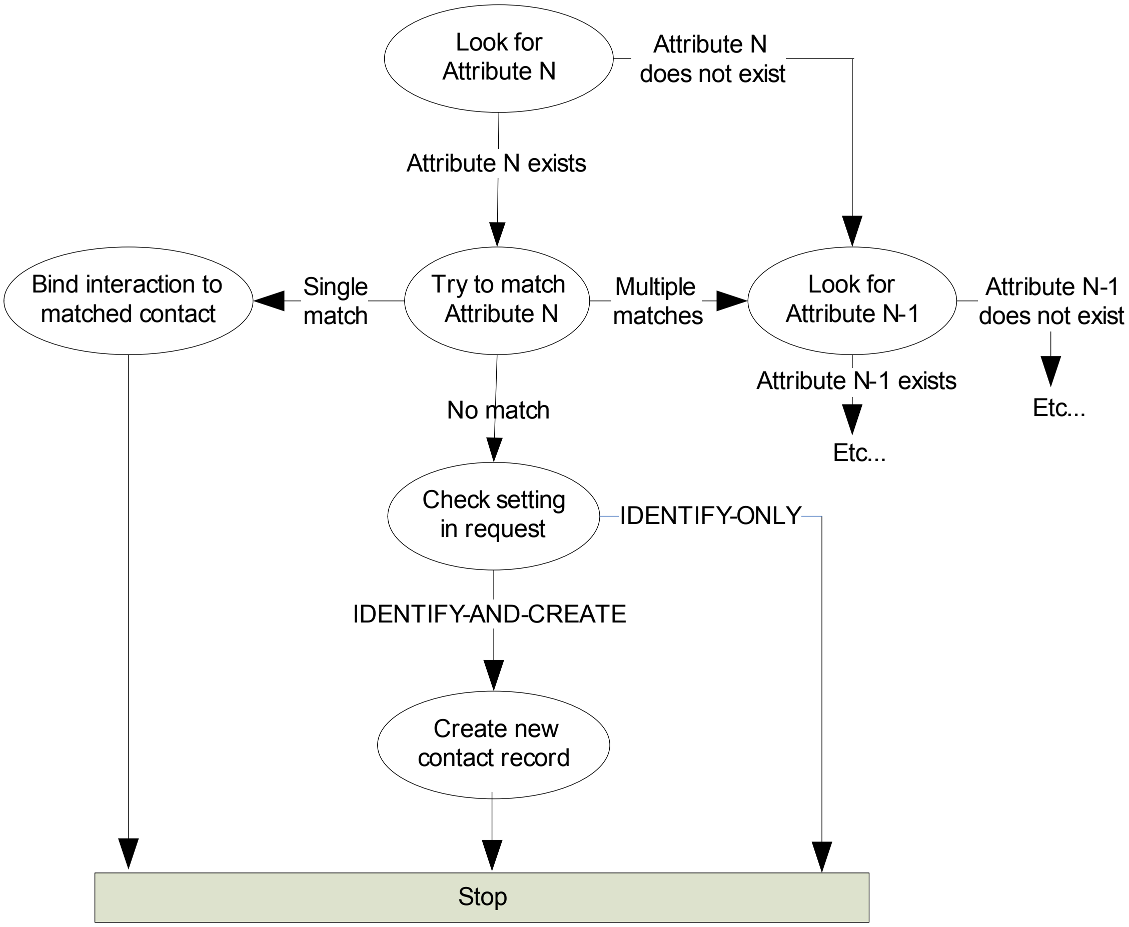


Figure 96: Default Procedure

Here is an example:

1. A strategy requests identification of a contact, with only the following data provided:

   EmailAddress=johnanddora@doefamily.net

   FirstName=john

   LastName=doe

2. UCS takes EmailAddress, the highest priority attribute, and searches for EmailAddress = johnanddora@doefamily.net.

3. UCS finds two matching contacts, one with FirstName = john and the other with FirstName = dora.

4. UCS moves on to the next highest priority attribute. This would be PhoneNumber if it were specified, but it is not. So UCS takes the next-highest priority attribute, of which there are two, FirstName and LastName, both with a priority of 2.

5. UCS searches on EmailAddress = johnanddora@doefamily.net, and restricts the result to FirstName = john and LastName = doe
6. UCS finds a single contact matching these three criteria. It returns the ID of the Contact found, plus additional information if available.

   ContactId = 56464FGG519EA03

   EmailAddress = johnanddora@doefamily.net

   FirstName = john

   LastName = doe

   PhoneNumber = 555-654-6303

There are the following additional considerations:

- If the client's request conveys no specification of what to do in the event of no match, UCS creates a new contact record.
- When two attributes have the same rank, as do `FirstName` and `LastName` in the default ranking, UCS tries to match both of them.

  If the interaction has values for both attributes, UCS returns only contact records that match both. But if one of equally-ranked attributes has no value, UCS returns any contact records that match the attribute that does have a value (this is equivalent to saying that an attribute with no value matches everything).

If UCS goes through all of the attributes that it has been specified to check and finds no values for any of them, it returns an error message.

If UCS goes through all of the attributes that it has been specified to check and still finds two or more matching contacts, the next step depends on the entity that requested the contact identification:

- If UCS is identifying contacts in response to a request from a media server:
  - With E-mail Server, UCS takes the first contact on the list and associates it with the interaction.
  - With Chat Server, UCS simply passes the interaction on for processing with no associated contact. It reports neither the list of contacts found nor the fact that multiple contacts were found. The agent handling the interaction can select a contact for it using the Agent Desktop.
- If UCS is identifying contacts in response to an IRD `Identify Contact` object, it depends on whether the object's `Return Unique` checkbox is selected:
  - If `Return Unique` is not selected, UCS returns a list of the matching contacts.
  - If `Return Unique` is selected, UCS only reports that multiple contacts were found but does not return the list of matching contacts.

  In either case, what happens next depends on the subsequent part of the strategy. The system may continue to process the interaction without any contact, until the interaction reaches the Agent Desktop, when the agent handling the interaction can select a contact for it.

Customizing

You can alter UCS's default contact identification behavior in both aspects (whether an attribute is used and, if it is used, what priority it has in matching). You can do this for all interactions globally or for interactions of a specific media type.

With these differing scopes of customization, the more specific scope takes precedence. UCS operates according to the following order:

1. If a media type has its own configuration for how a contact attribute is used in contact identification, that takes precedence over any other configuration ("Customize per media type" on page 232).
2. Configuration of a contact attribute itself is more general and applies only to those media types that lack their own configuration for contact identification ("Customize which attributes are checked" on page 231 and "Customize the priority of attributes" on page 232).
3. In any areas where items 1 and 2 do not apply, UCS follows the default behavior already described (page 228).

Procedure: Customize which attributes are checked

Start of procedure

1. In Configuration Manager, open the `Properties` window for the desired contact attribute.
2. On the `Annex` tab of the `Properties` window (to set Configuration Manager to show the `Annex` tab, see `Configuration Manager Help`), create a `settings` section if it does not already exist.
3. In the `settings` section, create an option called `is-searchable.` Set its value to `TRUE` to make UCS use this attribute in contact identification. Set its value to `FALSE` to keep UCS from using this attribute in contact identification.

   (The default value depends on the attribute. The five attributes used in UCS's default behavior (page 228) have the default value `TRUE.` All other attributes have the default value `FALSE`).

End of procedure

Notes: This procedure affects the value of the desired attribute's `IsSearchable` attribute in the UCS database. You must use only this procedure to do this. Never edit any attribute or value directly in the database.

There is also a way to control which attributes are searchable from the agent desktop, described in the "Making an Attached Attribute

Sortable" section of the "Interaction Package" chapter of *Selected Conceptual Data Models for the UCS Database.*

Procedure: Customize the priority of attributes

Summary

You can customize the priority of the attributes that UCS checks, but only when both of the following conditions are true:

- The media type is specified in the interaction's user data.
- The interaction's media type is *not* e-mail, chat, or callback. In the typical case, this would be because the interactions are being submitted by a custom media server (built using the Genesys Media Interaction SDK Java or Open Media Interaction SDK Web Services).

Start of procedure

1. In Configuration Manager, open the `Properties` window for the desired attribute and go to its `Annex` tab. Create a `settings` section if it does not already exist.
2. In the `settings` section, create an option called `search-order-level.` Give the option a numerical value to determine its priority in matching. Possible values are any integer in the range 0 (highest priority) to 127 (lowest priority). Two or more attributes can have the same priority.

End of procedure

Procedure: Customize per media type

Summary

For a finer level of granularity, you can control how contacts are identified for interactions of a specific media type (of course, the interactions must have a valid media type specified in their user data).

Start of procedure

1. In Configuration Manager, open the `Properties` window for the desired media type and go to its `Annex` tab.
2. Create a `contact-searchable-attributes` section.

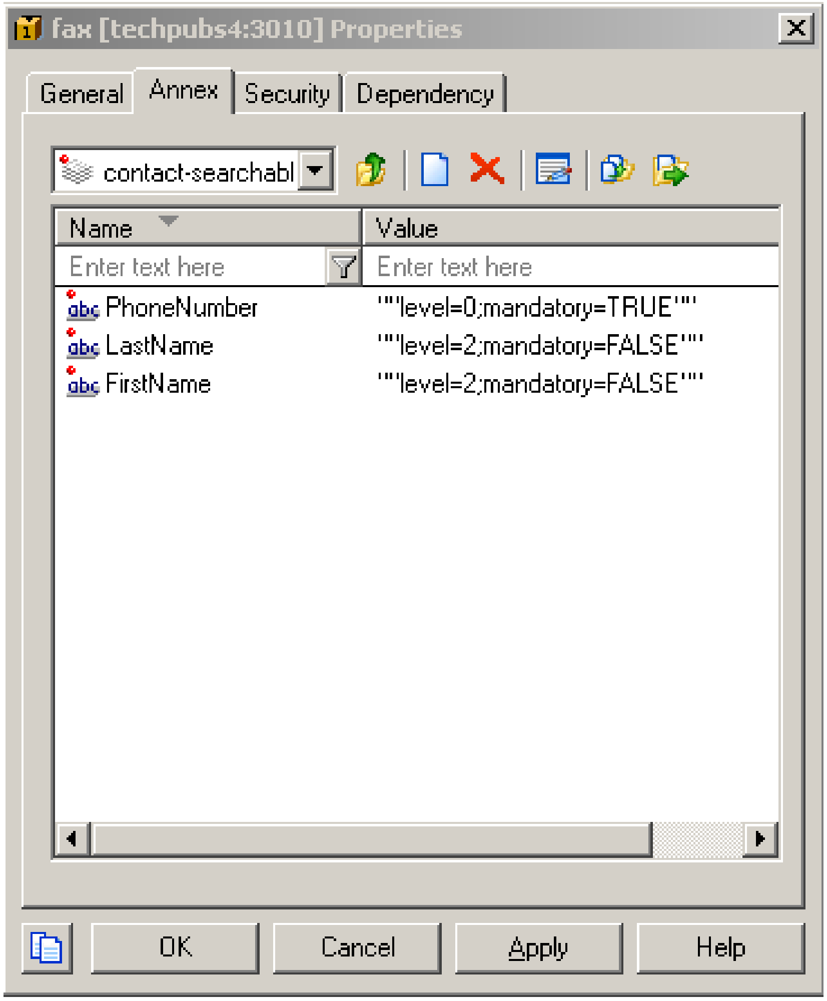
3. In this section, create an option for each contact attribute that you want UCS to use in contact identification for this media type.

   > **Note:** Any such attribute must either be one on the default list (page 228) or have the `is-searchable` option with a value of `TRUE`.

   The options that you create must have the following characteristics:

   - The name duplicates the attribute's system name (displayed in the `Name` box of the `Properties` window, not the `Display Name` box); for example, `PhoneNumber` rather than `Phone Number`.
   - The value has the form `"level=<integer>;mandatory=<Boolean>"`, where
     - `<integer>` is an integer from 0–127 setting the attribute's priority.
     - If `<Boolean>` is `TRUE`, this attribute must be present (that is, it must have a value) for matching to occur.
     - If `<Boolean>` is `FALSE`, this attribute need not be present (that is, it may lack a value) for matching to occur.

Figure 97 shows an example.



Name	Value
PhoneNumber	""level=0;mandatory=TRUE""
LastName	""level=2;mandatory=FALSE""
FirstName	""level=2;mandatory=FALSE""

Figure 97: Media Type Configured for Contact Identification

With the configuration in Figure 97,

- UCS returns an error if no `PhoneNumber` value is present. It does this even if `FirstName` and `LastName` values are available, because `PhoneNumber` is the attribute with highest priority (0) and is defined as mandatory.
- If multiple matching records are found with a given `PhoneNumber` value, UCS uses `FirstName` and `LastName` together to discriminate because they have the same priority (2).
- If there is no `LastName` value, UCS does not use `FirstName` alone, because `LastName` is specified as mandatory in this level. Instead, the list of records matching `PhoneNumber` only is returned. This avoids identification based on `FirstName` only.

End of procedure

Contact Creation

If UCS cannot find a contact in its database that matches the contact associated with a new interaction, it creates one in its database.

Default Behavior

The default behavior is for UCS to simply create a new contact record. This behavior can be overridden for an interaction in a routing strategy using the IRD `Identify Contact` object (see *Universal Routing 8.1 Interaction Routing Designer Help*). For e-mail interactions only, you can also modify the default behavior by using one of the non-default settings of E-mail Server's `contact-identification` option (see the *eServices 8.1 Reference Manual*).

Customizing

Procedure: Turning contact creation on and off

Summary

You can customize the default behavior for each media type.

Start of procedure

1. In Configuration Manager, go to `Business Attributes > Media Type > Attribute Values` and double click the media type that you want to adjust.
2. On the `Annex` tab of the resulting `Properties` window (to set Configuration Manager to show the `Annex` tab, see `Configuration Manager Help`), create a `settings` section if it does not already exist.
3. In the `settings` section, create an option called `create-contact`. Set its value to `FALSE` to block contact creation for this media type. (The default value is `TRUE`).

End of procedure

Procedure: Setting minimum attributes for creation

Summary

You can also define a minimum set of contact attributes that must be present (must have values) for a contact to be created.

Start of procedure

1. In Configuration Manager, open the `Properties` window for the desired media type and go to its `Annex` tab.
2. Create a `contact-minimum-attributes-set` section.
3. In this section, create an option for each contact attribute that you want to require for contact creation. The option's name must duplicate the attribute's database name (displayed in the `Name` box of the `Properties` window, not the `Display Name` box); for example, `PhoneNumber` rather than `Phone Number.` The option's value must be empty.

End of procedure

Genesys

Chapter

7 Searching the UCS Database

This chapter describes searching the UCS database, in these topics:

Overview

Starting with release 8.0, UCS supports full-text search of contacts, interactions, and standard responses using the Platform SDK (PSDK) Contact API.

To enable full text search, the stored data must be prepared, an operation known as indexing. Indexing converts, abbreviates, and sorts data to allow fast searching in large data sets.

There are various ways of preparing the data for the index; one way is to divide text data into words. Another way of preparing text data is *stemming,* or storing only the stem of a word rather than all its forms (for example, *run, running, runs,* and *ran* all stored as simply *run*). UCS uses *analyzers* to perform word division and stemming; see "Configuration" on page 239 for more information.

Many attributes are not directly searchable because of how they are stored. This chapter also describes how to make such attributes searchable.

More About Indexing

It is important to note that the UCS indexing service enables searching for text data, but does not guarantee its retrieval.

When an object is found it must be retrieved using standard PSDK methods. These methods are documented in the API References of the Platform SDK.

When designing an application, data coming from search results should be considered as cached data, like the preview that can be seen in web search engines. This data is optimized for search and cannot be used to update UCS database objects. For example, the StructuredText of an interaction is rendered from HTML to text. All formatting is lost if this data is re-inserted into the interaction.

Documents stored in the index will at least have the following retrievable fields:

- `id`—Unique identifier in the UCS database. This is what enables retrieval of the object from UCS.
- `IndexationDate`—The date this index was created. The format is UTC (yyyy-MM-dd'T'HH:mm:ss.SSS'Z').
- `DocumentType`—Type of the document, selected from the following:
  - EmailIn
  - EmailOut
  - PhoneCall
  - CallBack
  - CoBrowse
  - Chat
  - Interaction (Open Media)
  - Contact
  - StandardResponse

Example:

`id` = 00001a4EM4TD007K

`IndexationDate` = 2009.11.24.19.53.124

`Type` = chat

In the index, even dates and numbers are text. This format enables alphabetic sorting, so that, for example, 21 December 2008 comes before 1 June 2009.

Default configuration enables the indexing of Contacts, Interactions and Standard Responses.

Sizing

The index can be divided into two parts:

- The inverted index part that enables the full text search
- The data part that enables retrieval of the original data

If all of the original data is kept in the index, it can take as much space as the database itself; the default configuration does not compress it. The inverted index part is considerably smaller, but its size depends on the frequency of words. If the inverted index contains many unique values like IDs and timestamps it will be big. If it contains many common words such as those in the body of e-mails, it will be smaller.

Table 26 presents an example of sizing for a relatively large database.

Table 26: Example Sizing

Database Name	Database Size	Index Size	Duration
Standard responses	15 MB	6.5 MB	3 seconds
Contacts	5 GB	4.68 GB	2.5 hours
Interactions	20 GB	14 GB	2 hours

In this case the total size of the index files is about 19 GB.

During operations the index can grow to twice the size of the usable data.This is because index operations (such as internal reordering, purge of deleted documents, and concatenation) create new temporary files. To make these operations appear instantaneous, the system creates new files while it is still reading the old files. Then when it is finished creating the new files, it removes the old ones. Therefore, to be safe, free space on the disk hosting the indexes should be three times the size of the index.

Configuration

Basic options controlling indexing and searching are described in the *eServices 8.1 Reference Manual.* This section describes one further option that controls the use of analyzers.

Warning! Any change in configuration of indexing usually triggers reindexing the full content of all UCS indexed objects.

The default analyzers, supplied with UCS, are

- WhitespaceAnalyzer—Splits the text into tokens separated by white space characters (specifically, SPACE_SEPARATOR, LINE_SEPARATOR, PARAGRAPH_SEPARATOR, HORIZONTAL TABULATION, LINE FEED, VERTICAL TABULATION, FORM FEED, CARRIAGE RETURN, FILE SEPARATOR, GROUP SEPARATOR, RECORD SEPARATOR, or UNIT SEPARATOR).
- LowerCaseAnalyzer—Converts the text to lower case and splits the text into tokens separated by the white space character.
- SimpleAnalyzer—Divides text at non-letters and converts to lower case. This works well for languages in which words are separated by spaces, such as most European languages, but is of little use for languages in which words are not separated by spaces, such as many Asian languages.
- KeywordAnalyzer—Treats the entire text as a single token. This is useful for data like zip codes, IDs, and some product names.

In the default case, UCS search uses the LowerCaseAnalyzer for all fields in all tables in the database.

To override the default analyzer, use the following option.

<table_name>-field-analyzer<any>

Optional: Yes
Default value: LowerCaseAnalyzer
Valid values: See below
Changes take effect: After restart

Sets the analyzer used for any table or field. In the option name, `<table_name>` is one of the following tables in the UCS database:

Callback

Chat

CoBrowse

Contact

ContactAttribute

EmailIn

EmailOut

Interaction

PhoneCall

StandardResponse

`<any>` can be anything, including zero. Use it to differentiate among multiple `field-analyzer` options referring to the same table.

Values for this option have the general form

<field>=<analyzer>, <field>=<analyzer>, ...

where <field> is the name of a field in the table and <analyzer> is the name of a supported and installed analyzer. For example:

Option name: `interaction-field-analyzer`

Option value: `Text=GermanAnalyzer,StructuredText=StandardAnalyzer`

With this option name and value, when searching the `Interaction` table, the search operation applies `GermanAnalyzer` to the `Text` field and `StandardAnalyzer` to the StructuredText field.

You can achieve the same result by creating two options:

Name: `interaction-field-analyzer-01`, value: `Text=GermanAnalyzer`

Name: `interaction-field-analyzer-02`, value: `StructureText=StandardAnalyzer`

Supported Analyzers

Default Analyzers

WhitespaceAnalyzer

LowerCaseAnalyzer

SimpleAnalyzer

KeywordAnalyzer

Language-specific Analyzers

These are the same as SimpleAnalyzer but also remove *stop words:* words that are so common that there is little to be gained in searching for them or listing their occurrences.

As an example, the stop words used by StandardAnalyzer, the language-specific analyzer for English, are *a an and are as at be but by for if in into is it no not of on or such that the their then there these they this to was will with.*

The language-specific analyzers installed with UCS are:

BrazilianAnalyzer

ChineseAnalyzer

CJKAnalyzer (Chinese/Japanese/Korean; any language that uses Chinese characters/kanji/hanja)

CzechAnalyzer

DutchAnalyzer

FrenchAnalyzer

GermanAnalyzer

GreekAnalyzer

RussianAnalyzer

StandardAnalyzer (English)
ThaiAnalyzer
SpanishAnalyzer
ItalianAnalyzer

Search Syntax and Examples

Searching the UCS database uses the Lucene syntax, described at `http://lucene.apache.org/core/3_6_1/queryparsersyntax.html`.

Some samples follow.

```
FirstName:kristin*
```

This query searches for all records having a key `FirstName` whose value is any word starting with “kristin.”

```
FirstName:Kim AND LastName:Brown AND EmailAddress:hotmail
```

This query searches on three attributes: FirstName containing “Kim,” LastName containing “Brown,” and EmailAddress containing “hotmail.”

```
Text:complain*
```

This query searches for all records having an attribute `Text` which contains words starting with “complain.”

Making an Attribute Searchable from the Desktop

For both interactions and contacts, many attributes are not directly searchable from the desktop.

This section describes the way to make these attributes searchable from the desktop.

You do this by adding an option called `is-sortable` to the `Annex` section of the corresponding Business Attribute. Do not confuse this option with the `is-searchable` option that is described in “Customize which attributes are checked” on page 231 of Chapter 6. Table 27 compares the two.

Table 27: is-sortable and is-searchable

Option Added	Effect	Used with
is-sortable	Makes attribute searchable from desktop	Interaction attributes, contact attributes
is-searchable	Makes UCS use the attribute in identifying contacts (at runtime)	Contact attributes only

By default, all attributes that are attached to an interaction are stored in the `AllAttributes` attribute of the `Interaction` entity. You can make searchable any attribute that is represented in the Configuration Server database by a `Business Attributes` object of type `Interaction Attributes`. Examples are:

- Category
- Disposition Code
- Interaction Subtype
- Interaction Type
- Language
- Media Type
- Priority
- Reason Code
- Service Type
- StopProcessing Reason

The `Interaction` entity includes attributes `StrAttribute1`–`StrAttribute10` and `IntAttribute1`–`IntAttribute5`, which exist to enable you to make attached attributes searchable. These attributes `StrAttribute1`–`StrAttribute10` and `IntAttribute1`–`IntAttribute5` may be referred to collectively as *replicant attributes,* as explained below.

To be able to perform lookup on any of the attached attributes, use the following procedure.

Procedure: Making an attached attribute searchable

Start of procedure

1. In Configuration Manager, be sure that `Properties` windows show their `Annex` tabs. If they do not:
   a. Go to the `View` menu and select `Options`.

   b. In the resulting dialog box, select the `Show Annex tab in object properties` check box.

2. In the tenant for your UCS, go to `Business Attributes > Interaction Attributes > Attribute Values`.
3. Open the `Properties` window for the attribute that you want to make searchable (for example, `Service Type`).
4. On the `Annex` tab, create a section named `settings` if it does not already exist.
5. In this `settings` section, create an option named `is-sortable` and give it the value `true`.

Note: Although the option name refers to being *sortable,* its real effect is to make attributes *searchable.*

6. If the attribute is of type string, you are finished. If it is of type integer, you must create an additional option, also in the `Settings` section, named `type` with the value `integer`.

End of procedure

Once you have configured an attached attribute as searchable, UCS takes its value as stored in `AllAttributes` and copies it as the value of one of the replicant attributes. To find out which replicant attribute copies a given attached attribute, look at the content of the `IxnAttributeMetaData` table. For example, if you have configured the `ServiceType` attribute to be searchable, you can find out which replicant attribute copies its value by using the following SQL request:

```
select MappingColumnName from IxnAttributeMetaData where
TheName='ServiceType';
```

Please also note the following:

- This replication process only applies to interactions created or updated after you perform the configuration described in this section. The replication process is not applied to interactions retroactively. The replicant attribute in older records will remain empty.
- By default, the replicant attribute that replicates the Interaction attribute does not have any database index. To increase performance during queries, consider adding index(es) to those replicant attributes that contain copied attributes.
- Replicant attributes are read-only from outside UCS. UCS is responsible for synchronization of their content whenever `Interaction.AllAttributes` is updated.
- The mapping between a searchable interaction attribute and a replicant attribute is based on the type (string or integer) of the business attribute declared in the Configuration Server database (string by default). UCS

chooses from among the replicant attributes of the proper type that are not already associated with an attached attribute. It does this until no more replicant attributes are available.

- Once a replicant attribute has been used for a particular attribute, it is dedicated to that attribute: it cannot be used for another one. The only modification you can make is to configure a searchable attribute to be no longer searchable. The replicant attribute that copied this attribute's values will then retain those values for existing records and for any new records.

Using Full Text Search in a Primary/Backup Environment

This section describes best practices for using the full text search (FTS) functionality of Universal Contact Server in a primary/backup environment.

Note: The full text search is built on top of the Apache Lucene project.

Shared File System

The same index files can be shared by two primary/backup UCS instances by using a system shared folder with the `storage-path` configuration option in UCS applications.

Table 28 shows the mandatory configurations options for the index.

Table 28: Mandatory Index Options

Option	Comment
index/enabled=true	None.
index.contact/enabled=true	None.
index.contact/storage-path=<same location as backup>	The path does not have to be the same string, but must point to the same location.
index.interaction/enabled=true	None.
index.interaction/storage-path=<same location as backup>	The path does not have to be the same string, but must point to the same location.
index.srl/enabled=true	None.
index.srl/storage-path=<same location as backup>	The path does not have to be the same string, but must point to the same location.

Note: The following steps are required in order to have UCS running as a Windows service and able to use a network index:

1. Set the `storage-path` option of each index in the following form: `\\<host of index>\<end of path>`.
2. On the `Log On` tab of the `Properties` service, select `This account`, and provide the login and password to have access to the index location.

Limitations

- If UCS is rebuilding an index when the switchover occurs, it will stop building the index, and the other UCS instance will not continue to rebuild the index. In this case, Genesys recommends a full rebuild of the index.
- If the shared file system becomes unavailable when trying to update the index, the index might omit some data, or become corrupted or locked. To be notified of index update failures, you can set an SCS alarm on message `20105`.

  Here is an example of message `20105` in the UCS log:

  ```
  14:13:25.899 Std 20105 [Notifier-2] Failed to send notification 13
  to Persistent.
  ```

  In this case, Genesys recommends a full rebuild of the index.
- If the shared file system becomes unavailable, the search is not available.
- UCS will not be able to start if indexing is enabled and the shared file system where the index resides is not available. In this case, the shared file system must be made available, or UCS can be started by disabling indexation (by setting the `enabled` option in the `index` section to `FALSE`).
- If switchover occurs while the shared file system is not available, the switchover procedure will be reverted and both UCSs returned to their original states. During the switchover attempt, all UCS processing will be stopped. After switchover reversion, the original primary UCS will resume normal processing, except for indexation.
- If the primary UCS shuts down unexpectedly while the shared file system is not available, the successive attempts to switch the backup UCS to primary mode will fail. If such a failure occurs, the first critical recovery piece is either:
  - Make the shared file system available to the backup UCS so it can finish the switching procedure.
  - Disable indexation in the backup UCS application, and then restart the backup UCS. This UCS will start as the primary server without indexation.

Shared File System Recommendation

When implementing High Availability (HA), you may want to first build the index onto a local drive (solid state drive, if possible). Then, move the index to a Storage Area Network/Network Attached Storage (SAN/NAS) with replication/snapshot functionality (a NetApp FAS storage system, for example). Each UCS will access the index as either network storage (NAS) or fiber channel (SAN).

File System Synchronization

In the case of disaster recovery, a High Availability solution must provide FTS service after switchover, but the loss of some of the indexed documents may be acceptable.

A possible configuration is to configure the Primary and Backup UCS with the following:

- An independent storage path.
- A nightly synchronization of the file system from primary storage path to backup storage path.

In this scenario, a switchover results in uninterrupted UCS service with regard to searchable content. A few hours will be missing from the searchable content, which should be acceptable for disaster recovery.

Limitations

- The limitations regarding availability of the file system apply in the same manner as for a shared file system (see “Limitations” on page 246).
- You will need to schedule the recovery procedure described in the following section.

Recovery

Genesys recommends a full rebuild of the indexes, which should occur during off-production (recommended), or during slow contact center activity.

Two methods are available to rebuild the indexes:

Full build

1. Shut down all UCSs.
2. Delete the current indexes from the local directory reflected in the `storage-path` configuration option.
3. Start the UCSs with the option `index-rebuild="if-new"` for each deleted index.

Notes:

- The full build method is the most efficient, time-wise.
- If searches are expected by the clients during the build period, the response will not be accurate until the full rebuild is complete. This means service is not very useful until a majority of database content is rebuilt.

Full re-build

1. Shut down all UCSs.
2. Keep the index that was present before the switchover, or restore the most up-to-date backup.
3. Start the UCSs with the option `index-rebuild="on-start"` for each index that needs reconstructing.

Notes:

- The full-rebuild method obtains the most accurate search results immediately after UCS startup.
- Indexing will be significantly slower than a full build.

Genesys

Chapter

8 Interaction Properties

This chapter describes interaction properties and provides information on which of them are safe to modify. It covers these topics:

Overview

Genesys eServices interactions have a number of properties that take the form of key-value pairs. This chapter lists these properties and provides some information about them, including whether it is safe to change them.

Since you can set up your eServices system to change some of these properties as the interaction moves through the system, it is particularly important to know that some properties are safe to change and others are not. For example, you can change interaction properties by using the `Update`, `UpdateBusinessData`, and `UpdateInteractionData` functions in a routing strategy. But you should be very careful when considering using these functions to change any interaction property.

There are three types of properties that can be defined in terms of the following two characteristics

- Whether they exist as independent fields in the `interactions` table. If they do, you can refer to them when defining conditions, orders, and segments in Views in Business Processes.
- Whether they are used by Genesys media servers.

Table 29 summarizes this classification and specifies whether it is safe for users to change the properties.

Note: When this chapter says that you must not change an interaction property, that means that you must not change the value of existing properties *and* that you must not create a new object with this name.

Table 29: Three Property Types

Type	Independent Field	Used by Genesys Media Server	Change OK
System	+	-	No, except `ExternalID` and `ParentID`
Business	+/-	+	OK if not used by media server
Custom	-	-	Yes

Here is further information on these three property types:

- System properties. These are maintained solely by Interaction Server, as independent fields in the `interactions` table. The user cannot change them, with the two exceptions listed in Table 29. The definition of conditions, orders, and segments in Views can refer to them. Examples: `InteractionID, MediaType.` See also "System Properties" on page 251.
- Business properties. The values of these are set by media servers (including both Genesys and custom media servers), and these properties may be used by the media servers at later points in processing. Therefore you must not change the ones that are used by any media server in your solution. Some are stored as independent fields in the interactions table but most are stored in the `FlexibleProperties` field; see Table 31 on page 255. The definition of conditions, orders, and segments in Views can refer to `ServiceObjective` and `Priority`, which are the only ones that are stored as independent fields. They can also refer to the other Business properties if you create custom fields that correspond to them, as described in "Custom Properties" on page 256.
- Custom properties. These exist only if you define them, as described in "Custom Properties" on page 256. Media servers do not use them. The definition of conditions and orders in Views can refer to them.

Note: In setting conditions for Views and snapshots, besides directly using some interaction properties types, you can also use a set of functions called Translations. These provide database manipulation tools that are independent of the underlying database. See "Translations" on page 260.

Interaction properties show up in several places:

- Interaction Server logs show all of them.
- Some IRD objects (for example, `UpdateBusinessData`) use them as parameters and display them in drop-down lists.

Interaction Properties

This section lists the following about interaction properties:

- Property name as it appears in protocol messages
- Field name in the `interactions` table, if the property is stored as an independent field
- Data type
- Short description

System Properties

Table 30 lists the system properties. You must not change these properties, with the two exceptions listed in Table 29. You can use these properties on the `Condition`, `Order`, and `Segmentation` tabs of Views in Business Processes, except that properties with Timestamp data type cannot be used on the `Segmentation` tab.

Table 30: System Properties

Name	Name in Interactions Table	Type	Description
AbandonedAt	abandoned_at	Timestamp	Date and time that the media server set the `IsOnline` attribute to `0`. Simplifies calculation of some statistics. If the interaction is still online, this attribute is not set.
DeliveredAt	delivered_at	Timestamp	Date and time that the interaction was first offered to the agent
ExternalId	external_id	String	External interaction identifier (examples: chat session ID, e-mail mime ID)
FlexibleProperties	flexible_properties	Binary	Stores attached data and most business properties (see “Business Properties” on page 254).
InQueues	destinations	String	Suggested destinations for the interaction (if provided by URS)

Table 30: System Properties (Continued)

Name	Name in Interactions Table	Type	Description
InteractionId	id	String	Record identifier
InteractionState	state	Integer	0 = queued 1 = cached 2 = being processed by URS 3 = being handled by agent
InteractionSubtype	subtype	String	Defined as Business Attribute in Configuration Server
InteractionType	type	String	Defined as Business Attribute in Configuration Server
IsLocked	is_locked	Integer	0 = unlocked 1 = locked
IsOnline	is_online	Integer	0 = offline 1 = online This property applies to any media type (for e-mail, the value is always 0).
MediaType	media_type	String	Defined as Business Attribute in Configuration Server
MovedToQueueAt	moved_to_queue_at	Timestamp	Date and time that the interaction was first moved to a queue. If the interaction leaves the queue, then returns to it, the value of this property remains the same; that is, it shows the earliest time that the interaction entered the queue, without regard for later entry to or exit from the queue.
OutQueues	destinations	String	Suggested destinations for reply
ParentId	parent_id	String	Identifier of the parent interaction in the UCS database
PlacedInQueueAt	placed_in_queue_at	Timestamp	Date and time that the interaction was placed in the current queue, regardless of whether the interaction was in this queue previously

Table 30: System Properties (Continued)

Name	Name in Interactions Table	Type	Description
PlaceInQueueSeq	place_in_queue_seq	Integer	Event sequence number, denoting the chronological order of the last `EventPlacedInQueue` (for the interaction)
Queue	queue	String	Name of the queue that the interaction is in
ReceivedAt	received_at	Timestamp	Date and time that the media server received the interaction. If not provided by media server, date and time of submission to Interaction Server (same as `SubmittedAt`).
ScheduledAt	scheduled_at	Timestamp	Date and time before which the interaction must not be processed. See “Setting the ScheduledAt Property” in the “Creating Business Process Objects” chapter of the *Universal Routing 8.1 Business Process User’s Guide.*
SubmittedAt	submitted_at	Timestamp	Date and time that the interaction was submitted to Interaction Server
SubmittedBy	submitted_by	String	Name of the client application that submitted the interaction
SubmitSeq	submit_seq	Integer	Event sequence number, denoting the chronological order of `EventInteractionSubmitted` (for the interaction)
TenantId	tenant_id	Integer	Tenant associated with the interaction
Workbin	workbin	String	Indicates that interaction is in a workbin
WorkbinAgentGroupId	agent_group_id	String	One of four properties specifying the ID and type of the workbin that the interaction is in. Only one of the four properties is present.
WorkbinAgentId	agent_id	String	See `WorkbinAgentGroupId.`
WorkbinPlaceGroupId	place_group_id	String	See `WorkbinAgentGroupId.`
WorkbinPlaceId	place_id	String	See `WorkbinAgentGroupId.`

Table 30: System Properties (Continued)

Name	Name in Interactions Table	Type	Description
AbandonedAt	abandoned_at	Timestamp	Date and time that the media server set the `IsOnline` attribute to `0`. Simplifies calculation of some statistics. If the interaction is still online, this attribute is not set.
HeldAt	held_at	Timestamp	Time and date, set by Interaction Server, that the interaction was put on hold.
CompletedAt	completed_at	Timestamp	Date and time, set by Interaction Server, that the interaction was first placed into one of the queue specified in the Interaction Server `completed-queues` option.
AssignedAt	assigned_at	Timestamp	Date and time, set by Interaction Server, that the interaction was last delivered to the resource.
AssignedTo	assigned_to	String	Employee ID of the agent to whom the interaction was last delivered. If the agent is anonymous, concat('@',place name) is used instead.

Business Properties

Business Properties are set by media servers, so it is safe to change these properties only if they are not used by any media server in your solution.

Business Properties are stored in two ways. The two properties listed in Table 31 are stored as independent fields, so you can use them on the Condition and Order tabs of Views in Business Processes.

Table 31: Business Properties Stored as Independent Fields

Name	Name In Interactions Table	Type	Description
ServiceObjective	service_objective	Integer	Time objective for servicing the interaction. The contact center may define a service objective for each combination of customer segment, service type, and media type.
Priority	priority	Integer	Indicates whether e-mail should receive special processing

All other Business Properties, listed in Table 32, are stored in the flexible_properties field. To use them on the Condition, Order, or Segmentation tabs of Views in Business Processes, you must create Custom Properties that correspond to them, as described in "Custom Properties" on page 256.

Table 32: Business Properties Stored in FlexibleProperties

Name	Type	Description
CaseId	String	Case identifier. Use and meaning to be defined by user.
CategoryId	String	Category identifier obtained by routing strategy request for e-mail classification
ContactId	String	Customer identifier in the UCS database. Provided by UCS.
DispositionCode	String	Code for moving the interaction somewhere else
FromAddress	String	Taken from e-mail interaction
FromPersonal	String	Name of person who sent the interaction
Header_*	String	Content of the header of an e-mail. Do not change.
Mailbox	String	Mailbox of addressee
ReasonCode	String	Code for reason for the operation that caused the event; for example, normal, autoresponse, sent, forwarded, or redirected. Set by routing strategy Stop object.
CustomerSegment	String	Code for the customer's revenue potential; for example, Gold, Silver, Bronze. May be assigned as a result of database lookup based on sender name. Routing strategy sets this property.

Table 32: Business Properties Stored in FlexibleProperties (Continued)

Name	Type	Description
ServiceType	String	Code for type of service being requested; for example, `Sales`, `Service`, `Information`. Routing strategy sets this property.
Subject	String	Taken from e-mail interaction
To	String	Destination e-mail address of an e-mail or web form. Do not change. This property is required if you use the Chat Transcript object in routing strategies. It supplies the value for the `To` field of the outbound e-mail that sends the chat transcript. One way to give this property a value is in a routing strategy; for details, see “Chat Transcript” under “eServices (Multimedia) Objects” in the “Interaction Routing Designer Objects” chapter of the *Universal Routing 8.1 Reference Manual.*

Custom Properties

You can create new interaction properties (fields). The data type of these custom properties can be timestamp, string, or number. You can use these properties on the `Condition`, `Order`, and `Segmentation` tabs of Views in Business Processes, except that properties with Timestamp data type cannot be used on the `Segmentation` tab.

Note: In a multitenant environment, the configuration of interaction custom properties in one tenant applies to all tenants. This means that the values of custom properties are saved in separate corresponding database fields for all interactions, regardless of whether the interactions belong to the tenant in which the custom properties are configured.

Procedure: Configuring a custom interaction property

Start of procedure

1. Decide on an attached data key that will be the source of the content of the custom property.
2. Create a new field directly in the `interactions` database.

   Note: Data type varchar(max) is not supported for custom fields.

3. Create a new Business Attribute:
   - Name = `InteractionCustomProperties`

- Display name = `Interaction Custom Properties`
- Type = `Custom`

If such an attribute already exists go to the next step.

4. Expand `Interaction Custom Properties` and open its `Attribute values.`
5. Give it an `Attribute Value`, with a name exactly matching the attached data key name that you decided on in Step 1. The matching is case sensitive (you can create a separate display name).
6. In your new attribute value, go to the `Annex` tab and create a section called `translation.`
7. In the new `translation` section, create an option called `translate-to,` with its value duplicating the name of the new field you created in Step 2.

End of procedure

Next Steps

You can now use the new custom property to attach data to an interaction and to define conditions and orders of Views and snapshots.

Notes

You should be aware of the following points when defining custom interaction properties:

- While Interaction Server allows defining custom property names that contain spaces in their name, and will correctly map these properties to the custom database fields, it does not convert these property names into custom field names when they are used in the definition of a view condition, view order, snapshot condition or snapshot order.
- Genesys DB Server does not support custom fields of type varchar(max).
- If you specify a custom field as not null, you must ensure that applications always provide some data to that field upon creation of an interaction (RequestSubmit). If no data is provided, the request will fail because Interaction Server sends NULL for empty fields, and that will be rejected by the DBMS. This also means that the default value trigger for such fields cannot be used.

  As a workaround, you can create fields in the interactions database without mapping them to custom properties of Interaction Server. Such fields are hidden from Interaction Server but can be used by third-party applications.

Business Attributes

Business Attributes differ from interaction properties, although the two share some names (for example, `MediaType` and `Case ID`).

- Interaction properties are fields in the `interactions` table. Their main purpose is to identify and describe interactions during processing.
- Business Attributes are objects in the configuration database. They have two purposes.
  - To represent data concerning areas other than traditional telephony. Each Business Attribute contains a range of possible values for a data category. In this sense, Business Attributes are like a dictionary.
  - To convey data between software components that have no other means of communication. Examples are Category Structure and Screening Rules. These are among the Business Attributes that you must not change in Configuration Manager.

Category Structure is an example of a Business Attribute that users must not change. You create and edit categories in Knowledge Manager. The categories that you create also appear in Configuration Manager, in `Resources > Business Attributes > Category Structure`. It is possible to select a category in Configuration Manager, open its `Properties` dialog box, and make changes as you would to any configuration object. But this could create severe problems and the system would probably not operate properly when trying to handle this category. Knowledge Manager is the only safe tool to use for creating and editing categories.

Table 33 lists the Business Attributes and Business Attribute Values that are relevant to eServices but are not represented in the `interactions` table.

Table 33: Business Attributes That Are Not in the Interactions Table

Name	Change
CategoryStructure	Prohibited
ContactAttributes	Prohibited
ContactAttributes/AccountNumber	Allowed
ContactAttributes/LastName	Prohibited
ContactAttributes/PhoneNumber	Prohibited
ContactAttributes/PIN	Allowed
ContactAttributes/Title	Prohibited
EmailAccounts	Allowed
InteractionAttributes	Prohibited
Language	Allowed
RootInteractionID	Prohibited

Table 33: Business Attributes That Are Not in the Interactions Table (Continued)

Name	Change
ScreeningRules	Prohibited
StopProcessingReason	Allowed

Table 34 repeats two of the Business Attributes from Table 33 for which change is allowed, and adds information. Note that both of these Business Attributes must be present for eServices to work properly.

Table 34: Business Attributes, Change Allowed

Name	Default Value	Comment
EmailAccounts	None	The eServices configuration wizard creates two values: one for the account that E-mail Server pulls e-mails from in order to bring them into the eServices system, and one for an external agent. You may want to use Configuration Manager to create additional values, for example if you want additional e-mail addresses to be available for `Forward` and `Redirect` objects in routing strategies. Configure the address for this value on the `Annex` tab > `General` section > `address` option. The address must comply with RFC2822 and therefore it must be encoded according to RFC2047. Examples: legal@mycompany.com ""Legal Dpt"" <legal@mycompany.com> Type = custom.
Language	English	Used by Knowledge Manager (along with `Tenant`) to group category trees. You must add the value `unknown` if you want to train a model that classifies by language. See “Training” on page 90 and “Notes on Language” on page 121.
StopProcessingReason	AutoResponded Forwarded Normal Re-directed Sent Terminated	Must be selected in the IRD `Stop Interaction` object. You can manually add other values by creating them as Attribute Values of `StopProcessingReason`, under `Business Attributes` in Configuration Manager.

Translations

In IRD, on the `Conditions` and `Order` tabs of a View object in an Business Process, you can write statements that refer to the three types of interaction properties that are represented as independent fields in the `interactions` table: system, business, and custom. You can also use built-in translations, or functions, which provide database manipulation tools that are independent of the underlying database.

Translations enable you to use the interaction property names as presented in this chapter rather than database table field names. They provide a unified representation, regardless of the underlying database, of a collection of functions. Some deal with time and date, others with attributes of the specific interaction; for example, `_age()` calculates the interaction's age in seconds, regardless of database type used.

Notes: Bear in mind the following:

- Translation does not hide the original database syntax. Interaction Server translates whatever it can, leaves the rest unchanged, and transmits it all to the database. Therefore you can also use database field names and any database-specific constructs on the `Conditions` and `Order` tabs.
- Many of these translations are time sensitive. As such they have a dependency on the `freeze-interval` option of the Interaction Queue View object. For details, see the discussion of this option in the "Interaction Server Options" section of the "Configuration Options" chapter of the *eServices 8.1 Reference Manual.*

_timestamp

Specifies date constants in conditions. The single argument is a character constant that represents a date in the common form used by Interaction Server.

Usage

```
_timestamp('yyyy-mm-ddThh:mi:ssZ')
```

OR

```
_timestamp('yyyy-mm-dd hh:mi:ss')
```

Translations Performed by Database

For Oracle 9 and 10:

```
TO_DATE('yyyy-mm-dd hh:mi:ss', 'YYYY-MM-DD HH24:MI:SS')
```

For Microsoft SQL:

```
CONVERT(DATETIME, 'yyyy-mm-ddThh:mi:ssZ', 102)
```

For DB2:

```
TIMESTAMP('yyyy-mm-dd hh:mm:ss')
```

_current_time

Calculates the current UTC date-time.

Use this function to avoid the confusion caused by mixing UTC and local times. Since Interaction Server manipulates only UTC time, and all dates in the database are saved as UTC time, there is no standard function to get the current local date-time.

Database-specific functions can also be used to get the local time, such as Microsoft SQL's `getdate()` or Oracle's `sysdate`.

Usage

```
_current_time()
```

Translations Performed by Database

For Oracle 9 and 10:

```
cast(SYS_EXTRACT_UTC(SYSTIMESTAMP) as date)
```

For Microsoft SQL:

```
GETUTCDATE()
```

For DB2:

```
(CURRENT TIMESTAMP - CURRENT TIMEZONE)
```

_timestampdiff

Calculates the difference, in seconds, between two timestamps.

Both arguments should be of type timestamp, which differs depending on the database engine:

- Oracle—date
- Microsoft SQL—datetime
- DB2—timestamp

Usage

```
_timestampdiff(timestamp date1, timestamp date2)
```

Translations Performed by Database

For Oracle 9 and 10:

```
(((date1) - (date2))*86400)
```

For Microsoft SQL:

```
cast(cast(((date1) - (date2)) as float)*86400 as integer)
```

For DB2:

```
timestampdiff( 2, char((date1) - (date2)) )
```

_timestampadd

Method of increasing and decreasing dates. Generates a date increased or decreased by a specified number of seconds. The second parameter can be negative, to make the result a date earlier than the original date.

Usage

```
_timestampadd(timestamp date, integer seconds)
```

Translations Performed by Database

For Oracle 9 and 10:

```
(date + (seconds)/86400)
```

For Microsoft SQL:

```
dateadd(second, seconds, date)
```

For DB2:

```
(date + (seconds) second)
```

_age

Calculates the age in seconds of the interaction; that is, the difference between the current time in UTC and the interaction attribute `ReceivedAt`.

Usage

```
_age()
```

Translations Performed by Database

For Oracle 9 and 10:

```
((cast(SYS_EXTRACT_UTC(SYSTIMESTAMP) as date) - received_at)*86400)
```

For Microsoft SQL:

```
cast(cast((getutcdate() - received_at) as float)*86400 as integer)
```

For DB2:

```
timestampdiff( 2, char((current timestamp - current timezone) -
received_at))
```

_time_in_queue

Calculates the time, in seconds, that the interaction has spent in the queue; that is, the difference between the current time in UTC and the attribute `PlacedInQueueAt`.

Usage

```
_time_in_queue()
```

Translations Performed by Database

For Oracle 9 and 10:

```
((cast(SYS_EXTRACT_UTC(SYSTIMESTAMP) as date) -
placed_in_queue_at)*86400)
```

For Microsoft SQL:

```
cast(cast((getutcdate() - placed_in_queue_at) as float)*86400 as
integer)
```

For DB2:

```
timestampdiff( 2, char((current timestamp - current timezone) -
placed_in_queue_at))
```

_time_in_same_queue

Same as `_time_in_queue` except that it is based on the `MovedToQueueAt` property, which is updated only when the interaction is placed into a queue that it has not been in before.

Provides a unified method of calculating the amount of time that the interaction has spent in the current queue. It calculates the difference in seconds between the current time in UTC and the attribute `MovedToQueueAt`.

Usage

```
_time_in_same_queue()
```

Translations Performed by Database

For Oracle 9 and 10:

```
((cast(SYS_EXTRACT_UTC(SYSTIMESTAMP) as date) -
moved_to_queue_at)*86400)
```

For Microsoft SQL:

```
cast(cast((getutcdate() - moved_to_queue_at) as float)*86400 as
integer)
```

For DB2:

```
timestampdiff( 2, char((current timestamp - current timezone) -
moved_to_queue_at))
```

_empty

Usage

```
_empty(string expression)
```

The result is a logical expression that is true if the value of the string expression is either null or an empty string, and is false otherwise.

Translations Performed by Database

For Oracle:

```
((value) is null)
```

For all other database platforms:

```
(((value) is null) or ((value)=''))
```

Note: The reason the Oracle condition is different is that Oracle interprets an empty string as null, and a condition like `(anything=null)` is always false.

_not_empty

Usage

```
_not_empty(string expression)
```

The result is a logical expression that is false if the value of the string expression is either null or an empty string, and is true otherwise.

Translations Performed by Database

For Oracle:

```
((value) is not null)
```

For all other database platforms:

```
(((value) is not null) and (not ((value)='')))
```

Chapter

9 Capture Points Functionality in Interaction Server

This chapter describes the integrated capture points functionality added to Interaction Server starting in release 8.0.2. It covers these topics:

Overview

The integrated capture points provide a mechanism for capturing new interactions from external source systems, and for issuing various requests to existing interactions.

The capture points, with the exception of the Web Service Capture Point, also produce notifications on changes to interactions.

JMS Capture Point

The integrated JMS (Java Message Service) Capture Point functionality is supported in Interaction Server starting in release 8.0.2. This functionality enables Interaction Server to capture requests to Interaction Server from a JMS-compliant message queue and to send Interaction Server replies and interaction event notifications to JMS-compliant message queues, in the form of XML documents. In iWD 7.6.1, the JMS Capture Adapter was a separate component, but its functionality was integrated into Interaction Server 8.0.2.

File Capture Point

The integrated File Capture Point is supported in Interaction Server starting in release 8.0.21. This capture point is similar to JMS Capture Point as it works with Interaction Server requests, replies, and notifications in the form of XML documents. However, it uses specified file directories to capture requests and to produce replies and notifications in the form of XML files. This capture point is compatible with the iWD XML file capture adapter by means of configuration options and transformation scripts.

For the JMS and File Capture Points, XML representation of requests and notifications enables two modes of operation:

- Native mode, in which requests and interaction events notifications are consumed and generated, respectively, in Interaction Server native XML format.
- iWD (intelligent Workload Distribution) compatibility mode, in which supplied transformation scripts convert between iWD XML representation and native Interaction Server XML representation, supporting the full iWD API functionality (such as task creation, updating, holding, canceling, and various task state change notifications).

Database Capture Point

The integrated Database Capture Point, introduced in Interaction Server 8.1.0, is functionally equivalent to Database Capture Adapter in iWD 8.0. It provides the ability to capture new interactions and to propagate updates for existing interactions in the form of user defined queries to external databases. It also provides a mechanism of propagating interaction event notifications to the external system in the form of user-defined queries. No transformation scripts are used with Database Capture Point.

Web Service Capture Point

The integrated Web Service Capture Point, introduced in Interaction Server 8.1.2, provides a web service interface for interaction-related requests such as `submit`, `stop`, `update`, `hold`, `resume`, and `get info`, as well as for ping requests. It fully supports web service definition of the Web Service Capture Point in iWD 8.0 when operating in iWD compatibility mode. It can be configured to work with either HTTP or Secure HTTP, in both native and iWD-compatible modes. No transformation scripts are used with Web Service Capture Point.

XML Representation

The integrated JMS Capture Point is capable of capturing interactions in the form of XML documents from JMS-compliant message queue providers. The File Capture Point also captures XML documents, but from a local or network directory. This section describes both inbound and outbound XML messages for these two types of capture points.

Inbound Messages

A correctly generated XML document can use different encodings and will contain encoding specification in the document header. For that reason, XML should be always treated as binary data, not text. An XML document should always be put in a message queue as a binary message or written to a file as binary data.

Message queue capture points, such as the JMS capture point, can accept binary messages and text messages (for backward compatibility). To avoid incorrect or unnecessary transcoding, ensure that the XML document uses the same encoding that a specific message queue provider uses to encode the text messages.

The following encodings of *inbound* XML documents are supported:

- UTF-8
- UTF-16
- ISO-8859-1
- US-ASCII

Note: All *outbound* XML documents are encoded using UTF-8.

Operation Elements and Root Element

An inbound XML document can contain multiple operations, but only a single root element. For maximum flexibility the name of the root element can be anything, and it is not taken into account. The transformation scripts use `messages` as the name of the root element. If the document contains a single operation, this operation can be a root element. Any operation item is specified by an `interaction` element.

The following are sample XML messages:

Sample 1

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="submit" ExternalId="SomeExternalId"/>
```

Sample 2

```
<?xml version="1.0" encoding="UTF-8"?>
<messages>
<interaction operation="submit" ExternalId="ExternalId2"/>
<interaction operation="submit" ExternalId="ExternalId3"/>
</messages>
```

Sample 3

```
<?xml version="1.0" encoding="UTF-8"?>
<myinteractions>
<interaction operation="submit" ExternalId="ExternalId4"/>
<interaction operation="submit" ExternalId="ExternalId5"/>
</myinteractions>
```

Operations

The operation type is specified by the `operation` attribute and can be one of the following:

- `submit`—Submit a new interaction
- `update`—Update or change interaction properties
- `hold`—Hold the interaction
- `resume`—Resume the interaction
- `stop`—Stop or delete the interaction
- `getinfo`—Request interaction properties

Properties Element

The `properties` element, which should be a direct child of the `interaction` element, specifies the interaction properties that are needed to perform the operation.

- For `submit`, the `properties` element specifies all of the interaction properties including any user data or custom properties. It also specifies standard attributes such as the tenant and queue to which interactions are submitted. Any attribute can have a default value specified in the capture point configuration.
- For `update`, the `properties` element specifies properties that need to be changed. This might include the `Queue` property, in which case the interaction will be moved into the specified interaction queue. For the `update` operation, configured default values are not used, and the attribute `InteractionId` or `ExternalId` must be specified.
- For `hold`, `resume`, and `stop`, the only attribute required or processed is `InteractionId` (or `ExternalId`), which specifies the interaction that is to be held, resumed, or stopped.

For simplicity, any child element of the `properties` element can be specified as an attribute of the `interaction` element. For example, to hold an interaction the following `interaction` element can be used:

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="hold" InteractionId="itx00777"/>
```

Interaction Server supports key-value lists (of any depth) as interaction properties. To specify such attributes, natural XML structure is used. Note the `CustomerInfo` group of properties in the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="submit">
  <properties>
    <ExternalId>SomeExternalId</ExternalId>
    <CustomerSegment>Gold</CustomerSegment>
    <CustomerInfo>
      <FirstName>William</FirstName>
      <LastName>Bell</LastName>
    </CustomerInfo>
  </properties>
</interaction>
```

Interaction Server supports spaces and some special characters in interaction property names. To allow for this in XML messages, any property can have a “real” name specified as a `name` attribute.

For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="submit">
  <properties>
    <property name="First Name">William</property>
    <property name="Last Name">Bell</property>
  </properties>
</interaction>
```

The following is a list of Interaction Server’s predefined properties and their meanings. Custom properties can also specified and attached to the interaction.

`<properties>`

`<InteractionId>`Interaction Identifier. Can be omitted and generated by Interaction Server`</InteractionId>`

`<ParentId>`Parent interaction identifier`</ParentId>`

`<ExternalId>`Identifier used by the external system`</ExternalId>`

`<TenantId>`Tenant identifier`</TenantId>`

`<MediaType>`Interaction media type`</MediaType>`

`<InteractionType>`Inbound|Outbound|Internal`</InteractionType>`

`<InteractionSubtype>`Interaction subtype out of the defined list for tenant`</InteractionSubtype>`

`<IsOnline>`0|1`</IsOnline>`

`<IsHeld>`0|1`</IsHeld>`

`<Queue>`Name of the queue in which the interaction is initially placed`</Queue>`

`<Workbin>`Initial workbin name; optional`</Workbin>`

`<WorkbinAgentId>`Initial workbin agent id`</WorkbinAgentId>`

`<WorkbinAgentGroupId>`Initial workbin agent group id`</WorkbinAgentGroupId>`

```
  <WorkbinPlaceId>Initial workbin place id</WorkbinPlaceId>
  <WorkbinPlaceGroupId>Initial workbin place group id</WorkbinPlaceGroupId>
  <ReceivedAt>YYYY-MM-DD HH:MM:SS</ReceivedAt>
  <Priority>Initial interaction priority</Priority>
  <ServiceType>Service type</ServiceType>
  <ServiceObjective>Service objective in seconds</ServiceObjective>
</properties>
```

Delete Element

The delete element, which must be a direct child of the interaction element, is used only for the update operation, and specifies the names of the properties that are to be deleted. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="update" InteractionId="itx00777">
  <properties>
    <property name="Last Name">Ball</property>
  </properties>
  <delete>
    <property name="Middle Name"/>
  </delete>
</interaction>
```

Reason Element

The reason element, which must be a direct child of the interaction element, can specify the reason for the operation. This attribute is optional and can be used with the hold, resume and stop operations. The reason element has the attributes name and description. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="hold" ExternalId="Loan1022011-02">
  <reason name="AwaitingInfo" description="Waiting for credit history
report"/>
</interaction>
```

Processed and Error Queues in JMS Capture Point

For message queue capture points, a copy of the original message is put either into the processed or error queues specified by the options processed-queue-name and error-queue-name, respectively. These options are configured in the settings section of the Capture Point Application object. No reformatting of the message takes place and no XML parsing or transformation is involved. This is an exact copy of the original message.

Outbound Notifications

The outbound XML encoding is UTF-8. For the JMS capture point, the message type of the outbound notification messages is controlled by the option `outbound-message-type`, and can be either `binary` (the default) or `text`.

The messages placed in the notifications queue (JMS Capture Point) or folder (File Capture Point) consist of outbound notifications and responses to capture point requests. The correlation identifiers in notification messages are not set because these are unsolicited notifications and not the replies. The correlation identifier is set for reply messages to correlate responses with requests.

Outbound notifications are generated as separate XML documents with the root element `interaction`. The `operation` attribute specifies the type of notification and can be one of the following:

- `changed` The interaction properties have changed.
- `stopped` The interaction has been stopped/deleted.
- `held` The interaction has been put on hold.
- `resumed` The interaction has been resumed from hold.
- `moved` The submitted interaction has been moved from one queue.
- `assigned` The interaction has been delivered to an agent or pushed to a strategy.

Note: Timestamps for outbound notifications and responses sent by the integrated capture points are in UTC (Coordinated Universal Time). This is inconsistent with iWD capture points. Outbound notifications and responses sent by iWD capture points are in local time.

Properties Element

The `properties` element specifies all current interaction properties. The following is the list of predefined Interaction Server properties. Note that any user data is also presented along the predefined properties.

```
<properties>
  <InteractionId>Interaction identifier</InteractionId>
  <ParentId>Parent interaction identifier</ParentId>
 <ExternalId>Identifier used by external system</ExternalId>
 <TenantId>Tenant identifier</TenantId>
 <MediaType>Interaction media type</MediaType>
 <InteractionType>Inbound|Outbound|Internal</InteractionType>
 <InteractionSubtype>Interaction subtype out of the defined list for tenant</InteractionSubtype>
 <IsOnline>0|1</IsOnline>
 <IsHeld>0|1</IsHeld>
 <Queue>Current queue name</Queue>
 <Workbin>Current workbin name, optional</Workbin>
 <WorkbinAgentId>Workbin agent ID</WorkbinAgentId>
 <WorkbinAgentGroupId>Workbin agent group ID</WorkbinAgentGroupId>
 <WorkbinPlaceId>Workbin place ID</WorkbinPlaceId>
 <WorkbinPlaceGroupId>Workbin place group ID</WorkbinPlaceGroupId>
 <SubmittedBy>Capture point name</SubmittedBy>
 <InQueues>List of suggested destination queues</InQueues>
 <ReceivedAt>YYYY-MM-DD HH:MM:SS</ReceivedAt>
 <SubmittedAt>YYYY-MM-DD HH:MM:SS</SubmittedAt>
 <DeliveredAt>YYYY-MM-DD HH:MM:SS</DeliveredAt>
 <PlacedInQueueAt>YYYY-MM-DD HH:MM:SS</PlacedInQueueAt>
 <MovedToQueueAt>YYYY-MM-DD HH:MM:SS</MovedToQueueAt>
  <AssignedTo>Agent ID (Place ID if no Agent ID in login)</AssignedTo>
 <AssignedAt>YYYY-MM-DD HH:MM:SS</AssignedAt>
 <Priority>Current interaction priority</Priority>
 <ServiceType>Service type</ServiceType>
 <ServiceObjective>Service objective in seconds</ServiceObjective>
</properties>
```

Changed and Deleted Elements

The `changed` and `deleted` elements are used only with the `changed` notification and specify changed and deleted interaction properties respectively. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="changed" event_time="2010-10-22T07:34:05Z">
  <properties>

    <InteractionId>05512B2CQRPPR001</InteractionId>
    <InteractionType>Inbound</InteractionType>
    <InteractionSubtype>InboundNew</InteractionSubtype>
    <TenantId>107</TenantId>
    <Queue>Inbound</Queue>
    <MediaType>workitem</MediaType>
    <SubmittedBy>CapturePointMSMQPerceptron</SubmittedBy>
    <State>3</State>
    <ReceivedAt>2010-10-19T23:47:32Z</ReceivedAt>
    <SubmittedAt>2010-10-19T23:47:32Z</SubmittedAt>
    <DeliveredAt>2010-10-22T07:33:05Z</DeliveredAt>
    <PlacedInQueueAt>2010-10-19T23:47:32Z</PlacedInQueueAt>
    <MovedToQueueAt="MovedToQueueAt="">2010-10-
19T23:47:32Z</MovedToQueueAt>
    <AssignedAt>2010-10-22T07:33:05Z</AssignedAt>
    <AssignedTo>a0001</AssignedTo>
    <ExternalId>MyExternalId</ExternalId>
    <LastName>Ball</LastName>
  </properties>
  <changed>
    <LastName>Ball</LastName>
  </changed>
  <deleted>
    <CustomerSegment/>
  </deleted>
  <actor type="agent" tenant="107" place="p0001" agent="a0001"/>
</interaction>
```

Reason and Actor Elements

The `reason` element specifies the reason for the operation, if it is provided by the server (and if, in turn, it was provided by the client in the request).

The `actor` element specifies the actor of the operation and can be one of the following types:

- `agent` The actor is an agent application and the `tenant`, `place`, and `agent` attributes specify the tenant identifier, place name and agent employee ID.
- `strategy` The actor is a strategy and the `tenant`, `strategy`, and `router` attributes specify the strategy.
- `mediaserver` The actor is a media server and the `mediaserver` attribute specifies the name of the media server.

The following is an example of the `actor` and `reason` elements (not all properties are included in this example):

```
<?xml version="1.0" encoding="UTF-8"?>
<interaction operation="held" event_time="2010-10-22T07:54:43Z">
  <properties>
    <InteractionId>05512B2CQRPPR001</InteractionId>
    <ExternalId>MyExternalId</ExternalId>
  </properties>
  <actor type="agent" tenant="107" place="p0001" agent="a0001"/>
  <reason name="AwaitingInfo" description="Waiting for credit report"/>
</interaction>
```

The following is an example of the `strategy` actor:

```
<actor type="strategy" tenant="107" strategy="InboundStrategy"
router="URServer"/>
```

The following is an example of the `mediaserver` actor:

```
<actor type="mediaserver" server="CapturePointJMS"/>
```

Party Element

The `party` element is used in `assigned` notifications and specifies a party to which the interaction has been assigned. A party can be either an agent or a strategy. The `type` attribute specifies the party type and can be either `agent` or `strategy`. The `tenant` attribute specifies the identifier for the tenant to which the party belongs.

The following is an example of an `agent` party (note the `place` and `agent` attributes):

```
<party type="agent" tenant="107" place="p0001" agent="a0001"/>
```

The following is an example of a `strategy` party (note the `strategy` and `router` attributes):

```
<party type="strategy" tenant="107" strategy="InboundStrategy"
router="URServer"/>
```

Responses to Capture Point Requests

The responses are formatted the same way as notifications. Everything that is applicable to notification messages also applies to response messages, except for:

- Correlation Id (JMS Capture Point)
- Response Types (JMS and File Capture Points)
- Error Notification (JMS and File Capture Points)

Correlation Id

In response messages the JMSCorrelationID parameter is set to the JMSMessageID of the request by default.

In Interaction Server 8.1.200 and later, you can change this default behavior using the JMS Capture Point’s `use-correlation-id-in-reply` option: with a setting of `true,` the JMSCorrelationID parameter of the reply message is set to the value of JMSCorrelationID parameter of the request. A setting of `false` retains the default behavior.

Response types

The following operation types are used in responses to capture point requests:

- `submitted`—Only as a response to the `submit` operation, and never as an unsolicited notification
- `changed`—As a response to a capture point’s `change` request or as a notification regarding changes to interactions submitted by this capture point

- `stopped`—As a response to the `stop` operation or as unsolicited notification if an interaction submitted by this capture point is stopped by another entity
- `held`—As a response to the `hold` operation or as unsolicited notification if an interaction submitted by this capture point is held by another entity
- `resumed`—As a response to the `resume` operation or as unsolicited notification if an interaction submitted by this capture point is resumed by another entity
- `info`—Only as a response to a `getinfo` request from a capture point
- `error`—In response to any failed request

Error Notification

The `error` element specifies the error code and (optionally) a description if an operation has failed. The following is an example of the `error` element:

```
<interaction operation="error" code="agent" description="107"/>
```

Transformation

The integrated capture point functionality in Interaction Server also supports message transformation. XML transformation can be applied to inbound and outbound messages to adjust to your specific format. Each incoming XML message and each outgoing message passes through transformation scripts, allowing integration with custom interaction definitions and XML formats.

A sample iWD compatibility transformation script is included with the installation of Interaction Server and can be used as a basis for customization. The sample script is a transformation script that allows you to format an XML message according to the iWD 8.0 schema, and then have it transformed into Interaction Server’s native message format. See “iWD Compatibility Transformation Scripts” on page 278 for more information.

Inbound vs. Outbound Transformations

The *inbound* transformation Groovy script (if specified by the `xsl-inbound-transform-path` option) is called when a capture point needs to transform an inbound XML message.

The *outbound* transformation Groovy script (if specified by the `xsl-outbound-transform-path` option) is called when a capture point needs to transform outbound XML message.

Configuration Options

In order to enable transformation, the following options must be configured in the `settings` section of the Capture Point Application object:

Note: For full descriptions of each of the following options, refer to the *eServices 8.1 Reference Manual*.

- `xsl-inbound-transform-path` String representation of a Uniform Resource Identifier (URI) that points to a shared Groovy script file containing the transformation script.
- `xsl-outbound-transform-path` String representation of a URI that points to a shared Groovy script file containing the transformation scripts for outbound notifications.

Transformers Interface

The interface for the transformation script is defined as follows:

```
package
com.genesyslab.eservices.interactionserver.capturepoints.xmltransfo
rmer;

public interface XmlTransformer
{
   void init(java.util.Properties parameters);
   byte[] transform(byte[] inputXml, java.util.Properties
parameters);
   void cleanup();
   void setLogger(Logger logger);
   void reconfigure(java.util.Properties parameters);
}
```

Any custom script should implement this interface in order to be usable by Interaction Server. For a good starting point for a custom script, refer to the sample scripts that are provided with Interaction Server. These scripts are described in “iWD Compatibility Transformation Scripts” on page 278.

When Interaction Server creates a transformer, it calls its `init` method and passes all of the parameters that are defined in the `inbound-transformer-parameters` section (for inbound transformation scripts) or in the `outbound-transformer-parameters` section (for outbound transformation scripts). The transformer object should store these properties for possible future use during transformation.

The main functional method `transform` transforms the `inputXML` XML message into the required form and returns the transformed XML message that will be either parsed by Interaction Server (for inbound messages) or, for outbound messages, put directly into notification message queue (for message queue capture points). Each call to the `transform` method by Interaction Server can be given a set of properties to account for during the individual transformation process of a single document. In 8.0.2, Interaction Server provides the following parameters to the transformation method:

- `CapturePointName` The name of the capture point invoking the transformation
- `CurrentTime` The current UTC timestamp in the format `YYYY-MM-DDTHH:MM:SSZ`

The `Logger` interface provided to the script allows for logging of any diagnostic or error messages into the Interaction Server log by the same means that Interaction Server logs messages. Logging configuration works the same as for any other Interaction Server messages, including logging to the console, a file, or network logging. The `Logger` interface is defined as follows:

```
package
com.genesyslab.eservices.interactionserver.capturepoints.xmltransfo
rmer;

public interface Logger
{
    public static enum LogLevel { DEBUG, TRACE, STANDARD };
    public void log(LogLevel level, String logMessage);
}
```

XML Encoding Considerations

Interaction Server can parse XML messages in the following encodings:

- UTF-8
- UTF-16
- ISO-8859-1
- US-ASCII

Outbound notification messages are encoded in UTF-8. This requires the transformation scripts to provide output in one of the supported encodings (for inbound transformations) and to be capable of parsing the UTF-8 encoded XML messages (for outbound transformation scripts).

Because Groovy scripts use the `XmlParser` class to parse XML messages, they have no difficulty processing UTF-8 and can support any encoding supported by Java (depending on the installed packages). The outbound transformation scripts can also produce outbound XML messages in any encoding if the appropriate Java packages are installed. The outbound transformer provided with Interaction Server generates output in UTF-8 and is capable of generating output (without any additional Java packages) in the following encodings:

- US-ASCII
- ISO-8859-1
- UTF-8

- UTF-16BE
- UTF-16LE
- UTF-16

Care should be taken to correctly handle encoding in Groovy. The recommended place to look for an example is the transformation scripts that are provided with the Interaction Server installation. The following pattern shows how to correctly generate XML in the required encoding and with appropriate XML declaration:

```
def outputDoc = new StreamingMarkupBuilder()
outputDoc.encoding = "utf-8"
def outputStream = new ByteArrayOutputStream()

new OutputStreamWriter(outputStream, outputDoc.encoding) <<
outputDoc.bind {
   mkp.xmlDeclaration()
   somecontent {
    }
}
return outputStream.toByteArray()
```

iWD Compatibility Transformation Scripts

There are two Groovy scripts provided for transformation of inbound and outbound messages to and from iWD message format. These scripts provide backward compatibility with iWD 8.0 message format considering the structure of the iWD specific business process provided with iWD 8.0. The iWD 8.0 message format is described in detail in the *iWD 8.0 Deployment Guide*. Only general rules of the transformation process are described here.

The provided transformation scripts are only for standard iWD messages as specified in the document. For custom messages, customization of these scripts is necessary.

Inbound Transformation Script

The inbound transformation script path is `iwd_scripts\iWD2IxnServerTransformer.groovy`. The script produces output in UTF-8 for Interaction Server to parse.

Inbound Script Parameters

The script uses the following parameters:

- `CompleteQueues` A comma-separated list of queue names for completed interactions (default `iWD_Completed`).
- `RestartQueues` A comma-separated list of queue names for new interactions (default `iWD_New`).

- `CancelQueues` A comma-separated list of queue names for canceled interactions (default `iWD_Canceled`).
- `ExtendedAttributes` A comma-separated list of attributes that must be present under the `<Ext>` tag of the `CreateTask` iWD message.
- `AllowAnyAttributes` If set to `true` or `yes`, the transformation script copies any unknown attributes to the transformed message.
- `CaseSensitiveAttributes` If set to `false` or `no`, the transformation script ignores the case of known attribute names (including `Ext` and `Data` section names).
- `CaseSensitiveActions` If set to `false` or `no`, the transformation script ignores letter case of action names.

Note: Interaction Server parser and interaction representation are case sensitive. The customized script must take care to produce the output in the correct case.

Root Element

The root element of iWD inbound message may be `GTLMessages` or `GTLMessage`. The script checks for the root element name and generates an error if the document root element is anything else. The root element of the transformed messages is always `messages` and the child elements describe the operations.

Transforming Actions

The iWD message action is the name of the tag of the child element of the root element. Possible actions and their translations are as follows:

- `CreateTask` Translates to `<interaction operation='submit'>`
- `GetTaskInfo` Translates to `<interaction operation='getinfo'>`
- `UpdateTask` Translates to `<interaction operation='update'>`
- `CompleteTask` Translates to `<interaction operation='update'>`
- `HoldTask` Translates to `<interaction operation='hold'>`
- `ResumeTask` Translates to `<interaction operation='resume'>`
- `RestartTask` Translates to `<interaction operation='update'>`
- `CancelTask` Translates to `<interaction operation='update'>`

Note: CompleteTask, RestartTask, and CancelTask are transformed into the update operation, which allows changing the queue for the interaction. The queue name is then added based on transformer parameters. Specifically, for the CompleteTask action, the first queue name from the transformer parameter CompleteQueues is added as the Queue property of the translated message. For the rest of these actions, the first queue name from the appropriate parameter is taken.

Transforming Properties

The following transformation takes place for the inbound iWD message:

- All known direct children of the action element are translated according to Table 35 and put into the properties tag of the transformed message.
- If any unknown tag is encountered, it is ignored if the AllowAnyAttributes option of the transformer is not set to true or yes. If the option is set to true or yes, the attribute is copied without any translation to the properties tag of the transformed message.
- If the Ext tag is encountered, all children of this tag are copied into the properties tag of the transformed message with the prefix IWD_ext_.
- If the Data tag is encountered, all children of this tag are copied into the properties tag of the transformed message without any changes.
- If the Reason tag is encountered, it is translated to the reason tag with the name attribute containing the value of the original Reason tag (for example <reason name="Original Reason"/>).

Note: Attributes will show up in the transformed message in the order describe above.

Table 35: Translation Table for Known Attributes (Inbound)

iWD Message Attribute	Attribute Name in Interaction Server	Notes
BrokerId	InteractionId	
CaptureId	ExternalId	
Actor		Ignored
ActionDateTime		Ignored
tenantId	IWD_tenantId	
solutionId	IWD_solutionId	
capturePointId	IWD_capturePointId	Same as SubmittedBy

Table 35: Translation Table for Known Attributes (Inbound) (Continued)

iWD Message Attribute	Attribute Name in Interaction Server	Notes
priority	Priority	
businessValue	IWD_businessValue	
channel	iWD_channel	
category	IWD_category	
activationDateTime	IWD_activationDateTime	No default value
dueDateTime	IWD_dueDateTime	No default value
expirationDateTime	IWD_expirationDateTime	No default value
processId	IWD_processId	
departmentId	IWD_departmentId	
reprioritizeDateTime	IWD_reprioritizeDateTime	
Hold	IsHeld	Changed to 0 or 1 (from false or true)

Outbound Transformation Script

The outbound transformation script path is `iwd_scripts\IxnServer2iWDTransformer.groovy`. The script produces output in UTF-8 for Interaction Server to put into the notification queue (or to deliver to an external system by other means).

The script uses the following parameters:

- `CompleteQueues` A comma-separated list of queue names for completed interactions (default `iWD_Completed`)
- `RestartQueue` A comma-separated list of queue names for new interactions (default `iWD_New`)
- `CancelQueues` A comma-separated list of queue names for canceled interactions (default `iWD_Canceled`)
- `RejectQueues` A comma-separated list of queue names for rejected interactions (default `iWD_Rejected`)
- `ExtendedAttributes` A comma- separated list of interaction attributes that has to appear under the `<Ext>` tag of the iWD notification messages

The script uses the following internal parameters (hard-coded as static member variables) to maintain the compatibility with previous versions of iWD:

- `includeWorkbinQueueName` If set to `true`, the default, the script includes the workbin queue name in `TaskDistributedQueue` messages, as is done in iWD. If set to `false`, the actual workbin name is included.
- `specificQueueNotifications` If set to `true`, the script generates specific unsolicited notifications based on the queue name (for example, `TaskCompleted`, `TaskCanceled`, `TaskRestarted`, `TaskRejected`, `TaskErrorHeld` are generated instead of a generic `TaskDistributedQueue`). The default value is `false`, which generates the generic `TaskDistributedQueue` as is done in iWD.

The two internal parameters described above can be changed in the script file. Changes take effect after restart.

Root Element

The notification messages produced by Interaction Server always contain a single notification. This notification is a root element. The outbound transformation script expects the root element to be `messages` and if it is, treats all of the child elements as notifications (which always have the name `interaction`). If the root element is not `messages`, then it is expected to be `interaction` and is treated as a single notification element. In all other cases the transformation fails. In output XML, the root element is always `GTLMessages` and child elements are iWD notification elements.

Transforming Actions

The Interaction Server operation is specified by the `operation` attribute of the `interaction` tag. Possible actions and their translations are as follows:

- `<interaction operation='created'>` Translates to `TaskCreated`
- `<interaction operation='changed'>` Translates to `TaskUpdated`
- `<interaction operation='stopped'>` Translates to nothing
- `<interaction operation='held'>` Translates to `TaskHeld`
- `<interaction operation='resumed'>` Translates to `TaskResumed`
- `<interaction operation='info'>` Translates to `TaskInfo`
- `<interaction operation='moved'>` Translates to one of `TaskCompleted`, `TaskRestarted`, `TaskCanceled`, `TaskRejected` or `TaskDistributedQueue`
- `<interaction operation='assigned'>` Translates to `TaskAssigned`
- `<interaction operation='error'>` Translates to `Error`

Note the transformation of the `moved` notification to different iWD notifications. The choice of the appropriate notification is made based on the `Queue` attribute of the original notification message as follows:

- If the value of the `Queue` attribute is included in the `CompleteQueues` parameter, then the `TaskCompleted` notification is generated.
- If the value of the `Queue` attribute is included in the `RestartQueue` parameter, then the `TaskRestarted` notification is generated.

- If the value of the `Queue` attribute is included in the `CancelQueues` parameter, then the `TaskCanceled` notification is generated.
- If the value of the `Queue` attribute is included in the `RejectQueues` parameter, then the `TaskRejected` notification is generated.
- Otherwise, the `TaskDistributedQueue` notification is generated.

Transforming Properties

The following transformation takes place for the outbound iWD message:

- All known direct children of the `properties` tag are translated according to Table 36 and put into the transformed message as direct children of the notification message.
- All direct children of the `properties` tag that begin with prefix `IWD_ext_` are put into the `Ext` tag of the transformed message as child elements with the same name, but without the prefix `IWD_ext_`.
- All other children of the `properties` tag are put into the `Data` tag of the transformed message as child elements with exactly same names.

Table 36: Translation Table for Known Attributes (Outbound)

Attribute Name in `properties`	iWD Message Attribute	Notes
InteractionId	BrokerId	
ExternalId	CaptureId	
SubmittedBy	CapturePointId	
IWD_CapturePointId		Ignored. `SubmittedBy` is used instead.
<actor>	Actor	Strategy, agent ID, or server name.
<reason>	Reason	Attribute `name of reason` tag.
event_time attribute of the notification	EventDateTime	If not present, it is set to the `CurrentTime` parameter of transformation.
IWD_tenantId	tenantID	
IWD_solutionId	solutionId	
IWD_departmentId	departmentId	
IWD_processId	processId	
IWD_channel	channel	

Table 36: Translation Table for Known Attributes (Outbound) (Continued)

Attribute Name in `properties`	iWD Message Attribute	Notes
IWD_category	category	
State, Queue, IsHeld	status	Based on a set of attributes.
IWD_businessCalendarId	businessCalendarId	
SubmittedAt	createdDateTime	
HeldAt	heldDateTime	Only if held, no translation in iWD 8.0
AssignedAt	assignedDateTime	
CompletedAt	completedDateTime	
IWD_activationDateTime	activationDateTime	
IWD_dueDateTime	dueDateTime	
IWD_expirationDatetime	expirationDateTime	
Priority	priority	
IWD_reprioritizeDateTime	reprioritizeDateTime	
IWD_businessValue	businessValue	
AssignedTo	assignedToUser	
Queue	Queue	
Workbin, WorkbinAgentId, WorkbinAgentGroupId, WorkbinPlaceId, WorkbinPlaceGroupId	QueueType	• If Workbin is empty InteractionQueue • If WorkbinAgentId is set AgentWorkbin • If WorkbinAgentGroupId is set AgentGroupWorkbin • If WorkbinPlaceId is set PlaceWorkbin • If WorkbinPlaceGroupId is set PlaceGroupWorkbin
WorkbinAgentId, WorkbinAgentGroupId, WorkbinPlaceId, WorkbinPlaceGroupId	QueueTarget	First not empty

Sample Configurations for JMS Capture Point

This section provides some sample configurations for JMS Capture Point applications working with the following JMS providers: OpenMQ, TIBCO, and WebSphere MQ.

Sample Configuration for OpenMQ Capture Point

Perform the following steps to set up queues using Open Message Queue Administration Console.

Procedure: Setting up queues with Open Message Queue Administration Console

Start of procedure

1. Connect to the OpenMQ broker that is running.
2. Add the following queues using the `Add Broker Destination` dialog: `Inbound`, `Processed`, `Error`, and `Notification`.
3. For each queue that you have added, set `Max Number of Producers` and `Max Number of Active Consumers` to `Unlimited`.
4. Add a new Object Store and set the following JNDI Naming Service Properties:
   a. Set `java.naming.factory.initial` to `com.sun.jndi.fscontext.RefFSContextFactory`.
   b. Set `java.naming.provider.url` to `file:///D:/OpenMQExample`.

   Note: This is the directory in which the `.bindings` file containing definitions will be saved.

5. Connect to the newly created object store.
6. Add a connection factory object using the `Add Connection Factory Object` dialog:
   a. Specify the lookup name, such as `ConnectionFactory`.
   b. Specify the `Factory Type` as `QueueConnectionFactory`.
   c. In the `Client Identification` tab, specify the `Default Username` and `Default Password` (for example, guest/guest).

d. Add destinations to the object store for all four queues that you defined previously:

   i. For the `Inbound` queue, specify the lookup name `inbound` and destination name `Inbound`.

   ii. For the other queues, set the lookup names as `processed`, `error`, and `notification`.

Note: The lookup names can be different from the destination names.

7. After the above steps have been completed, the folder `D:/OpenMQExample/` contains the `.bindings` file with connection factory and queue definitions. Open the file, examine it for the presence of the defined queues and connection factory, and save it with File format set to `UNIX` so that it is possible to use it on UNIX operating systems.

End of procedure

Next Steps

- Create a Capture Point `Application` in Configuration Manager.

Procedure: Creating a Capture Point application in Configuration Manager (OpenMQ example)

Start of procedure

1. Create a Capture Point `Application` in the Configuration Manager named `CP_OpenMQ_solaris`.
2. On the `Options` tab create a section `settings`. In this section add the following options:
   - `capture-point-type=jms`
   - `inbound-queue-name=inbound` (the same as in the queue lookup name above)
   - `error-queue-name=error`
   - `processed-queue-name=processed`
   - `notification-queue-name=notification`
   - `xsl-inbound-transform-path=./iwd_scripts/iWD2IxnServerTransformer.groovy` (points to the default iWD Compatibility scripts)
   - `xsl-outbound-transform-path=./iwd_scripts/IxnServer2iWDTransformer.groovy`
   - `username=guest` (as configured in the connection factory)
   - `password=guest`

- `jms-initial-context-factory= com.sun.jndi.fscontext.RefFSContextFactory`
- `jms-provider-url=file:///home/InteractionServer` (the path points to the folder where the `.bindings` file (in UNIX file format) is stored on the Interaction Server host)
- `jms-connection-factory-lookup-name=ConnectionFactory`

3. On the `Connections` tab add the Interaction Server that will use this JMS Message queue.

End of procedure

Next Steps

- Configure the Interaction Server options to load JVM and all of the required libraries.

Procedure: Configuring Interaction Server options to load JVM and all of the required libraries (OpenMQ example)

Start of procedure

1. In the options of the Interaction Server to which the Capture Point `Application` is connected, create a section called `java-config` and add the option:

   `jvm-path= /usr/local/java/jdk1.6.0_22/jre/lib/sparcv9/server/libjvm.so` (this is the full path to the `libjvm.so` (`jvm.dll` on Windows) on the host on which the Interaction Server is deployed)

2. Create a section called `jvm-options` and add the following three options:
   - `-Djava.class.path=./jms/jms_wrapper.jar:/home/OpenMQ_sol/mq/lib/imq.jar:/home/OpenMQ_sol/mq/lib/fscontext.jar:/home/OpenMQ_sol/mq/lib/jms.jar:./transformation/xml_transformer_capture_point.jar:./transformation/groovy-all-1.7.3.jar:./transformation/xercesImpl.jar:./transformation/xsltc.jar:`

     This option specifies the classpath to all of the Java archives that are necessary for JMS Capture Points on OpenMQ with iWD compatibility transformations to run. Note that the jar files `imq.jar`, `fscontext.jar`, and `jms.jar` are located in the Open MQ intallation directory and are *not* supplied in the Interaction Server installation package.
   - `-Xoss1m`
   - `-Xss1m`

Note: The options `-Xoss1m` and `-Xoss1m` must have empty values.

End of procedure

Sample Configuration for TIBCO Capture Point

This example assumes the following:

- The host of the TIBCO message queue service is called `tibhost`.
- Queues called `inbound`, `error`, `notification`, and `processed` are defined.
- Both user name and password are `guest`.
- The connection factory is called `tibconnectionfact`.

Procedure:
Configuring the Capture Point application in Configuration Manager (TIBCO example)

Start of procedure

1. On the `Options` tab create a section called `settings`. In this section add the following options:
   - `capture-point-type=jms`
   - `inbound-queue-name=inbound` (the same as the queue name)
   - `error-queue-name=error`
   - `processed-queue-name=processed`
   - `notification-queue-name=notification`
   - `xsl-inbound-transform-path=./iwd_scripts/iWD2IxnServerTransformer.groovy` (points to the default iWD Compatibility scripts)
   - `xsl-outbound-transform-path=./iwd_scripts/IxnServer2iWDTransformer.groovy`
   - `username=guest`
   - `password=guest`
   - `jms-connection-factory-lookup-name=tibconnectionfact` (the name of the connection factory on TIBCO)
   - `jms-initial-context-factory=com.tibco.tibjms.naming.TibjmsInitialContextFactory`
   - `jms-provider-url=tibjmsnaming://tibhost:7222`
2. On the `Connections` tab, add the Interaction Server which will use this JMS Message queue.

End of procedure

Next Steps

- Configure the Interaction Server options that are required to load JVM and the necessary libraries.

Procedure: Configuring Interaction Server options to load JVM and all of the required libraries (TIBCO example)

Start of procedure

1. On the `Options` tab of the Interaction Server Application, create a section named `java-config` and add the option:

   `jvm-path=/usr/local/java/jdk1.6.0_22/jre/lib/sparcv9/server/libjvm.so` (the full path to the `libjvm.so` (`jvm.dll` if the operating system is Windows) on the host on which the Interaction Server is deployed)

2. Create a section named `jvm-options` and add the following three options:

   - `-Djava.class.path=./jms/jms_wrapper.jar:/opt/tibco/ems/6.0/lib/jms.jar:/opt/tibco/ems/6.0/lib/tibjms.jar:./transformation/xml_transformer_capture_point.jar:./transformation/groovy-all-1.7.3.jar:./transformation/xercesImpl.jar:./transformation/xsltc.jar:`

     This option specifies the class path to all of the Java archives that are necessary for JMS Capture Points on TIBCO with iWD compatibility transformations to run. Note that the jar files `tibjms.jar` and `jms.jar` are located in the TIBCO installation directory and are *not* supplied in the Interaction Server installation package.
   - `-Xoss1m`
   - `-Xss1m`

Note: The options `-Xoss1m` and `-Xoss1m` must have empty values.

End of procedure

Sample Configuration for IBM WebSphere MQ

Perform the following steps to set up queues using IBM WebSphere MQ Explorer.

Procedure:
Setting up queues using IBM WebSphere MQ Explorer

Start of procedure

1. Start WebSphere MQ Explorer. Find the Object tree in the Navigator window.
2. Right-click the Queue Managers node and select New to create a new Queue Manager. Follow the steps in the resulting Wizard, choosing a name (for example, my_QManager) and unique listening port.
3. As the Object tree is updated, find the Queues node under the new Queue Manager. Right-click this node and select New > Local Queue.
4. Create Local Queues named mq_inbound, mq_notifications, mq_errors and mq_processed. Select Persistent for the Default Persistence setting.
5. With the Queues node selected in the Object tree, right-click mq_inbound in the Content pane and select Put Test Message. Enter any text of your choice in the Message data field, then click Put message. This test message will wait in the queue until the Capture Point retrieves it.
6. In the Object tree, right-click the JMS Administered Objects node and select Add Initial Context. Choose File system for the JNDI namespace location and select the directory where the corresponding storage file will be created.
7. The new node for initial context now appears in the Object tree. Select it and verify that the Connection Factories and Destinations nodes appear under it. If necessary, right-click and use the context menu to connect to the InitialContext object make these nodes visible.
8. Right-click Connection Factories and select New > Connection Factory.Enter or select the following values:
   a. Sample name—my_ConnFactory
   b. Messaging provider—WebSphere MQ
   c. Transport—MQ Client
   d. Base queue manager and Broker queue manager (last screen)—the Queue Manager that you created in Step 2.
   e. Host name and port—correct values for your environment
9. Right-click Destinations and select New > Destination and create four new Destinations that correspond to the queues that you created in Step 4:
   a. Set type = Queue
   b. Set names = jms-inbound, jms-errors, jms-notifications, and jms-processed.
   c. On the last screen, select the proper Queue Manager and Queue objects.

10. Find the file named `.bindings` at the location established in Step 6. It will be referred to later on the sample configuration.

End of procedure

Next Steps

Configure the JMS Capture Point Application object.

Procedure: Configuring the JMS Capture Point application in Configuration Manager (WebSphere MQ example)

Start of procedure

1. On the `Options` tab create a section called `settings`. In this section add the following options:
   - `capture-point-type=jms`
   - `inbound-queue-name=inbound` (the same as the corresponding Destination name)
   - `error-queue-name=jms-error`
   - `processed-queue-name=jms-processed`
   - `notification-queue-name=jms-notifications`
   - `xsl-inbound-transform-path=./iwd_scripts/iWD2IxnServerTransformer.groovy` (points to the default iWD Compatibility scripts)
   - `xsl-outbound-transform-path=./iwd_scripts/IxnServer2iWDTransformer.groovy`
   - `jms-connection-factory-lookup-name=my_ConnFactory` (the name of the connection factory that you created in WebSphere MQ)
   - `jms-initial-context-factory=com.sun.jndi.fscontext.RefFSContextFactory`
   - `jms-provider-url=file:///home/InteractionServer` (the path points to the folder where the .bindings file—in UNIX file format—is stored on the Interaction Server host)
2. On the `Connections` tab, add the Interaction Server which will use this JMS Message queue.

End of procedure

Next Steps

- Configure the Interaction Server options that are required to load JVM and the necessary libraries.

Procedure: Configuring Interaction Server options to load JVM and all of the required libraries (WebSphere MQ example)

Start of procedure

1. On the `Options` tab of the Interaction Server Application, create a section named `java-config` and add the option:

   `jvm-path=/usr/local/java/jdk1.6.0_22/jre/lib/sparcv9/server/libjvm.so` (the full path to the `libjvm.so`, or `jvm.dll` if the operating system is Windows, on the host on which the Interaction Server is deployed).

2. Create a section named `jvm-options` and add the following three options:
   - `-Djava.class.path=./jms/jms_wrapper.jar:./transformation/xml_transformer_capture_point.jar: ./transformation/groovy-all-1.7.3.jar:./transformation/xercesImpl.jar:./transformation/xsltc.jar:/usr/location/jms/mq/com.ibm.mq.jar:/usr/location/jms/mq/com.ibm.mqjms.jar: /usr/location/jms/mq/fscontext.jar:/usr/location/jms/mq/jms.jar:`
   - `-Xoss1m`
   - `-Xss1m`

Note: The options `-Xoss1m` and `-Xoss1m` must have empty values.

End of procedure

Note on the first of the three options in Step 2: this option specifies the class path to Java archives that are necessary for JMS Capture Points on WebSphere MQ with iWD compatibility transformations to run. Note that in the class-path the jar files `com.ibm.mq.jar`, `com.ibm.mqjms.jar`, `fscontext.jar`, and `jms.jar` are located in some user-defined location, and are not supplied in the Interaction Server installation package.These files are installed on your host by the WebSphere MQ installation, server or client, and are typically located in the subdirectory `.../WebSphere MQ/Java/lib`.

The connection to WebSphere MQ requires more than just the four jar files listed in the class-path, as these jar files depend on other jar files in the same directory. Therefore, the class-path should refer to them at the same location where they were placed by WebSphere MQ installation. Or, if they were copied, all files contained in the `./lib` directory should be copied to the new location.

Configuring JMS Capture Point to use SSL

This section provides two examples of enabling SSL, on OpenMQ and on TIBCO. These providers use different approaches for client configuration. However, in general, configuration of an SSL connection to these and other providers can be expected to consist of the following major steps:

1. Prepare the certificates.
2. Configure the JMS provider to operate in SSL mode.
3. Configure the options in Interaction Server's `jvm-options` section and add required JARs to the class path.
4. Configure the JMS Capture Point.

Procedure: Configure Capture Point to use SSL (OpenMQ example)

Prerequisites

This example assumes that an instance of Open MQ is configured and operating with a JMS Capture Point, without SSL.

Start of procedure

The first several steps involve configuring the OpenMQ broker

1. Generate a self-signed broker certificate:
   a. Run keytool to generate a key store (if one does not already exist) to generate a self-signed certificate:
      `<OpenMQ installation dir>\mq\bin>imqkeytool`
   b. Answer all the prompts and remember the chosen passwords. By default, the keystore will be called `keystore` and will be located in `<OpenMQ installation dir>\etc\mq.`
2. Add `ssljms` to active broker services:
   a. Locate the file `<OpenMQinstallation>\var\mq\instances\imqbroker\props\config.properties.`
   b. At the end of the file, add the following line:
      `imq.service.activelist=ssljms,admin,httpjms`
   c. Set the SSL port by adding the following line:
      `imq.ssljms.tls.port=1756`
   d. Restart the broker.

      The broker will prompt the user for a keystore password.

3. Update the connection factory properties: In the `.bindings` file, find the line

   ```
   {Your connection factory lookup name}/RefAddr/44/Content=
   ```

   and change it to

   ```
   {Your connection factory lookup
   name}/RefAddr/44/Content=mqssl\://{your broker host}\:1756
   ```

   where 1756 is the same port as that set in the broker properties.

   This operation can be done using the OpenMQ Administration Console by selecting the corresponding connection factory and adding `mqssl://{your broker host}:1756` to the Message Server Address List properties on its `Connection Handing` tab.

The next steps involve configuring Interaction Server.

4. Export the broker certificate to a trust store:
   a. Export the broker certificate with the following command:

      ```
      keytool -export -alias imq -keystore keystore -file
      openmqbroker.cer
      ```

   b. Copy the `.cer` file to Interaction Server's host and import it to a local trust store:

      ```
      keytool -import -keystore truststore.jks -file openmqbroker.cer
      -alias openmqbroker
      ```

5. Add the following to the Interaction Server `jvm-options` section:

   ```
   -Djavax.net.ssl.trustStore= {Path to the local trust
   store}/truststore.jks
   -Djavax.net.ssl.trustStorePassword={your local trust store password}
   -Djavax.net.ssl.trustStoreType=jks
   ```

   For debugging purposes, you can also add the following option, which prints debug information to the console:

   ```
   -Djavax.net.debug=ssl:handshake,data,trustmanager,record
   ```

6. Finally, configure the JMS Capture Point by adding the following to the `jms-additional-context-attributes` section:

   ```
   java.naming.security.protocol=ssl
   java.naming.security.authentication=simple
   ```

End of procedure

It should be noted that in this example, the JNDI naming service used has all of the relevant context stored in a `.bindings` file and does not have any mechanism of authorization and authentication. With other JNDI services, the user accessing JNDI may have to provide a username and a password, which can be different from the JMS connection credentials. If this is the case, the JMS Connection credentials must be specified in the JMS Capture Point `settings` section as `username` and `password`, while the JNDI username and password must be specified in the `jms-additional-context-attributes` section as `java.naming.security.principal` and `java.naming.security.credentials` respectively.

Procedure: Configure Capture Point to use SSL (TIBCO example)

Prerequisites

This example assumes that:

- An instance of TIBCO Enterprise Message Service is configured and operating with a JMS Capture Point, without SSL.
- TIBCO EMS 6.0 is running on a host named `tibcohost`.
- OpenSSL is present.

Start of procedure

The first several steps involve configuring the TIBCO EMS:

1. Use OpenSSL to generate the following certificates:
   a. Generate a server certificate:
   ```
   openssl req -x509 -days 365 -subj "/C=US/ST=California/L=Daly
   City/CN=tibcohost.genesyslab.com" -newkey rsa:2048 -keyout
   tibcoserver.key.pem -out tibcoserver.pem
   ```
   Note that the PEM password in this example is `tibcoserver`.

   b. Generate a client certificate:
   ```
   openssl req -x509 -days 365 -subj "/C=US/ST=California/L=Daly
   City/CN=tibcohost.genesyslab.com" -newkey rsa:2048 -keyout
   tibcoclient.key.pem -out tibcoclient.pem
   ```
   Note that the PEM password in this example certificate is `tibcoclient`.

   c. Export the generated certificate and the key into a client identity:
   ```
   openssl pkcs12 -export -in tibcoclient.pem -inkey tibcoclient.key.pem -out
   tibcoclient.p12
   ```
2. Configure TIBCO properties:
   a. New configuration file: this example assumes that the relevant certificates are copied into the folder `/opt/tibco/ems/6.0/samples/certs/`. Prepare a new TIBCO configuration file `tibemsd_ssl.conf` based on `tibemsd.conf` by adding or modifying the following lines:

```
listen = ssl://7243
ssl_require_client_cert = enabled
ssl_server_identity = /opt/tibco/ems/6.0/samples/certs/tibcoserver.pem
ssl_server_key = /opt/tibco/ems/6.0/samples/certs/tibcoserver.key.pem
ssl_password = tibcoserver
ssl_server_trusted = /opt/tibco/ems/6.0/samples/certs/tibcoclient.pem
```

b. Update factories configuration: In `factories.conf`, configure the following factory (or add a factory with a new name):

```
[SSLQueueConnectionFactory]
type = queue
url = ssl://tibcohost.genesyslab.com:7243
ssl_identity = //opt/tibco/ems/6.0/samples/certs/tibcoclient.p12
ssl_trusted = //opt/tibco/ems/6.0/samples/certs/tibcoserver.pem
```

c. Use the TIBCO EMS Administration tool to create a new user:

```
tcp://localhost:7222> create user genesys password=tibcoclient
```

Note: The user password must be exactly the same as the PEM password for the example client certificate. Note the following excerpt from the TIBCO EMS User's Guide (Chapter 18): “Because connection factories do not contain the `ssl_password` (for security reasons), the EMS server uses the password that is provided in the `create connection` call for user authentication. If the `create connection` password is different from the `ssl_password`, the connection creation will fail.”

d. Restart TIBCO with the new configuration:

```
tibemsd -config "{Path to tibemsd_ssl.conf}/tibemsd_ssl.conf"
```

3. Configure Interaction Server options: Add the following TIBCO EMS jars to the `-Djava.class.path` option in the `jvm-options` section: `jms.jar`, `tibjms.jar`, `tibcrypt.jar`, `slf4j-simple-1.4.2.jar`, `slf4j-api-1.4.2.jar`.

4. Configure the JMS Capture Point:

a. In the `settings` section, set options as follows:

- `jms-connection-factory-lookup-name=SSLQueueConnectionFactory`

  This option points to a new connection factory.
- `jms-provider-url=ssl://tibcohost.genesyslab.com:7243`

  The provider URL now points to a secure port.
- `password=tibcoclient`
- `username=genesys`

  The username and password correspond to those of the newly created TIBCO client.

b. In the `jms-additional-context-attributes` section, set options as follows:

- `com.tibco.tibjms.naming.security_protocol=ssl`
- `com.tibco.tibjms.naming.ssl_enable_verify_host=true`
- `com.tibco.tibjms.naming.ssl_enable_verify_hostname=false`
- `com.tibco.tibjms.naming.ssl_identity={Local path to certificates}\tibcoclient.p12`
- `com.tibco.tibjms.naming.ssl_password=tibcoclient`
- `com.tibco.tibjms.naming.ssl_trusted_certs={Local path to certificates}\tibcoserver.pem`

- `java.naming.security.credentials=tibcoclient`
- `java.naming.security.principal=genesys`

The following two options can be added for debugging:

- `com.tibco.tibjms.naming.ssl_debug_trace=true`
- `com.tibco.tibjms.naming.ssl_trace=true`

End of procedure

Database Capture Point

The integrated Database Capture Point provides the ability to capture interactions from databases, and also provides compatibility with the iWD Database Capture Adapter. The Database Capture Point provides the ability to create interactions based on a database query, and to update database records to propagate changes in interaction states or parameters.

The integrated Database Capture Point picks up updates for the interactions in the source database and applies these updates to the corresponding interactions. All relevant queries for selection and updates in the source database are configurable in the integrated Database Capture Point application settings.

The set of possible configurable queries in Database Capture Point includes the queries of iWD 8.0 Database Capture Adapter and introduces a number of new queries, corresponding to existing interaction events. iWD compatibility is achieved by configuring corresponding iWD-related queries and parameters.

This section describes the following:

- “Configurable Queries” on page 297
- “Query Language” on page 305
- “Error Handling” on page 308

Configurable Queries

The queries are written in SQL language, observing the semantics of the DBMS that you are using. When performing *select* queries, the columns should be named as standard interaction properties or user data keys (both case-sensitive). In *update* queries (using the interaction parameters or special keys) the interaction parameters and user data are case-sensitive as well.

When using parameters (such as "`externalid=<external id of the interaction>`"), write a question mark followed by the name of the parameter known to the interaction server in single quotes (such as

" externalid=?'ExternalId' "). The question mark must be followed by the parameter name in single quotes, with no spaces.

Note: Do **not** use the curly apostrophe/single quote symbol ('); use the straight single quote (').

Inbound Queries

Table 37 lists the three inbound queries. All three are required.

Table 37: Inbound Queries

Query parameter	Description
captureQuerySql	The database query that returns the result set in which each row will be captured as an interaction by Interaction Server. If a column name does not belong to the predefined interaction properties' names, its value will be attached to the user data of the interaction with a key corresponding to the column name. For example: `select externalid "ExternalId", stamp "ReceivedAt", tenantid "TenantId", priority "Priority", status "Status"  from inbound where status='new'`
capturedUpdateSql	The database query that updates the corresponding database record to reflect that certain data has been successfully captured as an interaction by Interaction Server. Besides the values available from the corresponding capture query, the 'InteractionId' value is available to this query if it has not been provided in the result set of the corresponding capture query. For example: `update inbound set interactionid=?'InteractionId', status='submitted' where externalid=?'ExternalId'`
errorUpdateSql	The database query that updates the corresponding database record to reflect that the associated interaction has not been captured by Interaction Server. Besides the values available from the corresponding capture query, additional values 'ErrorCode' (integer) and 'ErrorDescription' (string up to 256 characters) are available to this query. For example: `update inbound set status='error', errorcode=?'ErrorCode', errordescr=?'ErrorDescription' where externalid=?'ExternalId'`

Notification Queries

Notification queries will be invoked upon the corresponding reporting events being generated. All notification queries are optional. If no query exists in the configuration, then no action is performed when the corresponding event occurs. Notification queries are queued (up to a `batch-size`, or up to `storing-timeout`, both configurable options) and executed in one transaction. Table 38 lists the notification queries.

Table 38: Notification Queries

Query parameter	Description	Reporting event (and condition)
`assignedUpdateSql`	The database query that updates the database to reflect that the associated interaction has been assigned to an agent. Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'assigned', assignedto=?'AssignedTo', assignedat=?'AssignedAt' where interactionid=?'InteractionId'`	`EventPartyAdded` (party not `'strategy'`)
`completedUpdateSql`	The database query that updates the database to reflect that the associated interaction has been placed into a queue belonging to the `CompleteQueues` set specified in the `iwd-parameters` section of the configuration options (if the section and property are configured). Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'completed' where interactionid=?'InteractionId'`	`EventPlacedInQueue` (queue in `CompleteQueues`)

Table 38: Notification Queries (Continued)

Query parameter	Description	Reporting event (and condition)
canceledUpdateSql	The database query that updates the database to reflect that the associated interaction has been placed into a queue belonging to the CancelQueues set specified in iwd-parameters section of the configuration options (if the section and property are configured). Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'canceled' where interactionid=?'InteractionId'`	EventPlacedInQueue (queue in CancelQueues)
heldUpdateSql	The database query that updates the corresponding database record to reflect that the associated interaction has been put on hold. Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'held' where interactionid=?'InteractionId'`	EventHeld
queuedUpdateSql	The database query that updates the database to reflect that the associated interaction has been placed into any queue **not** belonging to the sets of iWD queues specified in the iwd-parameters section of the configuration options (such as CancelQueues, CompleteQueues, and so on). Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'queued', queue=?'Queue' where interactionid=?'InteractionId'`	EventPlacedInQueue (queue not in any iWD queues)

Table 38: Notification Queries (Continued)

Query parameter	Description	Reporting event (and condition)
errorHeldUpdateSql	The database query that updates the database to reflect that the associated interaction has been placed into a queue belonging to the ErrorHeldQueues set specified in the iwd-parameters section of the configuration options (if the section and property are configured). Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'errorheld' where interactionid=?'InteractionId'`	EventPlacedInQueue (queue in ErrorHeldQueues)
rejectedUpdateSql	The database query that updates the database to reflect that the associated interaction has been placed into a queue belonging to the RejectQueues set specified in the iwd-parameters section of the configuration options (if the section and property are configured). Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'rejected' where interactionid=?'InteractionId'`	EventPlacedInQueue (queue in RejectQueues)
restartedUpdateSql	The database query that updates the database to reflect that the associated interaction has been placed in the RestartQueues set specified in the iwd-parameters section of the settings (if the section and property are configured). Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'restarted' where interactionid=?'InteractionId'`	EventPlacedInQueue (queue in RestartQueues)

Table 38: Notification Queries (Continued)

Query parameter	Description	Reporting event (and condition)
stoppedUpdateSql	The database query that updates the database to reflect that the associated interaction has been stopped in Interaction Server. Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'stopped' where interactionid=?'InteractionId'`	EventProcessingStopped
routeRequestedUpdateSql	The query statement that updates the database to reflect that the associated interaction has been sent to a router. Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'routing' where interactionid=?'InteractionId'`	EventPartyAdded (party is 'strategy')

Table 38: Notification Queries (Continued)

Query parameter	Description	Reporting event (and condition)
`updatedUpdateSql`	The query statement that updates the database to reflect that the associated interaction has been updated in Interaction Server by some other entity (not this Database Capture Point). Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set priority=?'Priority' where interactionid=?'InteractionId'`	`EventPropertiesChanged`
`resumedUpdateSql`	The query statement that updates the corresponding database record to reflect that the associated interaction has been resumed from a hold. Values of all interaction properties and user data (except binary and kv-lists) of the corresponding interaction are available to this query. For example: `update inbound set status = 'resumed' where interactionid=?'InteractionId'`	`EventResumed`

Source Update Queries

Table 39 lists the source update queries. If `sourceUpdateQuerySql` is specified, the other two queries are required to be configured and correct. If no query exists in the configuration, then no action is performed when the corresponding event occurs.

Table 39: Source Update Queries

Query parameter	Description
`sourceUpdateQuerySql`	The database query that fetches a set of rows, where each row represents an update request. Each such update request may contain one or more columns that represent interaction properties. The name of the column represents the name of the interaction property and the value is the new value of that interaction property. Each row of the result set must contain either `'InteractionId'` or `'ExternalId'`. If both `'InteractionId'` and `'ExternalId'` are contained in a row, the value of `'InteractionId'` will be used to access the interaction, and the value of ‘`ExternalId'` will be treated as one of the interaction properties to update. For example: `select interactionid "InteractionId", stamp "SomeTime",  priority "Priority"  from updates where status='new'`
`sourceUpdatedUpdateSql`	The database update (or delete) query that will execute against a special table in the source database to mark a particular update as having been processed. For example: `update updates set status='applied' where interactionid=?'InteractionId'`
`sourceErrorUpdateSql`	This update is executed when there is an error executing an update request (the one that is fetched by `sourceUpdateQuerySql`). Besides the values available from the corresponding capture query, additional values `'ErrorCode'` (integer) and `'ErrorDescription'` (string up to 256 characters) are available to this query. For example: `update updates set status='error', errorcode=?'ErrorCode', errordescr=?'ErrorDescription' where interactionid=?'InteractionId'`

Additional Queries

Table 40 contains an additional query that the Database Capture Point supports.

Table 40: Additional Queries

Query parameter	Description
startupQuerySql	This optional query runs once, upon the Database Capture Point point establishing a connection to the database. It cannot take any parameters from Interaction Server.

Query Language

Reference Table 41 to set and get interaction properties and their data.

Table 41: Setting and Getting Interaction Properties and their Data

Interaction property	Can be provided in submit	Can be updated by source update query	Input data type	Output data type
InteractionId	Y	N	Varchar	Varchar(256)
ExternalId	Y	Y	Varchar	Varchar(256)
ParentID	Y	Y	Varchar	Varchar(256)
MediaType	Y	N	Varchar	Varchar(256)
InteractionType	Y	N	Varchar	Varchar(256)
InteractionSubtype	Y	N	Varchar	Varchar(256)
TenantId	Y	N	Varchar or Int	Int
Queue	Y	Y	Varchar	Varchar(256)
Workbin	Y	N	Varchar	Varchar(256)
WorkbinAgentId	Y	N	Varchar	Varchar(256)
WorkbinPlaceId	Y	N	Varchar	Varchar(256)
WorkbinAgentGroupId	Y	N	Varchar	Varchar(256)
WorkbinPlaceGroupId	Y	N	Varchar	Varchar(256)
IsOnline	Y	N	Varchar or Int	Int

Table 41: Setting and Getting Interaction Properties and their Data (Continued)

Interaction property	Can be provided in submit	Can be updated by source update query	Input data type	Output data type
ReceivedAt	Y	N	Datetime	Varchar(256)
SubmittedBy	N	N	Not applicable	Varchar(256)
State	N	N	Not applicable	Int
IsLocked	N	N	Not applicable	Int
SubmittedAt	N	N	Not applicable	Varchar(256)
DeliveredAt	N	N	Not applicable	Varchar(256)
SubmittedTo RouterAt	N	N	Not applicable	Varchar(256)
PlacedInQueueAt	N	N	Not applicable	Varchar(256)
MovedToQueueAt	N	N	Not applicable	Varchar(256)
AbandonedAt	N	N	Not applicable	Varchar(256)
IsHeld	N	N	Not applicable	Int
HeldAt	N	N	Not applicable	Varchar(256)
AssignedAt	N	N	Not applicable	Varchar(256)
AssignedTo	N	N	Not applicable	Varchar(256)
CompletedAt	N	N	Not applicable	Varchar(256)

Reference Table 42 for special column names or data keys.

Table 42: Setting and Getting Interaction Properties and their Data

Special column names or data keys	Can be provided in submit	Can be updated by source update query	Input data type	Output data type
Hold	Y	Y (but should not)	Int or Varchar	
ErrorCode	Y (but should not)	Y (but should not)	Not applicable	Int
ErrorDescription	Y (but should not)	Y (but should not)	Not applicable	Varchar(256)

Table 42: Setting and Getting Interaction Properties and their Data (Continued)

Special column names or data keys	Can be provided in submit	Can be updated by source update query	Input data type	Output data type
EventTime	N	N	Not applicable	Varchar(256) available to notification queries only
ActorType	N	N	Not applicable	Int
ActorMediaServerId	N	N	Not applicable	Varchar(256)
ActorStrategyId	N	N	Not applicable	Varchar(256)
ActorRouterId	N	N	Not applicable	Varchar(256)
ActorTenantId	N	N	Not applicable	Int
ActorPlaceId	N	N	Not applicable	Varchar(256)
ActorAgentId	N	N	Not applicable	Varchar(256)
ReasonSystemName	N	N	Not applicable	Varchar(256)
Reason Description	N	N	Not applicable	Varchar(256)
Operation	N	N	Not applicable	Int
ItxServerName	N	N	Not applicable	Varchar(256)
ItxServerDBID	N	N	Not applicable	Int
TenantsNames	N	N	Not applicable	Varchar(256)
TenantsDBIDs	N	N	Not applicable	Varchar(256)
ReportingEventSequenceNumber	N	N	Not applicable	Varchar(256), available to notification queries only

User Data All other column names not corresponding to interaction properties, special column names, or data keys are interpreted as user data keys.

Data Types The tables above refer to data types Datetime, Int, and Varchar. More formally, these data types are defined for each DBMS as shown in Table 43.

Table 43: Data Types Defined Per DBMS

DBMS	Int Types	Datetime Types	Varchar Types
Oracle	int, integer, smallint	date, timestamp	varchar2
DB2	numeric, decimal, smallint	timestamp	varchar, char
MSSQL	numeric, decimal, smallint, money, smallmoney	datetime, datetime2	varchar

Note: The values in columns of Datetime type are converted and attached to their corresponding keys as strings, therefore their values are available as Varchar type for output parameters. If they need to be inserted into actual datetime columns, either casting or conversion should be performed.

Error Handling

In situations where a capture or update query results in an error and cannot be executed, the values `ErrorCode` and `ErrorDescription` are provided to the corresponding error queries.

A returned `ErrorCode` can be equal to `0` for different `ErrorDescriptions.` This means that the error is not a protocol error and might not have a separate error code.

If an inbound (or source update) query results in an ODBC exception, the exception is reported in the logs, and the inbound (or source update) cycle pauses for the duration of the `inbound-exception-sleep-interval` parameter (for inbound queries) or the `updates-exception-sleep-interval` parameter (for source update queries). Both of these parameters are configuration options for the Database Capture Point.

Web Service Capture Point

The web service Capture Point provides a web service interface for interaction-related requests such as `submit, stop, update, hold, resume,` and `get info,` as well as for ping requests.

This section describes the following:

- Common Aspects, page 309
- Generating a Client, page 310
- Web Service Capture Point Client Over Secure HTTP, page 320

- Messaging in Native Mode, page 326
- Messaging in iWD Compatible Mode, page 330

Common Aspects

Service URL

The Web Service Capture Point service URL can be easily obtained from the Interaction Server startup log. Look for the following message and simply copy the URL:

```
11:17:58.814 Trc 23323 Capture point 'WSCapturePoint' will set
endpoint: '
http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_8
12_zoo/WebServiceCapturePoint'
```

You can also construct the URL using the template provided by the Web Service Capture Point application option `soap-endpoint`, whose default value is:

```
<Protocol>://<ServerName>:<ServerPort>/Genesys/Interaction/<CapturePoin
tName>/WebServiceCapturePoint
```

where

- Protocol is HTTP or HTTPS, as specified in the `protocol` option.
- Server Name is either specified in the `soap-hostname` option or is equal to the name of Interaction Server's host.
- Port is the port of the Web Service Capture Point Application object.
- CapturePointName is the name of the Application object.

This template can be changed, but generally it contains the four parts just listed. Note that none of the parameters are mandatory and the entire endpoint can be simply specified in its final form, which may be preferable in some cases.

WSDL URL

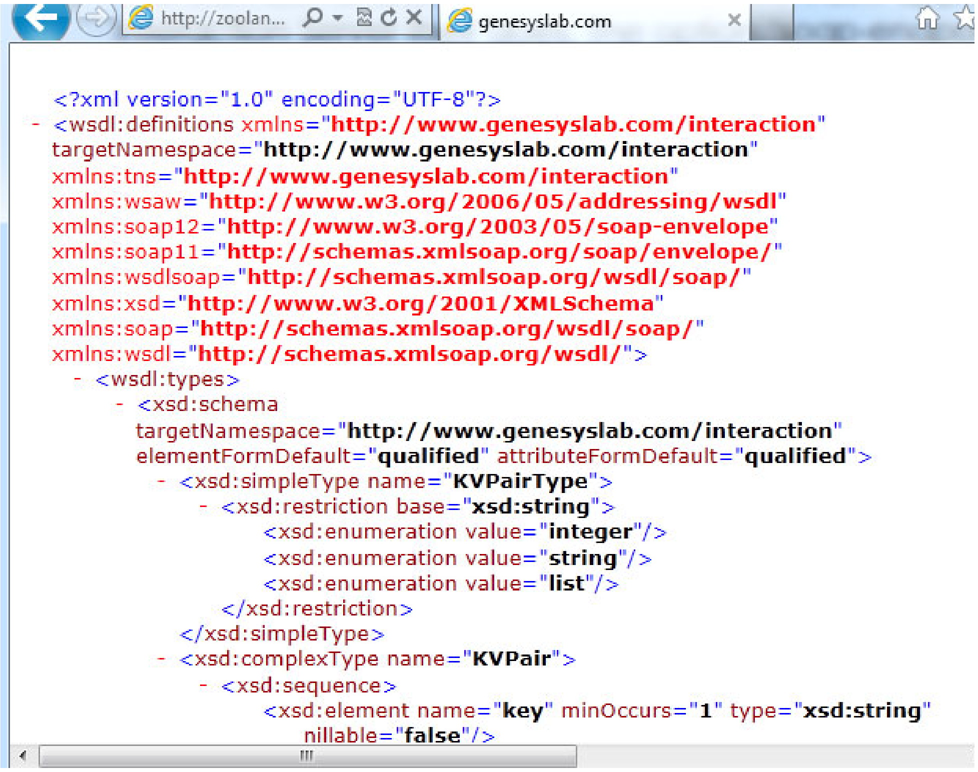
The WSDL URL is the service URL with `?wsdl` appended; for example:

```
http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_8
12_zoo/WebServiceCapturePoint?wsdl
```

Checking Connectivity and Inspecting WSDL

Once you get the service URL, you can use it in different tools to generate a Web Service client.

To check that the service is up and running and to inspect the service WSDL, first ensure that Interaction Server is running, then navigate to the WSDL URL using any web browser. Figure 98 shows WSDL in Internet Explorer.



```
  <?xml version="1.0" encoding="UTF-8"?>
- <wsdl:definitions xmlns="http://www.genesyslab.com/interaction"
  targetNamespace="http://www.genesyslab.com/interaction"
  xmlns:tns="http://www.genesyslab.com/interaction"
  xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
  xmlns:soap12="http://www.w3.org/2003/05/soap-envelope"
  xmlns:soap11="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  - <wsdl:types>
    - <xsd:schema
        targetNamespace="http://www.genesyslab.com/interaction"
        elementFormDefault="qualified" attributeFormDefault="qualified">
      - <xsd:simpleType name="KVPairType">
        - <xsd:restriction base="xsd:string">
            <xsd:enumeration value="integer"/>
            <xsd:enumeration value="string"/>
            <xsd:enumeration value="list"/>
          </xsd:restriction>
        </xsd:simpleType>
      - <xsd:complexType name="KVPair">
        - <xsd:sequence>
            <xsd:element name="key" minOccurs="1" type="xsd:string"
              nillable="false"/>
```

Figure 98: WSDL

This confirms that you have access to the WSCP service. You can inspect the WSDL or save it to a file to later use. Saving the WSDL is not required since most tools can simply access the WSDL URL directly, as long as Interaction Server is running.

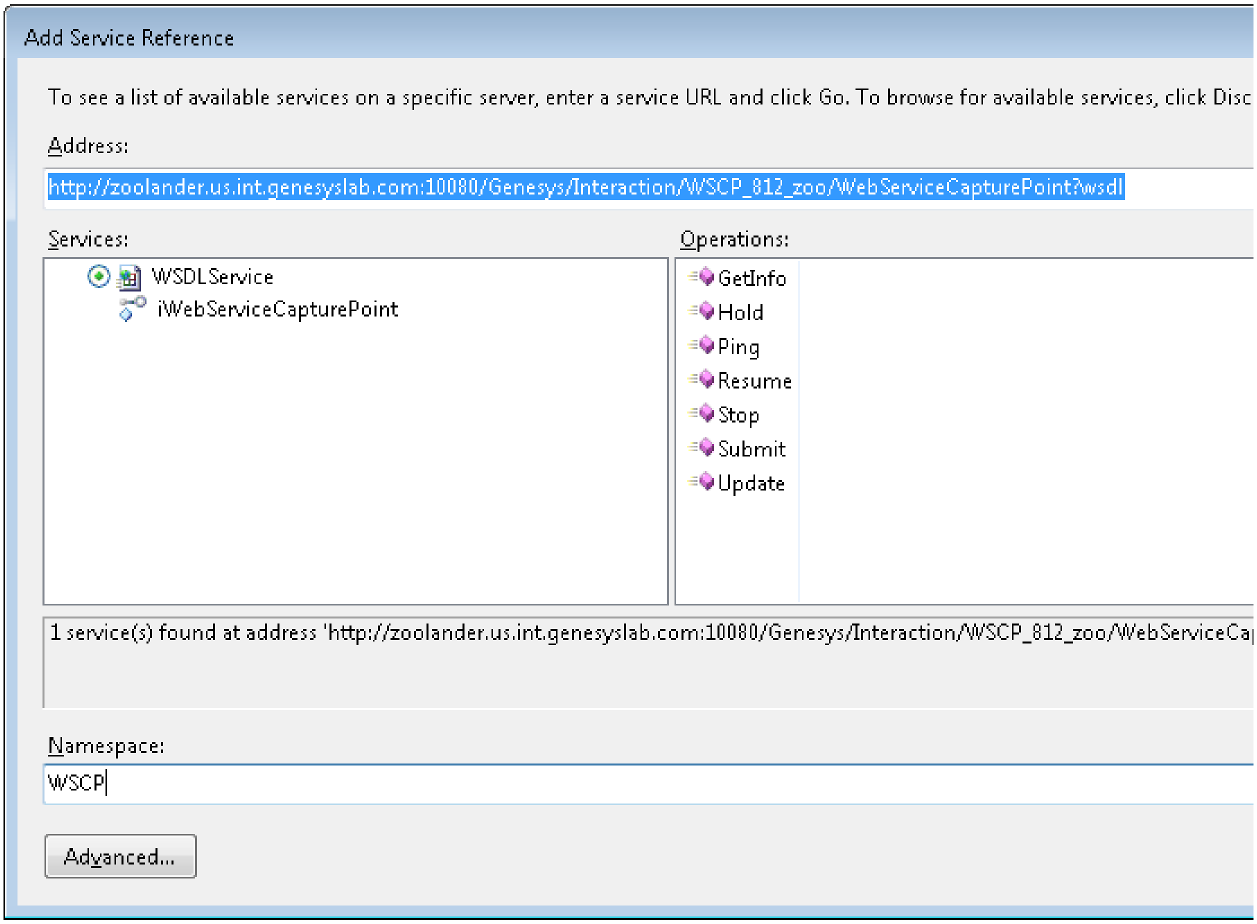
Generating a Client

The following tools were used to generate WSCP clients for this document:

- Visual Studio 2010
- JAX-WS 2.2
- Apache CXF
- Apache Axis2

.NET client

1. Open Visual Studio 2010 and create a C# Win32 console application.
2. In Solution Explorer right-click `References` and choose `Add Service Reference`. The dialog box of the same name appears, shown in Figure 99.


Add Service Reference

To see a list of available services on a specific server, enter a service URL and click Go. To browse for available services, click Disc

Address:
http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebServiceCapturePoint?wsdl

Services:
- WSDLService
  - iWebServiceCapturePoint

Operations:
- GetInfo
- Hold
- Ping
- Resume
- Stop
- Submit
- Update

1 service(s) found at address 'http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebServiceCa

Namespace:
WSCP

Figure 99: Add Service Reference

3. Enter the WSDL URL of the Web Service Capture Point
4. Enter the service namespace (for example, 'WSCP'):
5. Click Go

   Provided Interaction Server is running and the WSDL URL is specified correctly, `WebServiceCapturePoint` should appear in the `Services` list.
6. Click `OK` to generate the service client.
7. To test the service, open the `Program.cs` file and insert the following code in the `main` method:

```
WSCP.iWebServiceCapturePointClient client = new WSCP.iWebServiceCapturePointClient();

// This is optional step to reconfigure the client to use different endpoint.
// It's usually done using configuration setting for the application
//client.Endpoint.Address = new System.ServiceModel.EndpointAddress(
//    "http://localhost/Genesys/Interaction/MyCP/WebServiceCapturePoint");

// Create a key-value list of extensions and specify the signature,
// so we can recognize the request in Interaction Server log
var extension = new WSCP.KVList();
extension.Add(new WSCP.KVPair() { key = "signature",
value = new WSCP.KVPairValue() { ValueString = ".Net WSCP test client" } });

// We expect ping info back in Ping response
WSCP.KVList userdata = null;
WSCP.KVList pinginfo = null;

try
{
    // Ping the server and get some statistics back
    client.Ping(out userdata, out pinginfo, ref extension);

    Console.Out.WriteLine(trace_list(pinginfo));
}
catch (FaultException<FaultMessage> ex)
{
    // process WSCP specific error code
  Console.Out.WriteLine("Error {0}: {1}",
ex.Detail.ErrorCode, ex.Detail.ErrorDescription);
}
catch (Exception ex)
{
    Console.Out.WriteLine(ex.ToString());
}
```

8. Add the method `trace_list` to your program to output the server response:

```
static string trace_list(WSCP.KVList list, string indent = "")
{
    StringBuilder result = new StringBuilder();

    list.ForEach((item) =>
    {
        result.Append(indent);
        result.Append(item.key);

        if(null != item.value.ValueString)
        {
            result.Append(" [string] = ");
            result.Append(item.value.ValueString);
            result.Append('\n');
        }
        else if (null != item.value.ValueList)
        {
            result.Append(" [list] = \n");
            result.Append(trace_list(item.value.ValueList, indent + "    "));
        }
        else
        {
            result.Append(" [int] = ");
            result.Append(item.value.ValueInt.ToString());
            result.Append('\n');
        }
    });

    return result.ToString();
}
```

This simple test prints Interaction Server statistics into a console window. You can then discover the service methods using autocompletion and the object browser.

JAX-WS

To generate a WSCP service proxy for Java, use `wsimport` utility, which is included in JDK:

```
wsimport -d <output directory> <WSDL URL>
```

For example:

```
wsimport -d c:\Temp\MyJSClient
http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WS
CP_812_zoo/WebServiceCapturePoint?wsdl
```

The tool generates a set of files for the proxy.

Create a simple Java console application to ping the service:

```
import java.net.URL;
import javax.xml.namespace.QName;
import javax.xml.ws.Holder;
import com.genesyslab.interaction.*;
public class Test {

   public static void main(String[] args) throws Exception {

      WebServiceCapturePoint service = new WebServiceCapturePoint(new
    URL("http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebServiceCaptu
    rePoint/?WSDL"),new QName("http://www.genesyslab.com/interaction", "WebServiceCapturePoint"));

      IWebServiceCapturePoint cp = service.getIWebServiceCapturePointHttpBinding();

      KVPairValue val = new KVPairValue();

      val.setValueString("I am coming from JAXWS client");

      KVPair pair = new KVPair();

      pair.setKey("Source");
      pair.setValue(val);
```

```
        KVList extList = new KVList();

        extList.getKvitem().add(pair);

        Holder<KVList> extension = new Holder<KVList>(extList);

        Holder<String> eventTime = new Holder<String>();
        Holder<KVList> userData = new Holder<KVList>();
        Holder<KVList> pingInfo = new Holder<KVList>();

        cp.ping(extension, eventTime, userData, pingInfo);

        System.out.println("Ping response time:" + eventTime.value);
        printKVList("PingInfo", pingInfo.value);
        printKVList("UserData", userData.value);
        printKVList("Extension", extension.value);

    }

    public static void printKVList(String name, KVList kvList) {
        printKVList(name, kvList, "");
    }

    private static void printKVList(String name, KVList kvList, String shift) {
        if (null == kvList) {
            System.out.println(shift + name + "[KVList]=null");
        } else {
            System.out.println(shift + name + "[KVList]=");

            for (KVPair pair : kvList.getKvitem()) {
                KVPairValue value = pair.getValue();

                if (value.getValueInt() != null) {
                    System.out.println(shift + "\t" + pair.getKey() + "[int]="
                            + value.getValueInt());
                } else if (null != value.getValueList()) {
                    printKVList(pair.getKey(), value.getValueList(), shift
                            + "\t");
                } else {
                    System.out.println(shift + "\t" + pair.getKey()
                            + "[string]=" + value.getValueString());
                }
            }
        }
    }
}
```

Apache CXF

Java Client

To generate WSCP service proxy for Java use the `wsdl2java` tool:

```
wsdl2java -frontend jaxws21 -d <output directory> <WSDL URL>
```

For example:

```
wsdl2java -d c:\Temp\MyJSClient
http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WS
CP_812_zoo/WebServiceCapturePoint?wsdl
```

The tool generates a set of files for the proxy.

Create a simple Java console application to ping the service:

```
import java.net.URL;
import javax.xml.ws.Holder;
import com.genesyslab.interaction.*;

public class Test {
   public static void main(String[] args) throws Exception {

      WebServiceCapturePoint service = new WebServiceCapturePoint(new
   URL("http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebServiceCaptu
   rePoint/?WSDL"));

      IWebServiceCapturePoint cp = service.getIWebServiceCapturePointHttpBinding();

      KVPairValue val = new KVPairValue();

      val.setValueString("I am coming from CXF client");

      KVPair pair = new KVPair();

      pair.setKey("Source");
      pair.setValue(val);

      KVList extList = new KVList();

      extList.getKvitem().add(pair);

      Holder<KVList> extension = new Holder<KVList>(extList);

      Holder<String> eventTime = new Holder<String>();
      Holder<KVList> userData = new Holder<KVList>();
      Holder<KVList> pingInfo = new Holder<KVList>();

      cp.ping(extension, eventTime, userData, pingInfo);

      System.out.println("Ping response time:" + eventTime.value);
      printKVList("PingInfo", pingInfo.value);
      printKVList("UserData", userData.value);
      printKVList("Extension", extension.value);

   }
   public static void printKVList(String name, KVList kvList) {
      printKVList(name, kvList, "");
   }

   private static void printKVList(String name, KVList kvList, String shift) {
      if (null == kvList) {
         System.out.println(shift + name + "[KVList]=null");
      } else {
         System.out.println(shift + name + "[KVList]=");

         for (KVPair pair : kvList.getKvitem()) {
            KVPairValue value = pair.getValue();

            if (value.getValueInt() != null) {
               System.out.println(shift + "\t" + pair.getKey() + "[int]="
                     + value.getValueInt());
            } else if (null != value.getValueList()) {
               printKVList(pair.getKey(), value.getValueList(), shift
                     + "\t");
            } else {
               System.out.println(shift + "\t" + pair.getKey()
                     + "[string]=" + value.getValueString());
            }
         }
      }
   }
}
```

Javascript Client

You can generate a Javascript client using the Apache CXF `wsdl2js` tool:
`wsdl2js -d <output directory> <WSDL URL>`

For example:

```
wsdl2js -d c:\Temp\MyJSClient
http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WS
CP_812_zoo/WebServiceCapturePoint?wsdl
```

The tool generates a single file that contains a proxy that can send requests to the service and receive replies asynchronously. You must also include the `cxf-util.js` file, which is part of Apache CXF.

The sample below does not require anything beyond HTML and Javascript (`wscp.js` is the file generated by `wsdl2js`):

```
<html>
<head>
```

```
<script type="text/javascript"
   src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>
<script type="text/javascript" src="cxf-utils.js"></script>
<script type="text/javascript" src="wscp.js"></script>

<script language="JavaScript" type="text/javascript">

var gCounter = 0;

function print_list(list, indent)
{
   var r = '';

   for(var i=0; i < list._kvitem.length; ++i)
   {
      r += indent;

      var pair = list._kvitem[i];
      r += pair._key;
      if( pair._value._ValueString )
      {
         r += " [str] = '";
         r += pair._value._ValueString;
         r += "'";
         r += "<br>";
      }
      else if( pair._value._ValueInt )
      {
         r += " [int] = ";
         r += pair._value._ValueInt;
         r += "<br>";
      }
      else
      {
         r += " [list] = ";
         if( pair._value._ValueList )
         {
            r += "<br>";
            r += print_list(pair._value._ValueList, indent + "....");
         }
         else
         {
            r += " EMPTY";
            r += "<br>";
         }
      }
   }

   return r;
}
```

```
function test()
{
   var svc = new _iWebServiceCapturePoint();
   svc.url =
   'http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebServiceCapturePo
   int';

   var extension = new _KVList();
   var items = new Array();

   var signature = new _KVPair();
   signature.setKey("signature");
   var signature_value = new _KVPairValue();
   signature_value.setValueString("JavaScript client generated with CXF");
   signature.setValue(signature_value);

   var counter = new _KVPair();
   counter.setKey("Request count");
   var counter_value = new _KVPairValue();
   counter_value.setValueInt(++gCounter);
   counter.setValue(counter_value);
   items.push(signature);
   items.push(counter);
   extension.setKvitem(items);

   svc.Ping(
      function(response)
      {
         var r = "Response timestamp: " + response.getEventTime() + ", ping info:<br>";

         r += print_list(response.getPingInfo(), "");

         $("#response_text").html(r);
      },
      function(status, statusText)
      {
         $("#response_text").html("Response failed: (" + status + ") " + statusText);
      },
      extension
   );
}

</script>
</head>
<body>
<p>Press the button to call the service...</p>
<p><input value="Ping the service" type="button" onclick="test()"/></p>
<p><div id="response_text"></div></p>
</body>
</html>
```

Apache Axis2/Java

Generate a WSCP service proxy using the Axis2 plug-in. The following sample demonstrates how to send a Ping request and to print out the contents of the PingResponse.

```
import com.genesyslab.www.interaction.WebServiceCapturePointStub.*;
import com.genesyslab.www.interaction.*;

public class TestWSCP {
   public static void main(String[] args) {
      try {
         WebServiceCapturePointStub serviceStub = new WebServiceCapturePointStub(

   "http://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebServiceCapturePo
   int");
         Ping ping = new Ping();

         KVList ext = new KVList();

         // create a string kv pair
         KVPair strPair = new KVPair();
         KVPairValue value = new KVPairValue();
         strPair.setKey("Source");
         value.setValueString("I am coming from axis2 client");
         strPair.setValue(value);

         // add this pair to the extension
         ext.addKvitem(strPair);

         // set extension
         ping.setExtension(ext);

         PingResponse response = serviceStub.Ping(ping);

         System.out.println("Ping response time:" + response.getEventTime());
         printKVList("PingInfo", response.getPingInfo());
         printKVList("UserData", response.getUserData());
         printKVList("Extension", response.getExtension());
      } catch (Exception e) {
         e.printStackTrace();
      }
   }

   public static void printKVList(String name, KVList kvList) {
      printKVList(name, kvList, "");
   }

   private static void printKVList(String name, KVList kvList, String shift) {
      if (null == kvList) {
            System.out.println(shift + name + "[KVList]=null");
        } else {
            System.out.println(shift + name + "[KVList]=");

            for (KVPair pair : kvList.getKvitem()) {
                KVPairValue value = pair.getValue();

                if (null != value.getValueList()) {
                    printKVList(pair.getKey(), value.getValueList(), shift
                            + "\t");
                } else if (null != value.getValueString()) {
                    System.out.println(shift + "\t" + pair.getKey()
                            + "[string]=" + value.getValueString());
                } else {
                System.out.println(shift + "\t" + pair.getKey() + "[int]="
                            + value.getValueInt());
}
            }
        }
    }
}
```

Web Service Capture Point Client Over Secure HTTP

This section provides an example of configuring a Web Service Capture Point, generating and importing certificates, and using .NET and Java clients over Secure HTTP. OpenSSL version 1.0.0g or better is assumed to be installed. This example configuration assumes the presence of a server host (`zoolander.us.int.genesyslab.com` in the example) and a client host (`clienthost.us.int.genesyslab.com` in the example). The server host has an Interaction Server with a Web Service Capture Point named `WSCP_812_zoo` connected to it.

Server Certificate

The server certificate is used for server authentication (by the client) and ensures that server can be trusted. The Web Service Capture Point requires a server certificate to support SSL.

Generate a server certificate, along with a private key:

```
openssl req -x509 -days 365 -subj "/C=US/ST=California/L=Daly 
City/CN=zoolander.us.int.genesyslab.com" -newkey rsa:2048 -keyout 
wscpserver.pem -out wscpserver.pem
```

The output file `wscpserver.pem` contains a private key along with a certificate. During the private key generation, the user is prompted for a password, which will be required later. The user will be asked to come up with a *PEM pass phrase,* which will be later used in the WSCP configuration, along with the generated `.pem` file.

The server certificate can also be a self-signed certificate or a certificate signed by any Certificate Authority (CA). The certificate generated for the server must be imported or copied into the client's trusted certificates store. Use the procedure and tools appropriate for your platform.

Note: The private key should **never** be copied or given to anyone. It should be password protected (encoded) and should be accessible to the server only. The client is given only the certificate (public key) to put into the trusted certificates store.

The following is a procedure for putting server certificates into client's trusted certificates store for Windows, using the `openssl` utility.

Procedure:
Put server certificate in client's store

Start of procedure

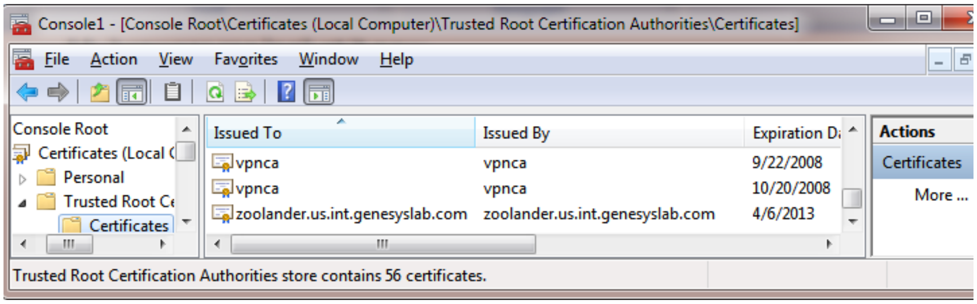
1. Convert the generated certificate to 'der' format:

```
openssl x509 -outform der -in wscpserver.pem -out wscpserver.cer
```

   The output file `wscpserver.cer` contains a public server certificate, which will be added to the trusted certificates of the client using the Web Service Capture Point.

2. Import the generated .CER server certificate into the trusted certificates store (for browser and .NET client):
   a. Start Microsoft Management Console.
   b. On the `File` menu, select `Add or Remove Snap-ins.`
   c. Choose `Certificates,` then click `Add.`
   d. When prompted, choose `Computer account` and `Local Computer.`
   e. Click `Finish,` then `OK.`
   f. Right-click `Certificates > Trusted Root Certification Authorities > Certificates.`
   g. Choose `All tasks > Import·`
   h. Choose `wscpserver.cer` for import.

The certificate is added to the trusted certificates, as shown in Figure 100.



Issued To	Issued By	Expiration Da
vpnca	vpnca	9/22/2008
vpnca	vpnca	10/20/2008
zoolander.us.int.genesyslab.com	zoolander.us.int.genesyslab.com	4/6/2013

Trusted Root Certification Authorities store contains 56 certificates.

Figure 100: Certificate Added to Trusted Certificates

3. For Java clients only, import the generated .CER server certificate into a Java keystore. Assuming that a standard JDK is present on the client host, add the server certificate to a trust store on the client host:

```
keytool -import -keystore truststore.jks -file wscpserver.cer -alias
wscpserver
```

End of procedure

Client Certificate for Browser and .NET Client

A client certificate is required for mutual SSL authentication. If the Web Service Capture Point is configured for server authentication only, the client certificate is not required.

The following procedure provides an example of generating the certificate on Windows using the openssl utility.

Procedure:
Deploy a client certificate for a .NET Client

Start of procedure

1. Generate a client certificate:

```
openssl req -x509 -days 365 -subj "/C=US/ST=California/L=Daly
City/CN=clienthost.us.int.genesyslab.com" -newkey rsa:2048 -keyout
wscpclientkey.pem -out wscpclient.pem
```

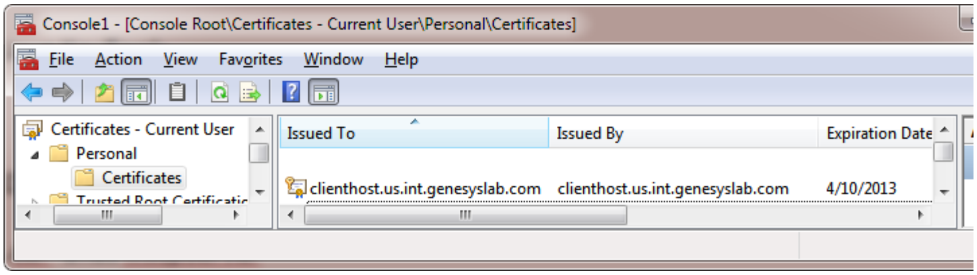
The output certificate without a private key, `wscpclient.pem`, will be given to the WSCP so that it can authenticate the client. The user will be asked to provide a PEM pass phrase, which is later used to export the certificate, along with the key, `wscpclientkey.pem`, to another format.

2. Export the generated client certificate and the private key into PFX format:

```
openssl pkcs12 -export -out wscpclient.pfx -inkey wscpclientkey.pem
-in wscpclient.pem
```

When exporting to PFX format, the user will be asked to provide a pass phrase (the same as the PEM pass phrase referred to in Step 1) and to set an Export Password, which will be used later.

3. Import the PFX certificate to Personal Certificates for Current User: Import the `wscpclient.pfx` with Microsoft Management Console and follow the same procedure as used to import the sever certificate (Step 2 on page 321), except that you must choose `My user account` rather than `Computer account` in Step d. The result will appear as in Figure 101.


Console1 - [Console Root\Certificates - Current User\Personal\Certificates]
File
Action
View
Favorites
Window
Help
Certificates - Current User
Personal
Certificates
Trusted Root Certificatio

Issued To	Issued By	Expiration Date
clienthost.us.int.genesyslab.com	clienthost.us.int.genesyslab.com	4/10/2013

Figure 101: Importing PFX Certificate

4. Copy the client certificate to the server host: Copy the contents of `wscpclient.pem` into a file named `wscp_clients.pem` on the server host.

End of procedure

Client Certificate for Java Client

The following procedure provides an example of generating the certificate using keytool.

Procedure: Deploy a client certificate for a Java client

Start of procedure

1. Generate a Java client key:

```
keytool -genkey -alias javawscpclient -keyalg RSA -keystore
keystore.jks -keysize 2048
```

This command generates a client key and places it in the local keystore.

2. Export the generated certificate from the keystore:

```
keytool -export -alias javawscpclient -keystore keystore.jks -file
javawscpclient.cer
```

3. Convert the exported certificate to .PEM format:

```
openssl x509 -inform der -in javawscpclient.cer -out
javawscpclient.pem
```

4. Copy the Java client certificate: Append the contents of `javawscplient.pem` to the contents of `wscp_clients.pem` on the server host.

End of procedure

Web Service Capture Point Configuration

In a Web Service Capture Point application, named, for example, `WSCP_812_zoo,` set the following options:

- `server-key-file=<Path to wscpserver.pem>\wscpserver.pem`
- `password=<'PEM pass phrase' for wscpserver.pem>`
- `protocol=https`
- `require-client-authentication=true`
- `cacert-file=<Path to wscp_clients.pem>\wscp_clients.pem`

Do not change any other options. You must restart Interaction Server for these option values to take effect.

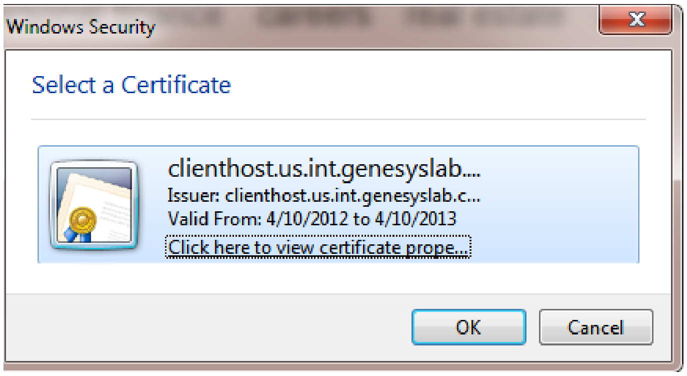
If client authentication is not required, set the option `require-client-authentication` to `false` and omit all procedures relevant to generation and manipulation of client certificates ("Client Certificate for Browser and .NET Client" on page 322 and "Client Certificate for Java Client" on page 323).

WSDL over HTTPS in the Browser

Assuming the client host has the server certificate in the trusted certificates, and the client certificate in personal certificate, you can request the WSDL from the client host by entering the URL that you obtained in "Service URL" on page 309:

```
https://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebServiceCapturePoint?wsdl
```

The browser then prompts the user to select a certificate, as shown in Figure 102:



Select a Certificate

clienthost.us.int.genesyslab....
Issuer: clienthost.us.int.genesyslab.c...
Valid From: 4/10/2012 to 4/10/2013
Click here to view certificate prope...

Figure 102: Select a Certificate

Select the imported certificate and click `OK`. The contents of the WSDL file should display in the browser.

Modifications for .NET Clients

Assuming that a .NET client has been previously configured without Secure HTTPS, and all the above procedures of generating, exporting, and importing certificates have been completed, the italicized changes are required in the existing .NET client's `app.config` to make it work over HTTPS:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <system.serviceModel>
      <behaviors>
        <endpointBehaviors>
          <behavior name="ProvideClientCertificate">
            <clientCredentials>
              <clientCertificate storeLocation="CurrentUser" x509FindType="FindByIssuerName"
    findValue="clienthost.us.int.genesyslab.com"/>
            </clientCredentials>
          </behavior>
        </endpointBehaviors>
      </behaviors>
        <bindings>
            <basicHttpBinding>
                <binding name="iWebServiceCapturePointHttpBinding" closeTimeout="00:01:00"
                    openTimeout="00:01:00" receiveTimeout="00:10:00" sendTimeout="00:01:00"
                    allowCookies="false" bypassProxyOnLocal="false"
    hostNameComparisonMode="StrongWildcard"
                    maxBufferSize="65536" maxBufferPoolSize="524288" maxReceivedMessageSize="65536"
                    messageEncoding="Text" textEncoding="utf-8" transferMode="Buffered"
                    useDefaultWebProxy="true">
                    <readerQuotas maxDepth="32" maxStringContentLength="8192" maxArrayLength="16384"
                        maxBytesPerRead="4096" maxNameTableCharCount="16384" />
                    <security mode="Transport">
                        <transport clientCredentialType="Certificate"/>
                    </security>
                </binding>
            </basicHttpBinding>
        </bindings>
        <client>
            <endpoint
    address="https://zoolander.us.int.genesyslab.com:10080/Genesys/Interaction/WSCP_812_zoo/WebService
    CapturePoint"
                binding="basicHttpBinding" bindingConfiguration="iWebServiceCapturePointHttpBinding"
    behaviorConfiguration="ProvideClientCertificate"
                contract="WSCP.iWebServiceCapturePoint" name="iWebServiceCapturePointHttpBinding" />
        </client>
    </system.serviceModel>
</configuration>
```

Modifications for Java Clients

Assuming that all the procedures of generating, exporting, and importing certificates have been completed, the following modifications are required for the above Java client samples to run over HTTPS:

1. Update the URL of `WebService` or `WebService Stub` by replacing `http` with `https`.
2. Start your client with the following JVM options:
   - `-Djavax.net.ssl.keyStore="<Path to keystore.jks>/keystore.jks"`
   - `-Djavax.net.ssl.keyStorePassword="<Key store password, set when creating the keystore>"`
   - `-Djavax.net.ssl.keyStoreType=jks`
   - `-Djavax.net.ssl.trustStore="<Path to keystore.jks>/truststore.jks"`
   - `-Djavax.net.ssl.trustStorePassword=<Trust store password, set when creating the truststore>"`
   - `-Djavax.net.ssl.trustStoreType=jks`

Messaging in Native Mode

This section presents details of requests and responses in Interaction Server native mode.

Requests

Request Submit

This request is used for creating a new interaction. It assumes that `Queue`, `TenantId`, `InteractionType`, `InteractionSubType`, and `MediaType` are either specified in the `default-values` section of the Web Service Capture Point or provided in the request parameters. Example `Submit` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:Submit xmlns="http://www.genesyslab.com/interaction">
<TenantId>101</TenantId>
<Queue>Queue1</Queue>
<ExternalId>Test00001</ExternalId>
<UserData>
<kvitem><key>StringKey</key><value><ValueString>StringValue</ValueString></value></kvitem>
<kvitem><key>IntKey</key><value><ValueInt>812</ValueInt></value></kvitem>
<kvitem><key>List1Key</key><value><ValueList>
   <kvitem><key>StringKeyL1</key><value><ValueString>StringValueL1</ValueString></value></kvitem>
   <kvitem><key>IntKeyL1</key><value><ValueInt>1812</ValueInt></value></kvitem>
   <kvitem><key>List2Key</key><value><ValueList>
      <kvitem><key>StringKeyL2</key><value><ValueString>StringValueL2</ValueString></value></kvitem>
      <kvitem><key>IntKeyL11</key><value><ValueInt>11812</ValueInt></value></kvitem>
</ValueList></value></kvitem>
</ValueList></value></kvitem>
</UserData>
</ixn:Submit>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Request Hold

This request is used for putting an interaction on hold. It must have either an `InteractionId` or an `ExternalId` argument.

Example `Hold` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:Hold xmlns="http://www.genesyslab.com/interaction">
<ExternalId>Test00001</ExternalId>
</ixn:Hold>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Request Stop

This request is used for stopping a running interaction. It is very similar to request Hold. It must have either an `InteractionId` or an `ExternalId` argument. Only existing, running, or held interactions can be stopped.

Example `Stop` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:Stop  xmlns="http://www.genesyslab.com/interaction">
<ExternalId>Test00001</ExternalId>
</ixn:Stop>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Request Resume

This request is used for resuming a held interaction. It is very similar to request Hold. It must have either an `InteractionId` or an `ExternalId` argument.

Example `Resume` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:Resume xmlns="http://www.genesyslab.com/interaction">
<ExternalId>Test00001</ExternalId>
</ixn:Resume>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Request Update

This request is used for changing interaction properties. It must have either an `InteractionId` or an `ExternalId` argument. For changing properties there are the following two structures:

- Changed—For changing existing fields or creating new ones
- Deleted—For removing fields from the interaction

Example `Update` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:Update xmlns="http://www.genesyslab.com/interaction">
<ExternalId>Test00001</ExternalId>
<Changed>
<kvitem><key>StringKey</key><value><ValueString>StringValueAfterChange</ValueString></value></kvitem>
<kvitem><key>IntKey</key><value><ValueInt>8120</ValueInt></value></kvitem>
</Changed>
<Deleted>
<kvitem><key>List1Key</key><value></value></kvitem>
</Deleted>
</ixn:Update>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Request GetInfo

This request is used for getting interaction properties. It must have either an `InteractionId` or an `ExternalId` argument.

Example `Getinfo` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:GetInfo xmlns="http://www.genesyslab.com/interaction">
<ExternalId>Test00001</ExternalId>
</ixn:GetInfo>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Request Ping

This request is used for heartbeat monitoring. It has no required parameters.

Sample `Ping` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:Ping>
</ixn:Ping>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample Responses

All requests except `GetInfo` return a structure called `RequestResponse`. For a successful request, this structure has the following characteristics:

- `Hold, Stop, Resume, Update`—The response is empty.
- `Submit`—The response's `Extension` field contains the Interaction ID returned by Interaction Server.
- `Ping`—The response contains Interaction Server and Capture Points statistics.

The `GetInfo` request returns a structure called `GetInfoResponse`, which contains various fields holding interaction properties.

Example error response:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<SOAP-ENV:Fault>
<faultcode>SOAP-ENV:Client</faultcode><faultstring>Required value is missing</faultstring>
<detail>
<ixn:FaultMessage>< ixn:ErrorCode>2</ ixn:ErrorCode>
< ixn:ErrorDescription>Missing InteractionId or ExternalId</ ixn:ErrorDescription>
</ixn:FaultMessage>
</detail>
</SOAP-ENV:Fault>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Example of a response to a successful `Submit` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:ixn="http://www.genesyslab.com/interaction">
<SOAP-ENV:Body>
<ixn:RequestResponse>
   <ixn:Extension>
      <ixn:kvitem>
         <ixn:key>InteractionId</ixn:key>
         <ixn:value><ixn:ValueString>02JH8H2FE3Q3T00E</ixn:ValueString></ixn:value>
      </ixn:kvitem>
   </ixn:Extension>
</ixn:RequestResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Messaging in iWD Compatible Mode

This section presents details of requests and responses in iWD Compatible Mode.

Requests

Request ping

Example of a `ping` request:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
```

```
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:iwd="http://webservice.capture.gtl.evo">
 <SOAP-ENV:Body>
<iwd:ping>
</iwd:ping>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample Request createTask

This sample shows how to specify two k-v pairs in the `data` part of the message and how to specify a `customerId` Task Extension in the `ext` part of the message.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ns2="http://webservices.evo"
xmlns:ns4="http://taskinfo.gtl.evo"
xmlns:ns3="http://broker.gtl.evo"
xmlns:ns1="http://evo"
 xmlns:iwd="http://webservice.capture.gtl.evo">
 <SOAP-ENV:Body>
<iwd:createTask>
<iwd:captureId>TestiWD_0002</iwd:captureId>
<iwd:data xsi:type="iwd:string2stringMap">
<iwd:entry><iwd:key xsi:type="xsd:string">Key1</iwd:key><iwd:value
   xsi:type="xsd:string">Value1</iwd:value></iwd:entry>
<iwd:entry><iwd:key xsi:type="xsd:string">Key2</iwd:key><iwd:value
   xsi:type="xsd:string">Value2</iwd:value></iwd:entry>
</iwd:data>
<iwd:ext xsi:type="ns3:TaskExt">
<ns3:customerId>My Best Customer</ns3:customerId>
</iwd:ext>
</iwd:createTask>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample Request getTaskByTaskId

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:iwd="http://webservice.capture.gtl.evo">
 <SOAP-ENV:Body>
<iwd:getTaskByTaskId>
<iwd:taskId>02JHNT2FEDRTR005</iwd:taskId>
</iwd:getTaskByTaskId>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample Request getTaskByCaptureId

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:iwd="http://webservice.capture.gtl.evo">
 <SOAP-ENV:Body>
<iwd:getTaskByCaptureId>
<iwd:captureId>TestiWD_0002</iwd:captureId>
</iwd:getTaskByCaptureId>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample request updateTaskByTaskId

This sample demonstrates how to update various interaction properties.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"  xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ns2="http://webservices.evo"
   xmlns:ns4="http://taskinfo.gtl.evo"
xmlns:ns3="http://broker.gtl.evo" xmlns:ns1="http://evo"
   xmlns:iwd="http://webservice.capture.gtl.evo">
 <SOAP-ENV:Body>
<iwd:updateTaskByTaskId>
<iwd:taskId>02JHNT2FEDRTR00B</iwd:taskId>
<iwd:priority>123</iwd:priority>
<iwd:dueDateTime>2012-03-28T20:20:18Z</iwd:dueDateTime>
<iwd:data xsi:type="iwd:string2stringMap">
<iwd:entry><iwd:key xsi:type="xsd:string">Key1</iwd:key><iwd:value
   xsi:type="xsd:string">NewValue1</iwd:value></iwd:entry>
<iwd:entry><iwd:key xsi:type="xsd:string">Key3</iwd:key><iwd:value
   xsi:type="xsd:string">NewKeyNewValue</iwd:value></iwd:entry>
</iwd:data>
<iwd:ext xsi:type="ns3:TaskExt">
<ns3:customerId>The same customer</ns3:customerId>
</iwd:ext>
```

```
</iwd:updateTaskByTaskId>
 </SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Sample Responses

WebserviceFault Error Response

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ns2="http://webservices.evo"
   xmlns:ns4="http://taskinfo.gtl.evo" xmlns:ns3="http://broker.gtl.evo" xmlns:ns1="http://evo"
   xmlns:iwd="http://webservice.capture.gtl.evo">
<SOAP-ENV:Body>
<SOAP-ENV:Fault>
<faultcode>SOAP-ENV:Client</faultcode>
<faultstring>Interaction Server protocol error</faultstring>
<detail>
<fault xsi:type="ns2:WebserviceFault">
<code>43</code>
<message>Unknown interaction identifier specified</message>
<severity>ERROR</severity>
</fault>
</detail>
</SOAP-ENV:Fault></SOAP-ENV:Body></SOAP-ENV:Envelope>
```

createTaskResponse

The only parameter returned in the `createTaskResponse` is the `out` string, which contains the interaction ID of the new interaction.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
   ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ns2="http://webservices.evo"
   xmlns:ns4="http://taskinfo.gtl.evo" xmlns:ns3="http://broker.gtl.evo" xmlns:ns1="http://evo"
   xmlns:iwd="http://webservice.capture.gtl.evo"><SOAP-ENV:Body>
<iwd:createTaskResponse>
<iwd:out>02JGQY2FEEP9P000</iwd:out>
</iwd:createTaskResponse>
</SOAP-ENV:Body></SOAP-ENV:Envelope>
```

Genesys

Chapter

10 Transitioning to eServices from ICS 6.x

This chapter describes the transition from Internet Contact Solution (ICS) 6.x to Multimedia 7.x through 8.0.0.

For transition to releases 8.0.1 and later, you must first run the transition tool, then take the resulting database and run the upgrade script provided with your UCS 8.0.1 or later.

Note: This chapter uses *Multimedia 7.x* to refer to releases 7.x through 8.0.0.

It includes the following topics:

Overview

Multimedia 7.x has much of the same functionality as Internet Contact Suite (ICS) 6.x. However, Multimedia (eServices) does not replace ICS. Similarly, there is no migration of ICS components to Multimedia components. You cannot migrate ICS configuration data to 7.0. You must configure and install a new Multimedia solution from scratch.

There is one aspect of ICS 6.x data that you can convert for use in Multimedia: the data stored in and handled by the Contact Server database (with a few exceptions). The Multimedia equivalent of Contact Server is Universal Contact Server. A transition tool, available with Multimedia 7.x, can convert Contact Server data from ICS 6.5.1 to Multimedia 7.x. This tool is a standalone, GUI-less Java utility.

Note: In release 8.0.1, the transition tool is delivered on its own separate CD. Contact your Genesys representative for details.

Transition Strategy

If you have ICS 6.0, 6.1, or 6.5.0, you must first migrate your Contact Server database to ICS 6.5.1, then use the transition tool to convert the database for use in Multimedia 7.x. As a third step, you can then re-create the items that cannot be converted.

To summarize, ICS 6.x objects fall into three groups in terms of transition:

- Contact and interaction records in the Contact Server database are converted with little or no change. Converting the database with the transition tool is described in "Converting the Contact Server Database" on this page.
- Other Contact Server database objects are converted with some changes due to differences between ICS 6.x and Multimedia 7.x. These include standard responses and Content Analyzer rules (match criteria). They are described in "Transition Results" on page 346.
- All other objects cannot be converted or reused, but must be re-created. These include
  - Contact Server database objects such as Content Analysis training models
  - Other objects such as strategies and event handlers.
  - Any objects created as the result of customizing the ICS database.

  Suggestions for re-creating some of these are described in "Other 6.5.x Objects" on page 351.

Converting the Contact Server Database

In general outline, the database transition process is as shown in Figure 103.

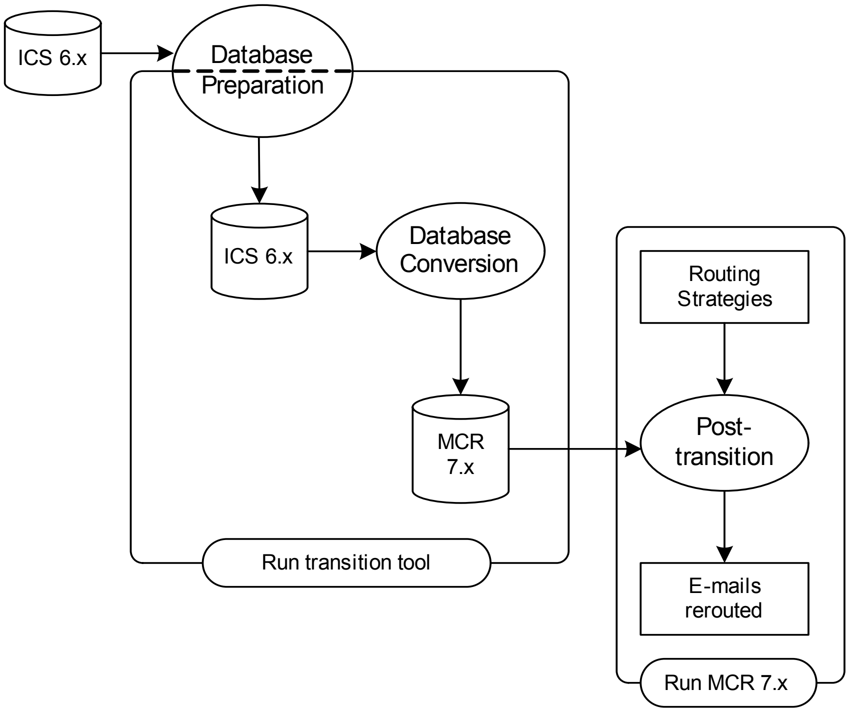


Figure 103: Outline of Transition Process

Table 44 lists the four overall stages in the database transition process, and provides links to the detailed description of the procedure for each stage.

Table 44: Overall Procedure for Database Transition

Stage	Description	Procedure
1. Preliminaries	Configure required objects and create required routing strategy or strategies.	"Preliminaries to database transition" on page 338
2. Preparation	Run two SQL scripts on the ICS 6.x database to prepare it for conversion.	"Preparing the database" on page 340
3. Conversion	Run the transition tool.	"Running the Transition Tool" on page 342
4. Post-transition	Start your Multimedia 7.x system. It uses the routing strategy you created in Step 1 to process any e-mails that were pending in the ICS 6.x database.	No separate description

Conversion Procedures

Procedure:
Preliminaries to database transition

Purpose: Configure required objects, if they do not exist already.

Start of procedure

1. Install Genesys Framework 7 (or later), including at least the following:
   - Configuration Server 7
   - Multimedia Third Party Components
   - Multimedia Universal Contact Server
   - Multimedia Universal Contact Server Transition Tool
   - Multimedia E-mail Server
2. Configure at least the following objects:
   - One Tenant.
   - Universal Contact Server (UCS), specifying the Tenant just described.
   - A Contact Server 6.5.1 Application object. This object must replicate your ICS 6.x Contact Server object: in particular, you must copy all options from your ICS 6.x Contact Server to this application. You can use the import and export functionality of Configuration Manager to transfer the option values.
   - E-mail Server, with a connection to UCS. This enables the system to reprocess e-mails which are pending in ICS 6.x.

   Note: Do not run E-mail Server until the transition process is complete.

   - All Persons that exist in your ICS 6.x configuration, with identical user names. This enables the system to convert the owner history of interactions. Persons are relevant to data conversion in the following ways:
     - Some data objects, such as agents’ personal spelling dictionaries and agent’s favorite standard responses, can only be converted if the Person identified as their Owner exists in the 7.x environment.
     - Some unfinished interactions are assigned to an agent in the ICS 6.x database (other unfinished interactions, such as those that have not been routed, may not be assigned). The transition tool can convert these interactions so that a post-transition routing strategy (described in Step 4) can assign them to the same agent. But such conversion can succeed only if both (a) the transition tool command

line does not include the parameter `-dontcheckperson` (described on page 343) and (b) the agent exists as a Person object in the 7.x environment.

  - Converted interactions include an Owner attribute that identifies the last agent who updated the interaction record. If this agent does not exist as a Person in the 7.x environment, the interaction is still converted, but the Owner attribute is blank.

3. Make a note of the following information, which you will need for the command-line arguments of the transition tool:
   - Host and port of Configuration Server
   - Name of the Contact Server 6.5.1 Application object
   - Name of the UCS Application object
   - Name of the E-mail Server Application object
4. Create and load a routing strategy to handle pending e-mails. In deciding how to configure this strategy, you should be aware of the following:

   After data conversion, E-mail Server 7.x submits to Interaction Server all e-mails which were pending in ICS 6.x. The queues that it submits them to are specified as follows:

   - Pending outbound e-mails go to the queue specified in the transition tool's `-outboundqueue` command-line parameter, which is described on page 343.
   - Pending inbound e-mails go to a queue defined as follows:
     - If the e-mail has a field `EmailIn.Mailbox` equal to the name of a `pop-client` section (in the options of the E-mail Server Application object in the Configuration Layer 7.x), E-mail Server submits the e-mail to the queue that is defined by the endpoint option managed in this `pop-client` section. See the *eServices 8.1 Reference Manual* for a description of endpoints.
     - If there is no endpoint configured, or if the inbound e-mail has a field `EmailIn.Mailbox` not equal to the name of a `pop-client` section, (because the pop-client sections differ in the 6.5 and 7.x configurations), E-mail Server works in 7.1 compatibility mode. In this mode, it submits the e-mail to the queue defined by the `email-processing` section's `default-inbound-queue` option.

   Also, in ICS 6.x, outbound e-mails can be of various types, such as *waiting for QA review, saved as draft by an agent, transferred to another location,* and so on. But Multimedia 7.x merges all of these types. So whatever strategy is loaded on the queue specified by the `-outboundqueue` command-line parameter (see Table 45 on page 342) treats all e-mails in that queue identically. If the strategy simply sends, it sends all of these pending outbound e-mails to customers, even those that were marked in ICS 6.x as drafts or as waiting for QA review. Because of this, Genesys recommends that you configure this strategy to route these pending e-mails to agents.

5. Plan and execute the preparation of your ICS 6.x database, as described in the following section.

End of procedure

Next Steps

- Review the limitations and restrictions described in the following section.
- Go on to prepare the database (page 340).

Limitations and Restrictions

Be aware of these limitations:

- Any customized columns that you have added to your ICS 6.5.x database cannot be converted. They will be lost.
- Once you begin the overall transition procedure, Genesys recommends that you make no configuration changes to either the ICS 6.5.x or the Multimedia 7.x environment. You are particularly discouraged from making any changes to Knowledge Management objects (Multimedia 7.x, using Knowledge Manager) or standard responses/categories/content analysis rules (ICS 6.5.x, using Content Analyzer and Response Manager).
- Genesys recommends that your UCS application not be connected (using its Connection tab in Configuration Manager) to StatServer. This prevents the system from creating inconsistent initial reporting data based on objects processed in ICS and converted to Multimedia.

Procedure: Preparing the database

Summary

The data conversion process is incremental: you can run the tool multiple times and each time it resumes the job where it left off. You can safely stop the transition tool using `Control-C`.

Note: Genesys strongly recommends against running pruning or archiving in between multiple conversion sessions. Use UCS Manager to ensure that no pruning or archiving is scheduled before transition is complete.

Start of procedure

1. Take your ICS 6.x system offline before you run the transition tool. There are two ways to accomplish this:
   - Maintenance windows—designate windows of time during which you can take your production ICS offline and run the transition tool on the database. This mode requires more time.
   - Database copy—copy your ICS database, then run the transition tool on the copy. This mode takes less time but requires more disk space for the duplicate database.
2. Locate the two SQL scripts that you will run on your ICS 6.x database before running the transition tool. These scripts are supplied, in either MSSQL or Oracle versions, in `GCTI\eServices 8.1\Universal Contact Server Transition Tool\sql_scripts\<database_type>`, where `<database_type>` is either `mssql` or `oracle`.
3. Proceed with either the maintenance windows or database copy method, as follows:

   a. Maintenance Windows

   iii. Take your production ICS database offline.

   iv. Optionally, make a backup copy of your ICS database.

   v. Run the preparation script `PrepareICSDBForTransition_<database_type>.sql` on the ICS database.

   vi. Run the population script `PopulateICSDBForTransition_<database_type>.sql` on the ICS database.

   vii. Run the transition tool on the ICS database. See "Running the Transition Tool" on page 342 for details.

   viii. When the maintenance window is over, stop the transition tool using `Control-C`.

   ix. Bring your production ICS database back online.

   x. At the next maintenance window, take your production ICS database offline and repeat Steps 1, 6, and 7.

   b. Database Copy

   i. Take your production ICS database off line.

   ii. Optionally, make a backup copy of your ICS database.

   iii. Make a working copy of the ICS database.

   iv. Run the preparation script `PrepareICSDBForTransition_<database_type>.sql` on both the original ICS database and the copy.

   v. Put the original database back on line.

   vi. Run the population script `PopulateICSDBForTransition_<database_type>.sql` on the copy.

vii. Run the transition tool on the copy. See “Running the Transition Tool” on page 342 for details.

viii. When the transition is complete, take the original database off line.

ix. Run the transition tool on the original (production) database. This converts only the records created, updated, or deleted since the original database was put back on line (in Step e).

End of procedure

Procedure: Running the Transition Tool

Start of procedure

1. Run the tool directly from the shell, on the target platform operating system. Use the command `UcsTransTool` with the parameters listed in Table 45.

Table 45: Transition Tool Command-Line Parameters

Name	Required?	Values
-app	Yes	Name of your ICS 6.x Contact Server Application object
-host	Yes	Host of your Configuration Server 7.x
-port	No	Port number of your Configuration Server 7.x. The default is 2020.
-ucsapp	Yes	Name of your UCS 7.x Application object
-esjapp	Yes	Name of your E-mail Server 7.x Application object
-user	Yes	Name of an existing Person object. It is used for the value of the Owner attribute of categories and standard responses.
-language	Yes	The language associated with standard responses and screening rules in Multimedia 7.x.
-archive	No	No values. If this parameter is not present, data conversion applies to the Main database. If this parameter is present, data conversion applies to the Archive database. Requires an Archive database in ICS 6.x and an Archive DAP in Multimedia 7.x.

Table 45: Transition Tool Command-Line Parameters (Continued)

Name	Required?	Values
-addresscheck	No	No values. If this parameter is present, the transition tool checks the addresses of e-mails for compliance with RFC 822 and does not convert noncompliant e-mails. If this parameter is not present, the transition tool does convert noncompliant e-mails.
-outboundqueue	Yes	Name of an existing queue. After the database transition, E-mail Server takes any outbound e-mails that are in a pending status in the ICS 6.x database and submits them to the queue specified by this parameter.
-iKnow	No	No values. If this parameter is present, the transition tool converts category feedback and creates an extra category to associate with each standard response. See "Standard Responses, Categories, Feedback" on page 347 for full description.
-tenant	Yes if UCS has multiple tenants	Tenant name. If UCS is associated with multiple tenants, this parameter selects one of them.
-dontcheckperson	No	No values. This parameter applies to interactions that are unfinished and are assigned to an agent (see also page 338). If this parameter is present, the transition tool converts these interactions without checking that the assigned agent exists as a Person object in the 7.x environment. If this parameter is not present, the transition tool does check and converts the interaction only if it finds the Person.

Example command line:

```
UcsTransTool -app ContServ651 -host Multimedia7host -port 7070
-ucsapp UCS701 -esjapp EmailSrvJav701 -user admin -language English
-outboundqueue TransitionQueue
```

Note: Genesys Technical Support may instruct you to use additional parameters for debugging purposes.

2. Upon launching, the tool displays the following on its console:
   - Number of records to convert per object type
   - Total number of records to convert
3. Then the tool starts to convert records, sequentially type after type, and for each record type being converted, it displays the following progress information:
   - Number of records converted
   - Predicted end time for this type of object
4. When all records for a given object type have been converted, it displays:

- Number of records successfully converted
- Number of errors encountered
- Time taken
- Average throughput

End of procedure

Next Steps

- Restart your Multimedia 7.x system. The system takes the routing strategy that you created in Step 4 on page 339 and uses it to process any e-mails that were pending in the ICS 6.x database.

Corruption Recovery

While running, the transition tool checks for corruption in the ICS 6.x database. If it detects corruption, it makes repairs on the corrupt data, with the repairs taking effect in the new Multimedia 7.x database. The tool gives no indication of this process.

Table 46 shows the types of corruption and the correction that the transition tool makes on them.

Table 46: Corruption Types and Corrections

Type of Corruption	Situation in ICS	Correction in Multimedia
Threading of interactions	The interaction has no parent interaction or an invalid parent interaction.	Set the interaction as the root interaction of a new thread of discussion.
Contact attributes	Of multiple `EmailAddress` or `PhoneNumber` attributes, there is either no primary attribute or multiple primary attributes.	Randomly select one and set it as primary.
Interaction contact	No contact is defined for an interaction or the contact is not available.	Use interaction data to create a new contact: • EmailIn: use e-mail's From address. • EmailOut: use e-mail's To address. • WebForm: use e-mail address. • PhoneCall: use phone number.

Table 46: Corruption Types and Corrections (Continued)

Type of Corruption	Situation in ICS	Correction in Multimedia
Interaction.DateSent	The DateSent field of the interaction is in the future.	Use value of DateCreated.
Value of isDone field is contradictory with other fields	isDone = true (interaction is finished) but Status = 1 (interaction is a draft assigned to an agent)	Convert Status to Pending. The interaction will be processed by a post-transition strategy.
	isDone = false (not finished) but the interaction is an automated response which has already been sent.	Convert Status to Stopped.

Errors and Recovery

If an error occurs, you can find a list of IDs of records that failed conversion in the file `log/UcsTransToolObjectsErrorList.log`.

The following are possible causes of errors during the conversion process:

1. Consistency error in the ICS 6.x record (for example, an invalid reference to a dependent object).
2. Format error in the ICS 6.x record (for example, an unknown charset or an unparsable email address).
3. System error (for example, no access to the ICS 6.x database or to UCS, or disk full).
4. Defect in UCS, the UCS API, the ICC API, or the transition tool itself.

To recover from (1) and (2), manually correct the ICS 6.x records, then run the transition tool again.

To recover from (3), fix the system error, then run the transition tool again.

To recover from (4), upgrade the defective component, then run the transition tool again.

After Transition

Inspect your database both before and after transition to check that no unwanted modifications are introduced by the transition process.

Transition Results

As mentioned on page 336, the transition tool converts contact and history data with little or no change. This is also the case with field codes.

Other objects in the Contact Server database undergo some change as the result of conversion. This section describes some differences between the pre- and post-conversion versions of these objects.

All of the objects described in this section acquire the following attributes upon conversion:

- Tenant—the first tenant in the list on the `Tenants` tab of the UCS Application object.
- Language—the value of the `-language` parameter in the transition tool command line.

The rest of this section presents information about individual object types.

Content Analysis Rules: Match Criteria (Screening Rules)

ICS 6.x Content Analysis rules cover several functionalities that are redistributed in Multimedia 7.x. In terms of the four tabs of ICS Content Analyzer, the distribution is as follows:

- The functionalities located on the `Match Criteria` tab correspond to Multimedia 7.x *screening rules,* controlled in Knowledge Manager and described in "Screening Rules" on page 59.
- The functionalities located on the `Routing Properties, Automated Actions,` and `E-Mail Properties` tabs are taken over by strategy objects in Multimedia 7.x.

The transition tool converts match criteria to screening rules, with the following changes:

- Any match criteria that contain the binary `NOT` operator (for example, `A NOT B`) are changed to use `AND NOT.` For example, `A NOT B` converts to `A AND NOT B.`
- Any special characters in the name of the Content Analysis rule are converted to underscore (_). For example, the name `Thanks&Redirect` would become `Thanks_Redirect.`

Content Analysis rule functionalities other than match criteria must be re-created in routing strategies. See also "Content Analysis Rules" on page 351.

Standard Responses, Categories, Feedback

Background

- In ICS 6.x, each standard response is associated with at least one category. You can create a standard response without a category but Response Manager assigns such standard responses to a category called Uncategorized.
- In ICS 6.x, the relation between standard responses and categories is many-to-many: one standard response can belong to many categories, and one category may have many standard responses. In Multimedia 7.x the relation between standard responses and categories is many-to-one: one standard response can belong to only one category, although one category may have many standard responses. This difference between ICS 6.x and Multimedia 7.x has consequences for conversion, as explained below.
- Feedback is data that records a relation between an object (either a category or a standard response) and a interaction, signifying that object O is a good match with interaction I. In ICS O is a standard response; in Multimedia O is a category. This difference also has consequences for conversion.

Name Conversion

As described on page 26, names of Knowledge Manager objects (categories, standard responses, and so on) in Multimedia 7.x must consist only of alphanumeric characters (A–Z, a–z, 0–9), hyphen (-), underscore (_), and space (names must also be no more than 64 characters long). If any ICS 6.x standard response or category has a name that violates this rule, the transition tool replaces the nonconforming characters according to the correspondences shown in Table 47.

Table 47: Replacement in Knowledge Manager Names

Original	Replacement	Original	Replacement
~	0	?	F
!	1	/	_
@	2	,	H
#	3	.	I
$	4	`	J
%	5	{	K
^	6	}	L

Table 47: Replacement in Knowledge Manager Names (Continued)

Original	Replacement	Original	Replacement
&	7	[	M
*	8	]	N
(	9	|	O
)	A	:	P
+	B	;	Q
=	C	"	R
<	D	'	S
>	E	\	T

Transition Process

The transition tool converts standard responses, categories, and feedback as follows (these numbered steps do not necessarily happen in chronological order):

1. It creates a root category named `ICS Migrated <appname> <langname>`, where `<appname>` is the value of the `-app` parameter and `<langname>` is the value of the `-language` parameter in the transition tool command line. An example might be `ICS Migrated myContactServer German`.
2. All categories existing in ICS 6.x become subcategories of the root category.
3. Standard responses remain associated with their categories. But any standard response that was associated with multiple categories becomes a set of duplicates, each associated with one category. Consider an example with three categories, called Cat1 Cat2 Cat3, and three standard responses, called SR1 SR2 SR3, associated as shown in Figure 104.

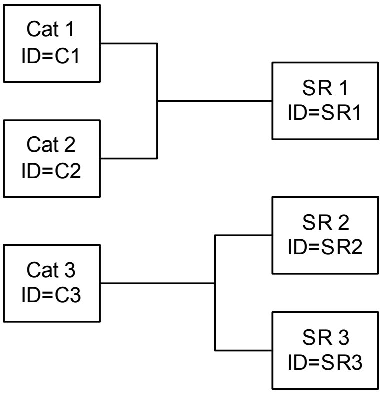


Figure 104: Example Standard Responses and Categories Before Conversion

Figure 105 shows the same categories and standard responses after conversion.

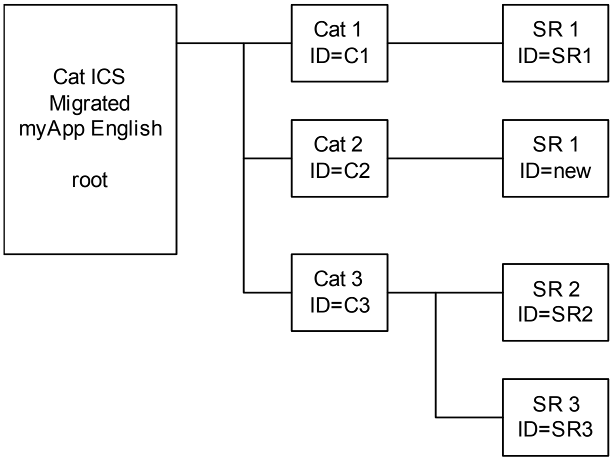


Figure 105: Example After Conversion

Notice SR 1. Before conversion it is a single standard response assigned to two categories. Because Multimedia 7.x does not support this type of relation, the transition tool replaces it with two identical standard responses (differing only in their database IDs), each assigned to a single category.

In this scenario, feedback is not converted. To convert feedback, you must include the parameter `-iknow` in the transition tool command line.

4. With the `-iknow` parameter, the output of the transition tool is different: the tool creates an additional layer of categories so that each standard response is associated with exactly one category. Figure 106 shows the results of converting the same example as Figure 104, but with the `-iknow` parameter.

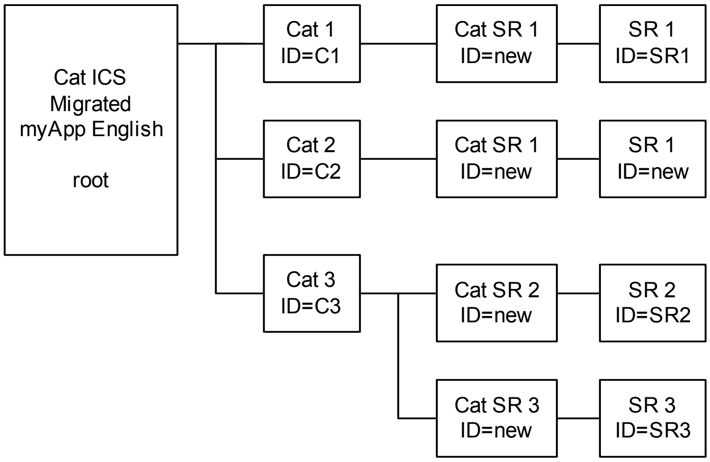


Figure 106: Example After Conversion, with the -iKnow Parameter

Notice SR 2 and 3. Before conversion, both belong to Cat 3. After conversion, each belongs to its own category, and it is these lower-level categories that both belong to Cat 3.

The advantage of this is as follows. Before conversion, feedback data could mark interactions as being good matches for SR2 and SR3 separately. After conversion, feedback data must mark interactions as good matches for categories, not standard responses—but before conversion, SR2 and SR3 belong to the single category Cat3. Taking the feedback data for SR2 and SR3 and merging it as feedback for Cat3 would lose the distinction between SR2 and SR3. Using the `-iknow` parameter adds a level of categories that have one-to-one relations with standard responses. Associating feedback data with this level of categories preserves the distinctions between standard responses like SR2 and SR3.

Other attributes of the converted standard responses are as follows:

- Name and Description—the name of the ICS 6.x standard response is copied into both of these attributes.
- Usage type—all three types are selected and specified as `Not Active.`
- Status—Approved.
- Owner—the value of the `-user` parameter in the transition tool command line.
- Date modified—date that the transition tool is run.
- Expiration date—empty.
- Version—0.

Field Codes

Field codes simply transfer their names and contents from ICS 6.x to Multimedia 7.x.

Other 6.5.x Objects

Objects that you created or customized in ICS 6.x cannot be directly migrated or transferred to Multimedia 7.x. This section lists some of the ICS 6.x objects and discusses what you can do to transition them to Multimedia 7.x.

Contact Center Objects

DNs

Unlike ICS 6.x, Multimedia 7.x does not use DNs that are specified for particular nonvoice media such as e-mail and chat. You can delete any such DNs from your configuration.

Places

Since DNs no longer serve to mediate the flow of interactions to agents, you must configure agent capacity rules. You can do this for either a tenant (the default) or individual agents.

Content Analysis Rules

These ICS 6.x rules have two general functions: scan an incoming message to see if it matches certain criteria, and perform specified actions if there is a match.

As stated in "Content Analysis Rules: Match Criteria (Screening Rules)" on page 346, the transition tool converts match criteria to screening rules.

However, the actions that were performed by the `Routing Properties` and `Automated Actions` tabs of ICS Content Analyzer are taken over by strategy objects in Multimedia 7.x. For example, Content Analyzer included an `Intelligently analyze the content of the message` check box. In Multimedia, this function is performed by the `Classify` and `Classify segmentation` strategy objects. See *Universal Routing 8.1 Reference Manual* for details on these and other IRD e-mail objects.

Strategies

There is no simple way to take routing strategies that you created for ICS 6.x and use them with Multimedia 7.x.

The best way to understand the relation between ICS 6.x and Multimedia 7.x is to study the Interaction Workflow Samples, a component of Multimedia 7.x. These exemplify many of the tasks that users commonly require (for example, sending interactions to QA review). If you have these or similar tasks set up in your ICS 6.x environment, you should compare your setup with the analogous part of the Interaction Workflow Samples.

Note also the following:

- In Multimedia 7.x, strategies must be contained in Business Processes. Business Processes are created and edited using Interaction Routing Designer; see *Universal Routing 8.1 Interaction Routing Designer Help* for details. If you want to use strategies from your ICS 6.x in Multimedia 7.x, you must embed the strategies in one or more interaction workflows.
- Universal Routing 7.0.1 contains many more e-mail routing objects than previous versions. Also some objects (such as `Target`) that were present in previous versions have significant differences in 7.0.1. See *Universal Routing 8.1 Reference Manual* for descriptions.

Other Objects

Event Handlers

Multimedia 7.x has no equivalent of custom event handlers.

Callback Records

The transition tool converts ICS 6.x callback records, so they are available in the Multimedia 7.x UCS database. But the callback media is not supported in Multimedia 7.x, with the following consequences:

- The transition tool converts records of all callbacks, including unfinished ones. Unfinished callback are converted as finished. They are not reprocessed in Multimedia 7.x.
- There will be no new callback records. The function of ICS Callback Server 6.x is performed by Genesys Voice Callback 7.0, which has its own storage system for callback requests. For details see the *Voice Callback 7 Getting Started Guide* and the resources that it cites.

iKnow Training Models

Classification Server (part of the optional iKnow package in 6.5.x) now classifies on Category objects rather than Standard Response objects as it did in 6.x. Because of this fundamental change, Classification Server 7.x cannot use any models created in ICS 6.x.

Web Sample and Web API

ICS 6.x Web Sample (WebStarterApp in older releases) cannot be migrated or converted at all. Web API Server likewise cannot be migrated or converted.

Glossary

This glossary provides an alphabetical listing of terms, names, and concepts used in the documentation for Genesys eServices 8.1. This glossary is not divided into sections.

Note: Items that apply only to Genesys Content Analyzer have definitions that begin with *(Genesys Content Analyzer)*.

accuracy

(Genesys Content Analyzer) A general term for how correct a model is in assigning text objects to categories. If you produce a model using cross-validation or test it on a training object, you obtain specific accuracy ratings called precision and recall. Contrast confidence, which is part of what a model does when assigning a text objects to categories: it produces a list of categories with a confidence level indicating the model's assessment of how likely it is that this text object belongs to this category.

acknowledgment

A possible use for standard responses: a message sent automatically to inform the sender that the message has been received.

actionability

A characteristic of social media interactions that reflects whether an interaction calls for attention from an agent.

agent capacity rule

Rule specifying the maximum number of interactions of each media type an agent can handle. Agent capacity rules are used in routing; if an agent is up to capacity for a given media type, the strategy may still send that agent further interactions of other media types. An agent may have different settings for different media types; for example, a capacity rule may classify the agent as busy for chat interactions (already engaged in chat) but not for e-mail interactions (may be sent additional e-mail interactions). You can configure agent capacity rules at various levels of generality (Agent, Tenant) in Configuration Layer. Configure using the Agent Capacity Wizard.

archiving, pruning

Means of keeping the size of the Universal Contact Server database under control. Archived records are removed from the Main database and stored in the Archive database. Pruned records are deleted and not stored.

autoresponse

A possible use for standard responses: a message sent as an automated response to an incoming e-mail.

business process

In Interaction Routing Designer (IRD), a set of objects (mainly queues and routing strategies) that are available for use in constructing interaction workflows. Objects in business processes function in an interaction workflow only if you have placed them into a workflow and connected them to at least one other object in the workflow.

cache

Database associated with Interaction Server that stores transient information about interactions. This information includes:

- operational data.
- queues through which the interaction passes during processing.

This cache has sometimes also been called Persistent Queue, Interaction Cache, or iCache.

category

An item in a system of categories and subcategories, called a category tree, created and edited using Knowledge Manager. A category, besides denoting a concept, may be associated with one or more standard responses and one or more screening rules. If an incoming e-mail is assigned to the category, the category's standard responses can serve either as the content for an automated reply or as suggestions for agents to use in their replies. A terminal or leaf category is one that contains no subcategories. A nonterminal category is one that does have subcategories.

co-browsing

The ability for agents and customers to simultaneously navigate shared web pages; sometimes also known as web collaboration or conavigation. In release 7.6 and later, this functionality is provided by two components, Genesys Co-Browsing Server and KANA Response Live Server. These components are provided as Genesys Web Collaboration, which is an option that you can add to either Genesys Chat or Genesys Inbound Voice. See also “eServices and the CIM Platform” on page 13.

confidence

(Genesys Content Analyzer) A numerical score, ranging from 1 to 100, indicating the likelihood, according to a particular model, that a text object belongs in a certain category. You set confidence level as one of the attributes of the IRD objects `Classify` and `Classification switch`. Contrast accuracy, which is an assessment, produced by testing, of the correctness of a model’s assignment of text objects to categories. Confidence expresses a model’s guess about a categorization; accuracy rates the correctness of that guess.

content analysis

(Genesys Content Analyzer) What Content Analyzer does: applies natural language-processing technology to analyze the content of incoming interactions. *See also* Genesys Content Analyzer and model.

e-mail

Knowledge Manager uses this term, interchangeably with *message,* to refer to interactions that it applies screening rules or content analysis to. Although most interactions that are screened or classified are expected to be e-mail messages, in fact these operations can apply to any interaction that has text somehow associated with it. The text can be the body of the interaction (e-mail, chat), or it can be more obliquely associated with it (as user data, for example).

ESP

External Services Protocol. Used by Interaction Server to communicate with servers that perform a specific service when requested to do so. Such servers are called ESP servers. Classification Server is an ESP server, as is E-mail Server when it generates an acknowledgment or autoresponse (when E-mail Server processes incoming or outgoing e-mails, it is a media server). You can create custom ESP servers using the Genesys Open Media Platform SDK.

field code

Formula using variables, constants, and operators; allows a standard response to be personalized relative to the contact it is addressed to. For example, a response beginning `Dear <$Contact.FirstName$>` may be sent to dozens of recipients; in each message *`<$Contact.FirstName$>`* is replaced by the first name for that contact as listed in the Universal Contact Server database. This process of replacement is called rendering.

Genesys Content Analyzer

Optional extension of Genesys Knowledge Management, activated by presence of the proper license key. Content Analyzer uses natural language-processing technology to scan incoming e-mails, assigning the e-mail to one or more categories with a percentage confidence rating. The category assignments can then be used to pull suggested responses from the Standard Response Library. Content Analyzer creates and refines its recognition algorithms by training.

interaction

In the broad sense: an attempted communication between a customer and a contact center, in either direction. The attempt may be successful or not; it has a media type; it may give rise to other interactions (as when an incoming e-mail gives rise to an automatic acknowledgement). It may belong to a series of related interactions, known as a *thread.*

In the narrow sense: a software object, created by a server, that represents an interaction in the broad sense.

interaction workflow

Created by IRD. Specifies the high-level flow of interactions between various contact center objects (mainly queues and routing strategies). Must begin with a single queue for inbound interactions. Normally uses the contact center objects belonging to one business process but may include objects from more than one. Any switching between business processes takes place within routing strategies.

IRD

Interaction Routing Designer; GUI component of Genesys Universal Routing used to design routing strategies that handle interactions as well as interaction workflows that encompass routing strategies, queues, and other objects.

media server

Component that interfaces with a particular media to bring interactions into the Genesys eServices system. Supported media are e-mail (E-mail Server), chat (Chat Server), Short Message Service (SMS Server), and social media (Social Messaging Server).

message

See e-mail.

model

(Genesys Content Analyzer) classification model; a resource that Classification Server uses to classify e-mails. A model is associated with a category structure; it contains a statistical representation of each category in the structure. To classify a new e-mail message Classification Server compares it with the representation of each category. Then the server returns a list of categories, each with a percentage rating of the confidence with which the e-mail in question can be assigned to that category. Model files are created and refined by training and are stored in the Universal Contact Server database.

natural language processing

(Genesys Content Analyzer) technology that operates on data that is in a language used by humans (as opposed to programming or other machine languages).

online

An online interaction is one that takes place online in real time. For example, chat is online; e-mail is not.

operational data

What the media server sends to Interaction Server for each incoming interaction: interaction ID, originating party, time received, and so on.

precision

(Genesys Content Analyzer) One mathematical expression of a model's accuracy. Given the following for a category X:

a = the number of items the model correctly assigns to X

b = the number of items the model incorrectly assigns to X

Then $precision = a /(a + b)$

See also "Average Results Subtab" on page 136.

pruning

See archiving, pruning.

queue

Or interaction queue: in eServices, a logical entity in the Interaction Server cache database. Typically, there is an Inbound queue, an Outbound queue, and various intermediary queues.

recall

(Genesys Content Analyzer) One mathematical expression of a model's accuracy. Given the following for a category X:

a = the number of items the model correctly assigns to X *c* = the number of items the model incorrectly rejects from X (that is, items that the model should assign to X but does not)

Then $recall = a / (a + c)$

See also “Average Results Subtab” on page 136.

rendering

The process of taking a formula contained in a field code, performing the operation described by the formula, and substituting the result of the operation for the formula text.

routing strategy

Created by Interaction Routing Designer (IRD). Uses strategy-building objects (routing rules, interaction data, business rules, and so on). Can be contained in an interaction workflow. Can apply logic (segmentation, conditional branching). Can deliver to an agent or other target. Used within an interaction workflow, can switch between business processes.

screening rule

Pattern matching for character strings in e-mails.

segmentation

Function and object of routing strategy, applying conditional branching to the routing of an interaction (if it has attribute A, do X; if B, do Y; otherwise do Z).

sentiment

A characteristic of an interaction that reflects the attitude that it expresses, generally classified as positive, negative, or neutral.

snapshot

A list of all the interactions in Interaction Server’s database that meet specified conditions at a given time. The agent application requests a snapshot from Interaction Server and uses the results to populate the list of interactions that display on the Agent or Supervisor desktop. See also the “Snapshot Operations” section of the “Basic Interaction Models” chapter of the *Genesys Events and Models Reference Manual.*

standard response

Item in the Standard Response Library, which stores prewritten responses for use as suggestion to agent, acknowledgment, and/or autoresponse. Each standard response is assigned to exactly one category in the system (however, a category may have zero or many standard responses assigned to it).

suggestion to agent

A possible use for standard responses: text displayed in an agent's desktop application as a suggestion for use in replying to an interaction. The agent can paste the text into an e-mail message or a chat reply, or read from it during a voice interaction.

terminal category

A category that contains no subcategories; a leaf on the category tree. A category that contains subcategories is a *nonterminal* category.

third party

It is important to distinguish Third Party *Components* from third-party *applications.*

- Multimedia Third Party Components is a collection of files, mostly Java libraries, that eServices uses for its normal operation. Third Party Components is installed as part of the integrated install, as described in the *eServices 8.1 Deployment Guide.* The Third Party Components installation package places these files in various locations on the host machine.

  Note: In the 8.0.1 release, this component name was changed to "Java Environment and Libraries for eServices and UCS."

- `ThirdPartyApplications` is a directory on the eServices product CD that contains the web server Apache, the application container Tomcat, and various supporting files. eServices requires a web server and an application container, but Apache and Tomcat are not the only supported types, and you are not required to install the particular copies in this directory; they are provided as a convenience.
- Third Party Media

training

(Genesys Content Analyzer) Process that the Content Analyzer follows to create and refine models: it works its way through a training object, which is a number of e-mails that have been classified according to a category tree. These may be actual e-mails that agents have classified, or they may be created especially to be used in training.

training object

A category tree plus a group of text objects that are classified according to the category structure. Typically the text objects are e-mails, but you can also include standard responses in the training object. Training operates on a training object to produce or refine a classification model, which can then be used to classify new e-mails.

view

A configurable node in an interaction workflow. Each queue defined in an interaction workflow is associated with one or more views. A view defines the queue-processing rules: what interactions should be selected from the queue and in what order.

VRP

Virtual Routing Point, a type of DN. A virtual device (not a device in a switch), not associated with any particular target, where customer interactions wait while Universal Routing Server makes routing decisions.

web collaboration

The ability for agents and customers to simultaneously navigate shared web pages; sometimes also known as co-browsing or conavigation. In release 7.6 and later, this functionality is provided by two components, Genesys Co-Browsing Server and KANA Response Live Server. These components are provided as Genesys Web Collaboration, which is an option that you can add to either Genesys Chat or Genesys Inbound Voice. See also “eServices and the CIM Platform” on page 13.

workbin

Contact center object holding interactions for later processing by a particular agent (or agent group, place, or place group). Agents can use workbins to store interactions that they have started working on and wish to continue working on at a later time. Interactions can also be distributed to workbins by Universal Routing Server. A workbin is like a queue in that it holds interactions. A workbin differs from a queue as follows:

- It is associated with a particular agent/place/group and its major function is to hold interactions for that agent/place/group to process.
- Agents can view the entire content of the workbin and pull interactions from it in any order. Agents can also pull interactions from queues, but only in the order defined by the queue.

workflow strategy

See interaction workflow.

Supplements

Related Documentation Resources

The following resources provide additional information that is relevant to this software. Consult these additional resources as necessary.

eServices

- *eServices 8.1 Deployment Guide,* which describes deployment procedures for all eServices components.
- *eServices 8.1 Reference Manual,* which provides a reference listing of all configuration options and of field codes used in standard responses.
- *eServices Social Media Solution Guide*, which provides information on deploying and using the Genesys Social Messaging Management product. It is available on the Genesys Documentation Wiki at `http://docs.genesyslab.com/wiki/index.php?title=EServices_Social_Media_Solution_Guide`.
- *eServices 8.1 Universal Contact Server Manager Help,* which is a guide to the Universal Contact Server Manager user interface.
- *eServices 8.1 Knowledge Manager Help,* which is a guide to the Knowledge Manager user interface.
- For a listing of classes, methods, fields, and constants of the Web API portion of the Web API Server component, see:
  - *eServices 8.1 .NET Web API Reference* for the .NET Web API
  - The API References of the Platform SDK for the Java-based Web API
- *eServices 8.1 Web API Client Developer's Guide,* which describes the structure of the Web API, explains the Simple Samples, and describes procedures for customizing them.
- "eServices Log Events" in *Framework 8.1 Combined Log Events Help,* which is a comprehensive list and description of all events that may be recorded in logs.

- Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at `http://genesyslab.com/support`.
- Documentation on the other three members of the Genesys Customer Interaction Platform: Universal Routing, Reporting, and Management Framework. Some of this is listed in the following sections.

Genesys Desktop

- *Genesys Desktop 7.6 Deployment Guide,* which describes deployment procedures for the Genesys Desktop.
- *Genesys Desktop 7.6 Developer's Guide,* which describes customizing the Genesys Desktop.
- *Genesys Desktop 7.6 Agent Help,* which is a guide to the Genesys Agent Desktop.
- *Genesys Desktop 7.6 Supervisor's Help,* which is a guide to the Genesys Supervisor Desktop.

Universal Routing

- *Universal Routing 8.1 Reference Manual,* which contains descriptions of all routing strategy objects, including those that are specific to eServices.
- *Universal Routing 8.1 Strategy Samples,* which describes the sample strategies supplied with Universal Routing.
- *Universal Routing 8.1 Business Process User's Guide,* which contains step-by-step instructions for using Interaction Routing Designer to design interaction workflows. It also describes the sample business processes supplied with eServices.
- *Universal Routing 8.1 Interaction Routing Designer Help,* which is a guide to Interaction Routing Designer, including the portion of it that designs interaction workflows and business processes for eServices.

intelligent Workload Distribution (iWD)

- *intelligent Workload Distribution 8.0 Deployment Guide,* which describes deployment procedures and configuration information for iWD components.

Genesys

- *Genesys Events and Models Reference Manual,* which includes a set of basic voice and interaction models, showing the components involved and the relevant event messages sent among them. For authoritative description of the event messages, see the next item.

- The API References of the Platform SDK, which provide the authoritative information on methods and functions for each SDK, including requests and events. The class `Message` includes all event and request messages.
- *Genesys Technical Publications Glossary,* which ships on the Genesys Documentation Library DVD and which provides a comprehensive list of the Genesys and computer-telephony integration (CTI) terminology and acronyms used in this document.
- *Genesys Migration Guide*, which ships on the Genesys Documentation Library DVD, and which provides documented migration strategies for Genesys product releases. Contact Genesys Technical Support for more information.
- Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at `http://genesyslab.com/support`.

Information about supported hardware and third-party software is available on the Genesys Technical Support website in the following documents:

- *Genesys Supported Operating Environment Reference Manual*
- *Genesys Supported Media Interfaces Reference Manual*

Consult these additional resources as necessary:

- *Genesys Hardware Sizing Guide,* which provides information about Genesys hardware sizing guidelines for the Genesys 8.x releases.
- *Genesys Interoperability Guide,* which provides information on the compatibility of Genesys products with various Configuration Layer Environments; Interoperability of Reporting Templates and Solutions; and Gplus Adapters Interoperability.
- *Genesys Licensing Guide,* which introduces you to the concepts, terminology, and procedures relevant to the Genesys licensing system.
- *Genesys Database Sizing Estimator 8.x Worksheets,* which provides a range of expected database sizes for various Genesys products.

For additional system-wide planning tools and information, see the release-specific listings of System Level Documents on the Genesys Technical Support website. These documents are accessible from the `system level documents by release` tab in the Knowledge Base `Browse Documents` Section.

Genesys product documentation is available on the:

- Genesys Technical Support website at `http://genesyslab.com/support`.
- Genesys Documentation wiki at `http://docs.genesyslab.com/`.
- Genesys Documentation Library DVD and/or the Developer Documentation CD, which you can order by e-mail from Genesys Order Management at `orderman@genesyslab.com`.

Document Conventions

This document uses certain stylistic and typographical conventions—introduced here—that serve as shorthands for particular kinds of information.

Document Version Number

A version number appears at the bottom of the inside front cover of this document. Version numbers change as new information is added to this document. Here is a sample version number:

80fr_ref_06-2008_v8.0.001.00

You will need this number when you are talking with Genesys Technical Support about this product.

Screen Captures Used in This Document

Screen captures from the product graphical user interface (GUI), as used in this document, may sometimes contain minor spelling, capitalization, or grammatical errors. The text accompanying and explaining the screen captures corrects such errors *except* when such a correction would prevent you from installing, configuring, or successfully using the product. For example, if the name of an option contains a usage error, the name would be presented exactly as it appears in the product GUI; the error would not be corrected in any accompanying text.

Type Styles

Table 48 describes and illustrates the type conventions that are used in this document.

Table 48: Type Styles

Type Style	Used For	Examples
Italic	- Document titles - Emphasis - Definitions of (or first references to) unfamiliar terms - Mathematical variables Also used to indicate placeholder text within code samples or commands, in the special case where angle brackets are a required part of the syntax (see the note about angle brackets on page 365).	Please consult the *Genesys Migration Guide* for more information. Do *not* use this value for this option. A *customary and usual* practice is one that is widely accepted and used within a particular industry or profession. The formula, x +1 = 7 where x stands for . . .
Monospace font (Looks like `teletype` or `typewriter text`)	All programming identifiers and GUI elements. This convention includes: - The *names* of directories, files, folders, configuration objects, paths, scripts, dialog boxes, options, fields, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages. - The values of options. - Logical arguments and command syntax. - Code samples. Also used for any text that users must manually enter during a configuration or installation procedure, or on a command line.	Select the `Show variables on screen` check box. In the `Operand` text box, enter your formula. Click `OK` to exit the `Properties` dialog box. T-Server distributes the error messages in `EventError` events. If you select `true` for the `inbound-bsns-calls` option, all established inbound calls on a local agent are considered business calls. Enter `exit` on the command line.
Square brackets ([ ])	A particular parameter or value that is optional within a logical argument, a command, or some programming syntax. That is, the presence of the parameter or value is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information.	`smcp_server -host [/flags]`
Angle brackets (< >)	A placeholder for a value that the user must specify. This might be a DN or a port number specific to your enterprise. **Note:** In some cases, angle brackets are required characters in code syntax (for example, in XML schemas). In these cases, italic text is used for placeholder values.	`smcp_server -host <confighost>`

Index