

IBM Cúram Social Program Management
Version 7.0.2

Cúram System Configuration Guide



Note

Before using this information and the product it supports, read the information in [“Notices” on page 24](#)

Edition

This edition applies to IBM® Cúram Social Program Management v7.0.2 and to all subsequent releases unless otherwise indicated in new editions.

Licensed Materials - Property of IBM.

© **Copyright International Business Machines Corporation 2012 , 2018.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

- List of Figures..... v**
- List of Tables..... vi**

- Chapter 1. Configuring the system.....1**
 - Introduction..... 1
 - Purpose..... 1
 - Audience..... 1
 - Prerequisites..... 1
 - Chapters in this Guide..... 1
 - Application Configuration..... 2
 - Introduction..... 2
 - Configuring Application Properties..... 2
 - Configuring Code Tables..... 3
 - Working with Configurable Validations..... 4
 - Configuring Language and Locale Mappings..... 6
 - Configuring Participant Nicknames for Search..... 6
 - (deprecated) Configuring Spell Checking..... 7
 - Case Audit Selection Query Configuration..... 7
 - Introduction..... 7
 - Creating and Publishing a Dynamic Selection Query..... 7
 - Creating and Publishing a Fixed Selection Query..... 8
 - Communication Template Configuration..... 8
 - Introduction..... 8
 - Managing Microsoft Word Templates..... 8
 - Managing XSL Templates..... 9
 - Assigning Communication Templates to Case and Participant Types..... 9
 - Batch Processing Configuration..... 9
 - Introduction..... 9
 - Adding a New Batch Process to the Application..... 9
 - Organizing Batch Processes into Groups..... 10
 - Submitting a Batch Process for Execution..... 10
 - Creating a Batch Process Error Code..... 10
 - Security Configuration..... 11
 - Introduction..... 11
 - Types of Security for Application Elements..... 11
 - User Security Profiles..... 12
 - Online Security Process Logging..... 13
 - Business Intelligence Configuration..... 14
 - Introduction..... 14
 - Configuring Business Intelligence Reports..... 15
 - Configuring the Business Intelligence Report Viewer..... 15
 - Target System Configuration..... 16
 - Introduction..... 16
 - Creating a Target System..... 16
 - Content Management Interoperability Services Configuration 17
 - Introduction..... 17
 - Enabling Integration with a Content Management System..... 17
 - Configuring Metadata for Attachments..... 17
 - Conclusion..... 18
 - Summary..... 18
 - Additional Information..... 19

Technical Information.....	19
Inserting Fields into a Microsoft Word Template.....	19
Introduction.....	20
Creating a Microsoft Word Template.....	20
Writing Server Code to Populate Communication Data.....	20
(deprecated) Configuring FILE_EDIT widget with Google Chrome.....	22
Notices.....	24
Privacy Policy considerations.....	25
Trademarks.....	25

List of Figures

- 1. Sample Microsoft Word Template Content.....21
- 2. Sample Code to Return Data to Populate a Microsoft Word Communication..... 22
- 3. Structure of the Returned Object from Code Sample..... 22

List of Tables

- 1. BIRT Report Configuration Options..... 15
- 2. BIRT Report Viewer Configuration Options..... 15
- 3. Metadata Elements..... 18

Chapter 1. Configuring the system

A broad range of functions in the Cúram application can be administratively configured. The following can be configured: application properties, case audit selection queries, communication templates, batch processes, security settings, business intelligence reports, target systems, content management interoperability services.

Introduction

Purpose

The purpose of this guide is to provide an overview of the configuration options that are available to system administrators for managing parts of the application. To completely understand administration services for the application, this guide should be read in conjunction with the Cúram Location Administration Guide and the Cúram Organization Administration Guide.

System administration includes functionality for managing a broad spectrum of elements that impact the operation of the application. System administration requires some familiarity with technical terms, as certain components of system administration can only be created during application development. For example, the execution of batch processes is requested from the system administration module; however, the batch processes themselves can only be designed and implemented as part of application development.

Other application components can be maintained as part of system administration, but must still be referenced in the application as part of application development. This includes code tables and rate tables.

In order to best understand these concepts, the guide should be read in full.

Audience

This guide is intended for business analysts and system administrators employed by the organization. It is assumed that this audience has a strong knowledge of the organization's business requirements. A thorough knowledge of the application is required to read this document. For the most part, this document assumes a low level of technical knowledge from its readers; however, there are certain aspects of system administration which link into the application, and thus, may introduce certain terms more familiar to a reader with a higher technical background.

Prerequisites

It is assumed that the reader is familiar with the basic concepts of Social Enterprise Management. In particular, it is assumed that the reader is familiar with administrative tasks necessary to manage a social enterprise organization, such as the management of system users, user security, and the organization's reporting hierarchy.

Chapters in this Guide

The following list describes the chapters within this guide:

Application Configuration

This chapter covers a range of configuration options including application properties, code tables, and language locale configuration. It also covers participant nickname configuration.

Case Audit Selection Query Configuration

This chapter covers case audit selection queries configuration.

Communication Template Configuration

This chapter provides an overview of the configuration options available for communication templates.

Security Configuration

This chapter provides an overview of configuration options for security administration.

Batch Processing Configuration

This chapter provides an overview of batch processing configuration options.

Business Intelligence Report and Viewer Configuration

This chapter provides an overview of the system configuration options for the business intelligence viewer and associated reports.

Target System Configuration

This chapter covers the basic configuration options available in the system administration application for target systems.

Application Configuration

Introduction

This chapter covers a range of configuration options specific to the running of the runtime application. This includes property configuration, code tables, and language locale configuration. It also covers participant nicknames.

Application properties are used in the application to configure certain parts of the runtime application. As such, they provide a means for the system administrator to customize the application to suit the organization's needs without having to build and redeploy the application.

Code tables contain codes for items that appear in drop-down fields. Code tables are used to save space in the application database. By storing drop-down field selections as codes rather than the full text of the selection, a great deal of space can be saved in the database. For example, rather than storing the ethnicity 'American Indian' or 'Alaskan Native' in the database, the application can store the code 'ETH4'. Code tables also allows for the localization of drop-down fields. Localization allows drop-down fields to contain values appropriate to a user's language and dialect.

Locales identify a specific language and geographic region. The localization of the application is supported in a number of different languages. Each supported language is specified by a language-locale mapping. For example, English is mapped to the en locale.

Configuring Application Properties

The below sections describe how application properties can be configured. Application properties are variables that are used by the system in a variety of ways; for example, some properties alter the functionality provided by the system and therefore allow the system to be configured to suit an organization's needs. The values for these variables can be maintained at runtime, thereby providing a mechanism for changing functionality dynamically without the necessity of going through a full development cycle to implement changes. An example of such a variable is the property that denotes the default date format used by the application, *curam.misc.app.defaultdateformat*. The value of this property can be 'Date_mdy_ext' or it can be changed to 'Date_dmy_ext'.

Browsing for a Property

Properties can be browsed for and filtered by locale and category. Property categories are broadly divided between Application and Infrastructure categories. Categories group similar property types together in order to simplify the management of related property types. For example, one type of property category is 'Application - Address Settings'. This property category contains all the properties that relate to address settings in the application.

Adding a Property to the Application

Properties can be added to the application. Information maintained for each property includes the locale, current value, default value, and the category of the property. The default value specifies the value that the application property will be reset to if a user resets the property defaults for the application. The locale is primarily used to distinguish the language for the property description and display name, e.g., *en-US* (US English). The display name is the property name displayed to a user. For example, the e-mail server property that is used by the application would have a display name, *curam.notification.notificationemailserver*. The description provides more detailed information about the functionality of the property. Both the display name and the description should be written in the language described in the locale setting.

Creating a Property Description

Property descriptions are used to provide multi-language descriptions for application properties. A property description includes the locale, display name, and the description for the property. Using multi-language property descriptions ensures that users in multiple locales will be able to understand the application properties. Only one property description can be entered for each locale.

Publishing Property Changes

Changes made to properties are not propagated to the application until the changes are published. Application properties have a dynamic setting, which determines whether or not published changes to the application property will dynamically affect the system. If a property is defined as static, then published changes to this property will not take effect until the system is rebooted. The reason that static property changes do not take effect until reboot is that static properties contain information that cannot be updated while the application is running. An example of a static property would be 'curam.db.type=DB2', which indicates a connection to a DB2® database. This connection cannot be broken while the application is running. Thus, if the value is changed to 'curam.db.type=ORACLE', which indicates a connection to a Oracle database, this change cannot be implemented until the server is rebooted.

Resetting to Default Properties

Application properties can be reset to their default property values. Application properties that have a dynamic setting will be changed immediately. Static properties will be reset to their default value after the server is rebooted.

Configuring Code Tables

A code table is made up of a number of code table items; each code table item represents a selection in a drop-down field. The bulk of code table information is contained within code table items. A code table item contains the actual code that will be stored in the application database when that code table item is selected in a drop-down field in the runtime application. In addition, it contains a description, which is the text that will actually be displayed in a drop-down field, and the language setting, which contains information relating to the localization of the code table item.

A code table has a default code table item. This is the code table item that a drop-down field defaults to.

Adding a New Code Table to the Application

New code tables can be added to the application. A unique name must be entered. Once named, code table items can then be added to the table. The order in which the code table items are displayed can be specified. Code table items can be set as selectable. If the selectable indicator is set, the code table item will appear in the drop-down field populated by the parent code table. A language locale can also be set for the code table items.

Changes made to code tables or code table items are not propagated to the application's drop-down fields until the changes are published (or until the application server is rebooted).

Localizing Code Tables

Code table drop-down fields can be localized. Localized drop-down fields contain values appropriate to a user's language and country. The combination of language and country is known as a *locale*. Examples of locales include en-US (US English), en-GB (UK English), and es-US (US Spanish).

By using locales, the application can display different drop-down lists for users with different locales. For example, if a user's locale is set to US-Spanish, a drop-down field for the days of the week can display the values Lunes, Martes, Jueves, etc. Conversely, if another user's locale is set to US-English, then the same drop-down field can display the values Monday, Tuesday, Wednesday, etc.

There are two code table item settings that apply to drop-down field localization. The first is the description setting. This is the text that a user will actually see in a drop-down field. The second setting is language; this setting refers to the locale of the code table item. Note that although this setting is named language in order to make it more understandable, it actually refers to a locale, which contains both language and country information.

In multi-locale environments, a locale-specific version of each code table item should be recorded for all code tables. For example, imagine a code table that described the days of the week in an environment where both English and Spanish were used. This code table would require two code table items with a code value of DAY1 - an English code table item with a description of Monday, and a Spanish code table item with a description of Lunes. Having a language-specific code for each day of the week ensures that users are presented with all days of the week, no matter what language is displayed by the application.

Code Table Hierarchies

Code tables can be used to group other code tables in a hierarchy. Any number of code tables can be included in a code table hierarchy. A code table hierarchy allows the values available for selection in the drop-down field for one code table to be determined by the value selected in the drop-down field for another code table. For example, the values available when selecting the type of special caution to be recorded for a participant can be derived from the category of special caution selected. Code table hierarchies can be viewed and modified from the system administration application. For more information on code table hierarchies, please consult the *Server Developers Guide*.

Working with Configurable Validations

Validations are used throughout the application to maintain control over data that is entered by users, for example, to enforce data integrity or to prevent the entry of inconsistent data. An example of a validation is "The Case Participant Role start date must not be later than the Case end date - "%1d"." This is displayed when a user attempts to add a case member to a case with a start date later than the end date of the case.

Although all validations included in the application are run during processing by default, some validations are verified to not be required as they have no impact on system processing if they are not run. These validations are pre-defined as configurable validations and can be disabled by an agency.

The system administration application provides an agency with the ability to search for and disable validations that are identified as configurable validations. All other validations cannot be maintained within the application and are run during processing.

An organization might have a requirement to disable a validation that is not identified as a configurable validation. The organization can create a Product Enhancement Request (RFE) for the validation to be made configurable. The validation is then analyzed and it is determined whether it can be reclassified as a configurable validation.

For information about how to create a Product Enhancement Request (RFE), see the related link.

Related tasks

[Requesting a Product Enhancement \(RFE\)](#)

Enabling/Disabling Configurable Validations

Validations that are identified as configurable validations can be disabled or enabled by an administrator. The unique validation reference for a validation, e.g. `bpcaseparticipantrole.err_caseparticipantrole_xfv_from_date_caseheader_end_date|a|`, is used to search for and retrieve a validation. The validation reference for a validation can be identified using the Validation Messages HTML Documentation as described later in this section.

In order for the application to uniquely distinguish validations, the validation reference for each configurable validation is made up of a combination of the message catalog ID and an alphabetic constant that is used to distinguish between the validations used across different modules. In addition, the validation reference for any validations that use the same message text as other validations within a module will also have a numeric constant appended at the end. These constants are arbitrarily assigned with no specific meaning to the letters or numbers used.

The search available in the system administration application will only return validations that are identified as configurable validations and are an exact match to the unique validation reference entered.

An administrator can view information about whether the configurable validation is currently enabled or disabled, and can select to disable or enable the validation as required. Disabled validations will not be executed during system processing.

Administrators can also view a list of all validations that are currently disabled by leaving the search field blank.

The next section describes the documentation that the organization can use to identify what validations have been verified and identified as configurable validations. This documentation also provides the unique validation reference for each validation that can be used to search for the validation in the system administration application.

Using the Validations Messages Documentation

In order to determine what validations are identified as configurable validations that can be disabled, organizations can refer to the Validation Messages documentation that is provided in HTML format as part of the development installers. On running the development installer, this documentation is located in the top level 'Doc' folder in the installed code base. The home page of the Validation Messages documentation is accessed by selecting the `index.html` file located in the `ValidationMessages/html` folder.

A list of the validations referenced by each method of a facade class can be viewed by selecting the "Facades A-Z Index" link from the home page of the documentation, selecting a Facade Class Name, and then selecting a method.

A list of the validations referenced by each method of a screen can also be viewed by selecting the "Screens A-Z Index" link from the home page of the documentation, selecting a Screen Name, and then selecting a method.

For each validation, the validation reference and validation message text are provided. For example, for the validation used to prevent the start date of a case participant role from being later than the end date of a case, the validation reference of `BPCASEPARTICIPANTROLE.ERR_CASEPARTICIPANTROLE_XFV_FROM_DATE_CASEHEADER_END_DATE` and validation message text of "The Case Participant Role start date must not be later than the Case end date - "%1d"." are displayed.

For each validation, the documentation also displays whether or not the validation is configurable. All validations that are configurable have been verified to have no impact on system processing if they are not executed, and can be disabled by an agency. If the validation is not indicated to be a configurable validation, then the organization cannot disable the validation within the system administration application, and should raise a support case if there is a requirement to be able to disable the validation.

Additionally, a Configurable Messages section lists all validations that are identified as configurable validations, including the facades that the validations are referenced from. This section is accessed by selecting the "Configurable Messages A-Z Index" link from the home page of the documentation. The organization can use this list to determine the unique validation reference that is required to search for and disable the validation as part of system administration.

For example, if the organization wishes to identify the validations that exist and are configurable as part of the modify case member process, it can select the "Screens A-Z Index" option from the home page of the documentation, enter "modifyCaseMember" to filter the list of screens, select the "Case_modifyCaseMemberFromList.uim" screen, select the "Case.modifyCaseMember" facade reference, and then view a list of validations, including whether or not each validation is a configurable validation.

If the organization finds that the BPOCASEPARTICIPANTROLE.ERR_CASEPARTICIPANTROLE_XFV_FROM_DATE_CASEHEADER_FROM_DATE validation is configurable, it can then cross reference the validation with the list of configurable validations to identify the unique validation reference ID for the validation. This is done by using the Facade Reference and the Message Reference filters. If the "modifyCaseMember" method name is entered in the Facade Reference filter and the BPOCASEPARTICIPANTROLE.ERR_CASEPARTICIPANTROLE_XFV_FROM_DATE_CASEHEADER_FROM_DATE validation reference is entered in the Message Reference filter, a unique validation reference of `bpcaseparticipantrole.err_caseparticipantrole_xfv_from_date_caseheader_from_date|a|` is displayed for the validation.

If more than one unique validation reference is displayed for a particular Facade Reference and Message Reference combination, the organization can determine which unique validation reference to disable by enabling the display of the unique validation reference within the application alongside the validation message text that is displayed when business processing is executed that will result in the validation being called. This will in most cases result in the display of only one validation reference, which is the validation that should be disabled. In the unlikely event that more than one validation reference is displayed when the business process is executed, this represents a situation in which the same validation is being performed twice during the same business process and both should be disabled.

Note that all message catalog entries that are referenced by a method are displayed for each method in the Validation Messages documentation; therefore other types of messages such as infrastructure messages and messages used for logging may also be displayed along with the validation messages. In addition, for each method the documentation also displays all messages that are referenced by any method called by the method, and so in some cases, messages are listed that are not necessarily called by the screen that uses the method.

The following section provides more detail about how to enable the display of the unique validation reference ID within the application.

Enabling the Display of the Unique Validation Reference

Organizations can also identify the unique validation reference for a validation by enabling the application property `curam.validationmanager.displayreference.enabled`. This application property enables the display of the unique validation reference beside the validation message text displayed in the application. The unique validation reference can then be used to search for and disable or enable the validation if it is a configurable validation. Note that in certain circumstances, such as for validations that are not controlled by the validation manager, no validation reference will be displayed even when the application property is enabled. Any such validations are not configurable validations.

Configuring Language and Locale Mappings

Language and locale mappings are used to customize the user interface language. They are critical to many culturally and linguistically sensitive data operations, for example, locale information is used when generating pro forma communications.

Each language has a single locale associated with it. The choice of languages available for creating a new language locale mapping is itself populated from the list of languages that are available on the system.

Configuring Participant Nicknames for Search

A thesaurus of nicknames for an individual can be managed. The thesaurus allows the organization to define common nicknames that are associated with a name. For example, a person with the name "James" may also go by the name "Jimmy" or "Jamie". Defined nicknames can be used as search criteria when searching for a person and/or prospect person. The nickname search default setting is set in the administration application via property settings.

For more information on searching for persons by nickname, see the [Cúram Participant Guide](#).

(deprecated) Configuring Spell Checking



A spell checker can be configured for use within the rich text editor. The spell checking functionality is presented through a standard plug-in button in the rich text editor toolbar. The spell checker checks the contents of the text field for spelling errors.

Organizations can enable or disable the spell checker by using the application property `curam.spellcheck.enable`. If this application property is enabled, the spell checker button is displayed within the rich text editor toolbar. If it is disabled, the spell checker button is not displayed.

When spell checking is enabled, a user is able to select what spell checking dictionary to use from within the rich text editor toolbar. An agency can use the code table `SpellcheckLocale` to enable and disable the dictionaries that are presented to the user for selection. It can also be used to indicate what dictionary is presented as the default dictionary. The code table is populated with the locale values for the following dictionaries that are available:

- American English
- British English
- French
- Canadian French
- German
- Italian
- Spanish
- Portuguese

Case Audit Selection Query Configuration

Introduction

This chapter covers case audit configuration options available in the system administration application. Case audits are used to examine and evaluate cases. The random list of cases which are produced for a case audit are generated using selection queries. A selection query consists of an SQL statement and selection criteria used to validate the query and return information from the database. Two types of selection queries exist: fixed and dynamic.

A dynamic selection query is a flexible way to produce a list of cases for an audit. The coordinator can choose one or any combination of criteria to produce the list of cases. For example, the audit coordinator can select to generate a list of all cases with a status of open. Alternatively the audit coordinator could choose several criteria to produce the list of cases. For example, criteria of case start date and gender would return a more specific group of cases.

A fixed query is less flexible than a dynamic query, in that the values for the criteria form part of the query. An audit coordinator does not input the parameters for a fixed query. Fixed queries are reusable however, and are easier to run as no criteria selection by the audit coordinator is necessary. An example of a fixed query would be 'All open cases for males aged 18-35'.

For more information on selection queries, and case audits in general, please consult the [Cúram Case Audits Guide](#). For detailed information on selection queries and the SQL statements required to run a selection query, consult the [Case Audits Developer Guide](#).

Creating and Publishing a Dynamic Selection Query

New dynamic selection queries can be created by a database administrator or a system administrator. Once created, they are associated with a case audit configuration by an administrator. Development effort

is required to produce the new selection criteria page that an audit coordinator uses before the new selection query can be associated with a case audit configuration. A sample dynamic query is provided for each of the standard case types: integrated case, benefit product delivery, liability product delivery, and investigation case.

Page names for the manual search and random search pages are required when creating a dynamic selection query. These are the pages that the audit coordinator views when creating a list of cases for an audit. The system administrator must also enter the SQL statement for the selection query which will be used to query the database for the list of cases to be returned.

As part of creating a selection query, selection criteria are recorded to ensure the query is valid. The system administrator then publishes the selection query, making it available to be added to a case audit configuration by an administrator. This allows an audit coordinator to generate a list of cases for audit using the selection query. The selection criteria are used to return the list of cases.

When configuring a case audit, an administrator must associate one (and only one) predefined dynamic query with a case audit configuration.

Creating and Publishing a Fixed Selection Query

Fixed queries are used in conjunction with dynamic queries. If configured, an audit coordinator can choose which type of query to use when generating the list of cases for an audit. Page names do not have to be specified when creating fixed queries. This is because fixed queries are predefined. As such, audit coordinators are not required to enter any parameters for selection criteria that make up the query; therefore the pages to display the selection criteria are not required.

Otherwise, fixed queries are created similarly to dynamic queries, with an SQL statement that is validated using selection criteria. Once published, the administrator can then associate the fixed query with a case audit configuration. An audit coordinator can choose any fixed query that has been configured for a case audit. When run as part of the case list generation of an audit plan, this will return a list of cases for the audit coordinator in the runtime application.

An administrator can associate one or more fixed queries with a case audit configuration.

Communication Template Configuration

Introduction

This chapter provides an overview of the configuration options for communication templates. Two types of templates are supported: Microsoft Word and XSL templates. XSL templates are stylesheets used for generating pro forma communications; Microsoft Word templates are used to create Microsoft Word communications. XSL templates are used to generate bulk communications, while Microsoft Word templates are used to communicate more specific information to clients and participants, and can be edited individually according to the individual needs of the caseworker.

For more information on communications templates, please consult the *Cúram Communications Guide*.

Managing Microsoft Word Templates

Microsoft Word templates are basic document templates that allow a certain level of personalization for individual client communications.

Microsoft Word templates do not require any specialist technical knowledge and can be created in Microsoft Word. The template itself can be browsed for locally and uploaded. When uploading an Microsoft Word template, a name and template document ID for the template must be entered. A locale for the template must also be set. This allows the user to choose from the different templates based on the locale of the concerning participant when creating an Microsoft Word communication.

Fields are inserted into an Microsoft Word template so that data such as correspondent address information can be automatically populated into the Microsoft Word communication when it is created. Note that in order to populate the fields that are inserted into an Microsoft Word template with client data, some development effort is required. For more information on how to insert these fields, see [“Inserting Fields into a Microsoft Word Template”](#) on page 19.

Managing XSL Templates

XSL templates are used to generate pro forma documents and letters printed by the application using a combination of XML and XSL stylesheets.

XSL stylesheets are used to format the XML data for printing. XSL templates can be created using any XSL editor. XSL templates can then be uploaded and stored in the application database. When uploading an XSL template, a description and template ID for the template must be entered. A locale for the template must also be specified. This allows the user to choose from the different templates based on the locale of the concerning participant when creating a pro forma communication. Only one XSL template using the same template ID and locale may be created.

XSL templates can be checked out and downloaded. Checking out the template ensures that previous versions of the template are not lost. Templates can be checked out by more than one person at a time. System administrators can choose to ignore other checkouts on a template. Template version control ensures that templates are not accidentally overwritten. An XSL stylesheet developer is responsible for the creation and maintenance of XSL templates. When a new version is ready to be uploaded, the system administrator can choose to check in and upload the XML file.

For more information on XML and generating documents from XML and XSL templates, please consult the [Cúram XML Infrastructure Guide](#).

Assigning Communication Templates to Case and Participant Types

XSL and Microsoft Word templates can be assigned to a defined case or participant type. This is because certain templates may only be applicable for specific types of participants or cases. For example, an appeal decision template is only applicable to participants who are engaged in an appeal. Administrators can apply templates to specific case and participant types based on a category. For example, the category case includes a number of cases types, including income support, screening, etc. When configured, the template will only be available to the user when creating communications for the case type specified.

Fields are inserted into an Microsoft Word Template so that data such as address information can be populated into an Microsoft Word communication when it is created. Please refer to Appendix A for a description of how these fields are inserted.

Batch Processing Configuration

Introduction

This chapter provides an overview of batch processing configuration options. Batch processes are executables or 'mini-programs' that process a large number of records according to set parameters. Due to the potentially large processing nature of batch jobs, they are often scheduled by organizations for off-peak times, e.g., night-time, weekends, etc.

Adding a New Batch Process to the Application

New batch processes can be added to the application. Before the batch process can be utilized at runtime, the related process operation in the application model is made available as a batch process during development. This is done by assigning a batch stereotype to the process. When the model is generated, an SQL executable for the batch process is created. This executable can then be added to the application by a system administrator. A batch executable can be associated with a single batch process only. The system administrator selects the required batch process from the list of available batch processes. A name and description should be added, and the type specified. Batch processes can be of a

reporting batch type or an archiving batch type. The batch process type relates to a coded description of the batch process, which is used to group similar batch processes.

For more information on creating a new batch process, see the *Cúram Batch Processing Guide*.

Organizing Batch Processes into Groups

Batch process groups organize batch processes into logical groups. For example, financial batch processes can be grouped together so that users do not have to search the entire list of batch processes to find a set of financial batch processes to execute. Batch processes are grouped together simply by adding the batch processes to the same group and assigning a name for this batch process group. Batch process groups provide flexibility for an organization to manage and maintain their list of batch processes; batch processes can be grouped according to the needs of the organization.

Submitting a Batch Process for Execution

Batch processes can be submitted for execution by selecting to execute a batch process from the list of available batch processes. Depending upon the batch process, a number of parameters must be entered before the batch can be executed. The batch request will then be processed once the batch launcher is run. For details of the required parameters and running the batch launcher, consult the *Cúram Batch Processing Guide*.

The user can define the values for set parameters when submitting a batch process. This limits the information that will be processed. An example of a batch process is `DetermineProductDeliveryEligibility`. It is used to activate a large number of cases simultaneously and is therefore run as a batch process so as to defer this case processing to off-peak time and therefore minimize system impact. The batch process is configured to accept the parameter `product`. Setting the parameter to a specific product means that only cases of that product will be processed. Note that the values for some parameters must be set in order for a batch process to execute (setting other parameters is optional). A default value may also be set for a parameter. This will apply every time the batch process runs unless the user sets a different value.

The order in which batch processes are submitted must also be considered as some batch processes will not function unless others have been run previously. For example, `DetermineProductDeliveryEligibility` must be run before `GenerateInstructionLineItems` because instruction line items can only be generated for cases that have already been activated.

After the system administrator has submitted the batch processes, it is held in a batch queue until the batch launcher runs. The batch launcher is a separate program that executes the batch processes in the order in which they were submitted. Note that batch jobs can have a processing date specified. Typically the system date is used as the business processing date. Where the processing date is specified, this date overrides the system date.

Creating a Batch Process Error Code

Batch process error codes allow users to specify the error codes that will be returned by the application batch launcher in the event of a batch process failure. Information that is recorded for a batch error code includes the batch error code ID and the batch error code.

When a batch process fails, it will output an error message which is passed to the application batch launcher. The application batch launcher searches for a batch error code that matches the ID on the error message. If found, the application batch launcher then initiates the action that must be performed for the particular error. These actions are configured by an application developer.

For example, if the error code ID returned by a failed batch process is `CANNOT_CONNECT_TO_DATABASE`, then the batch launcher compares that to all batch process error codes that are stored in the system. If `CANNOT_CONNECT_TO_DATABASE` is found, then the batch launcher will retrieve the batch error code associated with this batch error code ID, e.g., "11". The batch launcher passes this batch error code to a task scheduler. The task scheduler then examines its own configuration files to determine what to do in the event of receiving error code 11. For information on the batch launcher and other aspects of batch process administration, see the *Cúram Batch Processing Guide*.

Security Configuration

Introduction

This chapter provides an overview of configuration options for security administration. At a high level, application security ensures that only valid users can access the application. It also defines specifically what a user can view and change in the application when logged in. Security administration is divided into two main categories: authentication and authorization. Authentication ensures that only valid users can access the application by requiring that the user provide a valid user name and password at login time. Authorization is used to secure functions in the Cúram application once a valid user has successfully logged in. Authorization defines a user's ability to perform actions and to access information within the Cúram application.

For more information on users, security roles, security groups, and the development implementation of security in the application, consult the Cúram Server Developers Guide.

Types of Security for Application Elements

A security identifier represents a protected resource. Every secured element in the application is given a SID that is unique across the entire application. They are used to secure administrative functions, fields on a screen, organization units, locations, case audits, and programs offered by the organization.

The most common type of security identifier is the functional SID, also known as a function identifier or FID. FIDs are used to secure business processes. An example of a function identifier is the FID assigned to the register person business process. Another type of security identifier is the field SID. Field security is used to secure specific information displayed in a field on an application page or set of pages. An example of a field SID is the SID used to secure the participant bank account balance field.

Securing Application Functions

Server functions are secured using FIDs. During application development, when a method is made publicly accessible; a unique security identifier is automatically generated for that function. In the deployed application, the methods contained in the model are generated as server functions. If security for a process method is switched off at design time in the model, a function identifier is still generated but will not be available to be used. Once functions are generated, FIDs can be created and added to the security hierarchy by a systems administrator by searching for a function and associating it with a FID. Only functions which are not already associated with a FID will be available for selection. Any changes made to a FID will only take effect when the changes are published.

Securing Application Fields

Field security governs the user's ability to view information in specific fields. Like functions, every field in the application can be secured using SIDs that a user must have in their security profile in order to view or access that field. During application development, developers create SIDs for fields that require security. By default, there is no security set on a field - it is up to a developer to indicate that a specific field requires a SID. The SID is then added to the database. This SID must still be added to the security hierarchy by a system administrator who also ensures that the SID is added to the appropriate user profiles. Any changes made to a SID will only take effect when the changes are published.

Securing Organization Units, Locations, and Programs

SIDS can be created by a system administrator to secure access to organization units, locations, case audits, and programs, including products and service plans. For example, a system administrator can create a SID of type product which can then be used by an administrator to secure read access to products of a particular type. Likewise, a system administrator can create a SID of type organization unit that can then be used by an administrator to control which users are able to maintain information about a particular organization unit. Any changes made to a SID will only take effect when the changes are published.

For more information on organization, location and product security, see the *Cúram Location Administration Guide*, the *Cúram Organization Administration Guide* and the *Cúram Integrated Case Management Guide*.

Grouping Related FIDs and SIDs together

A security group is the grouping of a set of related security identifiers. This level in the security hierarchy allows an administrator to group the large number of security identifiers into a smaller number of manageable groups. Any users who have a specific security group assigned to their security role will have access to all the resources represented by the security identifiers belonging to the security group. For example, users are authorized to register a person when their security role includes the security group which includes the register person security identifier.

User Security Profiles

User security profiles are defined by a hierarchy of security identifiers (SIDs). They apply to both internal and external users. SIDs are the building blocks of a user's security profile. They are used to secure administrative functions, fields on a screen, organization units, locations, case audits. They are also used to secure programs offered by the organization, including products and service plans.

The primary focus of the user security profile is to ensure that all users are authorized to access the information that they need to carry out their jobs in the organization while at the same time restricting those users from accessing secured information. The secondary focus of the user security profile is finding a way to best manage these profiles so that the work of the system administrator does not become a repetitive task.

Identifying Security Roles

The first step in creating user security profiles is to identify the organization's required roles. As an organization can be quite large, with many users, it makes sense to create security profiles for users who share the same security access. At the same time, it is important to distinguish the different skill levels between users with similar profiles. While both a trainee user and an advanced user both work with cases, there would be limitations to certain business operations that the trainee user could perform. For example, a trainee user is unlikely to conduct case reviews and would not be responsible for case approvals. Therefore, it is not only the primary functions that define a user role, but also the different levels of users. By organizing SIDs into a hierarchical structure, similar business processes that are shared between various users can easily be distributed without having to manually declare all of the secured elements for each user profile.

Limiting a Users Access to Application Elements Using Security Profiles

Security profiles apply to both internal and external users. By organizing SIDs into a hierarchical structure, similar business processes that are shared between various users can easily be distributed without having to manually declare all of the secured elements for each role. Each security role can be made up of any number of security groups, which in turn are made up of related security identifiers. Any changes made to a security role must be published before they can take effect.

Authorization evaluates a user's access to secured elements in the application based on his or her user security role. Every authorized user is assigned a security role; therefore, it is possible to authorize every user against any secured element of an application. External users are more restricted than internal users in what they can access.

Optimizing Authorization using the Security Cache

The security cache is an in-memory structure created to hold the security information associated with user roles. Security information is held in this cache to optimize the performance of the authorization process.

The cache is refreshed when the application reboots; it can be refreshed when a system administrator uses the cache refresh facility. The cache must be refreshed whenever any changes have been made to the user roles. This includes changes to security identifiers, security groups, and security roles. However,

the addition of a new user, if there are no other associated security changes (e.g. to roles or groups), does not require a security cache refresh.

Online Security Process Logging

To assist with troubleshooting and analysis, online security process details can be captured and written to the server log file. To enable this, set the application property `curam.online.data.based.security.logging` to `TRUE`. The default value of this property is `FALSE`, which prevents this information from being written.

Each log file message consists of the string `SECURITY_DIAG` followed by the below fields:

- `operationName` - The type of security check that has taken place.
- `loggedInUser` - The name of the logged in user.
- `concernRoleID` - The ID of the current concern role, if relevant.
- `caseID` - The ID of the current case, if relevant.
- `locationID` - The ID of the current location, if relevant.
- `securityCheckType` - The type of security check for a case that has taken place, if relevant. If applicable, a value of 1,2 or 3 can be outputted.
 - 1 = Maintenance Security Check for a case.
 - 2 = Approval Security Check for a case.
 - 3 = Read Security Check for a case.
- `locationAccessType` - The location access type, if relevant. If applicable, a value of LA1 and LA2 can be outputted.
 - LA1 = Maintain.
 - LA2 = Read.
- `caseSecurityCheckKey` - A key that is used to identify records that have been added to the cache. Can be used to add and retrieve records from the cache, if relevant.
- `cacheDetails` - A code depicting the current status of the cache.
- `securityPermDetails` - A code depicting the current security permissions.

Note: If a particular field is not relevant, for example if there is no `locationID` set for the current Note sensitivity security check, a value of N/A will be inserted in the log message.

There are a number of different codes that can be returned for the `cacheDetails` and the `securityPermDetails` fields. Each code gives information on the current status of the cache and security permissions respectively.

The following are the codes that can be inserted into the `cacheDetails` field:

- CD1 = cache on, cache hit, cache not updated.
- CD2 = cache on, cache not hit, cache updated.
- CD3 = cache on, cache hit, cache updated.
- CD4 = cache on, cache not hit, cache not updated.
- CD5 = cache disabled.
- N/A = not applicable.

The following are the codes that can be inserted into the `securityPermDetails` field:

- SPD1 = correct security permissions, not restricted, not readOnly.
- SPD2 = correct security permissions, not restricted, readOnly.
- SPD3 = correct security permissions, restricted, not readOnly.
- SPD4 = correct security permissions, restricted, readOnly.
- SPD5 = incorrect security permissions, restricted, readOnly.

- SPD6 = incorrect security permissions, restricted, not readOnly.
- SPD7 = incorrect security permissions, not restricted, readOnly.
- SPD8 = incorrect security permissions, not restricted, not readOnly.
- SPD9 = correct security permissions, not applicable, not applicable.
- SPD10 = incorrect security permissions, not applicable, not applicable.

Below are some sample messages from the server log:

- SECURITY_DIAG Concern Sensitivity and Location Security Check,superuser,101,N/A,N/A,N/A,LA2,101LA2superuser,CD2,SPD1
- SECURITY_DIAG Note Sensitivity Check,superuser,N/A,-7619527619557457920,N/A,N/A,N/A,N/A,CD5,SPD9
- SECURITY_DIAG Case Security and Location Security Check,superuser,101,-3007841601130070016,N/A,3,LA2,superuser101-30078416011300700163superuser,CD1,SPD5
- SECURITY_DIAG Case Creation Security Check,superuser,101,2501,N/A,N/A,N/A,N/A,CD4,SPD9

The above messages indicate the following events respectively:

- A concern sensitivity and location security check has occurred. The logged in user is superuser and the concern role ID is 101. There are no relevant caseID, locationID or securityCheckType. The locationAccessType is a read. The caseSecurityCheckKey is 101LA2superuser. The cache is enabled, but the result was not retrieved from the cache. The newly retrieved security result was then added to the cache. The user has the correct security permissions and the related case or participant is not restricted or read only.
- A Note sensitivity check has occurred. The logged in user is superuser. There is no relevant concern role ID. The note's ID is -7619527619557457920 and there is no relevant locationID, securityCheckType, locationAccessType and caseSecurityCheckKey. The cache is disabled and there was no cache hit and the cache was not updated. The user has the correct security permissions and it is not relevant whether the related case or participant is restricted or read only.
- A case security and location security check has occurred. The logged in user is superuser and the concern role ID is 101. The caseID is -3007841601130070016 and there is no relevant locationID. The securityCheckType is a read security check. The locationAccessType is a read and the caseSecurityCheckKey is superuser101-30078416011300700163superuser. The cache is enabled and the security result was retrieved from the cache. The cache was not updated. The user does not have the correct security permissions, and the related case or participant is restricted and read only.
- A case creation security check has occurred. The logged in user is superuser and the concern role ID is 101. The caseID is 2501 and there is no relevant locationID, securityCheckType, locationAccessType and caseSecurityCheckKey. The cache is enabled but there was no cache hit, and the cache was not updated. The user has the correct security permissions and it is not relevant whether the related case or participant is restricted or read only

Business Intelligence Configuration

Introduction

This chapter provides an overview of the system configuration options for Business Intelligence (BI) in the application. BI provides decision support information for case workers, supervisors, and senior managers in the organization. The information needs for each role is different and these are reflected in the BI tools that are available to each role.

BI consists of embedded analytics, and interactive dashboards and reports. Embedded analytics are used to represent data that has been pulled from the database and displayed to the user. Interactive dashboards are used to publish graphically intuitive displays of information, including dials, gauges, and traffic lights style graphs. These displays indicate the state of the performance metric compared with a

goal or target value. Report functionality is used to create formatted and interactive reports with highly scalable distribution and scheduling capabilities.

The Business Intelligence and Reporting Tools (BIRT) Report Designer is an Eclipse plug-in which allows developers to create custom BI reports which can then be imported into the application. A wide range of graphs using the BIRT charting engine is supported, as well as data listing. BI content can be displayed in the application pages when required, a licensable BI dashboard is also available to be used. Once the reports have been created and are available on the system, they can be displayed in the runtime application using the BIRT Report Engine which renders the report design. It can produce output in a number of formats including HTML and PDF. The aggregated data of a BI report is displayed in such a way that the user can interact with it.

For more information on building and deploying BI reports, please consult the *Cúram BIRT Developers Guide*.

Configuring Business Intelligence Reports

BI comes with a number of preconfigured sample reports. These sample reports demonstrate the infrastructure of the reports and how they can be used to read and deploy information held in the database. Development work is required to make the reports available in the runtime application. Once development is complete, the reports should be copied to the BI content directory on the application server.

BIRT report options that can be configured via the system administration application include:

This table describes BIRT report options

Report Configuration Option	Description
Report name	This is the display name of the report. Reports can have a number of different configurations with different display names.
Report file name	The actual file path to the report design on the report server.
Report category	The category is used to classify the report for search purposes, for example, case, participant, etc.
Report servlet	This is the report servlet used to render the report.
Parameters	These refer to the frame that contains the chart rather than the chart itself. The width and height can be set, as well as whether scrolling is allowed for the report or not. A border can also be optionally displayed around the report.

Additional parameters can be added to BI reports. These can be from a set recognized by BIRT or business specific parameters that the report can be programmed to handle.

Configuring the Business Intelligence Report Viewer

A BIRT viewer is available for use to display BI reports in the runtime application. Options that can be configured for the viewer include:

This table describes BIRT viewer options

Report Viewer Configuration Option	Description
Viewer name	The name of the report viewer engine.
Servlet	The report viewer configuration servlet used to render the reports. Each report configuration can have its own servlet setting. The default option is 'run'.
Report name parameter	This specifies the report viewer type, The default is '__report'.
Context	This defines how the URL for the report is built.
Root	Specifies the report root configuration string. This can include the http specifier, server name, and port the report server is running on.

Additional default parameters can be added to the report viewer but these need to be from a set that BIRT recognizes in order to be used by the report viewer.

Target System Configuration

Introduction

This chapter covers the basic configuration options available in the system administration application for target systems. A customer may have several different system installations within their environment. The application supports several services that can communicate and interact with these other systems. When using one of these services, the system initiating the interaction is known as the source system, and the system(s) with which it is communicating are known as target systems.

For example, a customer might setup two separate installations of the application for two distinct business operations. The customer may then wish to share evidence data between the two systems in order to improve their operating efficiency. In this scenario the two systems can be set up to communicate with one another and use the Cúram Evidence Broker to share evidence.

Creating a Target System

In order for the services on a source system to communicate with target systems, the system administrator first needs to setup and configure the details for the target system on the source system. This is achieved via a dedicated target system configuration tab in the system administrator workspace.

Adding a Service to a Target System

There can be multiple services associated with a target system. In the example described in the introduction, the Evidence Broker is the service being used. A URL must be defined for every service that is associated with a target system. The URL identifies and interacts with the service in the target system. The URL is generated by combining the root URL of the target system, consisting of the system hostname and port, and the extension URL for the associated service. For example, a URL `http://shell.example.com:9082/<servername>/services/EvidenceBroker` can be generated for the Evidence Broker service in a target system by combining the root URL (`http://shell.example.com:9082/`) of the target system and the extension URL (`<servername>/services/EvidenceBroker`) of the associated Evidence Broker service.

When adding a service to a target system, users can optionally specify a user name and password. The password is encrypted by Curam encryption algorithms during the creation of the target system service. The plain text password is never stored and Curam only ever compares the encrypted values for

authentication. Subsequent editing of the service does not retrieve the password. If you edit the service, it displays an empty box for the **Invoking User Password** field. For more information, see *Cryptography in Curam*.

Cúram Configuration Transport Manager (CTM) is another example where target systems are used. In CTM a target system is used to support the automatic transport of configuration data between source and target systems. When defining the target system for CTM, the Configuration Transport Manager service is used.

Related concepts

[Cryptography in Cúram](#)

Content Management Interoperability Services Configuration

Introduction

This chapter describes the configuration options specific to the integration of the application with a content management system. This includes the application properties that are used to enable integration and the configuration of metadata information.

When the application is enabled for integration with a content management system, the documents associated with attachments and communications are stored in and retrieved from the content management system. Metadata information about the document, such as document type, can be stored with the attachment documents in the content management system.

For more information on how the application can be integrated with a content management system, see the Cúram Content Management Interoperability Services Integration Guide.

The next section describes how integration with a content management system is enabled.

Enabling Integration with a Content Management System

Integration with a content management system is enabled through the use of a group of application properties located under the 'Application - Content Management settings' category. Three application properties are available to control the level of integration with a content management system.

The `curam.cms.enable` application property is used to specify whether or not the storage location for certain files should be in the configured content management system instead of the application database. When the application property is enabled, two additional properties can then be used to control what files should be stored in the content management system. The `curam.cms.attachment.enable` property is used to specify whether files categorized as attachments should be stored in the content management system. This includes attachments that are associated with recorded communications and Microsoft Word communications. The `curam.cms.proforma.enable` property is used to specify whether the files categorized as Pro Forma communications should be stored in the content management system. This includes all files associated with Pro Forma communications with the exception of those Pro Forma communications created as a result of batch processing.

The next section describes the configuration options available for the storage of metadata information.

Configuring Metadata for Attachments

When documents associated with attachments are created within the application and stored in the content management system, metadata information about the document can also be stored. This includes attachment documents associated with recorded communications and Microsoft Word communications.

The metadata information that can be stored about the document depends upon the context in which the attachment was created, for example, if an attachment is created in the context of a case, information about the case in which the attachment was created can be stored with the document; however, if the

attachment is created in the context of a participant, no case information can be stored, but participant information may be stored instead.

If information about a document, such as the document receipt date, is subsequently modified in the application, the related metadata information may also be updated. Whether or not an update can be made to a particular metadata element will again depend upon the specific context that resulted in an update to the attachment.

The application can be configured to store a number of pre-defined metadata elements with the attachment document. By default, each metadata element is enabled, and can be individually disabled so that the information will not be stored along with the document in the content management system.

Enabling or disabling a metadata element will not affect any metadata that was previously stored in the content management system. Any changes to the OOTB metadata configuration will only come into effect when future attachments are created or when existing attachments are updated by either preventing metadata elements from being stored or allowing additional metadata elements to be stored, depending on the configuration setting.

Each metadata element has a display name and a description that are displayed to the administrator. Multiple display names and descriptions can be created for each metadata element to provide multi-language support and the existing descriptions and display names can be modified if required.

The following metadata elements are available for documents categorized as attachments, including attachments associated with recorded communications and Microsoft Word communications:

This table shows the metadata elements available for attachments.

Metadata Element	Description
Case Reference	The case reference number
Participant Reference	The participant reference number
Participant First Name	The first name of a person or a prospect person
Participant Last Name	The last name of a person or a prospect person
Participant Date of Birth	The date of birth of a person or a prospect person
Document Type	The type of document
Document Type Code	The code for the document type
Document Receipt Date	The date on which the document was received
Communication Date	The communication date for attachments associated with recorded communications

An organization may also choose to implement additional metadata elements to meet their business requirements. For more information on how additional metadata elements are supported, see the *Cúram Content Management Interoperability Services Integration Guide*.

Conclusion

Summary

The following is a summary of the main concepts covered in this guide:

- Application security is configured in the system administration application. Security profiles govern how users interact with the application and are built upon security identifiers which are used to secure functions, fields, products, appeals, and service plans.

- Two types of communication template are supported and are configurable in the system administration application: Microsoft Word and XSL templates.
- Case audit selection queries are configured via system administration.
- Various configuration options for batch processes are available in the system administration application. Batch processes are used to process a large number of records according to set parameters.
- Business intelligence report and viewer configuration options are available in the system administration application.
- The use of target systems which allows data to be shared between different systems is supported.
- Configuration options for integration with a content management system are available in the system administration application.

Additional Information

Additional information on the topics covered in this guide are covered in several related documents:

Cúram Organization Administration Guide

This guide covers the basic concepts of organization administration functionality.

Cúram Location Administration Guide

This guide covers the basic concepts of organization location administration functionality.

Cúram Participant Guide

This guide covers the basic concepts of participant functionality.

Cúram Integrated Case Management Guide

This guide covers the basic concepts of case processing.

Cúram Communications Guide

This guide provides an overview of communication functionality.

Cúram Case Audits Business Guide

This guide provides a business overview of case audits.

Technical Information

The following is a list of technical documents referenced in this guide:

Cúram Server Developer Guide

This guide provides technical information on the following areas: application properties, security, and code tables.

Cúram Batch Processing Guide

This guide provides information on batch process development.

Cúram Operations Guide

This guide provides an overview of operations including application properties.

Cúram BIRT Developers Guide

This guide details development required for business intelligence.

Cúram Case Audits Developers Guide

This guide covers case audits development.

Cúram XML Infrastructure Guide

This document presents all aspects of the XML functionality provided with the Server Development Environment (SDEJ), from modelling to development to run-time management.

Cúram Content Management Interoperability Services Integration Guide

This guide outlines the available configuration options for integrating Cúram with a content management system.

Inserting Fields into a Microsoft Word Template

Introduction

This appendix provides instructions on how to create a Microsoft Word template that contains fields to hold variable data, such as a correspondent's address. This appendix also provides instructions on how to write the server code that populates the template with the variable data when a communication that is based on the template is created by a user.

Creating a Microsoft Word Template

About this task

A Microsoft Word template is created using the Microsoft Word application. Therefore, Microsoft Word template creation requires Microsoft Word application knowledge. As part of Microsoft Word template creation, fields are inserted into the template as placeholders that will be replaced with variable data that is specific to a correspondent, when the communication is created.

Note, that once a Microsoft Word communication is created from a template, the correspondent-specific data becomes part of the communication itself. For example, if the Microsoft Word template includes variables for the correspondent's name and address, the correspondent's actual name and address, rather than the variables, are stored as the communication text.

To insert a field in a Microsoft Word template that will be replaced with the variable data returned from the server, complete the following steps:

Procedure

1. Open a new Microsoft Word document. Note that this document must be opened independent of the application, i.e., locally.
2. Create new custom Document Properties fields (As per “Sample Microsoft Word Template Content” on [page 21](#) below, create the following fields: AddressLine1, AddressLine2, AddressLine3, personName, userName).
3. Insert the field you created into the template as follows:
 - a) Click where you want to insert a field.
 - b) On the Insert tab, in the Text group, click Quick Parts, and then click Field.
 - c) In the Categories list, select the Document Information category
 - d) In the Field names list, select DocProperty and select the field you created from the Field Property list.

Results

When the placeholders for the variable data have been inserted as fields in the Microsoft Word template, the content of the body of the communication that will remain the same for all communications that are generated using this template can be added to the document directly. The file is then saved as a normal Microsoft Word document and can be browsed for and uploaded in the standard fashion using the application.

For more information on browsing and uploading Microsoft Word templates, see “[Managing Microsoft Word Templates](#)” on [page 8](#).

Writing Server Code to Populate Communication Data

Once the template has been created and is available within the application, the user can select the template when creating a communication. The user then enters all the other details required to create the communication such as the correspondent name, communication name etc.

When the communication is opened, the server action that is called returns the variable data to be inserted into the fields in the Microsoft Word template, i.e., the actual name and address of the correspondent is returned.

The server action returns the variable data as an object of data type blob. The object is made up of name-value pairs with the "name" being the fields in the Microsoft Word template and the "value" being the data to be inserted for the field when the document is created.

Sample Microsoft Word Template Content

This is an example of Microsoft Word template. The AddressLine1, AddressLine2, AddressLine3, personName and userName are the custom document properties that will be replaced with the correspondent-specific data that is retrieved from the server and hence will vary in each communication. However the content in the body of the template will remain the same for all the communications created using this template.

```
{ DOCPROPERTY AddressLine1 }  
{ DOCPROPERTY AddressLine2 }  
{ DOCPROPERTY AddressLine3 }
```

Dear { DOCPROPERTY personName }

This is just an example of Microsoft Word Template.

Thanks,
{ DOCPROPERTY userName }

Figure 1: Sample Microsoft Word Template Content

Sample Code to Return Data to Populate a Microsoft Word Communication

The following sample code snippet illustrates how to write the code to build the values into a blob object and return it in order to insert the values into the Microsoft Word document. Note that values are inserted in the form of name-value pairs using the org.jdom.Element.

The NAME attribute in the name-value pair is the name of the DocProperty inserted in the template. The VALUE attribute is the correspondent-specific data that will replace the field in the created Microsoft Word communication.

```

org.jdom.Element rootElement = new org.jdom.Element("ROOT");
org.jdom.Element fieldsElement = new org.jdom.Element ("FIELDS");

org.jdom.Element fieldElement = new org.jdom.Element ("FIELD");
fieldElement.setAttribute ("NAME", "personName");
fieldElement.setAttribute ("VALUE", "James Smith");
fieldsElement.addContent (fieldElement);

org.jdom.Element fieldElement1 = new org.jdom.Element ("FIELD");
fieldElement1.setAttribute ("NAME", "AddressLine1");
fieldElement1.setAttribute ("VALUE", "1074, Park Terrace");
fieldsElement.addContent (fieldElement1);

org.jdom.Element fieldElement2 = new org.jdom.Element ("FIELD");
fieldElement2.setAttribute ("NAME", "AddressLine2");
fieldElement2.setAttribute ("VALUE", "Fairfield, Midway");
fieldsElement.addContent(fieldElement2);

org.jdom.Element fieldElement3 = new org.jdom.Element ("FIELD");
fieldElement3.setAttribute ("NAME", "AddressLine3");
fieldElement3.setAttribute ("VALUE", "UTAH");
fieldsElement.addContent (fieldElement3);

org.jdom.Element fieldElement4 = new org.jdom.Element ("FIELD");
fieldElement4.setAttribute ("NAME", "userName");
fieldElement4.setAttribute ("VALUE", "Caseworker");
fieldsElement.addContent (fieldElement4);

rootElement.addContent (fieldsElement);

return new curam.util.type.Blob (
    new org.jdom.output.XMLOutputter
        .outputString(rootElement).getBytes());

```

Figure 2: Sample Code to Return Data to Populate a Microsoft Word Communication

For more information on how to write server code, see the [Cúram Server Developer Guide](#).

Structure of the Returned Object from Code Sample

The following example illustrates the structure of the returned object that is built by the system as binary name-value pairs from the above code snippet:

```

<ROOT>
<FIELDS>
<FIELD NAME= "personName", VALUE="James Smith" />
<FIELD NAME= "AddressLine1", VALUE= "1074, Park Terrace"/>
<FIELD NAME= "AddressLine2", VALUE= "Fairfield, Midway" />
<FIELD NAME= "AddressLine3", VALUE="UTAH" />
<FIELD NAME= "userName", VALUE="Caseworker" />
</FIELDS>
</ROOT>

```

Figure 3: Structure of the Returned Object from Code Sample

(deprecated) Configuring FILE_EDIT widget with Google Chrome

IBM Cúram's FILE_EDIT widget was currently implemented as a Java applet which works with all supported browsers. In September 2015, Google Chrome dropped support for Java with version 45. This effectively means that as of version 45 the FILE_EDIT widget will cease to function in Google Chrome. IBM Cúram have provided an alternative, Google Chrome-specific, implementation to allow customers to continue to use functionality. This implementation is based around the *extension* and *native messaging*

features of Google Chrome. There is no additional server configuration required to use the Google Chrome-specific solution. When the IBM Cúram application detects that the browser being used is Google Chrome then the Google Chrome-specific implementation will be invoked. However, there are two additional software installations required on all end-user's machines:

1. Google Chrome extension

The Google Chrome extension can be downloaded from the Chrome web store. All users wishing to use the FILE_EDIT widget must have this extension installed.

2. Host application

The host application is delivered as a Microsoft installer application, IBMCuramWordIntegrationAssistant.msi. The installer file can be found in the following installation location: CuramCDEJ\lib\curam\installers\.

All users wishing to use the FILE_EDIT widget must have this extension installed. Once the installation has completed successfully a new registry key will have been created: Computer \HKEY_CURRENT_USER\Software\Google\Chrome\NativeMessagingHosts \curam.fileedit.chrome.nativebridge.

The value of this new key will be the qualified path to a .json file e.g. C:\Users\user\IBM\WordBridge \hostmanifest.json. The contents of this .json file will be similar to:

```
{
  "name": "curam.fileedit.chrome.nativebridge",
  "description": "Word Integration Bridge App",
  "path": "launchbridge.bat",
  "type": "stdio",
  "allowed_origins": [
    "chrome-extension://xxxxxxxxxxxxxxxxxxxxxxxx/"
  ]
}
```

launchbridge.bat is the file used to start the host application. The directory containing hostmanifest.json will also contain all of the .jar files required by the host application.

Customer's wishing to use the Google Chrome-specific solution should carefully consider the implication of these additional software installations on their deployment plans.

The end-user experience is the same for those using the applet or the Google Chrome implementations. It is possible to configure the IBM Cúram application to invoke the applet based implementation of this feature with Google Chrome. This can be done by setting the value of the fileedit.chrome.messaging.enabled property, found in ApplicationConfiguration.properties, to false.

Please note that this configuration option is only applicable for IBM Cúram installations where all of the end-users are using a version of Google Chrome less than 45.

Additionally, for Google Chrome versions 42 to 44 (inclusive) NP-API is disabled by default and must be manually re-enabled.

Notices

This information was developed for products and services offered in the United States.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing Legal and Intellectual Property Law IBM Japan Ltd. 19-21, Nihonbashi-Hakozakicho, Chuo-ku Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Privacy Policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies or other similar technologies that collect each user's name, user name, password, and/or other personally identifiable information for purposes of session management, authentication, enhanced user usability, single sign-on configuration and/or other usage tracking and/or functional purposes. These cookies or other similar technologies cannot be disabled.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy> and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other names may be trademarks of their respective owners. Other company, product, and service names may be trademarks or service marks of others.



Part Number:

(1P) P/N: