CA Identity Governance

Programming Guide

12.6.02a

CA Technologies Product References

- This document references the following CA Technologies products:
- CA Identity Governance
- CA Identity Manager
- CA Single Sign On
- CA User Activity Reporting
- CA SDM
- CA IAM Connector Server

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: CA Identity Governance Web Services

This section contains the following topics:

Web Services Overview

CA Identity Governance exposes web services that support a range of queries and interactions, including the following:

- **Data update and exchange**—endpoints can submit provisioning changes directly to a CA Identity Governance configuration, and receive provisioning updates from CA Identity Governance.
- **What-if queries**—provisioning platforms can use CA Identity Governance analytical tools to test proposed provisioning changes for compliance or pattern issues.
- **Service extension and integration**—external applications or resources can be integrated with CA Identity Governance workflows. For example, certification and other processes can interface with SMS or other messaging platforms to send users task or status notifications.

Javadoc of the public classes, methods, and services is contained in the **CA-RCM-*rel#*-Language-Files.zip** file of the CA Identity Governance installation package.

Available Web Services

The following web services are available with CA Identity Governance:

CertificationService

Performs all read and write operations on certifications. This includes certification lifecycle operations, queries on certification tasks, and actions on certification tasks.

DeltaBPRService

Tests Out of Compliance privileges in 'what-if' scenarios, using BPR definitions.

PatternAuditService

Tests Out of Pattern privileges.

SageBrowsingService

Reads users, roles, resources, and queries from CA Identity Governance. This service also supports link attributes.

SageModificationsService

Provides create, update, and delete operations on users, roles, and resources within CA Identity Governance. This service also supports link attributes.

SuggestService

Suggests additional roles and privileges using pattern-based audit.

TMSService

Provides web services to Workpoint processes. Any web service called from a WorkPoint process must implement this interface.

TranslationService

Exposes the mapping fields between CA Identity Manager and CA Identity Governance for users and roles.

Chapter 2: Model Event Notification API

This section contains the following topics:

Model Event Notification Overview

The Model Event Notification API provides a JMS topic for all model changes. You can then write custom code on external clients that handles this model event information.

How to Use the Model Event Notification API

To capture JMS topic information for model events, configure model event notifications.

Follow these steps:

1. Log in to the CA Identity Governance Portal as an administrator.
2. Go to Administration, Settings, Property Settings.
3. Set the following properties:

   modelEvent.producer.fireEvent

   Specifies what events trigger change notification. Valid values include:

   Atomic

   Triggers an event after every approval during an update request.

   Aggregate

   Triggers an event when all approvals complete for an update request.

   modelEvent.notify.topic

   Determines if you enable a Java Messaging Service (JMS) topic for model event notification.

   approval.isModelChangeNotificationOn

   Indicates whether you enable a model event notification. This parameter signals the availability of the Model Event Notification API to CA Identity Governance Workpoint processes that contain the Notify Model Change and Notify Aggregated Model Change building blocks.

4. Be sure that any 3rd-party clients listening to this JMS topic implement the ModelEventHandler interface.

Members in the Model Event Notification Message

The following members make up the model event notification message:

Member	Description
createDate	The time at which the event is triggered
requester	Who requested the update
universe	The universe where the update occurs
Source	The origin of the event
requestVOs	The entities that are updated
TicketId	The ticketId that handles the request

Chapter 3: Executing Batch (SBT) Files

This section contains the following topics:

Batch Processing Overview

When importing a large volume of data (which could involve several gigabytes in several source files), processing can take many hours. Therefore, a batch processing option enables processing during off-hours in accordance with a predetermined order defined by the Role Engineer.

The following list shows the typical order of commands in a batch file:

1. IMPORT RACF
2. IMPORT CSV
3. MERGE CFG
4. ENRICH UDB
5. ENRICH RDB
6. FILTER CFG

CA Identity Governance batch files adhere to standard XML file format. The file extension must be .sbt.

To run the batch file from a Microsoft Windows command prompt, execute the file using the following format:

`EurekifySageDNA-V32.exe` *`SBT_file_name`*

For example, EurekifySageDNA-V32.exe merge.sbt.

Alternatively, you can run the batch file by selecting Execute Batch File from the File menu.

How to Run Batch Files

You can submit batch files for processing in one of the following ways:

- Use the File menu commands of the client tools.
- Define a batch (SBT) connector job in the CA Identity Governance Portal.

When you define a batch connector job in the CA Identity Governance Portal, the batch file you specify must contain the following attribute:

USE_CONFIG

Defines the destination configuration file for batch file operations. Typically this value is the same as the DEST_FILE attribute.

Specify this attribute in addition to the DEST_FILE attribute in batch files that are submitted for processing through the CA Identity Governance Portal.

Example: Batch File Processed by the CA Identity Governance Portal

The batch file in this example can be specified in a batch connector job in the CA Identity Governance Portal because it includes the USE_CONFIG attribute.

```
<BATCH>
<COMMAND CONTINUE_ON_FAIL="true"
    ACTION="IMPORT SQL"
    SERVER="MSSQL_hostname"
    DATABASE="eurekify_sdb"
    SRC_FILE="SQL://BaseConfigModel.cfg"
    DEST_FILE="OUTPUT/BaseConfigModel.cfg"/>
<USE_CONFIG config="OUTPUT/BaseConfigModel.cfg"/>
</BATCH>
```

The CA Identity Governance Batch (SBT) Format

The following table details the format used when executing a batch file in CA Identity Governance.

Command	Attributes/Child Tags	Description	Status	Default
IMPORT RACF	CONFIG	Output CFG file path	Mandatory	
	USERS_DB	Output UDB file path	Mandatory	
	RES_DB	Output UDB file path	Mandatory	
	INPUT	Input RACF file path	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
	SUPP_HR	Input Supplementary HR file path	Optional	
	RACF_SYS_NAME	RACF system name	Optional	
	UACC	'UACC' flag	Optional	
	GROUPS_AS_ ROLES	'Import all groups as roles' flag	Optional	
	ADD_ACL_ ENTITIES	'Add ACL entities' flag	Optional	
	IGNORE_ REVOKED_USERS	'Ignore revoked users' flag	Optional	
IMPORT TSS	CONFIG	Output CFG file path	Mandatory	
	USERS_DB	Output UDB file path	Mandatory	
	RES_DB	Output UDB file path	Mandatory	
	TSSLISTFILE	Input TSS LIST file path	Mandatory	
	SUPP_HR	Input Supplementary HR file path	Optional	
	TSS_SYS_NAME	TSS system name	Optional	
	ADD_ACL_ ENTITIES	'Add ACL entities' flag	Optional	
	GROUPS_AS_ ROLES	'Import all groups as roles' flag	Optional	
IMPORT CSV	CONFIG	Output CFG file path	Mandatory	
	USERS_DB	Output UDB file path	Mandatory	
	RES_DB	Output UDB file path	Mandatory	
	ROLE	Input Roles CSV file path	Optional	
	USER_RES	Input User-Resource links CSV file path	Optional	
	USER_ROLE	Input User-Role links CSV file path	Optional	
	ROLE_RES	Input Role-Resource links CSV file path	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	ROLE_ROLE	Input Role-Role links CSV file path	Optional	
	USER_RES_ATTR	Input User-Resource link attributes CSV file path	Optional	
	USER_ROLE_ATTR	Input User-Role link attributes CSV file path	Optional	
	ROLE_RES_ATTR	Input Role-Resource link attributes CSV file path	Optional	
	ROLE_ROLE_ATTR	Input Role-Role link attributes CSV file path	Optional	
	SEPARATOR	CSV separator character	Optional	
	ROLEID_AS_NUM	'Role ID as number' flag: Used for backward compatibility; Obsolete	Optional	
	INPUT_UNICODE	Treats all input files as unicode. If a file does not have a unicode signature in the beginning (BOM: 0xFEFF), it is added to the file, and the original file is saved with a .backup extension.	Optional	False
EXPORT CSV	CONFIG	Input CFG file path	Mandatory	
	ROLE	Output Roles CSV file path	Mandatory	
	USER_RES	Output User-Resource links CSV file path	Mandatory	
	USER_ROLE	Output User-Role links CSV file path	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
	ROLE_RES	Output Role-Resource links CSV file path	Mandatory	
	ROLE_ROLE	Output Role-Role links CSV file path	Mandatory	
	USER_RES_ATTR	Output User-Resource link attributes CSV file path	Optional	
	USER_ROLE_ATTR	Output User-Role link attributes CSV file path	Optional	
	ROLE_RES_ATTR	Output Role-Resource link attributes CSV input file path	Optional	
	ROLE_ROLE_ATTR	Output Role-Role link attributes CSV file path	Optional	
	SEPARATOR	CSV separator character	Optional	
	ROLEID_AS_NUM	'Role ID as number' flag: Used for backward compatibility; Obsolete	Optional	
	UNICODE_OUTPUT	'Unicode output' flag	Optional	
MERGE CFG	FIRST_CFG	Input CFG file path	Optional	
	SECOND_CFG	Input CFG file path	Optional	
	TARGET_CFG	Output CFG file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
	CASE_SENSITIVE	'Case sensitive' flag	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	<SOURCE> (CFG path) </SOURCE>	Zero or more child tags. The tag's internal text contains an additional configuration path to merge. If FIRST_CFG and SECOND_CFG are absent at least 2 <SOURCE> tags should be present.	Optional	
MERGE UDB	FIRST_UDB	Input UDB file path	Mandatory	
	SECOND_UDB	Input UDB file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	CASE_SENSITIVE	'Case sensitive' flag	Optional	
MERGE RDB	FIRST_RDB	Input RDB file path	Mandatory	
	SECOND_RDB	Input RDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
MERGE AUDIT	TARGET_AUD	Output Audit card file path	Mandatory	
	MASTER_AUD	Input Primary Audit card file path	Mandatory	
	ADDED_AUD	Input Secondary Audit card file path	Mandatory	
TRIM CFGDB	INPUT_CFG	Input CFG file path	Mandatory	
	TARGET_CFG	Output CFG file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
FILTER CFG	SOURCE_CFG	Input CFG file path	Mandatory	
	TARGET_CFG	Output CFG file path	Mandatory	
	REMOVE_ONLY	'Remove Only' flag	Optional	
	<CONDITION>	1 or more child tags.		
	ENTITY	USER | ROLE | RESOURCE	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
	FIELD	If 'ENTITY' is USER: PERSON ID USER NAME ORGANIZATION ORGANIZATION TYPE (field name) If 'ENTITY' is ROLE: ROLE NAME DESCRIPTION ORGANIZATION OWNER TYPE CREATE DATE REVIEWER APPROVAL STATUS APPROVAL DATE FILTER ORGANIZATION 2 ORGANIZATION 3 If 'ENTITY' is RESOURCE: RES NAME 1/2/3 (field name) For all relevant entity types: DIRECT RESOURCES TOTAL RESOURCES DIRECT USERS TOTAL USERS DIRECT ROLES TOTAL ROLES CHILD ROLES PARENT ROLES	Mandatory	
	FROM	Minimum value	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	TOP	Maximum value	Optional	
	TYPE	REG_EXP | PATTERN | (none)	Optional	
	PATTERN	A text sample (exact string / pattern / regular expression)	Optional	
	</CONDITION>	Either FROM and TO or TYPE and PATTERN attributes may be present at the same time		
EVAL HR	SOURCE_UDB	Input CFG file path	Mutually exclusive. One should be present.	
	SOURCE_UDB	Input UDB file path		
	REP_FILE	Output report file	Mandatory	
ENRICH UDB	SOURCE_UDB	Input UDB file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	SUPP_FILE	Input Supplementary (Enrichment) file path	Mandatory	
	CONCATENATION_ CHARACTER	Filed concatenation character	Optional	
	CASE_SENSITIVE	'Case Sensitive' flag	Optional	
	CLEAR_EMPTY_ FIELDS	'Clear empty fields' flag	Optional	
	CLEAR_UNKNOWN_U SERS	'Clear unknown users' flag	Optional	
	<COLUMN>	0 or more child tags. This is an experimental feature not included in the official product definition.		
	POS	Position of the Enrichment file column	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	NAME	Title of the Enrichment file column	Optional	
	FIELD	Target field name in UDB	Optional	
	</COLUMN>			
	IDENTIFICATION_ FIELD	Valid when at least 1 <COLUMN> tag is present	Optional	
ENRICH RDB	SOURCE_RDB	Input RDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
	SUPP_FILE	Input Supplementary (Enrichment) file path	Mandatory	
IMPORT LDIF	MAPPING_FILE	Input Mapping XML file path	Mandatory	
	SOURCE_FILE	Input LDIF text file path	Mandatory	
	TARGET_CFG	Output CFG file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
IMPORT AD	MAPPING_FILE	Input Mapping XML file path	Mandatory	
	TARGET_CFG	Output CFG file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
	<SEARCH_ROOT> (Container DN) </SEARCH_ROOT>	0 or more child tags. If not present the default search root is used.	Optional	
	<DOMAIN>	1 or more child tags. If not present the <COMMAND> tag should include <DOMAIN>'s mandatory attributes	Optional	
	USER_NAME	AD User Name	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
	PASSWORD	AD Password	Mandatory	
	SERVER	AD Server Name / IP	Mandatory	
	PORT	AD Server Port number	Optional	
	</DOMAIN>			
EXPORT SQL	SQL_SERVER_ TYPE	MS SQL Server | ORACLE	Optional	
	SERVER	The SQL server network name/IP	Mandatory	
	DATABASE	The SQL database name	Mandatory	
	USER	The SQL user name. If not present default authentication is used (SSPI)	Optional	
	PASSWORD	The SQL password. Mandatory when USER is specified.		
	BULK_INSERT	'Bulk Insert' flag	Optional	
	SHARE_DIR	The shared directory path for the bulk files. Meaningless when BULK_INSERT is FALSE. Mandatory when TRUE.		
	SOURCE_CFG	Input CFG file path (on disk)	SOURCE_ file path is mandatory. 2 or more SOURCE_ are not allowed. TARGET_ is optional.	
	TARGET_CFG	Output CFG SQL file name		
	SOURCE_AUD	Input AUD file path (on disk)		
	TARGET_AUD	Output AUD SQL file name		
	SOURCE_UDB	Input UDB file path (on disk)		

Command	Attributes/Child Tags	Description	Status	Default
	TARGET_UDB	Output UDB SQL file name		
	SOURCE_RDB	Input RDB file path (on disk)		
	TARGET_RDB	Output RDB SQL file name		
	SOURCE_BPR	Input BPR file path (on disk)		
	TARGET_BPR	Output BPR SQL file name		
	OVERWRITE_ USERS_DB	'Overwrite UDB' flag. Valid when SOURCE_CFG is present	Optional	
	OVERWRITE_ RES_DB	'Overwrite RDB' flag. Valid when SOURCE_CFG is present	Optional	
IMPORT SQL	SQL_SERVER_ TYPE	MS SQL Server | ORACLE	Optional	
	SERVER	The SQL server network name/IP	Mandatory	
	DATABASE	The SQL database name	Mandatory	
	USER	The SQL user name. If not present default authentication is used (SSPI)	Optional	
	PASSWORD	The SQL password. Mandatory when USER is specified.		
	SRC_FILE	Input SQL file name. The command is intended to work with any types of files but currently implemented for the CFG files only.	Mandatory	
	DEST_FILE	Output file path.	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
RUN AUDIT	SOURCE_CFG	Input CFG file path	Mandatory	
	TARGET_AUD	Output Audit card file path	Mandatory	
	OVERWRITE_ TARGET_AUD	'Overwrite Audit card' flag (if FALSE new alert entities are added to the existing file)	Optional	
	IGNORE_INVALID_ ENTITIES	'Ignore invalid entities' flag.	Optional	
	MAX_TOTAL_ ALERTS	Maximum total number of generated alerts		
	MAX_ENTITY_ ALERTS	Maximum number of generated alerts per an entity		
	MAX_CATEGORY_ ALERTS	Maximum number of generated alerts per a category (audit type)		
	MAX_TOTAL_ ENTITY_ALERTS	Maximum number of generated alerts per entity and category		
	<SETTINGS>	Used for the Pattern Audit	Optional	
	FLAGS	The settings bitmask	Optional	
	TESTS	Comma separated list of pattern-based tests	Optional	
	<EVAL_WEIGHTS >		Optional	
	ORG ORG TYPE FIELD_N	User field evaluation weight		
	</EVAL_WEIGHTS >			
	Audit Parameters	According to the parameter	Optional	
	</SETTINGS>			
FIX AUDIT	TARGET_CFG	Output CFG file path	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
	TARGET_UDB	Output UDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
	TARGET_AUD	Output Audit card file path	Mandatory	
	ALERT_TYPES	See the alert types in the RUN AUDIT specification	Optional	
	FIX_TO_APPLY	1 | 2 (1 for primary, 2 for secondary fixes)	Optional	
COMPARE CFG	CURRENT_CFG	Updated CFG file path	Mandatory	
	PRIOR_CFG	Original CFG file path	Mandatory	
	TARGET_AUD	Output Audit card file path	Optional	
	DIFF_PATH	Output Diff Report file path	Mandatory	
	LOG_PATH	Output Diff Log file path	Mandatory	
	DIFF_NEW_USERS	'Report New Users' flag	Optional	
	DIFF_REMOVED_ USERS	'Report Removed Users' flag	Optional	
	DIFF_SAME_USERS	'Report Updated Users' flag	Optional	
	DIFF_NEW_RES	'Report New Resources' flag	Optional	
	DIFF_REMOVED_RES	'Report Removed Resources ' flag	Optional	
	DIFF_SAME_RES	'Report Updated Resources ' flag	Optional	
	DIFF_NEW_ROLES	'Report New Roles' flag	Optional	
	DIFF_REMOVED_ ROLES	'Report Removed Roles' flag	Optional	
	DIFF_SAME_ROLES	'Report Updated Roles' flag	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	DIFF_BY_ROLES	'Link Diff by Roles' flag	Optional	
	DIFF_BY_USERS	'Link Diff by Users' flag	Optional	
	UNICODE_OUTPUT	'Unicode Output' flag	Optional	
IMPORT REQUEST	TARGET_CFG	Target CFG to apply the requests	Mandatory	
	TARGET_AUD	Target Audit file to put the requests	Mandatory	
	SQL_SERVER_ TYPE	MS SQL Server | ORACLE	Optional	
	SERVER	The SQL server network name/IP	Mandatory	
	DATABASE	The SQL database name	Mandatory	
	USER	The SQL username. If not present, default authentication is used (SSPI)	Optional	
	PASSWORD	The SQL password. Mandatory when USER is specified.		
	REQUESTED_BY	Name of requester	Optional	
	REQUEST_CFG	Source CFG name in the database	Optional	
	TRANSACTION	Transaction name	Optional	
	CAMPAIGN	Campaign name	Optional	
	CATEGORY	Category name	Optional	
	VALUE	Value	Optional	
	TYPE	Request Type	Optional	
CSV MAPPER	SOURCE_CSV	Input CSV file path	Mandatory	
	TARGET_CFG	Output CFG file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
	PERSON_ID_ CASE_SENSITIVE	'Person ID case-sensitive' flag	Optional	
	IN_SEPARATOR	Input separator	Optional	
	OUT_SEPARATOR	Output separator	Optional	
	PERSON_ID	Person ID column index (1-based)	Mandatory	
	RES_NAME_X	Resource Name column index, where X may be 1, 2, or 3	For each X, either first or second is Mandatory	
	RES_NAME_X_ TEXT	Resource Name constant text		
BASIC ROLES SEARCH	ROLE_NAME_ PREFIX	Role name prefix	Optional	
	SOURCE_CFG	Input CFG path name	Mandatory	
	TARGET_CFG	Output CFG path name	Mandatory	
	MIN_NEWLY_ COVERED_ CONNECTIONS_NUM	Minimum number of newly covered connections	Optional	
	MIN_NEWLY_ COVERED_ CONNECTIONS_PCT	Minimum percent of newly covered connections	Optional	
	MIN_RESOURCES_ NOT_COVERED_NUM	Minimum number of resources not covered by new roles	Optional	
	MIN_RESOURCES_ NOT_COVERED_PCT	Minimum percent of resources not covered by new roles	Optional	
	MIN_USERS_NOT_ COVERED_NUM	Minimum number of users not covered by new roles	Optional	
	MIN_USERS_NOT_ COVERED_PCT	Minimum percent of users not covered by new roles	Optional	
	MAX_ROLES	Maximum number of new roles	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	MIN_RESOURCES	Minimum number of resources in a new role	Optional	
	MIN_USERS	Minimum number of users in a new role	Optional	
ITERATED SEARCH	ROLE_NAME_ PREFIX	Role name prefix	Optional	
	SOURCE_CFG	Input CFG path name	Mandatory	
	TARGET_CFG	Output CFG path name	Mandatory	
	DEFAULT_ DESCRIPTION	Default role description	Optional	
	MIN_NEWLY_ COVERED_ CONNECTIONS_NUM	Minimum number of newly covered connections	Optional	
	MIN_NEWLY_ COVERED_ CONNECTIONS_PCT	Minimum percent of newly covered connections	Optional	
	MIN_RESOURCES_ NOT_COVERED_NUM	Minimum number of resources not covered by new roles	Optional	
	MIN_RESOURCES_ NOT_COVERED_PCT	Minimum percent of resources not covered by new roles	Optional	
	MIN_USERS_NOT_ COVERED_NUM	Minimum number of users not covered by new roles	Optional	
	MIN_USERS_NOT_ COVERED_PCT	Minimum percent of users not covered by new roles	Optional	
	MAX_ROLES	Maximum number of new roles	Optional	
	MIN_RESOURCES	Minimum number of resources in a new role	Optional	
	MIN_USERS	Minimum number of users in a new role	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	SEARCH_MODE	USERS | RESOURCES | CONNECTIONS	Mandatory	
RULE-BASED SEARCH	ROLE_NAME_ PREFIX	Role name prefix	Optional	
	SOURCE_CFG	Input CFG path name	Mandatory	
	TARGET_CFG	Output CFG path name	Mandatory	
	SEARCH_MODE	USERS | RESOURCES | CONNECTIONS	Mandatory	
	IGNORE_NULL_ VALUES	'Ignore null values' flag	Optional	
	MIN_PERCENT_ WITHIN_GROUP	Minimum percent of role coverage within a group	Optional	
	MAX_RULES_ PER_GROUP	Maximum number of rules per a group	Optional	
	<ATTRIBUTE> (attribute name) </ATTRIBUTE>	1 or more attributes for the rule-based search	Mandatory	
	MIN_NEWLY_ COVERED_ CONNECTIONS_NUM	Minimum number of newly covered connections	Optional	
	MIN_NEWLY_ COVERED_ CONNECTIONS_PCT	Minimum percent of newly covered connections	Optional	
	MIN_RESOURCES_ NOT_COVERED_NUM	Minimum number of resources not covered by new roles	Optional	
	MIN_RESOURCES_ NOT_COVERED_PCT	Minimum percent of resources not covered by new roles	Optional	
	MIN_USERS_NOT_ COVERED_NUM	Minimum number of users not covered by new roles	Optional	
	MIN_USERS_NOT_ COVERED_PCT	Minimum percent of users not covered by new roles	Optional	

Command	Attributes/Child Tags	Description	Status	Default
	MAX_ROLES	Maximum number of new roles	Optional	
	MIN_RESOURCES	Minimum number of resources in a new role	Optional	
	MIN_USERS	Minimum number of users in a new role	Optional	
COPY ROLE	SOURCE_CFG	Input CFG path name	Mandatory	
	TARGET_CFG	Output CFG path name	Mandatory	
	<ROLE>	1 or more child tags.	Mandatory	
	NAME	Source role name	Mandatory	
	REPLACE	Replace role flag	Optional	TRUE
	COPY_LINKS	Copy links flag	Optional	TRUE
	TARGET_NAME	Target role name	Optional	(same as NAME)
	</ROLE>			
IMPORT UNIX PASSWORD	SOURCE_ PASSWORD	Input Unix Password file path	Mandatory	
	SOURCE_GROUPS	Input Groups file path	Mandatory	
	TARGET_CFG	Output CFG file path	Mandatory	
	TARGET_UDB	Output UDB file path	Mandatory	
	TARGET_RDB	Output RDB file path	Mandatory	
	RES_NAME_2	Default Res Name 2	Optional	(empty)
	RES_NAME_3	Default Res Name 3	Optional	(empty)
IMPORT ITIM 4.5	LOGIN_CONFIG	Login configuration parameter	Mandatory	
	APP_SRV_HOME	Application server	Mandatory	
IMPORT ITIM 4.6	DETAILS_XML	Details XML file path	Mandatory	
	MAPPING_XML	Mapping XML file path	Mandatory	
EXPORT ITIM 4.5	JAR_DIRECTORY	Java modules directory	Mandatory	
	ITIM_HOME	ITIM home directory	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
EXPORT ITIM 4.6	DIFF_FILE	Export differences file path (export commands only)	Mandatory	
IMPORT CA	VERSION	IM Version	Optional	0.0
	CFG_FOLDER	Configuration file folder (from 12.5)	Mandatory	
EXPORT CA	PASSWORD	Password	Mandatory	
	DIFF_FILE	Export differences file path (export only)	Mandatory	
	OUT_FOLDER	Output file folder (import only, from 12.5)	Mandatory	
	REPORT_ID	Report ID string (import only, from 12.5)	Optional	(empty)
	SETTINGS	Settings XML file path (before 12.5)	Mandatory	
	MAPPING	Mapping XML file path (before 12.5)	Mandatory	
	RES_DB	Input RDB file path (before 12.5)	Mandatory	
	USERS_DB	Input UDB file path (before 12.5)	Mandatory	
	ROLE	Input Roles CSV file path (before 12.5)	Mandatory	
	USER_RES	Input User-Resources CSV file path (before 12.5)	Mandatory	
	ROLE_RES	Input Role-Resources CSV file path (before 12.5)	Mandatory	
	USER_ROLE	Input User-Roles CSV file path (before 12.5)	Mandatory	

Command	Attributes/Child Tags	Description	Status	Default
	ROLE_ROLE	Input Role-Roles CSV file path (before 12.5)	Mandatory	
REMOVE REDUNDANT LINKS	SOURCE_CFG	Input configuration file path	Mandatory	
	TARGET_CFG	Output configuration file path	Mandatory	
BPR SOD REPORT	CFG	Input configuration file path	Mandatory	
	BPR	Input rules file path	Mandatory	
	OUTPUT	Output report file path	Mandatory	
IMPORT SAP	PASSWORD	SAP Password	Mandatory	
	SETTINGS	Settings XML file path	Mandatory	

Pattern-Based Audit Tests

CA Identity Governance provides pattern-based audit tests to identify and list suspicious users, roles, and resources in six categories: suspect entities, suspect connections, similar roles and role hierarchy, similar resources, in/out of pattern entities and entities with many or few connections.

Note: For more information about patter-based audit, see the *Client Tools Guide*.

Following is a list of pattern-based audit tests that correspond to tests available in the client tools:

- PROPOSE COLLECTOR USERS
- PROPOSE COLLECTIBLE RESOURCES
- PROPOSE SUSPECT ROLES
- PROPOSE RESOURCES WITH SUSPECTED USERS
- PROPOSE ROLES WITH SUSPECTED USERS
- PROPOSE EXTRA USER RESOURCE
- PROPOSE EXTRA USER RESOURCE BY HR
- PROPOSE EXTRA USER RESOURCE BY PRIVILEGES
- PROPOSE EXTRA USER RESOURCE DIRECT ONLY
- PROPOSE EXTRA USER RESOURCE DIRECT ONLY BY HR
- PROPOSE EXTRA USER RESOURCE DIRECT ONLY BY PRIVILEGES

- PROPOSE EXTRA ROLE RESOURCE
- PROPOSE EXTRA USER ROLE
- PROPOSE EXTRA USER ROLE BY HR
- PROPOSE EXTRA USER ROLE BY PRIVILEGES
- PROPOSE RELATED RES USERS
- PROPOSE RELATED RES HIERARCHY
- PROPOSE RELATED ROLES USERS
- PROPOSE RELATED ROLES RESOURCES
- PROPOSE RELATED ROLES SUBSET USERS
- PROPOSE RELATED ROLES SUBSET RESOURCES
- PROPOSE RELATED ROLES SUBSUMED USERS
- PROPOSE RELATED ROLES SUBSUMED RESOURCES
- PROPOSE RELATED ROLES HIERARCHY
- PROPOSE RELATED ROLES OVERLAP
- PROPOSE ADDITIONAL USERS
- PROPOSE ADDITIONAL RESOURCES
- PROPOSE NEW ROLES
- PROPOSE NEW RESOURCES
- PROPOSE NEW ROLES BY PRIVILEGES
- PROPOSE NEW RESOURCES BY PRIVILEGES
- PROPOSE EXCEPTION USERS GREATER THAN RESOURCES
- PROPOSE EXCEPTION USERS GREATER THAN TOTAL RESOURCES
- PROPOSE EXCEPTION USERS GREATER THAN ROLES
- PROPOSE EXCEPTION USERS LESS THAN RESOURCES
- PROPOSE EXCEPTION USERS LESS THAN TOTAL RESOURCES
- PROPOSE EXCEPTION USERS LESS THAN ROLES
- PROPOSE EXCEPTION RESOURCES GREATER THAN USERS
- PROPOSE EXCEPTION RESOURCES GREATER THAN TOTAL USERS
- PROPOSE EXCEPTION RESOURCES GREATER THAN ROLES
- PROPOSE EXCEPTION RESOURCES LESS THAN USERS

- PROPOSE EXCEPTION RESOURCES LESS THAN TOTAL USERS
- PROPOSE EXCEPTION RESOURCES LESS THAN ROLES
- PROPOSE EXCEPTION ROLES GREATER THAN RESOURCES
- PROPOSE EXCEPTION ROLES GREATER THAN TOTAL RESOURCES
- PROPOSE EXCEPTION ROLES GREATER THAN SUB ROLES
- PROPOSE EXCEPTION ROLES GREATER THAN USERS
- PROPOSE EXCEPTION ROLES GREATER THAN TOTAL USERS
- PROPOSE EXCEPTION ROLES GREATER THAN PARENT ROLES
- PROPOSE EXCEPTION ROLES LESS THAN RESOURCES
- PROPOSE EXCEPTION ROLES LESS THAN TOTAL RESOURCES
- PROPOSE EXCEPTION ROLES LESS THAN SUB ROLES
- PROPOSE EXCEPTION ROLES LESS THAN USERS
- PROPOSE EXCEPTION ROLES LESS THAN TOTAL USERS
- PROPOSE EXCEPTION ROLES LESS THAN PARENT ROLES
- PROPOSE DUAL USER RESOURCE LINKS
- PROPOSE DUAL USER ROLE LINKS
- PROPOSE DUAL ROLE RESOURCE LINKS
- PROPOSE DUAL ROLE ROLE LINKS
- PROPOSE DIRECT USER RESOURCE LINKS
- PROPOSE USERS MATCHING RULES
- PROPOSE USERS NOT MATCHING RULES

Pattern-Based Audit Test Parameters

When running a pattern-based audit test, you specify parameters associated with the audit test. To match the listed parameters with the correct pattern-based audit test, you can view the audit tests in the client tools and see which parameters are necessary for the test you want to perform.

Following is a list of parameters that correspond to the specific pattern-based audit test:

- COLLECTOR_MINSTD
- COLLECTOR_MAXP
- COLLECTOR_MAXRESOURCES

- COLLECTIBLE_MINSTD
- COLLECTIBLE_MAXP
- COLLECTIBLE_MAXUSERS
- SUSPECT_ROLES_MINSTD
- SUSPECT_ROLES_MAXP
- SUSPECT_ROLES_MAXUSERS
- EXCESS_USER_RESOURCE_MISMATCH
- EXCESS_USER_RESOURCE_MISMATCH_BY_PR
- EXCESS_USER_ROLE_MISMATCH
- EXCESS_USER_ROLE_MISMATCH_BY_PR
- EXCESS_ROLE_RESOURCE_MISMATCH
- RELATED_OVERLAP_USERS
- RELATED_OVERLAP_RESOURCES
- RELATED_ROLES_USERS
- RELATED_ROLES_RESOURCES
- RELATED_ROLES_SUBSET_USERS
- RELATED_ROLES_SUBSET_RESOURCES
- RELATED_ROLES_SUBSUMED_USERS
- RELATED_ROLES_SUBSUMED_RESOURCES
- RELATED_RES_USERS
- RELATED_RES_HIERARCHY
- ROLE_ADDITIONAL_USERS
- ROLE_ADDITIONAL_RESOURCES
- PROPOSE_NEW_ROLES
- PROPOSE_NEW_RESOURCES
- PROPOSE_NEW_ROLES_BP
- PROPOSE_NEW_RESOURCES_BP
- PROPOSE_EXC_USER_GT_RES
- PROPOSE_EXC_USER_GT_TOTRES
- PROPOSE_EXC_USER_GT_ROLES

- PROPOSE_EXC_USER_LT_RES
- PROPOSE_EXC_USER_LT_TOTRES
- PROPOSE_EXC_USER_LT_ROLES
- PROPOSE_EXC_RES_GT_USERS
- PROPOSE_EXC_RES_GT_TOTUSERS
- PROPOSE_EXC_RES_GT_ROLES
- PROPOSE_EXC_RES_LT_USERS
- PROPOSE_EXC_RES_LT_TOTUSERS
- PROPOSE_EXC_RES_LT_ROLES
- PROPOSE_EXC_ROLE_GT_USERS
- PROPOSE_EXC_ROLE_GT_TOTUSERS
- PROPOSE_EXC_ROLE_GT_RES
- PROPOSE_EXC_ROLE_GT_TOTRES
- PROPOSE_EXC_ROLE_GT_SUBROLES
- PROPOSE_EXC_ROLE_GT_PARENTROLES
- PROPOSE_EXC_ROLE_LT_USERS
- PROPOSE_EXC_ROLE_LT_TOTUSERS
- PROPOSE_EXC_ROLE_LT_RES
- PROPOSE_EXC_ROLE_LT_TOTRES
- PROPOSE_EXC_ROLE_LT_SUBROLES
- PROPOSE_EXC_ROLE_LT_PARENTROLES

Using Variables in Batch Processing

While using CA Identity Governance in production and using batch processing, you may need to specify dynamic parameters for filenames that are defined in the SBT files.

For example, instead of writing CONFIG="AD_Config.cfg", you may prefer that the filename has the date and time. If so, you can specify the current date and time in the filename as a variable. Each time you run the batch file the filename automatically updates with the current date and time.

Example:

```
<BATCH>
<COMMAND
ACTION="IMPORT CSV"
ROLEID_AS_NUM="true"
CONFIG="Config_%%DATE%%_%%TIME%%_%%USERNAME%%.cfg"
USERS_DB="UsersDB.udb"
RES_DB="ResDB.rdb"
ROLE="roles.txt"
USER_RES="user-res.txt"
USER_ROLE="user_role.txt"
ROLE_RES="role_res.txt"
ROLE_ROLE="role_role.txt"/>
</BATCH>
```

Filename variables in SBT files are converted when the SBT file is executed.

A variable is defined as a keyword surrounded by %% and %%, such as %%DATE%%.

The following table lists all supported variables in CA Identity Governance:

Variable Name	Output format
Any user environment variable	The parameter's value Example: If HOME= C:\HOME %%HOME%% will be replaced with C:\HOME
%%DATE%%	YYYYMMDD
%%TIME%%	HHMM

Using Special XML Characters in Batch Files

Many characters require special treatment if included as part of a string in XML files. The following table lists the characters that require special treatment in a batch file:

Character	Character Name	Character as Used in XML Files
&	ampersand	&amp;
<	Less than	<
>	Greater than	>
"	Quotation mark	"
'	Apostrophe	&#39;

Example:

The following is an invalid use of the ampersand (&) character in an XML string:

```
<Organization>Discovery & Auditing</Organization>
```

The following is a valid use of the ampersand (&) character in an XML string:

```
<Organization>Discovery &amp; Auditing</Organization>
```

SQL Server Types

You can implement CA Identity Governance databases on Microsoft SQL Server or Oracle Server. In batch files that address the database server, you set the following command attributes to specify the type of database server and the target on the database server:

SQL_SERVER_TYPE

(Optional) Specifies what type of database server hosts CA Identity Governance databases. The default server for batch processes is Microsoft SQL Server. Use this attribute only when an Oracle server hosts CA Identity Governance databases. Any string value other than uppercase ORACLE specifies Microsoft SQL Server.

Valid values: ORACLE

Note: This attribute corresponds to the SQL Server Type field in the Data Management Settings dialog of CA Identity Governance client applications (File, General Settings, SQL Connectivity).

SERVER

Defines the target on the database server:

- For a Microsoft SQL Server, this field specifies the host name of the database server instance.
- For an Oracle database server, this field specifies the net service name, as defined in the tnsnames.ora file in the Oracle service directory.

Note: This attribute corresponds to the Server field in the Data Management Settings dialog of CA Identity Governance client applications (File, General Settings, SQL Connectivity).

Examples: Connect to Microsoft SQL Server and Oracle Server

The following file references the CA Identity Governance database on a Microsoft SQL Server. The Server attribute contains the host name of the server. The Database attribute is required.

```
<BATCH>
<COMMAND CONTINUE_ON_FAIL="true"
ACTION="IMPORT SQL"
    SERVER="MSSQL_hostname"
    DATABASE="eurekify_sdb"
    SRC_FILE="SQL://BaseConfigModel.cfg"
    DEST_FILE="OUTPUT/BaseConfigModel.cfg"/>
</BATCH>
```

The following file references the CA Identity Governance database on an Oracle Server. The Server attribute contains the Oracle service name. The Database attribute is not used.

```
<BATCH>
<COMMAND CONTINUE_ON_FAIL="true"
ACTION="IMPORT SQL"
    SQL_SERVER_TYPE="ORACLE"
    SERVER="RCM_DB_service"
    USER="sdb_admin"
    PASSWORD="password"
    SRC_FILE="SQL://BaseConfigModel.cfg"
    DEST_FILE="C:\SBT\IMPORT_ORACLE\NewConf.cfg"/>
</BATCH>
```

Sample Batch Files

CA Identity Governance comes with a collection of sample batch files that you can either use directly, or open and customize. The sample batch files can be found in the **\Sage Demo\Batch Examples** folder under the CA Identity Governance installation directory.

The Sample batch files include the following:

Batch Name	Batch Name
AD_Import	Merge_CFG
AD_Import_multi_domain	Merge_RDB
AD_Import_SSPI	Merge_UDB
Audit_Batch	MultiAD_Windows_authentication_SSPI
BatchExamples	RACF_Import
BatchTests	Res_Enrichment
BPR_Batch	SQL_Export
CompareBatch	TrimDB
Compare_cfg	TSS_Import
CSV_Export	UnixPwd
CSV_Import	Users_Enrichment
Enrichment	
FilterTypes	
HR_Eval	
Import_Users_Request	
LdapBatch	
LDIF_Import	

Chapter 4: CSV File Import

This section contains the following topics:

Import Data to a Mixed Universe Using CSV Files

In CA Identity Governance, data can be imported into a single universe from the following three types of endpoints:

- Managed endpoints—data is managed by CA Identity Manager
- Discovered endpoints—data is retrieved by the standalone JCS
- Unmanaged endpoints—data is retrieved through a third-party tool

A universe with data from different types of endpoints is called a mixed universe. To import data into a mixed universe, all data from an unmanaged endpoint must conform to the same standards. The following section details how to import data from an unmanaged endpoint into CA Identity Governance using CSV files that are compliant with these standards.

Note: Legacy CA Identity Governance functionality using the old CSV converter file format is still supported, but to support a mixed universe, create CSV files that conform to the necessary standards.

Representing Endpoint Objects in CA Identity Governance

An endpoint object is anything on the application that a given user may or may not be privileged to. Endpoint objects are defined by the following elements:

- Endpoint Type—the type of application from which the data is coming from, for example, SAP
- Endpoint Name—the name of the application instance from which the data is coming from, for example, 'SAP-PROD'
- Object Type—the object type, such as SAP Role or UNIX Group

- Unique ID—any string that uniquely identifies the endpoint object among other objects of the same type. This ID can be a name, a path, or a DN.
- Friendly Name—any string to describe what the object is. This name does not have to be unique.

Once you determine what endpoint objects exist on the endpoint, identify if it should be represented as a role or a resource within CA Identity Governance. Note the following:

- If the endpoint object represents a single privilege, it is represented as a *resource* in CA Identity Governance.
- If the endpoint object represents a privilege that gives other privileges, it is represented as a *role* in CA Identity Governance.

Entity Files

Users Database File

The first row in the entity file must be a header row. Each subsequent row represents a single user, where the row contains the following fields:

- PersonID—the key, and must be unique
- UniqueID—should appear in the PersonID also, usually the login name
- UserName
- Organization
- Organization Type
- Endpoint Type—the type of application from which the data is coming from, for example, SAP
  - Endpoint Name—the name of the application instance from which the data is coming from, for example, 'SAP-PROD'
- Field 1 to Field n (optional)

Note: The UniqueID, Endpoint Type, and Endpoint Name fields do not have to appear in the Users database file for the connector defined as "As Users". However, for flexibility reasons, we recommend including them. Any connector defined as "As Accounts" or "As Users and as Accounts" must include these three required fields.

Resources Database File

The first row in the entity file must be a header row. Each subsequent row represents a single resource and contains the following fields, where a combination of Resource Name 1, 2, and 3 is the key and is assumed to be unique:

- Resource Name 1—the friendly name of the endpoint object.
- Resource Name 2—composed of the endpoint type and endpoint name separated by the delimiter [EP], for example, ActiveDirectory[EP]ADSServer1

- Resource Name 3—composed of the object type and unique ID separated by the delimiter [P], for example, UNIX Group[P]serverAdmins
- Field 1 to Field n (optional)

Roles File

If you want to use custom role attributes, the first row in the entity file must be a header row. Otherwise, you can use the previous roles file format that did not require a header row.

The following system properties define how the file is treated:

csvImport.roles.forceHeader.enable

Defines if the first line in the file is read as a header row. If this property is set to true, the product reads the first line as a header row. If this property is set to false, the product automatically detects if the first row is a header row, based on the value of the following property.

Default: False

csvImport.roles.headerRow.key

Defines the field name to match to detect if the first line in the file is a header row when csvImport.roles.forceHeader.enable is set to false. When csvImport.roles.forceHeader.enable is set to true, this property is not used.

Default: rolename

The file has one row per role definition, each with the following fixed fields:

- RoleName - must be the unique ID
- Description
- Organization
- Owner
- Type—the object type, such as SAP Role
- Creation Date
- Reviewer
- Approval Status
- Approval Date
- Rule
- (Optional) Organization 2—composed of the endpoint type and endpoint name separated by the delimiter [EP], for example, ActiveDirectory[EP]ADSServer1. For a business role, this field is empty.

- Organization 3—Internal CA Identity Governance Use Only
- Expiration Date

For custom role attributes, add them to the roles file after the Expiration Date field.

Relations Files

User-Resource Links

The User-Resource Links file does not require a header row. The file requires one row per connection, each with the following fields:

- PersonID
- Resource Name 1
- Resource Name 2
- Resource Name 3

Role-Resource Links

The Role-Resource Links file does not require a header row. The file requires one row per connection, each with the following fields:

- RoleID
- Resource Name 1
- Resource Name 2
- Resource Name 3

User-Role Links

The User-Role Links file does not require a header row. The file requires one row per connection, each with the following fields:

- PersonID
- Role Name

Role-Role Links

The Role-Role Links file does not require a header row. The file requires one row per connection, each with the following fields:

- Role Name (parent)
- Role Name (child)

Prevent Role-Role Cyclical Dependencies in CSV File Import

Symptom:

I tried to run an import in the portal using CSV files with cyclical links. After the import completes, I cannot open the Model or Master configurations.

Solution:

When importing data into CA Identity Governance using CSV file format, CA Identity Governance can detect and prevent role-role cyclic dependencies.

Set the following properties:

- import.csv.shouldCheckRoleRoleCyclicLinks=true
- import.csv.shouldCheckRoleRoleCyclicLinks.shouldFailOnRoleCyclic=true

Note: The second property is relevant only if the first property is set to true.

CSV File Generation Property Setting

When importing CSV files into CA Identity Governance, use the following property setting:

import.csv.reader.escape

Changes the escape character in a CSV file to a specific character not included in the CSV font set when the backslash character ("\") is included.

Prevent Role-Role Cyclical Dependencies in CSV File Import

Symptom:

I tried to run an import in the portal using CSV files with cyclical links. After the import completes, I cannot open the Model or Master configurations.

Solution:

When importing data into CA Identity Governance using CSV file format, CA Identity Governance can detect and prevent role-role cyclic dependencies.

Set the following properties:

- import.csv.shouldCheckRoleRoleCyclicLinks=true
- import.csv.shouldCheckRoleRoleCyclicLinks.shouldFailOnRoleCyclic=true

Note: The second property is relevant only if the first property is set to true.

Chapter 5: Transforming Data

This section contains the following topics:

Create and Apply Data Transformations

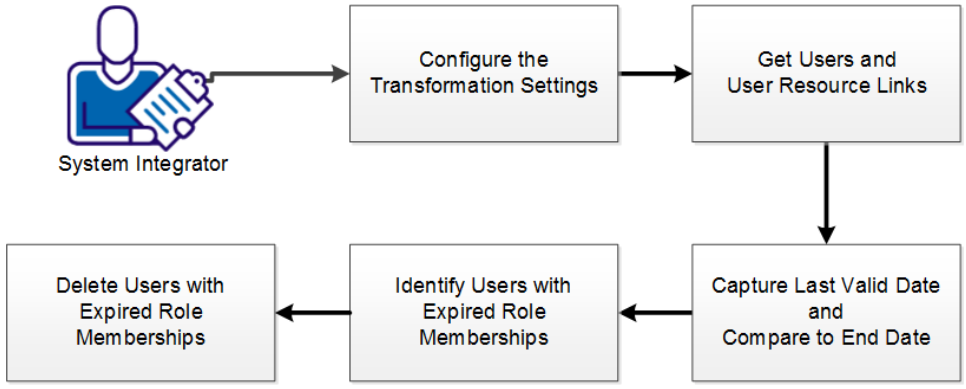
As a systems integrator, you add or modify CA Identity Governance data. You can modify data during an import (or an export) by running a transformation.

Use Case - Remove Expired Links

You are importing data from an Enterprise Resource Planning (ERP) application. In this application, users have roles membership. When the role membership of a user expires, CA Identity Governance maintains the account record and adds an end date value. When you import the data into CA Identity Governance, you can filter out users with expired role memberships based on the end date. This filter ensures that managers do not certify invalid access during a certification.

Note: This scenario outlines the steps to create the transformation sample KTR file **removeExpiredUserResourceLinks.ktr** that is located in the CA\RCM\Server\data-integration\samples\RCM directory. In this scenario, we remove expired links to SAP or Oracle roles that are modeled as resources in CA Identity Governance.

In PDI, a transformation comprises a series of steps. Each step represents an action that is performed on the data. A hop represents a data pathway that connects the steps and enables schema metadata to pass from one step to another.



Follow these steps:

1. Configure the transformation settings. (see page 48)
2. Get users and user-resource links (see page 49).
3. Capture last valid date and compare to end date (see page 50).
4. Identify users with expired role memberships.
5. Delete users with expired role memberships (see page 51).

Configure the Transformation Settings

For your transformation to connect to CA Identity Governance, configure the transformation settings in PDI.

Follow these steps:

1. Launch PDI, as follows:
   - **Windows:** Go to **Start**, **Programs**, **CA**, **Role & Compliance Manager**, **Server**, **Data Integration Tool (PDI)**.
   - **Linux:** Click the **Data Integration Tool (PDI)** icon in your home directory.

   Note: PDI is located in the following directory:

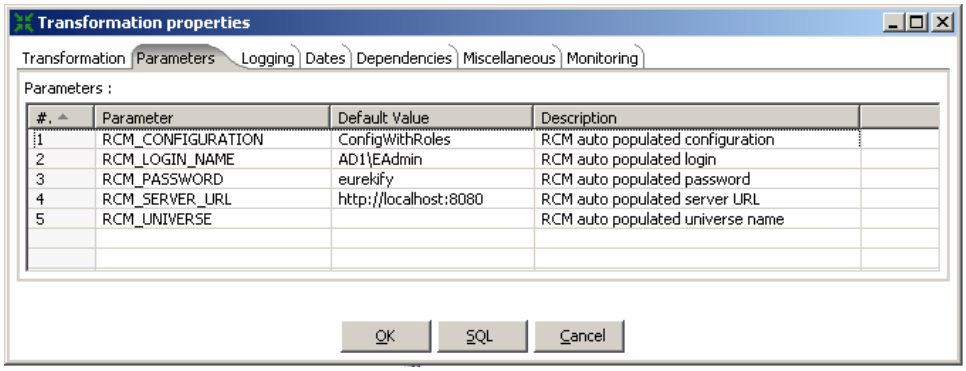
   `CA\RCM\Server\data-integration`
2. Go to **File**, **Open**, and browse to the following directory:

   `data-integration\samples\RCM`
3. Open the **template.ktr** file and save it under another name that is relevant to the transformation you want to create.
4. Press **CTRL-T**.

   The **Transformation properties** windows opens.
5. Click the **Parameters** tab, and enter the following CA Identity Governance information:
   - RCM_CONFIGURATION
   - RCM_LOGIN_NAME
   - RCM_PASSWORD

- RCM_SERVER_URL
- RCM_UNIVERSE

The following image is an example of the Parameters field with example server information:



6. Click **OK**.

   You have started PDI and configured the CA Identity Governance settings.

Now that PDI is configured, you can define the steps in your transformation.

Get Users and User Resource Links

To identify expired links, locate CA Identity Governance users, and obtain the user-resource links for users that have the start and end date information, use **Get Users and Links.**

Follow these steps:

1. In PDI, select and drag the **Input Users** step to the Transformation screen in the **Design** tab, under **RCM Input**.
2. Right-click the step and select **Edit step**.

   The I**nput Users!** window opens.
3. Under the **Fields** tab, click **Get fields**.

   All the available fields are populated.
4. Delete all fields except **PersonID**, and click **Ok**.
5. Select and drag the **Get User-Resource Links by Users** step to the Transformation screen in the **Design** tab, under **RCM Links**.
6. Right-click the **Get User-Resource Links by Users** step and select **Edit step**.
7. Under the **Link Fields (Keys and Attributes)** tab, click **Get Link Fields**.

8. Delete all fields except the following, and click **Ok**:
   - PersonID
   - ResName1
   - ResName2
   - ResName3
   - Start Date
   - End Date
9. Create a hop between the **Input Users** step and the **Get User-Resource Links by Users** step.

   You have added the step to Get Users and Links.

Capture Last Valid Date and Compare to End Date

To identify expired role memberships, add a field to define the last valid date when you run the transformation. The transformation filter compares this last valid date to the role membership end date to filter out users with expired role memberships.

Follow these steps:

1. In PDI, in the **Design** tab, locate and expand the **Transform** step, select **Add constant**, and drag it to the Transformation screen.
2. Right-click the **Add constants** step and select **Edit step**.
3. Add a field that is named '**LastValidDate' of type 'String**', and click **Ok**.
4. Create a hop between the **Get User-Resource Links by Users** step and the **Add Constants** step.
5. Click the **Transform** tab, select the **Set field value to a constant** step, and drag it to the Transformation screen.
6. Create a hop between the **Add constants** step and the **Set field value to a constant** step.
7. Right-click the **Set field value to a constant** step and select **Edit step**.

   The **Set field value to a constant** window opens.
8. Select the **Use variable in constant** option, and click **Get fields**.

9. Add the **LastValidDate** field and set the **Replace by value** field as follows:

   `${Latest_Valid_Date}`

   You provide this date when you run the transformation.

10. Click **Ok**.

    You have added a field to define the last valid date when you run the transformation.

Delete Users with Expired Role Memberships

To ensure that managers do not certify invalid role memberships, delete users with expired role memberships based on the end date.

Follow these steps:

1. In PDI, in the **Design** tab, locate and expand the **Flow** step, select **Dummy (do nothing)**, and drag it to the Transformation screen.
2. Create a hop between the **Filter Rows** step and the **Dummy (do nothing)** step.

   All valid links are sent to the **Dummy (do nothing)** step.

3. Select **Text file output**, and drag it to the Transformation screen.
4. Right-click the **Text file output** step and select **Edit step**.

   The **Text file output** window opens.

5. Under the **File** tab, provide a filename for the text output file.
6. Under the **Fields** tab, click **Get Fields**.
7. Delete all fields except for the following and click **Ok**:
   - PersonID
   - ResName1
   - ResName2
   - ResName3
   - Start Date
   - End Date
8. Create a hop between the **Filter rows** step and the **Text file output** step.

   All users with expired links are captured and sent to the Text file output.

9. On the **Design** tab, locate and expand the **RCM Output** step, select the **Delete User-Resource Links** step, and drag it to the Transformation screen.

10. Right-click the **Delete User-Resource Links** step and select **Edit step**.

    The **!Delete User-Resource Links!** window opens.

11. In the **General** tab, in the **RCM Configuration** section, map the following fields, and click **Ok**:

    - Person ID Field=PersonID
    - Res Name 1 Field=ResName1
    - Res Name 2 Field=ResName2
    - Res Name 3 Field=ResName3

    You have added the output steps to delete users with expired role memberships. When you run the transformation, it deletes all users with expired role memberships.

Add the Output Steps to Delete Users with Expired Role Memberships

To delete users with expired role memberships before certification, add the output steps to the transformation.

Follow these steps:

a. In PDI, in the **Design** tab, locate and expand the **Flow** step, select the **Dummy (do nothing)** step, and drag it to the Transformation screen.

b. Create a hop between the **Filter Rows** step and the **Dummy (do nothing)** step.

   All valid links are sent to the **Dummy (do nothing)** step.

c. On the **Design** tab, locate and expand the **Output** step, select the **Text file output** step, and drag it to the Transformation screen.

d. Right-click the **Text file output** step and select **Edit step**.

   The **Text file output** window opens.

e. Under the **File** tab, provide a filename for the text output file.

f. Under the **Fields** tab, click **Get Fields**.

g. Delete all fields except for the following and click **Ok**:

   - PersonID
   - ResName1
   - ResName2
   - ResName3

- Start Date
- End Date

h. Create a hop between the **Filter rows** step and the **Text file output** step.

All users with expired links are captured and sent to the Text file output.

i. On the **Design** tab, locate and expand the **RCM Output** step, select the **Delete User-Resource Links** step, and drag it to the Transformation screen.

j. Right-click the **Delete User-Resource Links** step and select **Edit step**.

The **!Delete User-Resource Links!** window opens.

k. In the **General** tab, in the **RCM Configuration** section, map the following fields, and click **Ok**:

- Person ID Field=PersonID
- Res Name 1 Field=ResName1
- Res Name 2 Field=ResName2
- Res Name 3 Field=ResName3

You have added the output steps to delete users with expired role memberships. When you run the transformation, it deletes all users with expired role memberships.

Use PDI Transformations on a Server with SSL Configured

To use PDI transformations on a CA Identity Governance server that is configured with SSL, use the following procedure.

Note: For more information on how to configure a CA Identity Governance server for SSL communication, see the *Installation Guide*.

Follow these steps:

1. Install OpenSSL.
2. Open a command prompt and run the following command:

```
openssl s_client -connect RCM_HOST:8443
```

3. Copy all text between the following (including the begin and end lines) to a cert.txt file:

```
----BEGIN CERTIFICATE----

----END CERTIFICATE----
```

4. Run the following command to verify the certificate:

```
"%JAVA_HOME%\bin\keytool" -printcert -file cert.txt
```

5. Create a **rcmServer.keystore** file by using the key tool, as follows:

   ```
   "%JAVA_HOME%\bin\keytool" -import -file cert.txt -keystore rcmSrever.keystore
   -storepass password
   ```

   The file is created under the java bin directory.

6. Copy the **rcmServer.keystore** file to the PDI home directory.

7. Edit the PDI application files, as follows:

   - **Windows:** Edit the **set-pentaho-env.bat** file and add the following:

     ```
     set PENTAHO_DI_JAVA_OPTIONS =%PENTAHO_DI_JAVA_OPTIONS%

     Djavax.net.ssl.trustStore=rcmSrever.keystore
     -Djavax.net.ssl.keyStorePassword=password
     ```

   - **Linux:** Edit the **set-pentaho-env.sh** file and add the following:

     ```
     PENTAHO_DI_JAVA_OPTIONS ="$ PENTAHO_DI_JAVA_OPTIONS

     Djavax.net.ssl.trustStore=rcmSrever.keystore
     -Djavax.net.ssl.keyStorePassword=password"
     ```

8. Open the **spoon.bat/sh** file and add the Input step for Users to a transformation.

9. Edit the Input Users step, and click **Edit the RCM Connection**.

10. Verify the following:

    a. The RCM connection information lists **https://servername:sslPort** as the connection URL.

    b. The **Accept any certificate** check box is cleared.

    c. Click **Test RCM Connection**.

    You have used PDI transformations on a CA Identity Governance server that is configured with SSL.

Chapter 6: Customizing Business Flows

This section contains the following topics:

Business Workflows

Role-based management of entitlements involves managers and resource owners throughout the company. Individuals can interact with the product at various levels.

At the highest level, an administrator can initiate a certification to review the entitlements assigned to workers, or the entitlements included in each role. Similarly, a manager can request changes to the entitlements of workers that they manage, or of roles that they own.

These high-level business goals are called *business workflows*.

For example, in a certification, each entitlement under review generates a certification item. For each entitlement, CA Identity Governance assigns relevant reviewers, tracks responses, and implements any changes indicated by the reviewers.

Ultimately, items require decisions by managers and other users. For example, a resource certification requires the owner of each resource to review the users that can access the resource, and the roles that grant access to the resource.

Workpoint Workflows

CA Identity Governance uses Workpoint process management software to conduct component tasks of a business workflow. Each item in a business workflow is associated with a Workpoint process. For each item in the business workflow CA Identity Governance creates a Workpoint job based on the corresponding process. CA Identity Governance populates the instance with parameter values specific to the workflow context.

The set of all possible items in each business workflow dictates a set of required Workpoint processes for the workflow, one process to handle each item.

Example: Items Generated by a Certification

When you initiate a certification workflow, the product generates certification items based on the data filters, optional audit cards, and other settings of the certification wizard.

A user certification includes only users with the TeamLeader role. In addition, an audit card provides suggested entitlements. Based on these settings the product generates the following items:

- Certify the TeamLeader role for the 35 users who have that role.
- Suggest new roles for four of these users who appear in the audit card.

To implement these items, CA Identity Governance creates Workpoint jobs using the following default Workpoint processes:

- The UCertURole Workpoint process certifies the link between a user and a role. CA Identity Governance creates 35 separate instances of the UCertURole process to certify that each user should have the Team Leader role.
- The USuggURole Workpoint process suggests a new role for a user. To handle the suggestion based on the audit card, CA Identity Governance creates four separate instances of the USuggURole process.

CA Identity Governance populates each job with unique parameter values that identify the user and role under review.

Each of the Workpoint jobs generates items for individual managers. For example, the manager of each user with the Team Leader role receives a review item. Similarly, the manager of each user for whom CA Identity Governance suggests a new role receives a review item.

Using Workpoint Designer

Use the Workpoint Designer application to edit CA Identity Governance Workpoint processes. The CA Identity Governance installer deposits a precompiled version of this application in the CA Identity Governance installation directory. This version of Workpoint Designer includes special tabs and other features that help you work with the heavily customized CA Identity Governance processes and process nodes.

You can copy this package to run Workpoint Designer on another workstation.

Note: For more information about running the customized Workpoint Designer package, see the *Installation Guide*.

Debug Workflow Processes

Use the Workflow Debug screen of the Portal to track and analyze the execution of business workflows. You can analyze the items a workflow generates.

To debug workflow processes

1. Go to Administration, Workflow Settings, Worfklow Debug.

   The Select Business Flow pane lists recent workflows. You can filter or sort this list by various parameters.

2. In the Select Business Flows pane, click a workflow.

   The Select Task pane lists the component tasks of the workflow. You can filter or sort this list by various parameters.

   These tasks also appear in a table at the bottom of the screen. This table shows more details about each task, using the following fields:

   Task Type

   Indicates the type of task. The task type correlates to the CA Identity Governance class invoked by the Workpoint.process.

   ParentID

   Indicates the sequential number assigned to the workflow by CA Identity Governance

   TaskID

   Indicates the sequential number assigned to the task by CA Identity Governance

JobID

Indicates the sequential number assigned to a Workpoint job when it is created. This value also indicates the database that is used to store and track the job. Typically this is the CA Identity Governance Workpoint database.

Entity1

Indicates the focal entity of the task. When the focal entity is a user, two columns display both the login and personID.

Entity2

Indicates the second entity of the task. Typically this is the second entity of a link under review.

Result

Displays the result of the task.

3. In the Select Task pane, click a task.

   The Select Context pane lists Workpoint jobs that handle the task. You can filter or sort this list by various parameters.

4. In the Select Context pane, click a job.

   The Select User Action pane lists user actions that the job generated. You can filter or sort this list by various parameters.

   These user actions also appear in the Edit User Action pane. You can approve or reject the action, and track its effect on the workflow.

How to Customize Workflow Processes

The product implements each item in a workflow by creating a job based on a Workpoint process. Create a customized Workpoint process to customize the behavior of each item.

In the Portal, map a set of processes to each workflow. You can then apply this set of customized processes to workflows.

Follow these steps:

1. In Workpoint Designer, clone and modify the default Workpoint processes you want to change.

   Each process implements a single type of task. The following process-level field identifies the CA Identity Governance task context associated with a Workpoint process:

   Implements

   Specifies the workflow task that a Workpoint process implements. The product creates a task context that exposes parameters and interactions relevant to the task.

   Note: The value of this parameter defines the only task to which you can map the process. To ensure correct mapping of this parameter and other parameters, base your new process on a copy of the standard process for the task.

2. In the Portal, go to Administration, Settings, Workpoint DB Administration and update the Workpoint processes in the database.

3. Create a new process mapping for the workflow.

4. Implement the new process mapping as follows:

   - For the entire CA Identity Governance server, make the custom process mapping the default for all workflows of this type in your deployment.
   - For a universe (see page 63), make the custom process mapping the default for all workflows of this type that are based on the universe.
   - For an individual workflow, specify the custom process mapping when you create a workflow. For example, you can override the default process mapping when you create a certification.

Create a Custom Process Mapping

When you apply a custom process mapping to a workflow, or define it as a system default, the product invokes the custom processes you specify to implement worklow tasks.

Follow these steps:

1. (Optional) If you want to customize workflow process mappings for the certification template wizard, go to Administration, Settings, Property Settings, and set the following properties:

   campaign.settings.allowModifiedCampaignProcesses

   Determines whether users can specify custom process mappings for individual certifications.

   campaign.settings.allowModifiedCampaign

   Determines whether users can specify custom campaign process mappings.

2. In the Portal, go to Administration, Workflow Settings, Workflow Process Mapping.

   The Process Mappings Administration screen appears. The table lists previously defined custom process mappings.

3. Specify settings for the new mapping under Add Process Mappings. The following fields are not self-explanatory:

   Workflow Type

   Specifies the category of workflow to which the mapping applies. The mapping only defines processes for tasks in the specified type of workflow.

   Flavor

   Defines the specific workflow to which the mapping applies.

4. Click Add.

   The Edit Process Mappings screen appears. A table lists the following information for all possible tasks of the workflow:

   Type

   Specifies the type of task that is mapped in each row of the table. The values in this column correlate to the value of the Implements parameter in Workpoint processes for the task.

   Process Reference

   Specifies the Workpoint process that is mapped to the task. CA Identity Governance invokes this process to implement the task. The drop-down list displays only Workpoint processes in the CA Identity Governance database whose Implements parameter value matches the task type.

   By default, the Edit Process Mappings screen displays the default mapping of tasks to standard workflow processes.

5. Click the Process Reference drop-down list beside a task, and select the custom process you defined.

   The task is mapped to the custom process. Workflows that use this mapping invoke the custom process to implement the task.

   Repeat this step to map custom processes for any or all tasks in the workflow.

6. Click Add.

   The process mapping appears in the table. You can apply this mapping to business workflows of the referenced type to implement modified behaviors.

Edit a Custom Process Mapping

CA Identity Governance implements each task in a workflow by invoking a Workpoint process. You can create custom Workpoint processes that implement nonstandard behaviors for one or more types of tasks in a workflow.

The set of possible tasks in each business workflow dictate a set of Workpoint processes that must be defined for the workflow. One process handles each type of task.

After you create custom Workpoint processes, use the following procedure to map a set of custom processes to each workflow.

When you apply a custom mapping to a workflow, or define it as a system default, the custom processes you specify are invoked to implement the process.

To edit a custom process mapping

1. In the CA Identity Governance portal, go to Administration, Workflow Settings, Workflow Process Mapping.

   The Workflow Process Mapping screen appears. The table lists previously defined custom process mappings.

2. Do *one* of the following:
   - Click Edit beside a process mapping to modify it.
   - Modify a copy of an existing mapping, as follows:
     a. Click Copy beside a process mapping.
     b. Enter a new, unique name for the copy and click OK.

        The copy appears in the table.
     c. Click Edit beside the copy.

   The Edit Process Mappings screen appears. A table lists the following information for all possible tasks of the workflow:

   Type

   Specifies the type of task that is mapped in each row of the table. The values in this column correlate to the value of the Implements parameter in Workpoint processes for the task.

   Process Reference

   Specifies the Workpoint process that is mapped to the task. CA Identity Governance invokes this process to implement the task. The drop-down list displays only Workpoint processes in the CA Identity Governance database whose Implements parameter value matches the task type.

   The screen displays the existing custom mapping of tasks to Workpoint processes.
3. Click the Process Reference drop-down list beside a task, and select the custom process you defined.

   The task is mapped to the custom process. Workflows that use this mapping invoke the custom process to implement the task.

   Repeat this step to map custom processes for any or all tasks in the workflow.
4. Click Save

   Changes are saved to the process mapping. You can apply this mapping to a business workflow of the referenced type to implement modified behaviors.

Define Default Workflow Processes for the CA Identity Governance Server

Use the Default Process Mapping screen to assign process mappings to workflows. The product uses the processes specified in these mappings to implement the tasks of workflows.

Note: You can override these default assignments when you define default mappings for a universe.

To access the Default Process Mapping screen, go to Adminstration, Workflow Settings, Default Process Mapping. The screen has the following sub-sections:

Certification

Lists business workflows related to certifications.

Access Request

Lists business workflows related to self-service requests.

Change Approval

Lists business workflows related to configuration changes implemented from CA Identity Governance client tools.

Each row represents a type of business workflow. A drop-down list displays available process mappings for that type of workflow.

Define Default Process Mapping for the Universe

To assign process mappings to CA Identity Governance business workflows within a universe, use the Default Process Mapping tab under Universes. CA Identity Governance uses the processes specified to implement business workflows.

Note the following:

- Universe mappings override global default mappings set under Administration, Workflow Settings.
- You can override these default assignments when you apply a specific process mapping to a workflow. Do this in the CA Identity Governance Portal by in Administration, Workflow Settings, Workflow Process Mapping.

To edit a universe default process mapping, click the Default Process Mapping tab. This tab has the following sections:

Certification

Lists business workflows related to certifications.

Access Request

Lists business workflows related to self-service requests.

Change Approval

Lists business workflows related to configuration changes launched from CA Identity Governance client tools.

Each row represents a type of business workflow. A drop-down list displays available process mappings for that type of workflow.

Apply Customized Process Mappings to a Workflow

CA Identity Governance implements the tasks in a workflow by invoking a set of Workpoint processes. You can customize workflow behavior by specifying a set of processes when you create a business workflow. The product uses the process mapping that you specify to implement the tasks of the workflow. This selection overrides default process mappings that are defined for the target universe, and system defaults.

Note: When you specify a process mapping during workflow creation, your selection only applies to this workflow. Universe and system defaults apply to subsequent workflows.

For certification workflows, specify the process mapping in the Execution screen of the certification wizard. The following options are available under Processes:

System defaults

Uses the default workflow processes installed with CA Identity Governance to implement the certification. Standard certification behaviors are executed.

Customized Processes

Uses the process mapping set you select from the drop-down to implement the certification.

Processes Applied

Displays the processes that the product invokes to execute the major tasks of the certification, based on your selection.

The following system property is used in the process for the default user for approval and decision making:

Workflow.defaultManager

Defines a default user for approval and decision making activities.

Value: User login

Custom Workflow Processes in a Certification

CA Identity Governance uses a set of predefined processes to execute items of a certification. Administrators can create alternative processes, which change how CA Identity Governance implements certification tasks. For example, administrators can define a set of processes that involve higher management levels in certification reviews. When you create a certification, you can specify which set of processes controls the execution of certification items.

Before you can apply alternative processes to your certification, administrators must create the processes, import them to CA Identity Governance, and map them to tasks of the certification business workflow.

Specify the process mapping for your certification in the Execution screen of the certification wizard. The following options are available under Processes:

System defaults

Uses the default workflow processes installed with CA Identity Governance to implement the certification. Standard certification behaviors are executed.

Customized Processes

Uses the process mapping set you select from the drop-down to implement the certification.

Processes Applied

Displays the processes that CA Identity Governance invokes to execute the major tasks of the certification, based on your selection.

The following system property is used in the process for the default user for approval and decision making:

Workflow.defaultManager

This property value is used as a default user for approval and decision making activities.

Value: User login

Add a Second Approval Step to Business Workflow

CA Identity Governance now defaults to a single-step certification process. To add a second approval step to a business workflow, add a second certification node to your workflow process. Do *not* use the legacy approval node. Use the approval node *only* for backward compatibility.

Chapter 7: Common Functionality in CA Identity Governance Workpoint Processes

This section contains the following topics:

Workpoint Process Nodes and CA Identity Governance Building Blocks

Building blocks are Workpoint node templates that invoke services and modular behaviors on the CA Identity Governance server. Each building block node exposes customized fields that support interaction with CA Identity Governance.

Workpoint processes that implement CA Identity Governance workflow tasks use nodes based on these building block templates to interact with CA Identity Governance. Generally, Workpoint jobs correspond to *tasks* of the CA Identity Governance workflow, and individual nodes and building blocks often generate *actions* for reviewers and other business users. One task can generate several actions.

Note: CA Identity Governance uses Workpoint processes as a tool for workflow implementation, but *CA Identity Governance retains overall control of the workflow*. CA Identity Governance initiates Workpoint jobs, and coordinates their progress. Workpoint processes define detailed task behaviors, but the building block nodes in these processes call functional modules in CA Identity Governance to implement the task. CA Identity Governance does not expose all server functionality in building blocks.

When CA Identity Governance initiates a business workflow, it creates and maintains a context that controls the Workpoint jobs and building block operations of the workflow. This context maintains the data set for the workflow, which can be passed to Workpoint jobs as parameter values.

For example, the CA Identity Governance context for a certification includes the scope of entities in the certification, member lists specified to assign reviewers, and other settings specified during certification creation. This information is passed to Workpoint jobs that implement tasks of the certification.

In some cases, these parameters are assigned values during the progress of the job. CA Identity Governance passes parameter values to the Workpoint job that reflect this control input, and advances the job to other nodes.

System, Workflow, and Task Parameters

CA Identity Governance exposes three kinds of parameters to Workpoint processes. Typically CA Identity Governance populates these parameters with values when it creates a Workpoint job.

- *System* parameters receive static values based on CA Identity Governance server system properties. These values are not unique to the workflow context. The *system.* prefix identifies these parameters.
- *Workflow* parameters receive unique local values based on the data set and other options in the business workflow context. For example, many settings in the certification creation wizard pass to Workpoint processes as workflow parameters. The *flow.* prefix identifies these parameters.
- *Task* parameters receive unique local values based on the workflow context. These values are necessary to complete the task associated with the Workpoint process. For example, the attributes that identify entities under review are necessary and specific to a link review task, and depend on the source configuration and data filters in the workflow context. The *task.* prefix identifies these parameters.

Examples: System, Workflow, and Task Parameters

The RoleSuggURole process suggests new roles during a role certification. This process declares the following parameters:

system.certification.useApprovers

Specifies which entities in a link under review generate reviewers. The value comes from the certification.useApprovers CA Identity Governance system property.

This value is not unique to the workflow, but applies to review tasks in all business workflows.

flow.userApproval.userMembersList

Specifies the member list that is used to assign reviewers for user entities.

This parameter reflects settings made by the user in the certification template wizard, and stored by CA Identity Governance in the workflow context. The *flow.* prefix indicates that it applies to *all* tasks of *this* worfklow that review a user entity.

task.personID

Specifies the user in the link under review.

task.roleName

Specifies the role in the link under review.

These task parameters contain information *unique to this task* of the workflow. Other Workpoint jobs of the workflow reference other links in the certification.

General Process Parameters

Most standard CA Identity Governance task processes declare the following parameters. When CA Identity Governance creates a job based on a process, it populates these parameters with values specific to the workflow and task.

Implements

Specifies the workflow task that a Workpoint process implements. CA Identity Governance exposes parameters and interactions relevant to the task.

universe

Specifies the reference universe of the task. Member lists, attribute fields, and other settings must adhere to the defaults defined for the specified universe.

task.configurationName

Specifies the reference configuration file of the task. Entity and RACI references address the specified configuration file and its RACI configuration files.

flow.affectedConfiguration

Specifies the configuration file in which changes are implemented. Typically, changes are implemented in the reference configuration, as indicated by the following default value:

```
flow.affectedConfiguration = {'task.configurationName'}
```

task.udbName

Specifies the user database of the target configuration file.

task.rdbName

Specifies the resource database of the target configuration file.

task.flowOwner

Specifies the user who initiated the workflow.

setResult

Assigns a value returned by CA Identity Governance to a process or node parameter.

Process Parameters for User, Role and Resource Attributes

Standard CA Identity Governance Workpoint processes declare binary data fields for user, role, or resource attributes. When CA Identity Governance initiates a Workpoint job, it populates these fields with actual attribute values of the entities involved in the job. These fields identify the entities under review in various operations. For example, you can use these fields to search member lists or the RACI configuration and assign reviewers.

The following data field identifies a user entity:

task.personId

Specifies a user entity in the source configuration by the value of the personID field defined for the universe.

The following data field identifies a role entity in processes that handle links between one role and a user or resource.

task.roleName

Specifies a role entity in the source configuration by the value of the roleName field defined for the universe.

Processes that handle links between parent and child roles use the following data fields instead of the task.roleName field:

task.childRoleName

Specifies the child role of a parent-child link in the source configuration by the value of the roleName field defined for the universe.

task.parentRoleName

Specifies the parent role of a parent-child link in the source configuration by the value of the roleName field defined for the universe.

The following data fields identify a resource entity:

task.resourceName1

Specifies a resource entity in the source configuration by the value of the ResName1 field defined for the universe.

task.resourceName2

Specifies a resource entity in the source configuration by the value of the ResName2 field defined for the universe.

task.resourceName3

Specifies a resource entity in the source configuration by the value of the ResName3 field defined for the universe.

The following data fields specify the values of entity attributes:

fieldName

Specifies the name of an entity or link attribute.

fields

Specifies a list of entity or link attributes and values. For example, the following value sets values for the Location and Organization parameters:

```
fields="Location=Houston;Organization=Marketing"
```

fieldsSeparator

Defines the characters that separate attribute statements in the fields parameter. By default, you can use commas and semicolons.

task.newRoleOwner

Specifies the user who owns the role created by a worfklow process.

startDate

Specifies the date from which a link is valid in the configuration.

endDate

Specifies the date at which a link ceases to be valid in the configuration.

Custom Workflow Parameters

You can define custom parameters for certification workflows.

The Properties screen of the certification wizard displays prompts for workflow parameters. The content of the Properties screen changes to reflect the Workpoint processes of the certification you create. CA Identity Governance scans the processes invoked for the certification, and identifies workflow parameters that are defined in these processes. CA Identity Governance displays prompts for these parameters in the Properties screen of the wizard.

You can also define prompt texts and labels for your custom parameters. These strings are displayed in the Properties screen of the certification wizard.

Define a Custom Workflow Parameter

Even though CA Identity Governance stores workflow parameter values in the general context of the workflow, use this procedure to declare the custom parameter locally in each process that needs it. You can define a different local default value in each process for the same parameter.

To define a customer workflow parameter

1. Run the Workpoint Designer application (see page 57) and access the CA Identity Governance Workpoint database.
2. Clone an existing CA Identity Governance Workpoint process and open the new process.

   Workpoint Designer displays the process in flowchart format.

3. Open the process-level properties dialog (right-click in the background of the process flowchart and select Properties).

   The Process Properties dialog appears.

4. Click the RCM Parameters tab.

   The tab lists system, workflow, task, and local parameters.

5. Click Add.

   The New RCM Property dialog appears.

6. Enter the name of the parameter. This string must match the naming format for custom workflow parameters (see page 72).

   Click OK

   The new parameter appears in the list.

7. (Optional) click the Value column beside the new parameter and specify a default value. You can select an existing parameter from the drop-down list.

8. Click Apply.

   The parameter is created. When a workflow uses this process, CA Identity Governance generates a prompt for this parameter based on the parameter type and text labels you embedded in the parameter name string. You can define text labels (see page 74) to enhance these prompts.

Naming Format and Parameter Types for Workflow Parameters

The naming convention for workflow parameters lets you define a set of related parameters. The naming convention also defines the type of the parameter. The type of the parameter determines which interface controls CA Identity Governance displays when it prompts for the parameter value.

Use the following naming format for workflow parameters:

`flow.Group_name.Param_type.Param_name`

Note: *Group_name* is the group to which this parameter belongs, *Param_type* is the type of the parameter, and *Param_name* is the name of the parameter.

CA Identity Governance supports the following parameter types:

Text

Declares a parameter that contains a text value. CA Identity Governance displays the name or text label of the parameter and a text box control.

Note: If you declare a parameter and do not specify a parameter type, a text parameter is created by default.

Boolean

Declares a parameter that contains a Boolean value. CA Identity Governance displays a selection checkbox control beside the name or text label of the parameter.

UserMembersList

Declares a parameter that specifies a member list which matches user attributes. CA Identity Governance displays the name or text label of the parameter and a drop-down list of user member lists defined in the universe.

RoleMembersList

Declares a parameter that specifies a member list which matches role attributes. CA Identity Governance displays the name or text label of the parameter and a drop-down list of role member lists defined in the universe.

ResourceMembersList

Declares a parameter that specifies a member list which matches resource attributes. CA Identity Governance displays the name or text label of the parameter and a drop-down list of resource member lists defined in the universe.

Configuration

Declares a parameter that specifies a configuration file in the universe. CA Identity Governance displays the name or text label of the parameter and a drop-down list of configuration files defined in the universe.

ConfigurationAll

Declares a parameter that specifies a configuration file. CA Identity Governance displays the name or text label of the parameter and a drop-down list of all configuration files defined in the CA Identity Governance database.

Universe

Declares a parameter that specifies a universe. CA Identity Governance displays the name or text label of the parameter and a drop-down list of universes defined in the CA Identity Governance database.

AuditCard

Declares a parameter that specifies an audit card in the universe. CA Identity Governance displays the name or text label of the parameter and a drop-down list of audit cards defined in the universe.

PortalUser

Declares a parameter that specifies a user defined in the CA Identity Governance portal. CA Identity Governance displays the name or text label of the parameter, the default user you specify for the parameter, and a drop-down list of all users in the permissions configuration of the CA Identity Governance server.

AccountableType

Declares a parameter that specifies how to search the RACI configurations for a reviewer. CA Identity Governance displays a selection check box, the name or text label of the parameter, and a pair of option buttons for the Manager and Manager's Manager AccountableType (see page 89) values. Use this parameter type to specify how a reviewer is selected for *role and resource* entities.

AccountableTypeUser

Declares a parameter that specifies how to search the RACI configurations for a reviewer. CA Identity Governance displays a selection checkbox, the name or text label of the parameter, and three option buttons corresponding to the standard AccountableType (see page 89) values. Use this parameter type to specify how a reviewer is selected for *user* entities only, because it offers the Self Attestation value.

Customize Text Labels for Workflow Parameters

By default, CA Identity Governance uses the group name and parameter name strings of the parameter declaration to display prompts for a workflow parameter.

For example, when parameters are declared as follows:

```
flow.CustomParamGroup.Boolean.PromptSelectOption
flow.CustomParamGroup.PortalUser.PromptForReviewer
flow.CustomParamGroup.AccountableType.PromptForRACI
```

CA Identity Governance generates interface elements and controls as shown in the following figure:

You can define text strings to replace these parameter declaration strings.

The following properties files store text strings for the CA Identity Governance portal interface:

- EurekifyBaseWebApplication.properties – core CA Identity Governance properties file.
- EurekifyBaseWebApplication_de.properties – localized German properties file.
- EurekifyBaseWebApplication_en.properties – localized English properties file.
- EurekifyBaseWebApplication_es.properties – localized Spanish properties file.

- EurekifyBaseWebApplication_fr.properties – localized French properties file.
- EurekifyBaseWebApplication_it.properties – localized Italian properties file.

When CA Identity Governance server runs on a JBoss application server, these files are in the following directory:

gm_install\Server\eurekify-jboss\server\eurekify\deploy\eurekify.war\WEB-INF\classes\com\eurekify\web\application

Note: *gm_install* is the CA Identity Governance installation directory.

Edit the relevant files to declare strings for your workflow parameters.

Note: Restart CA Identity Governance server to use these new strings in the portal.

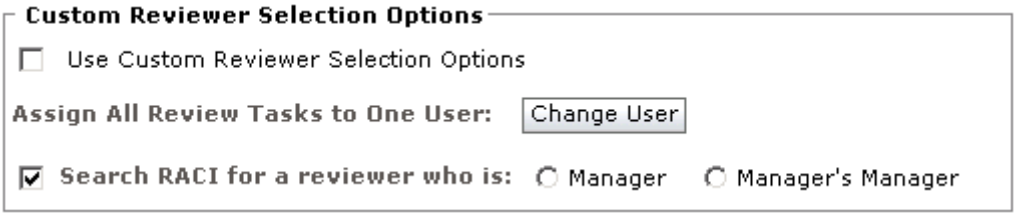
Example: Text Labels for Workflow Parameters

The following statements define strings for the custom parameters of the previous example:

```
flow.CustomParamGroup = Custom Reviewer Selection Options
flow.CustomParamGroup.Boolean.PromptSelectOption = Use Custom Reviewer Selection Options
flow.CustomParamGroup.PortalUser.PromptForReviewer = Assign All Review Tasks to One User
flow.CustomParamGroup.AccountableType.PromptForRACI = Search RACI for a reviewer who is
```

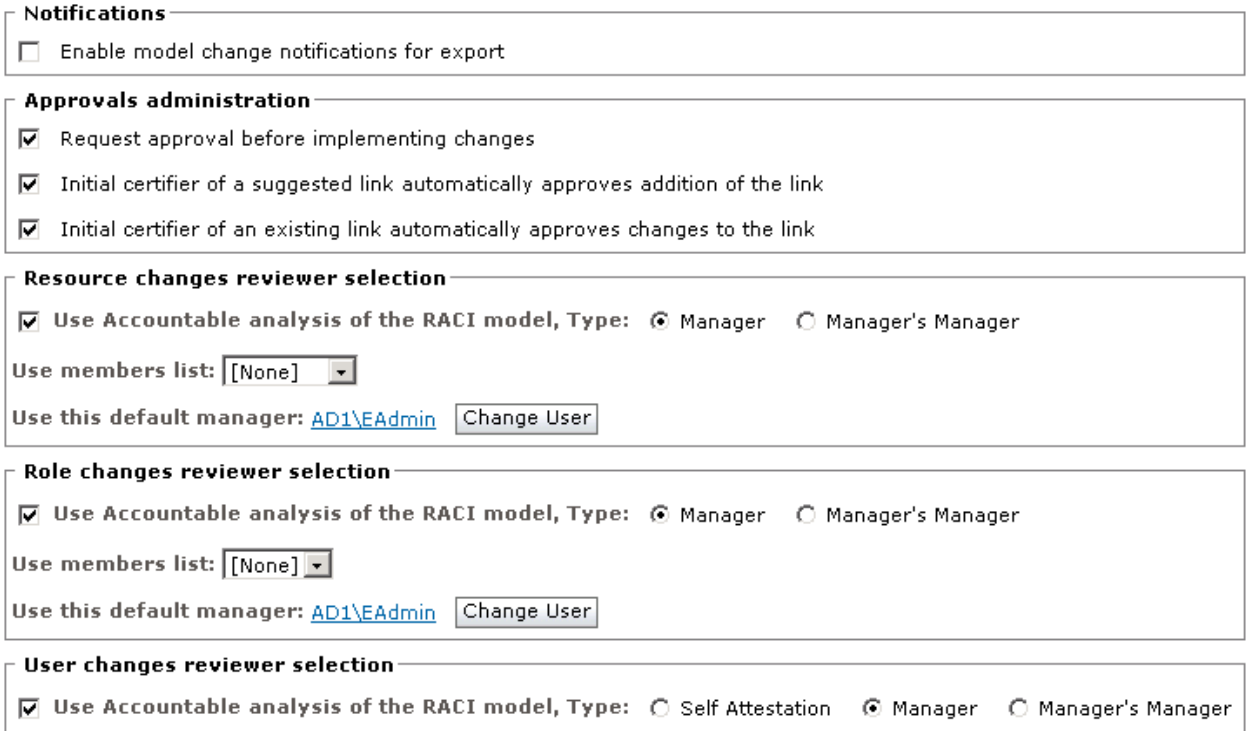
Note: Define a separate string for the group name, as shown.

Based on these string definitions, CA Identity Governance generates interface elements and controls as shown in the following figure:



Example: Workflow Parameters from Default CA Identity Governance Certification Processes

The following figure shows the Properties screen of the certification wizard, with prompts for workflow parameters drawn from default CA Identity Governance certification processes.



The fields in the Properties screen shown previously correspond to the following parameters in default CA Identity Governance Workpoint processes. These fields appear in the Properties screen when a certification uses these Workpoint processes, or any processes that declare these parameters.

- The Enable model change notifications... option under Notifications results from the flow.notifications.boolean.isModelChangeNotificationOn parameter.
- Under Approvals administration, the following options appear:
  - The Request approval before implementing changes option results from the flow.requestChangeApproval.boolean parameter.
  - The initial certifier of an suggested link... option results from the flow.requestChangeApproval.boolean.autoApproverSuggest parameter.
  - The initial certifier of an existing link... option results from the flow.requestChangeApproval.boolean.autoApproverCertify parameter.
- Under Resource changes reviewer selection, the following options appear:
  - The User Accountable analysis of the RACI model option results from the flow.resourceApproval.accountableType parameter.

- The Use member list option results from the flow.resourceApproval.resourceMembersList parameter.
- The Use this default manager option results from the flow.resourceApproval.portalUser.defaultManager parameter.

Similarly, the following parameters generate the Role changes reviewer selection options:

- flow.roleApproval.accountableType
- flow.roleApproval.roleMembersList
- flow.roleApproval.portalUser.defaultManager

Similarly, the following parameters generate the Role changes reviewer selection options:

- flow.userApproval.accountableTypeUser
- flow.userApproval.portalUser.defaultManager

Note: The radio button options generated for the flow.userApproval.accountableTypeUser parameter differ from the options generated for parallel role and resource accountableType parameters.

CA Identity Governance draws the text labels for these groups and options from the localization file, as in the following examples:

- These entries in the localization file apply to the Notifications area of the screen.

```
flow.notifications = Notifications
flow.notifications.boolean.isModelChangeNotificationOn = Enable model change
notifications for export
```

- These entries in the localization file apply to the Resource changes reviewer selection area of the screen.

```
flow.resourceApproval = Resource changes reviewer selection

flow.resourceApproval.accountableType = Use Accountable analysis of the RACI
model, Type:
flow.resourceApproval.resourceMembersList = Use members list:
flow.resourceApproval.portalUser.defaultManager = Use this default manager:
```

CA Identity Governance automatically generates check boxes, radio button options, and other controls based on the type specified for each parameter. For example, Boolean parameters display a single check box, and member list parameters display a drop-down list of related member lists.

Agent Scripts in CA Identity Governance Workpoint Processes

Nodes in standard CA Identity Governance task processes run agent scripts to evaluate parameter values, assign new values to parameters, and invoke task-related operations in CA Identity Governance. Most of these scripts are simple JavaScript routines.

Note: Some agent scripts invoke CA Identity Governance behaviors that are not controlled by Workpoint processes or system properties.

Many agent scripts set the following node parameter:

Result

Stores the result of logical operations by AreEqual, setResultToJobParameter, and other agent scripts.

Standard task processes often use the following agent scripts:

AreEqual

Compares the value of a parameter to a known string and writes a corresponding value to the result parameter.

CompareJobParameters

Compares the value of two parameters and writes a corresponding value to the result parameter.

endFlow

Instructs CA Identity Governance to terminate a workflow.

IsAggregatedExportEnabled

Tests the value of the system.modelEvent.producer.fireEvent system property to determine if model change events are aggregated, and writes a corresponding value to the result parameter.

rcmInvoker

Invokes the CA Identity Governance task context specified in the Implements parameter of the process.

runBatchWithData

Invokes batch processing using the web services API of the CA Identity Governance server.

runBatchWithURL

Invokes batch processing using the web services API of the CA Identity Governance server.

SetAutoApprovalNodeName

Maps the value of the autoApproverNode process parameter to a local data field.

setResultToJobParameter

Assigns a value to the result parameter.

SetRoleReviewer

Determines if the owner of a new role is defined. If no owner is defined, the system default reviewer receives role review and creation actions.

setSageBaseURL

Defines the URL of the web service API exposed by CA Identity Governance server.

Parent and Child Processes in a Workflow

Business workflows generate child tasks. For example, a self-service request generates individual approval tasks for each new link requested. CA Identity Governance initiates a Workpoint job for each approval task. These jobs are considered child processes within the workflow context that CA Identity Governance maintains.

Important! CA Identity Governance does not use native Workpoint parent-child capabilities to invoke child processes or transfer control between parent and child processes.

CA Identity Governance implements high-level workflow processes and low-level child processes as independent jobs in Workpoint. Parent and child jobs do not communicate directly with each other. Instead, they communicate with CA Identity Governance, which maintains the workflow context and controls all related Workpoint jobs.

Waiting for Child Processes

CA Identity Governance coordinates the execution of parent and child processes. For example, most certifications have distinct certification and change approval phases. CA Identity Governance pauses the high-level certification process until the child processes for certification actions complete. Then it advances the high-level certification process to the change approvals phase.

Parent and child processes pass a message flag between them through CA Identity Governance. However, CA Identity Governance monitors which jobs respond, and implements the logic that starts and stops parent and child jobs.

Process nodes that handle parent-child interaction use the following parameters:

waitKey

Causes CA Identity Governance to suspend a process at the current node. Any not null waitKey value causes CA Identity Governance to suspend the process.

This parameter value is passed to the workflow context in CA Identity Governancewhen the node runs the rcmInvoke agent script.

Note: This field can only be used in building blocks that pass the waitKey value to the CA Identity Governance workflow context.

In standard CA Identity Governance processes, the following values are used:

WAIT_PARENT

Causes CA Identity Governance to suspend a child process at the current node until the parent process sends the WAIT_PARENT string.

WAIT_CHILD

Causes CA Identity Governance to suspend a parent process at the current node until all child processes send the WAIT_CHILD string.

signalKey

Causes CA Identity Governance to resume a parent or child process. CA Identity Governance resumes a parent process only after all of its children return the correct signalKey value.

This parameter value is passed to the workflow context in CA Identity Governancewhen the node runs the rcmInvoke agent script.

Note: This field can only be used in building blocks that pass the signalKey value to the CA Identity Governance workflow context.

In standard CA Identity Governance processes, the following values are used:

WAIT_PARENT

Causes CA Identity Governance to resume child processes that were suspended with the waitKey value WAIT_PARENT.

WAIT_CHILD

Causes CA Identity Governance to resume a parent process that was suspended with the waitKey value WAIT_CHILD.

The waitKey and signalKey parameters are used together. The waitKey field defines the value of the signalKey field that causes CA Identity Governance to resume the suspended process. The two fields must specify the same text string.

More information:

Processes for Flow Control and Aggregation (see page 105)
Building Blocks for Parent-Child Process Handling (see page 109)

Parent Processes for Aggregated Tasks

When review actions are aggregated, all reviews of a certain stage of the workflow must complete before the workflow continues. For example, the default behavior in certifications is to wait for all initial certification reviews before starting change approval reviews.

Parent and child processes are used to support this behavior.

The following standard processes control aggregation behavior:

AggApproval (Aggregation of Approvals)

Implements logic to support aggregated approvals or rolling approvals in self-service workflows.

AggCert (Aggregation of Certifications)

Implements logic to support aggregated approvals or rolling approvals in certification certifications.

The following process parameter indicates whether the workflow aggregates review tasks:

flow.isAggregated

Specifies whether change approvals are aggregated in a workflow. CA Identity Governance populates this Boolean parameter when it creates an instance of the process.

In standard Workpoint processes for certification, suggestion, or approval tasks, the "Is Aggregate" node evaluates the flow.isAggregated parameter. If the workflow aggregates approvals, these processes signal the workflow context and wait at the "Wait for flow owner" node before they generate change approval actions.

Input from Workflow Control Actions

In addition to review actions, some standard business workflows create workflow control actions. When users submit these control actions, CA Identity Governance advances parent processes and invokes or terminates Workpoint jobs for child tasks.

Example: Create a New Role

When a user submits a new role request, the product creates an instance of the BBNewRole process.

The product steps through the process as follows:

1. At the Set Role Reviewer node, the product determines which user is the role owner.

2. At the Start Children node the product creates child Workpoint jobs to review the links that are part of the new role.
3. At the Open Approval Action node, the product places the Create Role workflow control action in the certification queue of the role owner.
4. Child jobs for certification actions continue in parallel.
5. When the role owner submits the Create Role workflow control action, the product advances the workflow to the Disable Open Approval Action nodes. The product terminates the child Workpoint jobs that are not yet complete, and removes these actions from the certification queues of certifiers.
6. The role is created with the approved privilege links.

Example: Initiate Change Approvals in a Certification

The following example shows how the product uses parent control processes, child task processes, and workflow control actions to control a complex workflow behavior.

To implement a certification workflow, the product creates an instance of the AggCert control process.

The product steps through the AggCert job as follows:

1. The Is Aggregate node evaluates to Yes.
2. At the Start Children node, the product creates child Workpoint jobs for the links under review. For example, a user certification generates jobs based on UCertURes, USuggURole, and other User Certification processes.

   This node also places a Start Approvals workflow control action in the certification queue of the certification owner.

   The Send Email node notifies the certification owner that the certification has begun.
3. The AggCert job waits at the Flow Owner Action node.

   Child jobs for certification actions continue in parallel. When a reviewer submits their Certify or Suggest action, the corresponding job stops at its Wait for flow owner node, which signals the product that it is waiting to continue.
4. When the certification owner submits the Start Approvals workflow control action, the product advances the AggCert job to the Disable Certify Action node. The product terminates the child Workpoint jobs that have not yet signaled that they are waiting to continue, and removes these actions from the certification queues of certifiers.
5. At the Signal Children node, the product signals the remaining child processes to continue.

   The change approvals phase of the certification begins. Child jobs advance to nodes that assign change approval actions to reviewers.

Assigning Reviewers

The most commonly used CA Identity Governance business workflows include one or more review tasks. Managers and resource owners review privileges to certify compliance with security mandates, and for ongoing role-based access control.

For example, standard processes for certifications subject the link under review to the following two reviews:

- Certification – the manager of an entity reviews the privilege links of the entity. This initial review can generate changes in the privileges of the entity, if the certifier rejects a link or approves a new link that CA Identity Governance suggests.
- Change Approvals – managers or users related to both entities of a link review proposed changes to the link. Changes are implemented only if these reviewers approve.

Self-service workflows typically implement a single change approvals review for each new or changed privilege link.

The following building blocks conduct initial certification review of the entity under review:

- OpenCertificationAction – certify nodes in default Workpoint processes.
- OpenSuggestAction – suggest nodes in default Workpoint processes.

Initial certification review focuses on a single entity and its links. In standard CA Identity Governance processes, there is *one* reviewer - typically, the manager of the entity under review. These code blocks match the entity under review to a user account in CA Identity Governance.

The following building block initiates change approval review for new or deleted links:

- OpenApprovalAction

Because privileges are expressed as links between two entities, change approval reviews involve *two* reviewers - one for each entity in the link under review. In standard CA Identity Governance processes, this code block matches each entity of the link to a user account in CA Identity Governance.

Note: Typically reviewers are owners or managers of reviewed entities.

You can customize these nodes to include more reviewers and more complex review processes. You can duplicate these nodes to create multistep review chains.

How to Define a Set of Reviewers

Define a set of reviewers for each certification, suggestion, or change approval node in a process.

Define the following settings for each reviewer in the set:

- The focal entity (see page 84) for the reviewer. CA Identity Governance uses this entity to select a reviewer, and presents the link under review as a property of this entity.
- The chain of sources, such as a member list, RACI query,and default reviewer, that CA Identity Governance uses to select the reviewer (see page 86).

Define the following optional behaviors for the set of reviewers:

- Aggregation and Weighting (see page 92): You can configure logic that requires a reply from any reviewer, all reviewers, or a subset of reviewers. In addition, you can assign a relative weight to the reply of each reviewer.
- Overruler (see page 93): You can define one reviewer in the set whose response overrides other reviewers and concludes the review.
- Automatic Approvers (see page 94): These reviewers are automatically assumed to approve the link or entity under review.
- A subset of reviewers (see page 96): You can activate a subset of the reviewers you defined.

Access these settings in the Properties window of the node, in one of the following ways:

- Use the form-based interface in the Building Blocks tab of the node.
- Directly edit building block parameters in the RCM Parameters tab of the node.

The Focal Entity of a Reviewer

In most review nodes, a link between two entities is under review. The product must know which of the two entities to reference when it searches for a reviewer. The product must also know how to present the link to the reviewer.

Example: Reviewers with Different Focal Entities

A standard change approval node submits a requested user-resource link for review to the following two reviewers:

- The *user* of the link under review is the focal entity for one reviewer. To select this reviewer, the product matches attributes of the *user* to a member list, or searches the RACI configuration for the manager of the *user* under review.

- The *resource* of the link is the focal entity for the other reviewer. To select this reviewer, the product matches attributes of the *resource* to a member list, or searches the RACI configuration for the manager of the *resource* under review.

Node Parameters for Focal Entities

In certification, suggestion, and change approval nodes, the following parameters define a numerical reference for each of the entities under review:

typeEntity1

Specifies the type of entity represented by the numeral 1 in entity fields of the node. In child processes of a certification, this parameter is typically the type of entity that is the focus of the certification.

Valid values: USER, ROLE, RESOURCE

typeEntity2

Specifies the type of entity represented by the numeral 1 in entity fields of the node.

Valid values: USER, ROLE, RESOURCE

The values of these two fields must describe the two entities of the link under review.

The following parameters use these numerical indexes to specify focal entity settings for each reviewer:

mainEntities

Defines the focal entity for each reviewer in the set. This field contains a comma-separated, ordered list of the numerals 1 and 2. Each entry represents the focal entity of a reviewer in the set. There must be one value for each reviewer in the set. Null values are not allowed.

entityForApproverSourceList

Defines the focal entity for each member list specified in the ApproverSource field. This field contains a comma-separated, ordered list of the numerals 1 and 2. Each entry represents the entity that is used to search the member list for one reviewer. There must be one value for each reviewer in the set.

Note: If no member list is specified for a reviewer, two consecutive commas indicate a null entry that corresponds to the null entry for this reviewer in the ApproverSource field.

Process-level parameters contain attribute values that identify the entities under review. These task-level parameters are mapped at the node level to fields named as follows:

`entity`*`X`*`.name`*`Y`*

Where X is 1 or 2, and Y is 1,2, or 3.

Example: Entity Declaration in User-Resource Certification

The UCertURes process certifies a User-Resource link in a user certification certification. Because the certification certifies users, the *user* entity is designated Entity1. The following fields and values are defined in both review nodes of the process:

```
typeEntity1=USER
typeEntity2=RESOURCE
entity1.name1={'task.personId'}
entity2.name1={'task.resourceName1'}
entity2.name2={'task.resourceName2'}
entity2.name3={'task.resourceName3'}
```

The Certify node in this process defines a single reviewer who is focused on the user entity, as expressed in the following parameter value:

```
mainEntities=1
```

The Open Approval Action node in this process defines a set of two reviewers. One reviewer is focused on each entity of the link under review, as expressed in the following parameter value:

```
mainEntities=1,2
```

Example: New Role Creation

The BBNewRole process creates a role. The owner of the new role must approve its creation. The following fields and values are defined in the Open Approval Action node:

```
typeEntity1=ROLE
entity1.name1={'task.roleName'}
```

In this special case, there is only one entity under review - the new role. Only one entity type is declared.

How CA Identity Governance Assigns Reviewers

To assign each reviewer in a set of reviewers, CA Identity Governance matches one of the entities under review to a user account in the CA Identity Governance data universe. CA Identity Governance matches entities to reviewers in the following ways:

- CA Identity Governance searches a member list to find a record that matches attribute values of the entity. When a match is found, the review action is assigned to the user that is specified in that record of the member list.
- CA Identity Governance queries the RACI configuration files of the universe, and assigns the review action based on the user who is Accountable for the entity.

  Note: In user certifications, CA Identity Governance first queries the Configuration user manager field defined in the target universe to identify the manager of each user.

- CA Identity Governance assigns the review action to a default reviewer defined for the universe.
- CA Identity Governance retrieves multiple reviewers from CA Identity Manager Role Owners or Role Administrators at runtime.

  Note: Role reviewers from CA Identity Manager are treated as one action and the result is taken from the first reviewer that submits their decision.
- CA Identity Governance assigns the review action to a reviewer based on an entity (user, role, or resource) attribute or link attribute.

When you define a reviewer, you can specify one method, or a combination of these options.

Important! Define a default reviewer to avoid process errors when a reviewer is not found.

When you combine multiple methods, CA Identity Governance evaluates them in the following order:

1. CA Identity Governance searches the approver attribute you specify.
2. CA Identity Governance searches the approvers from the CA Identity Manager role you specify.
3. CA Identity Governance searches the member list you specify.
4. CA Identity Governance searches the RACI configuration using the logic you specify.
5. CA Identity Governance uses the default reviewer you specify.

CA Identity Governance proceeds until one method matches a reviewer, and uses the first reviewer it finds.

This behavior is similar to options in the certification template wizard of the CA Identity Governance portal. However, these options are generalized and available in all certification and change approval nodes.

How to Specify Member Lists

You can specify member lists, RACI search logic, CA Identity Governance user accounts, and other values used to select reviewers in one of the following ways:

- **Explicit Declaration:** Specify the name of the member list or default user directly in the data field of the building block. This declaration is local, and does not persist in subsequent nodes of the process.

- **Process Parameters:** Use a parameter. Standard Workpoint processes with review nodes declare process-level parameters for member lists, RACI selection, and default reviewers. All nodes in the process can use the parameter values CA Identity Governance defines for the job.

Important! Verify that member lists you declare explicitly exist in the CA Identity Governance database, and that the fields they reference match the structure of user, role, or resource records in the target universe. Verify that the users you specify are defined in the CA Identity Governance database, and that the parameter values that CA Identity Governance passes to the Workpoint job support the desired behavior.

Process Parameters for Member Lists

Standard Workpoint processes for certification campaigns declare parameters for member lists. When CA Identity Governance initiates a Workpoint job, it populates these parameters with member list names based on settings from the certification template wizard or from defaults. All nodes in the process can use these parameters to reference the specified member lists.

Note: The lists you specify must exist in the CA Identity Governance database. Create and edit member lists in the CA Identity Governance portal, or using CA Identity Governance client tools.

One member list is specified for initial certification. Two lists are specified for change approvals, one for each entity in the link under review.

Example: Member Lists for a Standard User Certification Process

The standard UCertURes process certifies user-resource links in a user certification. This process declares the following parameters:

flow.memberListForCertify

Defines a member list that matches user attributes to find a reviewer for initial certification.

flow.userApproval.userMembersList

Defines a member list that matches user attributes to find a reviewer for approval of changes.

flow.resourceApproval.resourceMembersList

Defines a member list that matches resource attributes to find a reviewer for approval of changes.

Because the first two member lists match user attributes, you can specify either of them when you select a reviewer based on the *user* under review. When you select a reviewer based on the *resource* under review, only the flow.resourceApproval.resourceMembersList parameter references a relevant member list.

Example: Member List for a Standard Role Certification Process

The RoleCertRoleRole and RoleCertRolePaRo processes certify links between parent and child roles.Because both entities under review are roles, only one member list is specified for approval of changes. These processes declare the following parameters:

flow.memberListForCertify

Defines a member list that matches *role* attributes to find a reviewer for initial certification.

flow.roleApproval.roleMembersList

Defines a member list that matches role attributes to find a reviewer for approval of changes.

The flow.memberListForCertify parameter matches the focal entity of the certification in each case - in the first example, it matches user attributes, in the second example, it matches role attributes.

Process Parameters for RACI Accountable Types

Standard Workpoint processes for certifications declare process-level parameters for RACI query options. When CA Identity Governance initiates a Workpoint job, it populates these parameters based on user settings from the certification creation wizard or from defaults. All nodes in the process can use these parameters to reference the specified Accountable Type. These parameters can have the following Accountable Type values:

Self

Assigns the review action to the user under review. This option is only used to assign a reviewer based on a user entity.

Accountable

Assigns the review action to the user in the RACI configurations who is Accountable for the entity under review.

AccountableManager

Assigns the review action to the *manager of* the user in the RACI configurations who is Accountable for the entity under review.

One Accountable Type setting is specified for initial certification. Two Accountable types are specified for change approvals, one for each entity in the link under review.

The following process-level parameters receive RACI query option settings from CA Identity Governance:

flow.certificationApproval.accountableType

Defines the accountable type for initial certification.

flow.userApproval.accountableTypeUser

Defines the accountable type for approval of changes to a user in a certification.

flow.roleApproval.accountableType

Defines the accountable type for approval of changes to a role in a certification.

flow.resourceApproval.accountableType

Defines the accountable type for approval of changes to a resource in a certification.

You can use any of these parameters to specify an accountable type for any certification or change approval action. However, the Self accountable type can only be used to assign a reviewer for a user entity.

Process Parameters for Default Reviewers

Standard Workpoint processes that include review nodes declare process-level parameters for default reviewers. When CA Identity Governance initiates a Workpoint job, it populates these parameters with names based on user settings (such as options in the certification template wizard) or defaults. All nodes in the process can use these parameters to reference the specified default reviewer.

In processes that add or remove a link, a single default reviewer is specified.

In certification processes, one default reviewer is specified for initial certification, and two default reviewers are specified for change approvals, one for each entity in the link under review.

Note: In certification processes that handle links between parent and child roles, only one default reviewer is specified for approval of changes.

You can use any of these parameters to specify a default reviewer for any certification or change approval action. However, the default reviewer associated with one entity may not be knowledgeable or appropriate for other review actions.

The following process-level parameters receive default reviewer values from CA Identity Governance:

flow.defaultManager

Defines the default reviewer for initial certification in a certification. This parameter takes its value from the Workflow.defaultManager CA Identity Governance system property.

flow.userApproval.portalUser.defaultManager

Defines the default reviewer for approval of changes to a user in a certification.

flow.roleApproval.portalUser.defaultManager

Defines the default reviewer for approval of changes to a role in a certification.

flow.resourceApproval.portalUser.defaultManager

Defines the default reviewer for approval of changes to a role in a certification.

system.approval.defaultManager

Defines the default reviewer in general processes to add or remove links. This parameter takes its value from the approval.defaultManager CA Identity Governance system property.

CA Identity Manager Role Owners or Administrators as Approvers for CA Identity Governance Roles

To support CA Identity Manager approvers for roles within CA Identity Governance, the Certify building block should utilize the new source for approvers named IM Dynamic.

If you edit the building block and select the IM Dynamic source for approvers, CA Identity Governance dynamically retrieves the approvers for the role from CA Identity Manager at runtime.

Note: This feature only retrieves approvers for roles originating from an import from CA Identity Manager.

Approvers from Link or Entity Attributes

In addition to the existing member list, RACI, and default manager approvers, you can retrieve the approver from a link attribute or entity attribute.

For example, if the configuration has User-Resource Link attributes that include the personID of the approver, then we open a certification action for the approver from the link attribute.

Similarly, if the user or resource contains a manager attribute (this can be in addition to the default manager attribute, such as Manager2), this manager attribute can be used.

This feature affects the OpenCertificationAction and OpenApprovalAction building blocks. When you edit these building blocks in the Workpoint designer, a new field appears named Approver Attribute.

Note: For link attributes, proceed the attribute with 'link.*'. For example, if you state 'Link.Owner' as the approver attribute, the approver ID would be retrieved from the 'Owner' field on the link.

Review Completion Criteria: Aggregation and Weighting

You define a completion condition for each review node. When this condition is met, the product concludes the review, updates field values, and runs agent scripts of the node. The product also removes related review items from the certification queues of reviewers.

You can apply the following conditions for completion:

- **First Reviewer**: The review item concludes when one reviewer submits *any* decision (approve or reject) on the review item. Related review items are deleted from the certification lists of the other reviewers.
- **Any Reviewer**: The review item concludes when one reviewer submits an *approval* on the review item. This option is equivalent to a logical OR between all reviewers in the set. Related review items are deleted from the certification lists of the other reviewers.
- **All Reviewers**: The review item is approved only after all reviewers have approved it. If one reviewer rejects it, the review item immediately completes as rejected. This option is equivalent to a logical AND between all reviewers in the set.
- **Weighted Response**: You assign a numerical weight to the approval response of each reviewer. You also set a threshold value for approvals. When the weighted sum of approval responses meets or exceeds the threshold, the review item concludes. Conversely, when the weighted sum of rejection responses makes it impossible for approvals to meet the threshold, the review item concludes.

Note: When you define weighted approval criteria, automatic approval filters, or a subset of reviewers, the resulting logic can force the node to approve or reject the link under review in all cases. This forced response invalidates the purpose of the review node. Test the likely result when you apply these options to a set of reviewers.

The following examples clarify use of weighted reviews.

Example: *x* Out Of *y* Weighting

In a set of four reviewers, each reviewer has a Weight value of one. The minimum threshold value for completion is three.

The review concludes when the following conditions are met:

- When any *three* of the four reviewers *approve* the link, the threshold is met. The product concludes the review with a decision to *approve* the link under review.
- When any *two* of the four reviewers *reject* the link, it is impossible for the sum of approvals to meet the threshold value. The product concludes the review with a decision to *reject* the link under review.

Example: Percentage Weighting

In a set of six reviewers, the following Weight values are defined:

- Reviewer A has a Weight value of 20
- Reviewer B has a Weight value of 40
- Reviewer C has a Weight value of 70
- Reviewer D has a Weight value of 60
- Reviewer E has a Weight value of 30
- Reviewer F has a Weight value of 10

The minimum threshold value for completion is 100.

The review concludes when the following reviewers *approve* the link:

- Reviewers B and D, or
- Reviewers A, B, E, and F

Other combinations of responses conclude the review when the weighted sum of approval responses equals or exceeds 100. The result of the review is to *approve* the link under review.

CA Identity Governance concludes the review when the following reviewers *reject* the link:

- Reviewers B, C and D
- Reviewers B, C, and E

In these cases, it is no longer possible for the sum of approval responses to exceed the threshold of 100. The review concludes with the decision to *reject* the link under review.

The Overruling Reviewer

When you define a set of reviewers for a Certify, Suggest, or Approve Changes node, you can designate one reviewer in the set as the overruling reviewer. When this reviewer completes the review item the following occurs:

- Their response concludes the review item. Related review items are deleted from the certification lists of the other reviewers.
- Their choice determines the result of the review. Responses from other reviewers are ignored.

Typically the overruling reviewer is a higher-level manager.

Note: You can define only one overruling reviewer in a set of reviewers.

Automatic Approvers

You can define filters that identify when specific users receive a review item, and automate their response. These reviewers are automatically assumed to approve the link or entity under review. The review item is omitted from their certification queue. If weighting is defined, the weight of their response counts as if they approved the link.

One common use of this option is to prevent duplicate reviews. For example, the manager of an entity typically receives the initial certification review of a certification. If they reject a link of the entity, typical node logic selects the same manager to approve the change that they themselves requested. To prevent this, the change approval node handles reviewers from the previous certification node as automatic approvers.

You can use the following automated approval filters:

- **By PersonID**: This filter is a comma-separated list of personID values or process parameters that resolve to personID values.

  Define this filter in the Auto Approvers field in the form-based interface of the Building Blocks tab, or in the autoApprovers data field of the node.

- **By Node**: This filter is a comma-separated list of certify, suggest, or approval nodes that precede the current node in the Workpoint process. CA Identity Governance builds a filter that contains all reviewers from the listed nodes.

  Define this filter in the Auto Approvers from Activity Name field in the form-based interface of the Building Blocks tab, or in the ignoreApproversFrom data field of the node.

After the product applies the logic you defined to select reviewers, this filter is applied. Reviewers that appear in the filter lists are treated as automatic approvers.

Note: When you define automatic approval filters, weighted approval criteria, or a subset of reviewers, the resulting logic can force the node to approve or reject the link under review in all cases. This forced response invalidates the purpose of the review node. Check the likely result when you apply these options to a set of reviewers.

Automatic Approvers by Node in Standard Certification Processes

Standard processes for certifications handle node-based automatic approver filters using a chain of parameters.

The following process-level parameter controls use of automatic approval in processes that certify existing links:

flow.requestChangeApproval.boolean.autoApproverCertify

Defines whether the Open Approval Action node of a certification process treats reviewers from the initial Certify node as automatic approvers. This boolean parameter is set when CA Identity Governance initiates a job based on the process.

Similarly, the following parameter controls use of automatic approval in processes that suggest new links during certification review:

flow.requestChangeApproval.boolean.autoApproverSuggest

Defines whether the Open Approval Action node of a certification process treats reviewers from the initial Suggest node as automatic approvers. This boolean parameter is set when CA Identity Governance initiates a job based on the process.

In addition, the following local variable is defined:

autoApproverNode

Specifies the node whose users are treated as automatic approvers.

These processes have two review nodes: an initial node named Certify or Suggest, and a second Open Approval Action node for approval.

When an initial review generates changes that require secondary review, the Set Auto Approvers node evaluates the process-level boolean parameter described above. When this parameter has the value True, the Set Auto Approvers node populates the autoApproverNode variable with the name of the initial node as follows:

```
autoApproverNode=Certify
```

or:

```
autoApproverNode=Suggest
```

Control then passes to the Open Approval Action node. In this node, the ignoreApproversFrom parameter is defined parametrically as follows:

```
ignoreApproversFrom={'autoApproverNode'}
```

The ignoreApproversFrom parameter resolves to the value inserted in the autoApproverNode variable by the Set Auto Approvers node, as follows:

```
ignoreApproversFrom=Certify
```

or:

```
ignoreApproversFrom=Suggest
```

The Open Approval Action node treats reviewers from the previous Certify or Suggest node as automatic approvers.

How to Define a Subset of Reviewers

You can activate a subset of the reviewers you defined. You can create a node with a general set of reviewers, but use different reviewers in each instance of the node, or in different jobs.

Use the following node data field to define a subset of reviewers:

useApprovers

Defines a subset of reviewers that participate in the review actions of the node. This field contains a comma-separated list of numbers that identify reviewers by their position in the set of reviewers. This value corresponds to the position of each reviewer in the table displayed in the Building Block tab, and to the position of each reviewer in the string data fields that define the matrix parametrically.

Example: A Subset of Reviewers

A set of five reviewers is defined. The useApprovers field is defined as follows:

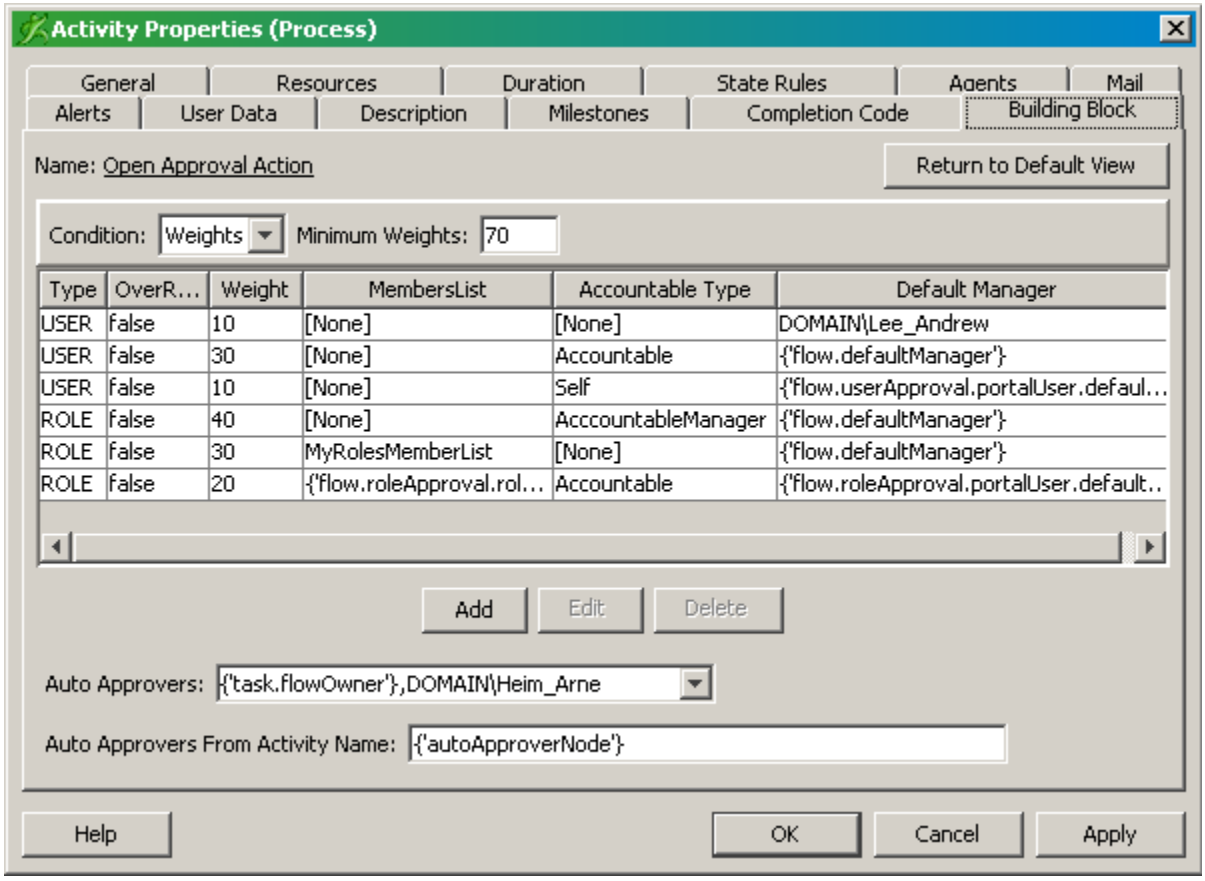
useApprovers=1,3,5

The first, third, and fifth reviewers participate in the review. CA Identity Governance ignores the second and fourth reviewer definitions.

Note: When you define automatic approval filters, weighted approval criteria, or a subset of reviewers, the resulting logic can force the node to approve or reject the link under review in all cases. This forced response invalidates the purpose of the review node. Check the likely result when you apply these options to a set of reviewers.

Example: Forms-Based Representation of a Set of Reviewers

Think of a set of reviewers as a matrix of settings. The form-based interface of the Building Blocks tab presents the set you define as a table. The following figure shows an OpenApprovalAction code block that defines a set of reviewers for a user-role link.



Each line of the table defines a reviewer.

The first three approvers reference the user entity under review, as follows:

- The first approver is always Andrew Lee.
- The second approver is selected by finding the user in the RACI configuration who is Accountable for the user entity under review.
- The third approver is the user under review, as indicated by the Self value in the Accountable Type column.

The last three reviewers reference the role entity under review, as follows:

- The fourth approver is the *manager of* the user in the RACI configuration who is Accountable for the role under review, as indicated by the AccountableManager value in the Accountable Type column.

- The fifth approver is selected by matching attributes of the role under review to the locally defined myRolesMemberList member list.
- The sixth approver is selected by matching attributes of the role under review to the member list specified by the flow.roleApproval.roleMemberslist parameter. If no reviewer is found, CA Identity Governance searches the RACI configuration for the user who is Accountable for the role under review.

A Default Manager is defined for all entries that search member lists or RACI to assign reviewers. If these methods do not find a reviewer, the review action is assigned to the default manager specified. No default manager is defined for the third approver (Accountable Type = Self) because the entity under review always exists.

None of the reviewers in this example is assigned Overrule privileges.

Weighted values are assigned to the reviewers, as indicated by the Weights value of the Condition field, and the values in the Weight column. CA Identity Governance concludes the review when the weighted sum of reviewers that approve exceeds the threshold value 70 shown in the Minimum Weights field.

Two automatic approval filters are defined, as follows:

- **By PersonID:** The Auto Approvers field lists two users who are treated as automatic approvers.
- **By Node:** The Auto Approvers from Activity Name field uses a process parameter to specify a node. The reviewers defined in that node are treated as automatic approvers.

Example: Parametric Representation of a Set of Reviewers

The set of reviewers shown in the previous example is defined by the following node-level user data values:

```
typeEntity1=USER
typeEntity2=ROLE
entity1.name1={'task.personId'}
entity2.name1={'task.roleName'}
mainEntities=1,1,1,2,2,2
entityForApproverSourceList=1,1,1,2,2,2
approversSource=,,,,myRolesMemberList,{'flow.roleApproval.roleMembersList'}
fallbackAccountableType=,Accountable,Self,AccountableManager,,Accountable
defaultManager=DOMAIN\Lee_Andrew,{'flow.defaultManager'},{'flow.userApproval.port
alUser.defaultManager'},{'flow.defaultManager'},{'flow.defaultManager'},{'flow.ro
leApproval.portalUser.defaultManager'}
aggregation=Weights
overruler=,,,,,
autoApprovers={'task.flowOwner'},DOMAIN\Heim,Arne
ignoreApproversFrom={'autoApproverNode'}
useApprovers=1,2,3,4,5,6
```

The following data fields correspond to fields and columns in the forms-based interface:

- mainEntities corresponds to the Type column.
- overruler corresponds to the Overruler column.
- approversSource corresponds to the MembersList column.
- fallbackAccountableType corresponds to the Accountable Type column.
- defaultManager corresponds to the Default Manager column.
- aggregation corresponds to the Condition field.
- autoApprovers corresponds to the Auto Approvers field.
- ignoreApproverseFrom corresponds to the Auto Approverse from Activity Name field.

The useApprovers field defines a subset of active reviewers. This field has no equivalent in the forms-based interface.

Optimized Handling of Workpoint Processes

Workflows such as large-scale certifications can involve many individual Workpoint processes. To improve performance, the product enacts the following two optimization behaviors. Consider these behaviors when you create custom processes.

- Progressive Launch - the product handles the first node of Workpoint processes before it actually initiates a job based on the process. The product immediately populates the certification queues with initial review items, and initiates Workpoint jobs as users respond to these items.

  Progressive launch works well with standard Workpoint processes, because most of these processes begin with Certify or Suggest nodes. When custom processes do not begin with a review item node, the product first initiates a job for each Workpoint process before further workflow handling.

  The following system property controls progressive launch of Workpoint processes:

  lazy.workflow.job.creation.enabled

  Specifies whether the product uses progressive launch. When this Boolean property is true, the product generates review items based on the initial review node of Workpoint processes before it creates a Workpoint job instance. The default value is True.

- Unilateral Cancel - During a workflow, an item can be canceled or become irrelevant. For example, a manager can terminate a request for several new privileges when only some of them are approved. The product marks these items internally as canceled, but does not continue to manage the corresponding Workpoint process. When a custom process includes conditions and nodes to handle a canceled process, the product does not return the cancelation status to the Workpoint process, and does not handle those nodes.

  The following system property controls unilateral cancelation of Workpoint processes:

  aggregated.workflow.job.cancel.enabled

  Specifies whether the product maintains Workpoint processes that correspond to canceled workflow items. When this Boolean property is true, the product continues to interact with Workpoint processes after their workflow items are canceled. The default value is False.

Chapter 8: Standard CA Identity Governance Workpoint Processes

This section contains the following topics:

Overview of Standard Workpoint Processes

Each type of task in a business workflow is associated with a Workpoint process. In a business workflow, CA Identity Governance creates a Workpoint job based on the process for each task of that type in the business flow.

This section provides an overview of the Workpoint processes that are installed with CA Identity Governance. These process support standard business workflow behaviors, and are mapped by default to the tasks of business workflows.

Workpoint Processes for Entity Certification

The following standard Workpoint processes implement individual review tasks of certification. Typically, each task includes the following logic:

- An initial Certify or Suggest node implements primary certification review.
- An optional Change Appovals section implements secondary approval of new links or changes to the existing link under review. This section uses the following nodes:
  - Need Approval
  - Set auto approval node
  - Open Approval Action
- A final section that executes requested changes in the target configuration. The node used reflects the workflow task implemented by the process.
- Nodes that export configuration changes to related endpoints. The Run Export and Export nodes support this optional functionality.

The logic of these processes is almost identical. They are distinguished by parameter declarations that reflect the entities under review in each process.

In addition to these processes for individual certification tasks, CA Identity Governance also implements the AggCert control process in the certification workflow context. This control workflow handles aggregation of child tasks during the certification.

The following processes implement certification tasks in User Certification certifications:

UcertUAcc

Certifies user-account link

USuggURole

Suggests new user-role link

USuggURes

Suggests new user-resource link

UCertURole

Certifies user-role link

UCertURes

Certifies user-resource link

The following processes implement certification tasks in Role Certification certifications:

RoleSuggURole

Suggests new user-role link

RoleSuggRoleRole

Suggests a new link between a role and its child role

RoleSuggRoleRes

Suggests new role-resource link

RoleSuggRolePaRo

Suggests a new link between a role and its parent-role

RoleCertURole

Certifies a user-role link

RoleCertRoleRole

Certifies a link between a role and its child role

RoleCertRoleRes

Certifies a role-resource link

RoleCertRolePaRo

Certifies a link between a role and its parent-role

The following processes implement certification tasks in Resource Certification certifications:

ResSuggURes

Suggests new user-resource link

ResSuggRoleRes

Suggests new role-resource link

ResCertURes

Certifies user-resource link

ResCertRoleRes

Certifies role-resource link

The following processes implement certification tasks in Account Certification certifications:

AccCertAUser

Certifies user links for each account

More information:

Assigning Reviewers (see page 83)

Processes for Approval and Implementation of Changes

The following standard Workpoint processes implement tasks for self-service business workflows. Typically, requests to change privileges correspond to individual, independent tasks that include the following logic:

- An initial Open Approval Action node submits the proposed change for review.
- A final section that executes requested changes in the target configuration. The node used reflects the task implemented by the process.
- Nodes that export configuration changes to related endpoints. The Run Export and Export nodes support this optional functionality.

In addition to these independent tasks, high-level processes implement more complex workflows that involve child tasks, such as creation or deletion of a role.

The following processes implement tasks to create a role:

BBNewRole

Controls the role creation workflow

NewRoleUser

Connects a user to a new role

NewRoleRole

Connects a role to a new role

NewRoleRes

Connects a resource to a new role

The following processes implement tasks to delete a role:

BBDelRole

Controls the role deletion workflow

DelRoleUser

Disconnects User-Role link

DelRoleRole

Disconnects Role-Role link

DelRoleRes

Disconnects Role-Resource link

The following processes implement general self-service request tasks:

BBURole

Connects User to Role

BBURes

Connects User to Resources

BBUpdateRole

Updates Role

BBRoleRole

Connect Role to Role

BBRoleRes

Connects Role to Resource

BBDelURole

Disconnects User-Role link

BBDelURes

Disconnects User-Resource link

BBDelRoleRole

Disconnects Role-Role link

BBDelRoleRes

Disconnects Role-Resource link

Processes for Flow Control and Aggregation

In both campaign and self-service flows, review tasks can be aggregated: all reviews of a certain stage of the workflow must complete before the workflow continues. The following standard processes control aggregation behavior:

AggApproval

Implements logic to support aggregated approvals or rolling approvals in self-service workflows.

AggCert

Implements logic to support aggregated approvals or rolling approvals in certification campaigns.

Similarly, the following high-level parent processes control the progress of child tasks:

BBNewRole

Controls the workflow to create a role

BBDelRole

Controls the workflow to delete a role

BBUpdateRole

Controls the workflow to update a role

In response to workflow control actions (see page 81) submitted by the workflow owner, CA Identity Governance advances these control processes and initiates aggregated child tasks.

More information:

Parent and Child Processes in a Workflow (see page 79)

Chapter 9: Standard CA Identity Governance Building Blocks

This section contains the following topics:

Overview of Building Blocks

Building blocks are Workpoint node templates that invoke services and modular behaviors on the CA Identity Governance server. Each building block exposes customized fields that support interaction with CA Identity Governance. CA Identity Governance packages these building block templates in a customized version of the Workpoint Designer application, which is installed with the CA Identity Governance server.

Workpoint processes that implement CA Identity Governance workflow tasks use nodes based on these building block templates to interact with CA Identity Governance. Generally, Workpoint processes correspond to *tasks* of the CA Identity Governance workflow, and individual nodes and building blocks often generate *actions* for reviewers and other business users. One task can generate several actions.

This section describes the building blocks exposed by CA Identity Governance.

Complete Task Building Block

Use this building block to create a node that concludes the process. This building block signals CA Identity Governance when the process concludes.

Note: End every process with a node based on this building block.

Building Blocks for Business User Actions

This section describes building blocks related to review, certification, and approval of access privileges by managers, resource owners, and other business users.

Open Certification Action

Use this building block to create a node that submits an entity or link for review. Typically this building block is used for the initial review in a certification workflow.

Note: For more information about this node and the parameters that it exposes, see the section on assigning reviewers (see page 83).

Open Suggest Action

Use this building block to create a node that submits suggested new entities or links for review. Typically this building block is used to supplement initial review in a certification workflow.

Note: For more information about this node and the parameters that it exposes, see the section on assigning reviewers (see page 83).

Open Approval Action

Use this building block to create a node that submits proposed changes to an entity or link for review. Typically this building block is used in a certification workflow to review changes to the configuration that result from initial review.

Note: For more information about this node and the parameters that it exposes, see the section about assigning reviewers (see page 83).

Decision Button

Use this building block to create a node that prompts users for workflow control input.

This building block exposes the following parameters:

decisionOwner

Specifies the personID of the user that receives the workflow control action. If this field is null, the flow owner receives the control action.

availableResults

Specifies the options that are presented to the reviewer in a workflow control action. This value is a comma separated list of options such as Approve, Reject, Cancel, Create Role.

userActionMessage

Specifies the strings that label the options presented in a workflow control action. This value is a comma separated list of keys in the CA Identity Governance server localization files. Each item in the list corresponds to a value in the availableResults parameter.

setResult

Specifies a node parameter that receives the result value chosen by the reviewer. CA Identity Governance returns this value when the reviewer submits the workflow control action.

Get Approvers from Node

Use this building block to create a node that retrieves a list of reviewers from other nodes in the process.

This building block exposes the following parameter:

nodesNames

A comma-separated list of nodes in the process that generate review actions.

Building Blocks for Parent-Child Process Handling

This section describes building blocks related to parent-child process interaction to support parallel or aggregated tasks in a workflow.

Note: For more information, see the section on Parent and Child Processes (see page 79).

Start Children

Use this building block to create a node that launches child processes.

More information:

Parent and Child Processes in a Workflow (see page 79)

Signal Children and Wait

Use this building block to create a node that sends a signal key to child processes and waits for a return key. Typically this building block is used to initiate change approvals or other subsequent nodes in the child process.

More information:

Parent and Child Processes in a Workflow (see page 79)

Signal Parent / Signal Parent and Wait

Use these building blocks to create a node in a child process sends a signal key to its parent process.

Note: For more information, see the section on Parent and Child Processes (see page 79).

More information:

Parent and Child Processes in a Workflow (see page 79)

Wait for Parent

Use this building block to create a node that suspends a child process until its parent process sends a signal key.

Note: For more information, see the section on Parent and Child Processes (see page 79).

More information:

Parent and Child Processes in a Workflow (see page 79)

Disable User Actions

Use this building block to create a node that terminates incomplete review actions in child processes.

This building block exposes the following parameter:

SourceNodeNames

Specifies the nodes in the child processes that the active review actions you want to end.

Building Blocks for Universe-Level Data Manipulation

This section describes building blocks related to the target universe and configuration files of a workflow process.

Get Universe Master / Get Universe Model

Use these building blocks to create a node that retrieves the filename of the master or model configuration.

Get Configuration UDB / Get Configuration RDB

Use these building blocks to create a node that returns the filename of the user database or resource database that a configuration references.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Create Sub-configuration

Use this building block to create a node that creates a subset of a configuration.

This building block exposes the following parameters:

subConfigurationName

Defines the name of the sub-configuration that CA Identity Governance creates in the universe.

usersFilter

Specifies the filter that selects user entities for the sub-configuration.

rolesFilter

Specifies the filter that selects role entities for the sub-configuration.

resourcesFilter

Specifies the filter that selects user entities for the sub-configuration.

Compare Audit Cards

Use this building block to create a node that submits two audit cards in a configuration to CA Identity Governance for comparison. CA Identity Governance writes the results to another audit card in the configuration.

This building block exposes the following parameters:

auditCard1

Specifies one of two audit cards that CA Identity Governance compares.

auditCard2

Specifies one of two audit cards that CA Identity Governance compares.

resultingAuditCard

Specifies the name of an audit card that CA Identity Governance creates in the universe. This audit card contains the results from comparison of two audit cards.

Building Blocks for Entity and Link Operations

This section describes building blocks used to define, query, and modify entities and links, and their attributes.

Add User / Role / Resource

Use these building blocks to create a node that adds a user, role, or resource entity to a configuration file.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Delete User / Role / Resource

Use these building blocks to create a node that removes a user, role, or resource entity from a configuration file.

These nodes use general process parameters (see page 69), user, role, and resource parameters (see page 69), and the following parameter:

totallyRemove

Determines whether a user or resource entity is removed only from the target configuration, or from both the configuration file and the user and resource it references. When this Boolean parameter is true, CA Identity Governance removes the entity from the user or resource database when no other configuration file in the universe contains the entity.

Update User / Role / Resource

Use these building blocks to create a node that updates the attribute values of an entity.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Get User / Role / Resource Field

Use these building blocks to create a node that retrieves the value of a user, role, or resource attribute.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Connect User Role / User Resource / Role Resource / Roles

Use these building blocks to create a node that creates a link between two entities.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Disconnect User Role / User Resource / Role Resource / Roles

Use these building blocks to create a node that removes a link between two entities.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Update Link User Role / User Resource / Role Resource / Roles

Use these building blocks to create a node that updates the attribute values of a link between two entities.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Get User Role / User Resource / Role Resource / Roles Link Field

Use these building blocks to create a node that retrieves the value of a link attribute.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

Are User Role / User Resource / Role Resource / Roles Connected

Use these building blocks to create a node that checks whether two entities are linked in a configuration.

Are User Role / User Resource / Role Resource / Roles Connected

Use these building blocks to create a node that checks whether two entities are linked in a configuration.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69) and the following parameter:

associationType

Specifies the type of link to verify between two entities. Valid values include:

Direct

The entities are linked directly

Indirect

The entities are linked through other entities.

Dual

The entities are linked both directly and indirectly.

A null value checks for any type of link between the entities.

Enrichment

Use this building block to create a node that enriches user and resource endpoint data with supplementary data.

This node uses the following parameters:

userEnrichmentFile

Specifies the file that contains user enrichment data.

userEnrichmentMatch

Specifies the match between the users database and the enrichment file. For example, PersonID and UserID.

userEnrichmentOptions

Enrichment options for user data. For example, clear, update, case sensitive.

resourceEnrichmentFile

Specifies the file that contains resource enrichment data.

resourceEnrichmentMatch

Specifies the match between the resources database and the enrichment file. For example, resName3 and UserID.

resourceEnrichmentOptions

Enrichment options for resource data. For example, clear, update, case sensitive.

roleEnrichmentFile

Specifies the file that contains role enrichment data.

roleEnrichmentMatch

Specifies the match between the roles database and the enrichment file. For example, resName3 and UserID.

roleEnrichmentOptions

Enrichment options for role data. For example, clear, update, case sensitive.

Building Blocks for RACI and Stakeholder Operations

This section describes building blocks related to the RACI configurations of the target universe, and the managers associated with an entity.

Get User / Role / Resource Manager

Use these building blocks to create a node that retrieves the manager of a user, role, or resource attribute.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69).

The Get User Manager parameter returns the value of the managerID field for the user record.

The Get Role Manager and Get Resource Manager building blocks query the RACI configurations, and expose the following parameter:

managementType

Specifies the RACI relationship between the entity and the manager. Valid values are R, A, C, and I.

Set User / Role / Resource Manager

Use these building blocks to create a node that defines the manager of a user, role, or resource attribute.

Note: The Set User Manager parameter changes the value of the managerID field in the specified configuration file, but not in the user database.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69) and the following parameter.

managerPersonId

Identifies the manager by their personID in the user database.

Add Role / Resource Stakeholder

Use these building blocks to create a node that defines a RACI stakeholder for a role or resource.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69) and the following parameters

managerPersonId

Identifies the manager by their personID in the user database.

relationshipType

Specifies the RACI relationship between the entity and the manager. Valid values are R, A, C, and I.

Remove Role / Resource Stakeholder

Use these building blocks to create a node that deletes a RACI stakeholder for a role or resource.

These nodes use general process parameters (see page 69) and user, role, and resource parameters (see page 69) and expose the following parameter:

relationshipType

Specifies the RACI relationship between the entity and the manager. Valid values are R, A, C, and I.

Building Blocks for External Interactions

This section describes building blocks that communicate with provisioning endpoints, mail servers, and other external platforms.

Send Mail

Use the Send Mail building block to create a node that sends email to a group of users. CA Identity Governance processes the request using its default email target and other settings.

This building block exposes the following parameters:

emailTemplateName

Specifies which template to use for the email. Of the templates defined in the CA Identity Governance server, this building block uses only templates associated with the Customized Process Email event.

Note: This building block sends the same email text to all recipients. Do not include personal data fields in email templates that you use with this building block.

The following default templates, installed with CA Identity Governance, are used for aggregated workflows:

- CommitChanges - Concludes a self-service request and implements the review decisions that were submitted.
- StartApproval - Initiates the change approvals phase of a certification.

To

Defines a list of users that receive the email. This value is a comma-separated list of email addresses or CA Identity Governance users. You can also specify parameters that contain lists of email addresses or CA Identity Governance users.

emailParams

Defines values for variable data fields in the email template. This value is a comma-separated list of variable=value statements. For example, if the template contains the ${campaignType} and ${flowName} variables, you can assign static values as follows:

```
campaignType="Annual Privileges Review", flowName="Managers Phase"
```

Notify Model Change / Notify Aggregate Model Change

Workflows can result in changes to the model configuration of a universe. You can export these changes to provisioning endpoints. CA Identity Governance communicates these changes using the Model Event Notification API (see page 11).

Workpoint processes query the following CA Identity Governance system parameters to determine the behavior that is enabled on the CA Identity Governance server:

approval.isModelChangeNotificationOn

Indicates whether you enable the model event notification.

modelEvent.producer.fireEvent

Specifies what events trigger change notification. Valid values include:

Atomic

Triggers an event after every approval during a workflow.

Aggregate

Triggers an event when all approvals complete for a workflow.

Use the Notify Model Change building block to create a node that exports changes that result from a single review task. Your process logic activates this node when modeEvent.producer.fireEvent is Atomic.

Use the Notify Aggregated Model Change building block to export all the changes that result from an entire workflow. Your process logic activates this node when modeEvent.producer.fireEvent is Aggregate.

The Notify Model Change building block exposes the following standard parameters:

- A set of node parameters that specify the focal entities (see page 85) of the process.
- General process parameters (see page 69) that identify the target universe and configuration file.

In addition, this building block exposes the following unique parameters:

Action

Describes the change to the specified link or entities. Valid values include:

Add

Indicates a new entity, attribute value, or link between specified entities.

Delete

Indicates that the entity, attribute, or link specified is removed from the configuration file.

More information:

Model Event Notification API (see page 11)

Building Blocks for General Logical Operations

This section describes building blocks used to evaluate mathematical expressions, regular expressions for text patterns, and perform other calculations or logical operations.

Evaluate Binary Expression

Use this building block to create a node that submits a binary expression to CA Identity Governance for evaluation.

This building block exposes the following parameters:

setResult

Indicates the result of a calculation or logical evaluation in a node. Typically this parameter receives the result of an agent script.

binaryExpression

Specifies a binary statement.

Evaluate Math Expression

Use this building block to create a node that submits a mathematical expression to CA Identity Governance for evaluation.

This building block exposes the following parameters:

setResult

Indicates the result of a calculation or logical evaluation in a node. Typically this parameter receives the result of an agent script.

mathExpression

Specifies a mathematical statement.

Does Value Match Regular Expression

Use this building block to create a node that submits a regular expression to CA Identity Governance for evaluation. Typically this building block is used to apply a pattern-based filter or test to attribute values.

This building block exposes the following parameters:

setResult

Indicates the result of a calculation or logical evaluation in a node. Typically this parameter receives the result of an agent script.

regularExpression

Specifies a regular expression statement that is used to test a string value.

Note: For information about how CA Identity Governance evaluates regular expressions, see the *Configuration Guide* and the *Client Tools Guide*.

value

Specifies a text string that is tested against a regular expression statement.

Building Blocks for Analytics

This section describes building blocks used to evaluate analytical processes.

Does Violate Rule-Based Role

Use this building block to create a node that defines if a rule-based role violation was triggered. Typically this building block is used in Add New Role, Update Role, and Add User-Role Link processes.

Transformation Building Blocks

The following three new workflow building blocks can be added to a workflow process by using the Workpoint Designer:

- ExecutePDIFile—accepts a transformation file stored locally on the file system. Provide the full path for the file name parameter. This option is a fast way to deploy a PDI transformation.
- ExecutePDIPackage—runs a transformation that has been loaded onto the server. The name of the package in the PDI_BUNDLE_NAME parameter should be the same name you use when uploading the transformation to the server. This option is more suitable for cluster deployments, as the package does not need to be local.
- ExecuteCMD—calls a command line executable that runs locally on the server.

Providing parameters for the two PDI building blocks (ExecutePDIFile and ExecutePDIPackage) is done with the PARAMS field. It accepts a list of parameters that are separated by commas, and start with "-param:". For example:

```
"-param:configurationName=model","-param:target=newModel"
```

In the previous example, the PDI transformation gets two parameters, "configurationName" with the value "model" and "target" with the value "newModel".

The parameters for the ExecuteCMD building block are as follows:

- CMD_SCRIPT_FILE—the name of the command. For example, "run.bat". The CMD_SCRIPT_FILE must exist in the file system. CMD commands (like "copy") cannot be used.
- CMD_WORKING_FOLDER—the name of the folder used as the starting point for the command.
- PARAMS—the list of parameters separated by commas. For example:

  ```
  PARAMS = ASGFASDGSDFG@$#!@","-value2"," ASGFASDGSDFG@$#!@"
  ```

Encrypting Values

You can provide encrypted values for passwords or other sensitive data that you do not want to save as clear text.

Follow these steps:

1. As an Administrator in the Portal, go to Administration, Workflow Settings, Encryption.
2. Enter the password or sensitive data in the Clear text field.
3. Click Encrypt.

4. Copy the resulting value in the field on the right and use this encrypted value as the value of the parameter.
5. State which parameters are encrypted by providing the position of encrypted parameters in the ENCRYPTED_PARAMS_INDICES_ZERO_BASED field.

   For example, if you have the following PARAMS value:

   PARAMS =
   "-param:pass="ASGFASDGSDFG@$#!@"","-param:configurationName=RCM","para
   m:pass2="ASGFASDGSDFG@$#!@""

   The value of ENCRYPTED_PARAMS_INDICES_ZERO_BASED must be as follows:

   ENCRYPTED_PARAMS_INDICES_ZERO_BASED = 0,2

   The previous line indicates that the first (starting from zero) and the third parameters are encrypted.

Appendix A: CA Identity Governance Data Files

CA Identity Governance uses three separate but related files in a text-based, comma-separated format to represent a configuration.

The user and resource database files contain the basic details of users and resources. The configuration file contains the dynamic parts of a configuration; that is, the role and relationship information.

This section contains the following topics:

User Database File

User database file names end with the .udb suffix. Each user is represented in this file by one line, which includes comma-separated values for the following fields (in this order):

- PersonID (the key)
- User name
- Organization name
- Organization type
- (Optional) an unlimited number of additional fields.

Although they are optional, CA Identity Governance requires you to specify fields for the following types of user information when you define a universe. Define these fields in .udb files that form the basis for a configuration file in a universe.

- LoginID
- User email
- ManagerID

Example: User Database File

The following sample .udb file contains 3 user records.

```
PersonID,UserName,OrgName,OrgType,Country,Location,ManagerID,email,LoginID,
"52656727","Rodman Adam","System
Management","Corporate","US","Pennsylvania","54672910","52656727@company.com","IB
MR50\\Rodman Adam",
"54672910","Cooper Amos","IT
Security","Corporate","US","Pennsylvania","64646410","54672910@company.com","IBMR
50\\Cooper Amos",
"64646410","Herman Barbara","Operations","Corporate","US","New
Jersey","64646410","64646410@company.com","IBMR50\\Herman Barbara",
```

Resource Database File

Resource database file names end with the .rdb suffix. Each resource is represented in this file by one line, which includes comma-separated values for the following fields (in this order):

- Resource Name 1 (ResName1)
- Resource Name 2 (ResName1)
- Resource Name 3 (ResName1)
- (Optional) An unlimited number of additional fields

The ResName fields typically map to the endpoint or application group of the resource.

Although they are optional, CA Identity Governance requires you to specify fields for the following types of resource information when you define a universe. Define these fields in .rdb files that form the basis for a configuration file in a universe.

- Application
- ManagerID

Example: Resource Database File

The following sample file contains 3 resource records.

```
ResName1,ResName2,ResName3,Description,ManagerID-Owner,Location,
"SYS1","RACFPROD","RACF22","Production RACF","77292450","Irvine,CA",
"Domain Users","NT5AVE","WinNT","Active Directory ","91236370","Houson,TX",
"DEVELOP","RACFPROD","RACF22","Production RACF","77292450","Irvine,CA",
```

Configuration File

Configuration file names end with the .cfg suffix. The configuration file refers to a user database file and a resource database file. It contains role definitions and links between users, roles, and resources.

Note: Multiple configurations may share the same user and resource database files.

The configuration file contains the following elements:

- A header section lists the owner and modification history of the file. The first two lines in the file specify the user and resource database files that the configuration references. These lines have the following format:

  UsersDB,*udb_pathname*
  ResDB,*rdb_pathname*

  Note: *udb_pathname* is the pathname of the referenced user database file, and *rdb_pathname* is the pathname of the referenced resource database file.

- User entity declarations define a subset of users from the referenced user database file. Each line defines a single user, with the following format:

  User, *udb_record*, *PersonID*

  Note: *udb_record* is the index value of a record in the user database file. The first user record in the .udb file has an index value of zero. *PersonID* is the value of the PersonID field in the referenced user record.

- Resource entity declarations define a subset of resources from the referenced resource database file. Each line defines a single resource, with the following format:

  Res, *rdb_record*, *ResName1*, *ResName2*, *ResName3*

  Note: *rdb_record* is the index value of a record in the resource database file. The first user record in the .rdb file has an index value of zero. *ResName1*, *ResName2*, *ResName3* are the values of the corresponding mandatory fields in the referenced resource record.

- Role declarations define a role in terms of users, resources, or other roles in the configuration. Each declaration defines a single role in one line, with the following format:

  `Role,roleID,roleName,roleDescription,roleOrganization,roleOwner`

  Note: *roleID* is the numerical identifier CA Identity Governance assigns to the role, *roleName* is the unique name of the role, *roleDescription* is a text description of the role, *roleOrganization* is the organization associated with the role, and *roleOwner* is the user that owns the role.

- Link declarations define role contents and user privileges as a set of links between the declared user, role and resource entities. Each line defines a single link, with the following format:

  Link_type,Entity1,Entity2

  Note: *Link_type* specifies the type of link. *Entity1* and *Entity2* specify the linked entities, using the record index of a user or resource entity, or the roleID of a role entity.

  The *Link_type* string can have the following values:

  - User-Res: user-resource link
  - User-Role: user-role link
  - Role-Res: role-resource link
  - Role-Role: role-role link (parent-child link within the role hierarchy)

  Entities must be listed in order. For example, in a User-Res declaration, the first entity is a user record, and the second entity is a resource record. In a Role-Role link, the first entity is the roleID of the parent role, and the second entity is the roleID of the child role.

Example: Configuration File

Configuration files are typically much larger than this sample. In this example, role 1001 has only one resource, role 1014 has two resources, and role 1015 includes both role 1001 and role 1014 as children.

```
UsersDB,.\UsersDB.udb
ResDB,.\ResDB.rdb
CreateDate,03/09/2007 12:27
ModifyDate,03/09/2007 12:27
StatusDate,17/04/2007 15:36
Owner1,Ilan Sharoni
Organization1,Company
```

```
Owner2,
Organization2,
Operation1,
Operation2,
Operation3,
Status,
ParentConfigName,SQL://(local).sdb/ConfigWithRoles.cfg
User,0,"45489940"
User,1,"47868650"
User,2,"52656727"
Res,0,"APPLDEV","RACFTEST","RACF22"
Res,1,"BRLIMSYS","RACFPROD","RACF22"
Res,2,"DEVELOP","RACFPROD","RACF22"
Role,1001,"BASIC ROLE","Basic role - for all IT users","Enterprise","82922230","Org
Role","","45489940","Approved","09/05/2007 10:36","No
Rule","Enterprise","Corporate",""
Role,1014,"Title - Product Manager","Characteristic Role (50%)","Title - Product
Manager","99883135","Org Role","","45489940","Approved","09/05/2007
10:36","Title=Product Manager;","Title","Corporate",""
Role,1015,"Title - Operator","Characteristic Role (50%)","Title -
Operator","45489940","Org Role","","45489940","Approved","09/05/2007
10:36","Title=Operator;","Title","Corporate",""
User-Res,0,2
User-Res,0,1
User-Role,1,1001
User-Role,2,1014
Role-Res,1001,0
Role-Res,1014,1
Role-Res,1014,2
Role-Role,1015,1014
Role-Role,1015,1001
```