Cúram 8.1.2

Workflow Overview Guide

Note

Before using this information and the product it supports, read the information in Notices on page 33

Edition

This edition applies to Cúram 8.1, 8.1.1, and 8.1.2.

Contents

1 Cúram Workflow overview

Workflow supports the automation of business processes and the routing of work among individuals and departments. The main components of the system are workflow process definitions, the workflow engine, inbox and task management tools, and the process definition tool.

1.1 Workflow Management System Overview

An overview of the main components of the workflow management system. This includes process definitions, the workflow process definition tool (PDT), the workflow engine, the tasks and inbox management, and the workflow administration.

Workflow Management System Functions

The workflow management system serves two main functions. The first is to support the automation of business processes. The second is to facilitate the routing of work among individuals and departments.

The main components of this system are workflow process definitions, the workflow engine, the inbox and task management, and the workflow Administration application, which includes the process definition tool (PDT). Each one of these plays a role in one or both of these functions.

The automation of a business process starts with the design of a workflow for a business process. To map a business process to workflow process definition, key decisions must be made. For example, there is an expectation that manual steps are involved in the business process or the business process may require configuration over the lifetime of the process. The workflow PDT is used to create the workflow process definition based on this design. By using the PDT, a workflow developer defines the workflow activities, the transitions between these activities, and the information that passes through the workflow process.

The following are some of the activity types available in a process definition: manual, automatic, loop, event wait, route, notification and subflow. The PDT is used to depict the sequence of a workflow process using the above activity types. The above activities are automatically preceded with a start process activity and conclude with an end process activity type by the system, followed by any transitions or looping activities in between the workflow processes.

The workflow engine performs the runtime execution of the workflow process definition, that is, process enactment. As a result of enactment, a workflow process instance is created. These workflow process instances can be monitored and controlled using the workflow Administration functions that are provided.

The routing of work among individuals and departments also starts with the workflow process definition. Certain activity types represent work that needs to be completed by an individual or department; for these activities, the workflow process definition includes a strategy for assigning this work. The workflow engine evaluates the allocation strategy to determine who should complete this work, creates tasks for this work, and assigns the tasks to the appropriate users, organizational objects, for example, organization units, positions or job, or work queues. The Inbox provides a user with information about the tasks that they need to action.

When to Use the Workflow Management System

The workflow management system provides the following key benefits. This information also describes the trade-offs to consider when you decide to use workflow to complete a business process.

The following are the key benefits.

- Allocation - It can handle complex allocation logic for assigning work to users.
- Consistency - It provides process automation which can improve consistency of outcomes.
- Flexibility - It provides software control over processes which enables re-design in line with changing business needs.
- Traceability - It provides a visual representation of when actions were taken, by whom and what stage a given process is at.
- Customizability - Customers can re-sequence workflows provided by the application to suit their process needs.

However, you must also take the following trade-offs into account.

- Performance - There is state maintenance overhead in enacting workflow processes.
- Atomicity - Workflows are not atomic as each step can execute within its own transaction. This makes failure in later transactions difficult to recover from.

Since workflow is the automation of business processes, consider the following guidelines when deciding when to use workflow management to automate specific business processes.

- Don't do any design until you have the requirements. Documented business process requirements are a prerequisite when considering when to apply workflow.
- A workflow should be considered when a business process requires human interaction.
- A workflow should not be used if a process can be automated end-to-end with no human interaction because using workflow in this scenario provides no benefit. It does not add any consistency, customizability or traceability above straight API calls and such the process contains no steps that require allocation. Therefore, unless there is a flexibility requirement to be able to re-sequence the steps at runtime, workflow should not be considered for such scenarios.
- When presented with sequences of automated steps in a business process, consider providing APIs for each step and composing those API calls into a single method (which can then be called from the workflow)

The Workflow Process Definition Tool

Use the workflow process definition too (PDT) to create workflow process definitions. The PDT also contains a library for the business methods, both Business Process Object (BPO) and entity methods, that are available for use by the workflow process definitions.

The main function of the PDT is to create workflow process definitions by defining the activities in a workflow and the transitions between them. There are various activity types to choose from when creating a process definition, each one of which performs a different function. What occurs between the activities in a workflow depends on their transitions, the conditions for these transitions, and the data that is passed between them. For further details on workflow process definitions see 1.2 Workflow Process Definitions on page 13.

The PDT includes a visualization tool which allows the workflow developer to view a version of the workflow process definition.

The PDT also validates workflow process definitions before the workflow developer releases them. It will check the process definition against a series of validations and report any errors for the workflow process on the whole, for activities, or for transitions. These validations assist a developer in producing a valid and well formed workflow.

Workflow Data Objects

Workflow Data Objects (WDOs) are the set of variables that transport data between activities in a workflow process. WDOs are mapped to activities in order to define the parameters for business processes which are called by the activity. For example, if information regarding a case and its owner is needed to perform an activity, a WDO can be mapped to that activity. This WDO would include two attributes: one for the case identifier and the other for the case owner.

WDOs must be added to a workflow process definition so that they can be mapped to the activities. Templates can be created for WDOs that are used in a number of workflow process definitions, for example, a template can be created for the case identifier. This eliminates the need to manually add the same WDOs to every workflow process definition.

Method References Library

The Method References Library contains references to all the BPO and entity methods from the application codebase which are available for use by the PDT when creating workflow process definitions. For example, a business method that is required to be executed by an automatic activity in a workflow process first needs to be added to this library before it may be selected for use in the PDT. The same rule applies to those methods used as allocation functions or deadline handler functions in a process definition.

The Workflow Engine

The workflow engine manages the process instance lifecycle by executing activity instances and evaluating transition rules. During the process instance lifecycle, the workflow engine continues to respond to events, such as completing a task, which tell it to resume the execution of a process instance.

The workflow engine creates tasks instructing users on the work that needs to be completed manually, and evaluates the allocation strategy to determine which users should be assigned these tasks. The workflow engine also creates notifications for users to inform them of the progress or status of a workflow process instance.

The workflow engine manages each process instance until the end process activity for that instance is reached. The execution of this activity indicates the completion of the process instance. If the workflow process instance fails, the workflow engine will record information about the failure. A workflow administrator can then use this information to retry the workflow process instance from the failure point.

For further details on the workflow engine, see 1.3 The Workflow Engine and Enacting Workflow Processes on page 22.

Tasks and Inbox Management

Tasks are used to assign and track the manual work of users in the application. The workflow engine creates tasks to complete manual activities and assigns these tasks to users based on an allocation strategy. All users manage their task loads from the Inbox.

Task deadline strategies can also be created to ensure that the task is actioned and completed in a timely manner. These deadline strategies can be defined to deal with any tasks that exceed defined deadlines.

The Inbox provides views to allow users to see the tasks they are currently working on and tasks that are available to them to work on. Notifications are also presented to the user in their Inbox. Users may also create and run task queries to better filter the tasks that are available to them to work on. Users can also subscribe to work queues and retrieve tasks to work on from work queues in the inbox.

The task management system also provides functions to allow users to manage individual tasks. Example of these functions include forwarding a task, reallocating a task and changing the amount of time worked on a task. 1.4 Managing My Inbox and Tasks on page 24 provides more details on the Inbox and task management area of the workflow management system.

Workflow Administration

Each time a workflow process definition is enacted in the application, the workflow engine creates new process instances. Administrators can monitor and control these process instances in Workflow Administration.

An administrator can search for process instances based on details of the process definition used to create that instance, the tasks related to a process instance, or by the events that the activity instances contained within the process instance are waiting on.

Administrators can also view the details of a process instance. The details include a graphical view of the process instance which displays all of the activities in the associated process definition as well as highlighting those that have actually been executed, that is, the path through the process. Administrators can also see details of the data that is used in that process instance as well as the data for each activity instance within the process instance.

The state of a process instance can also be managed as functions available to suspend a process instance, resume a process instance or indeed abort a process instance. Errors may also occur during the lifecycle of a process instance and these are also displayed here. An administrator can view the details of the error and based on this information can retry or discard the associated error.

Manual and decision activities both have allocation strategies which determine the users who will be assigned to the tasks created by the workflow engine for these activity types. Workflow allocation targets are one type of allocation strategy that can be used for this purpose and the creation and maintenance of these targets are managed here. Events are raised by application functions and the workflow engine uses them to transition workflows that are waiting on those events. These are also created and maintained by using the available Workflow Administration functions.

For more information about the available administration functions, see 1.5 Administering Workflow on page 30.

1.2 Workflow Process Definitions

Use this information to learn about workflow process definitions, and their main components such as activity types, transitions, and notifications. How to analyze a business process and design a workflow process definition by using the results of the analysis are also described.

A workflow process is enacted at runtime based on a workflow process definition. A workflow process definition describes the flow of a business process in terms understood by the workflow engine. It defines the activities in the workflow and the transitions between them. It also defines the data that is passed into and out of an activity during the execution of a workflow process instance.

Business processes that must be fully or partially automated have workflow process definition requirements. Depending on the type of processing, the workflow might contain various different types of activities. For example, where manual steps are required, a task needs to be generated to represent this piece of work, therefore the workflow process definition must contain a manual activity. Some business processes can be very complex and require a combination of different activity types that need to be executed in a specific sequence. The workflow process definition allows the designer to define the path between activities using transitions. For more information about the metadata associated with a process definition, see the *Workflow Reference Guide*.

Activities

An activity in a workflow process definition defines a discrete piece of work that must be completed before a workflow process can progress. Use this information to learn about the different types of activities and activity notifications.

Start Process and End Process

All workflow process definitions have a start and end activity which are automatically added by the PDT when a new workflow process definition is created. When a workflow is enacted, the transitions from the start activity are the first components to be evaluated. A workflow is completed when the transitions to the end activity are evaluated and the end process activity is executed.

Start activities must have at least one outgoing transition but no incoming transition. End activities must have at least one incoming transition but no outgoing transition. For more information about these activities, see the *Workflow Reference Guide*.

Automatic Activities

Automatic activities are steps in a workflow that are executed without any human intervention. When a step in a business process needs to be executed by the system, the workflow that represents that business process requires an automatic activity.

An automatic activity invokes a method of any BPO or entity method. Its definition specifies the fully qualified name of the method to call, which parameters to pass to it, and what results to retrieve from it. Input mappings specify which workflow data should be passed as parameters to the business process. Output mappings are used to map data calculated, retrieved, and/or modified by the method back into the workflow data. It can then be used anywhere in the remainder of the process, for example, the data can be used in the following activities or transitions.

For more information about the metadata associated with automatic activities, see the *Workflow Reference Guide*.

Event Wait Activities

When a business process is required to wait for any specified reason (i.e. some information is required before the next step of the process may proceed), then an event wait activity is required. An event wait activity is similar to a manual activity in that it specifies an event which the Workflow Engine waits for to signify that the activity has been completed. Event wait activities, however, do not create tasks for users. Once the specified event has been raised in the application, the Workflow Engine completes the event wait activity and proceeds to transition to the next activity in the process definition.

For more information about metadata associated with evidence wait activities, see the *Workflow Reference Guide*.

Route Activities

A route activity is an activity that performs no business functionality and its execution does not affect the application data nor the business process in any way. The primary purpose of the route activity is to assist in flow control. Route activities are often used as branch (split) and synchronization (join) points. They are also useful when the activities required by a business process do not naturally form a valid block structure that the workflow engine can execute.

Since all activity types can have notifications associated with them, route activities can be used to provide the effect of a pure notification that is not connected to any other functionality.

For more information about metadata associated with route activities, see the *Workflow Reference Guide*.

Subflow Activities

When designing a complex business process it may become too large to manage as one monolithic process definition. A sub-flow activity allows another process definition to enacted as part of another process. To enact a process as a subflow, the subflow activity must identify the process that will be enacted by name. As with the other process enactment mechanisms, the latest released version of the process is the one that will be enacted.

Subflows can be enacted synchronously. This means that the branch of the parent workflow containing the subflow activity that started the subflow process waits for that subflow process to finish before continuing. Alternatively, a subflow may be enacted asynchronously. This means that once the subflow activity starts the subflow process, the branch containing that subflow activity continues immediately with the outcome of the subflow process having no effect on the parent process.

For more information about the metadata associated with subflow activities, see the *Workflow Reference Guide*.

Loop Activities

Where a business requirement exists to carry out a piece of processing multiple times a loop activity may be used. The boundaries of a loop are defined by a Loop Begin and Loop End activity. The type of iteration and number or iterations depend on the loop type and conditions set for that loop. An example of loop types are *while* and *do/while*.

The Loop Begin activity is a control activity which specifies the loop type and the conditions under which the loop should reiterate or stop executing. The loop type indicates whether the conditions attached to the loop are evaluated up front (*while*) or after a full iteration of the loop (*do/while* or loop back). If the type is *while*, the conditions are checked before the first iteration of the loop. If they evaluate to `true`, then the loop doesn't have to go through an iteration. Otherwise, the loop will go through at least one iteration before the loop exit conditions are checked.

For more information about the metadata associated with loop activities, see the *Workflow Reference Guide*.

Decision Activities

When a business process requires a user to make a decision on what should happen next, for example, by answering a specific question, the workflow process definition contain a decision activity. A decision activity is an activity that asks a human user a question and allows the user to choose the answer from a list of presented options or alternatively enter a piece of free form text. It defines a task with one action which is a generic action that allows a user to answer a question. The format of the question with the answer options can either be multiple choice, which provides the user with options to select from, or else free form text, which provides the user with a text box to enter the answer in.

Decision activities contain an allocation strategy which defines the user or group of users assigned to the task of reaching a decision. Decision activities may also have a deadline strategy which defines what happens if an assigned user does not answer the question within a given time period.

For more information about the metadata associated with decision activities, see the *Workflow Reference Guide*.

Parallel Activities

A parallel activity acts as a wrapper around certain activities. The effect of using a parallel activity at runtime is that multiple instances of the wrapped activity are executed in parallel. To date, the only supported types of wrapped activity are *Manual* and *Decision* activities. Therefore, executing a parallel activity equates to the creation and allocation of multiple tasks in parallel.

A list workflow data object must be associated with a parallel activity. The number of items in this list workflow data object will determine the number of instances of the wrapped activity that will be created by the workflow engine.

For more information about the metadata associated with parallel activities, see the *Workflow Reference Guide*.

Activity Notifications

A notification is simply information that is sent to a users when a step in the process executes. Notifications manifest themselves as alerts in a user's *Inbox* or as emails. The users to which the notification must be sent are determined by the allocation strategy specified for the notification. The details that are displayed to the user in the alert or email are specified as part of the activity in the workflow process definition.

Notifications can be attached to any activity in a workflow process definition. The notification is created and sent using the specified delivery mechanism when the Workflow Engine executes the activity containing the notification.

The metadata associated with activity notifications is described in the *Workflow Reference Guide*.

Transitions

Transitions are used to link the various types of activity together in a workflow process. Their primary function is to dictate the order in which activities are executed. There are three types of transitions in a process definition.

The primary function of a transition is to dictate the order in which activities are executed. They determine how branch points and synchronization points relate to each other. Branch points can be of type *XOR* (Choice) or *AND* (Parallel). Corresponding branch and synchronization points must be of the same type. A branch point of type *XOR* indicates that the first transition that can be followed will be. A branch point of type *AND* indicates that all transitions that can be followed will be.

Transitions can optionally have a condition to decide whether or not a given transition will be followed. A condition is a list of expressions that perform logical operations.

For more information about the metadata associated with transitions, see the *Workflow Reference Guide*.

Workflow Data Objects

Data is maintained in the workflow engine as workflow data objects and list workflow data objects. These are logical objects defined in the process definition that have a name and a list of attributes of various types to which data can be assigned.

They are conceptually similar to objects in programming languages although their manifestation in the workflow system is of course quite different. Workflow data object values may be written at process enactment or from the output of various activity types.

Workflow data object instances and list workflow data object instances exist as soon as the process is enacted and exist until the process completes. As such they are available to be used in the activities and the transitions throughout the lifetime of that process instance. It is therefore the responsibility of the process designer to ensure that attributes of workflow data objects are populated before they are used. Attempts to use workflow data object attributes before they are populated will result in failures at runtime.

Context Workflow Data Objects

Context workflow data objects are those that are not explicitly defined in the workflow process definition metadata but are made available by the PDT and workflow engine at various places during the execution of a process. Examples include the `Context_RuntimeInformation` WDO which made available and maintained by the workflow engine, the `Context_Task` WDO which is made available for use in various mappings associated with a *Manual* activity and the `Context_Loop` WDO which is made available for use in some of the mappings associated with a *Loop* activity.

The metadata associated with workflow data objects and all of the available context workflow data objects is described in the *Workflow Data Objects* chapter of the *Cúram Workflow Reference Guide*.

Designing Workflow Process Definitions

Now that the main components of a workflow process definition have been outlined, use the information that follows to learn how to analyze a simple business process and convert it into a workflow process definition. The main considerations when designing workflow to take into account are also described.

Considerations for Business Process Analysis

It is the role of a workflow designer to analyze a business process in order to determine and specify the workflow process contained within it.

When analyzing a business process, the following considerations should be made:

- **Identify the intrinsic steps to the business process**
  These are the steps that are mandatory for the business process to succeed. If the workflow is to contain these steps, then additional considerations must be taken into account as described below.
- **Consider data integrity and traceability**
  Data passed through the workflow needs to be kept intact and there needs to be traceability for that data, i.e., the ability to determine how the data has changed as it has moved through the workflow. This may include the traceability of data through other integrated systems.
- **Determine if there are alternative ways to complete intrinsic steps that fail**
  It is very important to determine if there are alternative ways to complete any intrinsic steps that fail, either manual, or through some other automatic work around. If there is no way to complete an intrinsic step, should it fail, then it should not be included in the workflow.
- **Determine the steps in the business process likely to be altered**
  Steps likely to be altered in a business process are potential candidates for the workflow. The workflow is an easily configurable mechanism for handling activities. Whenever the steps in the business process need to be changed, the activities can be re-ordered or removed as necessary.
- **Identify non divisible tasks**
  There may be two or more steps in a business process which cannot be divided. For example, part of a business process might involve writing a person's social security number to one system and his or her salary details on different tables. These may be seen as two steps in the business process; however, they are non divisible tasks, i.e., one is invalid without the other. Non divisible tasks should not be in a workflow unless they can be combined into one activity.
- **Determine if step involves a notification**
  A notification may be added to any activity type in a workflow process definition. The notification will be delivered when the activity is executed by the workflow engine.
- **Determine if step involves a piece of work to be carried out by a user**
  A manual activity must be created for work that is carried out by a user.
- **Identify Data Required**
  This includes the type of data required and what it is used for.

Once these considerations have been taken into account, the workflow designer should be ready to design the workflow for the business process.

Selecting and Analyzing the Business Process for Workflow

The first step in workflow design is to select and analyze the business process by taking into account the considerations listed above, as well as any other additional considerations the

organization defines for its workflow designers. An example of a business process that will be used to demonstrate how this might be done is the *Close Case* business process.

At a high level, the Close Case business process closes a case and the case's associated records, including its case reviews, case referrals, and open case events. A closure communication is printed for the service supplier(s) for any case referrals. A closure communication is also printed for the primary client on the case. The case identifier (caseID) of the case being closed is required data to perform this business process.

At a lower level, this business process starts off with a series of steps which include checking case security and validations, updating the case header status to closed, setting the case status end date to the current date. There are then three new records inserted: a case status record, a case closure record, and a case event record. All of these steps can be identified as intrinsic to the Close Case business process. Note, however, without any one of these steps, either the data integrity for the case is compromised or the traceability of the attempt to close the case would be incomplete. As all of these steps are required and would be difficult to implement should the workflow fail, these steps should therefore not be moved to the workflow process.

The next step in the Close Case business process example is the reassessment of the case. This identifies any over or under payments, and as such, is intrinsic to the business process, as the case should not be closed if any over or under payments are found. Thus, the step should not be included in the workflow.

If the reassessment results in an over or under payment, the case owner is notified of the over or under payment, provided the case owner is not the user who is closing the case. As this step involves a notification based on a condition, this step may be configurable as part of the close case workflow.

If reassessment does not result in an over or under payment, the close case process continues. The system checks if there are any active case reviews and cancels them while notifying the case reviewers. The system checks if there are case reactivation events on the case and closes these events. The system checks if there are any active case referrals, it cancels them, and generates a communication to the service supplier(s) impacted by the canceled referrals. The system also prints a closure communication for the primary client of the case. All of these steps can be part of the workflow, and where relevant, notifications can be defined.

Note that the communications for the service supplier(s) and the closure communication for the primary client need to be manually put in an envelope and sent to the communication recipients. This is an additional step that might need to be added to the close case business process and would require a manual activity.

The final step in the close case business process is to determine if the case owner needs to be notified that the case is closed. This is only done if the user closing the case is not the case owner. This step can also be put into the workflow and the relevant notification defined.

In summary, there appears to be seven steps to be included in the workflow. A number of these steps will require notifications. Additionally, case and reassessment data are required.

Identifying Workflow Participants

Participants can play various roles in the completion of a workflow. For example, an activity can require that information is passed to a certain user. An important aspect of business process

automation is the capability to manage the assignment of work to the resources required to perform the work.

Manual activities can, for example, be assigned to users or groups of users that contribute to the execution of that activity. Decision activities can be assigned to the users who are responsible for answering a particular question.

Identifying workflow participants is helpful in determining the users that may be the targets of activities and activity notifications. Also, it may be necessary to communicate with participants who are not users as part of the workflow. Identifying these participants is the first step in finding ways to communicate with them as part of the workflow.

The participants in the Close Case process are identified as follows:

- User closing the case
- Case owner responsible for the case (this can be the user closing the case) - this user is notified of any over or underpayments or else notified when the case is closed
- Case reviewers - these users are notified when the scheduled reviews are canceled
- Service suppliers - communications are created for these participants when their referrals are canceled
- Primary client - a communication is created for this participant when the case is closed
- User who sends printed communications - this includes the communications for the service suppliers and the primary client

Designing Workflow

The main steps to design a workflow are to list its activities, including each activity's split and join type, to define all the transitions between the activities and their conditions, and to identify the workflow data object attributes required

All activities except for the End Process activity must have at least one outgoing transition. This is the transition from the activity to the next activity in the process. Activities with a split type of Parallel (*AND*) or Choice (*XOR*) will have multiple transitions.

Listing Activities and Split/Join Types

Each step in a business process that is implemented in the workflow must be related to an activity. Additional activities might be required to validate the workflow. For example, you can use route activities to ensure that a workflow is well formed.

All activities in the workflow must have an activity type, as well as a split and join type. Activities which include notifications and manual activities must have at least one participant that is the target of the notification or manual activity task. It may also be helpful to identify any other participants who play a role in an activity.

The following are the activities required for the Close Case workflow:

Table 1: Close Case Activities

Activity Number	Activity Description	Activity Type	Participants	Join Type	Split Type
1	Start workflow	Start	n/a	n/a	Choice XOR
2	Notify case owner case not closed	Route	Case owner	None	None

Activity Number	Activity Description	Activity Type	Participants	Join Type	Split Type
3	Check case reviews and notify reviewers	Automatic	Case reviewers	None	None
4	Check case reactivation	Automatic	n/a	None	None
5	Check case referrals and create communications for service suppliers	Automatic	Service supplierss	None	Choice XOR
6	Route if user closing case is case owner	Route	n/a	None	None
7	Notify case owner if case is closed	Route	Case owner	None	None
8	Create communication for primary client	Automatic	Primary client	Choice XOR	None
9	Mail service suppliers and primary client communications	Manual	Case owner	None	None
10	End workflow	End	Not applicable	Choice XOR	n/a

Defining Transitions and Their Conditions

All activities except for the End Process activity must have at least one outgoing transition. This is the transition from the activity to the next activity in the process. Activities with a split type of Parallel (*AND*) or Choice (*XOR*) have multiple transitions.

The workflow designer must define the transitions for all activities in a workflow. The conditions for these transitions must also be defined, if applicable. The workflow engine uses these conditions to determine the route through a workflow process.

The following are the transitions for the above Close Case activities and their conditions:

Table 2: Close Case Transitions

Transition Description	From Activity Number	To Activity Number	Transition Condition
Over or underpayment exists	1	2	If OverUnderPmtInd = true
Non-closure notification sent	2	10	
No over or underpayment	1	3	If OverUnderPmtInd = false
Case review(s) closed and reviewer(s) notified	3	4	
Case reactivation check complete	4	5	
Case referral(s) closed and case owner IS the user closing case	5	6	If caseOwner = userClosingCase
Case owner does not need to be notified	6	8	
Case referral(s) closed and case owner is NOT the user closing the case	5	7	If caseOwner not = userClosingCase

Transition Description	From Activity Number	To Activity Number	Transition Condition
Closure notification sent to case owner	7	8	
Communication created for primary client	8	9	
Service supplier and primary client communications mailed	9	10	

Designing a Graph

When you identify a list of activities and their transitions, it is good practice to create a graphical view of the workflow. This allows you to check the validity of the design before it goes to development.

To create the graph, you can choose your preferred tool, for example, Microsoft® Visio.

Identifying Workflow Data Object Attributes

For a workflow to enact and progress, the required data must successfully pass into the workflow process at enactment time and between activities during the lifetime of the process instance. The data is defined as workflow data objects for the workflow process definition and must map to the appropriate activities.

The following are the workflow data object attributes that are required during the course of the Close Case workflow process:

Table 3: Close Case Workflow Data Object Attributes

Data Item	Purpose
caseID	Required throughout the workflow to identify the case being closed
overUnderPmtInd	This is used for the transition condition to determine if the case can be closed or if the case owner must be notified that reassessment has discovered an over or under payment
caseOwner	Username of the case owner - this is used for the transition condition to determine if a case closure notification should be sent to the case owner
userClosingCase	The name of the user who initiated the Close Case business process - this is used for the transition condition to determine if a case closure notification should be sent to the case owner
listOfCaseReviewers	A list of user names of case reviewers for whom notifications must be sent regarding case reviews being canceled
listOfServiceSuppliers	A list of concern role names of service suppliers for whom communications must be printed and mailed regarding referrals being canceled

Determining the Start of the Workflow Process Instance

The workflow developer must determine what starts a workflow process instance for the new workflow. This is the stage in the business process where the workflow is enacted. When

designing a workflow, this is the step in the business process that is just before the first main step that is added to the workflow.

The first main step in the workflow is to determine whether an over or underpayment occurred as part of reassessment, and if so, notify the case owner that the case cannot be closed.

The step before this is the reassessment of the case itself. Since reassessment has not been added as an activity to the workflow, this is a likely candidate for where in the business process the workflow should be enacted from.

In the Close Case business process, an event can be added just after reassessment which enacts a process instance of the Close Case workflow. Alternatively, the source code can be updated to directly enact the Close Case workflow.

Additional Considerations

Learn about the additional considerations to take into account when analyzing a business process and designing a workflow. They are listed as follows.

- **Determine if steps can be broken down into smaller steps**
  This can be an alternative workaround for steps that fail. It is also a good way to break complex activities down into smaller, less complex activities. The less complex the activity, the easier it is to complete. If the workflow fails, it is easier to undo completed activities when these activities are smaller and more manageable.
- **Identify source code changes to support workflow**
  Existing BPO methods in the application might need to be refactored to allow them to be specified in a workflow process definition. This refactoring may include the breaking down of a specified method into smaller and more discrete steps, each with a predefined, well understood function that can easily be incorporated into a workflow.

1.3 The Workflow Engine and Enacting Workflow Processes

Use this information to learn about the workflow engine and the ways that you can enact a workflow process in the workflow management system.

The Workflow Engine

The workflow engine provides the runtime execution environment for a process instance. When a process is enacted, the workflow engine examines the specified process to enact and uses the latest released version of that process definition to create the process instance to run.

The enactment mappings in a process definition specify the data that is required to enact the specified workflow. These are mappings from struct attributes in the application to workflow data object attributes that have been marked as required for enactment. When the process is enacted, the data in the specified struct attributes is mapped to the workflow data object attributes and persisted so that it is available for use elsewhere in the workflow.

The workflow engine manages the process instance lifecycle, executing activity instances and evaluating transition rules. During the process instance lifecycle, the workflow engine continues to respond to events, such as the completion of a task, which tell it to resume the execution of a process instance. The workflow engine creates tasks instructing users on the work that needs to be completed manually, and evaluates the allocation strategies to determine which users should be

assigned these tasks. The workflow engine also creates notifications for users to inform them of the progress or status of a workflow process instance.

The workflow engine manages each process instance until the end process activity for that instance is reached. The execution of this activity indicates the completion of the process instance. If the workflow process instance fails, the workflow engine will record information about the failure. A workflow administrator can then use this information to retry the workflow process instance from the failure point.

Enacting Workflow Processes

Enacting a process definition creates an instance of that process. Four enactment mechanisms are supported by the workflow management system. Most process definitions require a minimum set of initial data. All enactment mechanisms must have a way to set the input data for a given process at enactment time.

Enacting from Code

The most direct way to enact a process is to identify a location in the application from where a process instance must be started. Code must then be inserted at that point to call the enactment service API.

The API allows the developer to specify the name of the process to start and to supply the enactment data required by the process.

While enacting a process in this way is simple and intuitive, it does have the draw back of being hard coded in the application logic. This being the case, alterations such as removing the enactment, changing the process to start or indeed even minor changes to the required enactment data will require code changes and redeployment of the application.

For more information about enacting processes from code, see the *Workflow Reference Guide*.

Enacting by Raising Events

Events provide a mechanism for loosely-coupled parts of the application to communicate information about state changes in the system. To use this functionality, events have to be defined, application code must raise these events, and event handlers must be defined and registered as listeners to such events.

When one module in the application raises an event, one or more other modules receive notification of that event having occurred provided they are registered as listeners for that event.

Developers must write and register event handlers, which are classes that perform some action when an event is raised. Optionally they can write event filters which are logic that determines whether or not to invoke the handler for a given event. The workflow management system provides a Process Enactment Event Handler that is automatically registered to listen for events associated with workflows. Where a process has been configured to be enacted from an event, the data from the event is mapped into the enactment data of the process, and the process is started.

It is possible to start a process in response to an event being raised. This requires the setup of some configuration data (either through an administration interface or as preconfigured database entries). The configuration specifies the process or processes to start in response to a specific event being raised. Mappings of event data to the enactment data required by the process can also be configured in this way.

Process enactment event configuration is stored on the database and a user interface is supplied to allow the manipulation of this data. As such process enactment created in this way can be enabled, disabled, changed and even removed at runtime. The main drawback of this approach is that since events have a finite amount of information, only process definitions that require such a small amount of enactment data can be enacted in this way.

For more information about enacting processes by raising events, see the *Workflow Reference Guide*.

Enactment as a Subflow

To enact a process as a subflow, you must create a subflow activity in the parent process that identifies the process that will be enacted by name. As with the other process enactment mechanisms, the latest released version of the process is enacted.

Subflows can be enacted *synchronously*. This means that the branch of the parent workflow containing the subflow activity that started the subflow process waits for that subflow process to finish before continuing.

Alternatively, a subflow may be enacted *asynchronously*. This means that once the subflow activity starts the subflow process, the branch containing that subflow activity continues immediately with the outcome of the subflow process having no effect on the parent process.

For more information about enacting a process as a subflow, see the *Workflow Reference Guide*.

Enacting with Web Services

You can expose the workflow processes as web services by setting metadata values. When the web services application is deployed, the Web Service Definition Language (WSDL) and the service for the process definitions are available in the normal way for the application web services.

For more information about enacting processes with web services, see the *Workflow Reference Guide*.

1.4 Managing My Inbox and Tasks

An overview of the inbox and task functionality that is available to users to manage their workload. This information describes the task lists that are available in the inbox and the functions that are available to users to manage their assigned tasks to completion.

Managing My Tasks

An overview of the **My Open Tasks** and **My Deferred Tasks** lists.

My Open Tasks

As an application user, to action a task, you must first add it to your **My Open Tasks** list. This is done by using the **Add to My Tasks** action which is available from the **Available Tasks** search page results list or the **Task Home** page.

The **My Open Tasks** list displays all of the tasks with a status of Open. Tasks that appear in this list no longer appear in your available tasks list and are also not available to any other user in the application.

Note: There is also a view of **My Open Tasks** from the application home pages.

My Deferred Tasks

When a task is listed in the your open tasks list, it can be deferred until a later date. A deferred task is moved from your open tasks list into the your deferred task list. If a restart date is specified for the task, the task may be automatically returned to your open tasks list on the specified date. This is achieved by running a batch job (`RestartTask`) delivered as part of the application. The task may also be returned to the your open tasks list by using the **Restart Task** action.

Managing My Inbox

An overview of the actions that users can perform from their Inbox. These are mainly shortcut actions for retrieving tasks to work and are accessed from the Actions menu in the title bar.

- **Get Next Task**
  This action moves the next available task from the list of tasks available to the user's **My Open Tasks** list.
- **Get Next Task from Preferred Org Unit**
  This action moves the next task from the user's preferred organization unit, to the user's **My Open Tasks** list. The user can specify their preferred organization unit in the Task Preferences section of the Inbox.
- **Get Next Task from Preferred Queue**
  This action moves the next task from the user's preferred work queue, to the user's **My Open Tasks** list. The user can specify their preferred work queue in the Task Preferences section of the Inbox.
- **New Task**
  This action allows the user to create a new manual task. A task subject must be provided and the task must be either added to the user's **My Open Tasks** list or assigned. A priority and a deadline for the task may also be specified. The newly created task may also be associated with a participant on the system and/or a specified case.

Note: How the next task is determined is configurable but typically it is the highest priority task with the earliest assignment date. For information about how to change the default Inbox customization, see the *Workflow Reference Guide*.

Searching Available Tasks

From their Inbox, a user can search for tasks assigned to them, assigned to their organizational objects (organization units, positions, or jobs), or assigned to any work queue that the user is subscribed to.

The user can select one or more of the assigned to search criteria. The granularity of the filter specified for the available task search can also be further enhanced by selecting one or more task priorities (High, Medium, and Low) and/or one or more of the provided task deadlines filters (Overdue, Due Today, Due This Week, Due This Month, Due After This Month).

Once a user has completed the Available Tasks search, their criteria is saved and will be used on subsequent visits to the page by the user. The search criteria can also be updated at any time by revisiting the page and changing the criteria.

Note: There is also a view of Available Tasks from the application home pages.

Working on Tasks

An overview of the functionality that is available to users to manage each task throughout its lifecycle

Task Home Page

The task home page displays the details of a specific task. From the task home page, users can complete the principal actions associated with the task. Tasks can have supporting information links and if available for a task, these allow the user to navigate to pages within the application containing supplemental information about the primary actions of the task. The task home page also provides access to the actions that allow the user to manage a task throughout the task lifecycle.

Task History and Comments

The task history and comments page displays all of the events that occurred during the lifecycle of the task. For each event, the user name, the date and time that the lifecycle event occurred and change information are recorded. A task history item is recorded for the following task lifecycle events:

- Task Created
- Comment Added
- Time Worked Changed
- Priority Changed
- Deadline Changed
- Added to My Tasks
- Made Available
- Forwarded
- Allocation Failed
- Allocated To Default Queue
- Reallocated
- Deferred
- Restarted
- Deadline Expired
- Closed

Task Assignments

The task assignments page displays the current assignments for the task. A task can be assigned to a user, an organizational object or a work queue and the details displayed here include the name of the assigned to object and also its type.

Task Graphical View

This page displays a graphical view of the process instance associated with the task. Each activity in the process definition is displayed as well as the transitions between them. The currently executing activity is also highlighted. Details of activities that have already been executed

are displayed in this view including the activity name, the date and time that the activity was executed and the status of any task associated with the execution of the activity.

Task Actions

To manage a task, the following actions are available to a user.

- **Add Comment**
  This function allows the user to add a comment to a task. A comment can be added for a task by the user at any time as this task action does not require the task to be part of the user's My Tasks list to be invoked. The task history is updated when a comment is added.
- **Add to My Tasks**
  On creation of a task, depending on the allocation strategy defined, a task is assigned to either a user, organization object or to a work queue. In order to action the task, a user must first add the task to their My Open Tasks list. This makes the task unavailable to other users. The task history is updated when this action is performed on a task.
- **Update Time Worked**
  This function allows a user to change the total time worked on a task. When this value is changed, a task history record is created which includes the total time before the change was made and the new total time.
- **Edit Priority**
  This function allows a user to change the priority of a task. When this value is changed, a task history record is created which includes the priority before the change was made and the new priority.
- **Edit Deadline**
  This function allows a user to edit the deadline date time of a task. When this value is changed, a task history record is created which includes the deadline before the change was made and the new deadline
- **Make Available**
  This action makes the task available to one or more previous assignees to work on. The previous assignee can be a user, organizational object or, work queue. When this action is performed, the user no longer sees the task in their My Open Tasks list. The task history is updated when this action is performed on a task.
- **Forward**
  This function allows a user to forward a task to a user, organization unit, position, job or work queue. The task history is updated when this action is performed on a task.
- **Reallocate**
  This function re-allocates a task by re-running the allocation strategy that originally allocated the task. Note since the allocation strategy is run again there is no guarantee that the task will be allocated to the same users, organizational objects or work queues as before (as the specified allocation targets may have been updated). The re-allocation of the task depends entirely on the logic of the allocation strategy. The task history is updated when this action is performed on a task.
- **Defer**
  Deferring a task effectively parks the task until a later date. When a task is deferred it moves from the user's My Open Tasks list to the My Deferred Tasks list. The task, however, is not made available to other users. The task can be scheduled for automatic restart. This means that the system will automatically revert the task status from deferred back to open on the date specified and the task will appear again in the user's My Open Tasks list. The task history is updated when this action is performed on a task.

- **Restart**
  This restarts a deferred Task and returns it to the user's My Open Tasks list. This performs the same function as the automatic restart done by the system on the restart date via a batch job, but can be done at any time by the user. The task history is updated when this action is performed on a task
- **Close**
  This closes manual tasks that were created by users in the Inbox. The action raises the specific event that such tasks are waiting on to indicate that they have been completed. Other tasks generated outside of the Inbox by the application are closed when the event those activities are waiting on are raised.

Searching Tasks

Users can search for any task in the application. The tasks are not required to be in the user's My Open Tasks list or be available to the user. Details of closed tasks can also be retrieved by using this search.

Running Task Queries

Task query functionality allows users to create, run, and save task search related queries. The same query may be executed over and over again.

The task query functionality allows the user to search for tasks based on a number of different criteria including tasks that are in the **My Open Tasks** list as well as tasks that are available to the user. The result of a task query may be further filtered by specifying the task status, task priority, task deadline date time, task creation date and task restart date.

For example, each morning the working pattern for a specified user ensures that they search for tasks available to them whose deadline expires that week. Therefore it makes sense for the user to create a query which they can run when required without having to re-enter the search criteria each day. In this case, the user creates a task query, they name the query This Week's Tasks, they select their user name in the Assigned To field and the Due this Week option in the Task Deadline field. They then click the save button and the query is saved.

Managing My Notifications

During the execution of a workflow process, the workflow engine might create notifications. Notifications can be delivered to the user by email or through alerts. The user can delete the notifications as needed.

Notifications are used to inform a user that an event has occurred. Unlike tasks, there is no expectation that the user needs to do any work when they receive a notification.

Notifications that are delivered using the alert mechanism are displayed in the **My Notifications** page. A notification will remain in this list until acknowledged by the user. The user can delete these notifications one by one or many at a time. It should be noted that alerts are physically deleted from the `Alert` database table when a user performs this action.

My Work Queues

The My Work Queues section is divided into two lists: the work queues that the user is subscribed to and the work queues that the user's organizational objects are subscribed to.

User Subscribed Work Queues

This list displays the work queues that the user is currently subscribed to. From here the user can view the tasks assigned to a work queue. They can also add the next task assigned to a work queue to their **My Open Tasks** list. Users can also subscribe to another work queue or unsubscribe from a work queue from this page.

Other Subscribed Work Queues

The **Other Subscribed Work Queues** list displays a list of work queues to which the user's organization objects (organization units, positions or jobs) are currently subscribed. From here, the user can view the tasks assigned to a work queue. They may also add the next task assigned to a work queue to their **My Open Tasks** list. A user cannot subscribe or unsubscribe to or from a work queue as that is the function of an administrator or a supervisor user only when the type of work queue subscription is an organizational object.

Setting Task Preferences

Users can set their task preferences in the general settings area of their inbox.

Setting the Preferred Queue and Preferred Organization Unit

A a user can set their preferred queue to indicate the work queue whose tasks the user works on most often. Similarly, when a user sets a preferred organization unit, it indicates the team whose tasks the user works on most often.

Once the user has specified a preferred queue or organization unit, they can use the **Get Next Task from Preferred Queue** and **Get Next Task from Preferred Organization Unit** shortcut Inbox actions.

Redirecting Tasks

A user can redirect tasks to another user or organizational object. This function is useful for situations such as annual leave. The user can specify the dates where the redirection start and ends. Task redirection must be set up to start at a future date, running either indefinitely or for a limited period. While the redirection period is active, no new tasks are assigned to the user. Instead, the tasks are assigned to the selected user or organization object. Any tasks in the user's **My Open Tasks** list remain on the list.

Once a redirection period is active, all tasks currently assigned to the user may be removed from the user's inbox, and added to the assigned tasks list of the user or organizational object specified in the task redirection. This is done in a batch job named `ScanActiveTaskRedirections`.

The user may also view a list of active, pending and expired redirections on the task redirection page. Active or pending task redirections may be removed by the user at any time.

Blocking Task Allocation

A user can stop tasks from being assigned to them without specifying another user to assign the tasks to. Task allocation blocking enables the user to ensure that no tasks are assigned to them from a specified time and date. Task allocation blocking, like task redirection, must be set up to start at a future date, and can run until a specified end time, or run indefinitely. No new task assignment records are created for the blocked user until the task allocation blocking period is cleared or expires.

Any task assignments that exist for the blocked user may be reallocated. This is done in a batch job named `ScanActiveTaskAllocationBlockingPeriods`.

A list of current, pending and expired allocation blocks may be viewed on the Task Allocation Blocking page. Active or pending allocation blocks may be removed by the user at any time.

1.5 Administering Workflow

A overview of Workflow Administration which includes the maintenance of workflow process instances, and the definition and maintenance of events, work queues, and allocation targets.

Monitoring Workflow Processes

Administrators can monitor workflow processes, view process instance information, and monitor process instance errors.

You can use the Process Instances and Process Instance Errors views to monitor workflow and deferred process instances.

- Use the Process Instances view to see the status of each workflow status instance. By searching and filtering, you can see the current workflow processes and their status. You can search on process, task, or event details. From these search results, you can suspend, resume, or abort a process instance.
- Use the Process Instance Errors view to see workflow and deferred process instance errors. By searching and filtering, you can easily identify which processes have failed. You can view the full stack trace for each failed process. From this view, you can retry, or abort failed workflow process instances. Deferred processes that have failed can be marked as resolved.

Maintaining Workflow Events

Workflow events are sent by application functions and are used to transition activities in workflows that are waiting on those events or to enact workflow processes.

Manual, event wait, and decision activities use events to progress a workflow process instance. The application raises an event when the action associated with the specified activity (task) is complete or when a particular event defined for an event wait activity has been completed. These events signal the workflow engine to complete the associated activity instance that is waiting on that event and to progress the workflow process instance by evaluating the next set of transitions and executing the next activity in the workflow.

The application can also use events to enact a workflow. When an action associated with an enactment event occurs in the application, the workflow engine processes this event and enacts a new instance of the specified workflow process that is defined in the process enactment event configuration data.

Administering Work Allocation

When a task is created, or a notification being delivered, as a result of the execution of an activity, that task or notification must be routed to a specific user or group of users for it to be actioned. This process is called work allocation and a set of allocation rules will be associated with the activity for this purpose.

There are four types of allocation strategy: functions, classic rules, CER rules, or allocation targets.

Maintaining Allocation Targets

Administrators can maintain allocation targets. Allocation targets are logical grouping of users or work queues to which tasks and notifications can be assigned. This allows tasks and notifications to be assigned to a cross section of users in the organization in a manner that can be customized by the workflow administrator.

Each allocation target acts as a container for one or more target items. These target items determine the users or work queues to whom the task or notification should be assigned. A target item can be any one of the following:

- Job - the task is assigned directly to the job. The notification is sent to any users assigned to the job via their position assignment.
- Organization unit - the task is assigned directly to the organization unit. The notification is sent to any users assigned to the organization unit also via their position assignment.
- Position - the task is assigned directly to the position. The notification is sent to any users assigned to the position.
- Work Queue - tasks are assigned directly to the work queue. This enables users to subscribe to the work queue, view and work on any tasks that have been assigned to that work queue. Notifications are sent to any users who are subscribed to the work queue.
- User - the task is assigned or the notification is sent to the specified individual user.

Maintaining Work Queues

Administrators can create, edit or remove work queues from the system. A work queue is a subscription-based list that can be assigned as a target item for an allocation target.

Work Queues must have an assigned administrator. The administrator may subscribe users to the work queue if required. Otherwise, if the work queue definition specifies that users are allowed subscribe to the work queue, the users themselves can subscribe to the work queue. Work queues can also have a sensitivity level which restricts user access to the work queue by comparing the sensitivity specified for the work queue against the user's sensitivity.

Work queues can be defined for particular jobs or roles, for particular departments, or any other arrangement, for example, the “Claim Approval Work Queue” and the “Claims Department Work Queue”. Tasks that are assigned to a work queue are visible to any user who subscribes to that work queue.

Translating Work Queue Names

If your Cúram application supports more than one language, you can add translation text for work queue names to support users who use that language. You can also modify existing translation text for a work queue name.

About this task

The application uses the correct language at runtime based on the user's locale. If Cúram supports a single language, but that language is not English, you can edit the work queue names in that language by selecting the **Edit** action for the work queue name you want to update.

Procedure

1. Log in to the Cúram application as an administrator.
2. Click **Administration Workspace** > **Shortcuts** > **Workflow** > **Work Queues**.
3. Locate the work queue whose name you want to translate.
4. Click the Translate Text icon. A list of existing translations for the work queue name is displayed.
5. Do one of the following:
   - Add a new translation.
     1. Click **Add Translation**.
     2. Select the language for the work queue name translation.
     3. Enter the translation for the work queue name text.
     4. Click **Save**.
   - Modify an existing translation.
     1. Click the **Edit** action for the translation you want to modify.
     2. Edit the translation for the work queue name text.
     3. Click **Save**.
6. Click **Close**.

Results

The work queue name is translated with the changes you made everywhere it appears in the application. For example, log in to Cúram as a caseworker and click **Inbox** > **Work Queues** > **My Work Queues**. The work queue names are displayed in the language based on the user's default locale.

For more information about configuring text translations, see the *System Administration Guide*. For information about developing localizable text translations, see the *Web Client Reference Manual*.

Notices

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the Merative website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of Merative

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of Merative.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

Merative reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by Merative, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

MERATIVE MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Merative or its licensors may have patents or pending patent applications covering subject matter described in this document. The furnishing of this documentation does not grant you any license to these patents.

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

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

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

The licensed program described in this document and all licensed material available for it are provided by Merative under terms of the Merative Client Agreement.

COPYRIGHT LICENSE:

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

Privacy policy

The Merative privacy policy is available at https://www.merative.com/privacy.

Trademarks

Merative ™ and the Merative ™ logo are trademarks of Merative US L.P. in the United States and other countries.

IBM®, the IBM® logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

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

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

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

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

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