Merative Social Program Management 8.1

Cúram Provider Management Developers Guide

Note

Before using this information and the product it supports, read the information in Notices on page 65

Edition

This edition applies to Merative™ Social Program Management 8.0.0, 8.0.1, 8.0.2, 8.0.3, and 8.1.

Contents

Note..... iii

Edition..... v

1 Developing with Provider Management..... 9

1.1 Using Strategy Patterns to Customize CPM..... 9
Provider Implementations..... 9
Placement and Contract Implementations..... 11
Training Implementations..... 12
Service Invoice Implementations..... 13
Custom Rates and Reassessment..... 15
Roster Implementations..... 16
1.2 Using Events to Add Custom Processing to CPM..... 18
Provider Customization Points..... 18
Provider Group Customization Points..... 23
Service Offering Customization Points..... 23
Service Authorization Customization Points..... 24
Placement Customization Points..... 26
Contract Customization Points..... 27
Service Invoice Customization Points..... 29
Attendance Customization Points..... 33
Financial Customization Points..... 37
Referral Customization Points..... 39
Service Delivery Customization Points..... 40
1.3 CPM Workflow Process Definitions..... 41
Introduction..... 41
External Enquiry Workflow..... 41
Home Study Approval Workflow..... 42
New Invoice Created Workflow..... 43
Service Invoice Exception Processing Workflow..... 43
Service Invoice Line Item Approval Workflow..... 44
Service Invoice Line Item Correction Approval Workflow..... 45
Supervisor Request Decision Workflow..... 46
Supervisor View New External User Task Notification Workflow..... 47
Roster Exception Processing Workflow..... 48
New Client Added to Roster Workflow..... 48
Roster Line Item Approval Workflow..... 49
Roster Line Item Correction Approval Workflow..... 50
1.4 CPM Products and Rule Sets..... 51
Products..... 51

Rule Sets..... 56
1.5 CPM Financials..... 56
Payment Types..... 56
1.6 Service Deliveries..... 58
Product Design and Configuration..... 58
Rule Set Creation..... 59
Evidence and Evidence Maintenance..... 59
Custom Rates..... 59
Service Delivery Creation..... 60
Display of Product Delivery Information..... 60
1.7 Compliancy for Provider Manager..... 60
Miscellaneous Entities..... 61
1.8 Appendix A..... 61
1.9 Appendix B: Schema definitions of the XML fragments created by Import process..... 63
The *UseReference.xsd* file..... 63
The *RelatedConcept.xsd* file..... 63
The *ExternalTerm.XSD* file..... 64

Notices..... 65
Privacy policy..... 66
Trademarks..... 66

1 Developing with Provider Management

Provider Management can be customized and configured to meet the requirements of the organization. Customization options include strategy patterns, events, workflows, and rule sets. CPM provides a number of service layer interfaces that are designed for customization. A number of application properties are available that allow administrators to configure Provider Management.

Purpose

The following pages describe the options for customizing the Cúram Provider Management (CPM) component. Its scope includes the customization of Strategy Patterns, Events, Workflows, Products, and Rule Sets. Customization can be distinguished from configuration. Customization means that developers can modify, extend, or replace source code to suit the agency's requirements. Configuration means administrators can manage the information that is displayed on application pages or to alter the behavior of the application in certain predefined ways.

The customization or extension points provided with the CPM component are outlined. Generic extension points, such as persistence events, are not outlined.

Audience

The target audience is developers who are responsible for customizing CPM.

Related information

1.1 Using Strategy Patterns to Customize CPM

CPM provides a number of service layer interfaces that are specifically designed for customization.

A new custom implementation can be provided for any of the interfaces listed below. It is worth noting that default implementations are provided for these interfaces. For information about how to provide an implementation for a service layer interface, see the *Developing with the Persistence Infrastructure* related link. The default implementations of these interfaces can be replaced with a new custom implementation by creating a new Guice module class and adding a corresponding entry in the MODULE table. For more information, see the *Creating a Guice module* related link.

Related information

Provider Implementations

Table 1: Provider Implementations

This table describes customizable implementations.

Functionality	Interface	Description
Provider Enrollment Date	curam.provider.impl.Provider	This interface allows agencies to enroll a Provider or Provider Group in CPM with an enrollment date which is in the past. This allows the agency to use the original date of enrollment while enrolling providers. The default implementation is to use the current date.
Provider Reference Number	curam.provider.impl.ProviderReferenceNumberStrategy	This interface allows agencies to generate Provider Reference Numbers according to their preferred format.
Provider Group Reference Number	curam.provider.impl.ProviderGroupReferenceNumberGenerator	This interface allows agencies to generate Provider Group Reference Numbers according to their preferred format.
Provider Enquiry Reference Number	curam.provider.impl.ProviderEnquiryReferenceNumberGenerator	This interface allows agencies to generate Provider Enquiry Reference Numbers according to their preferred format.
License Reference Number Generation	curam.provider.impl.LicenseNumberGenerator	This interface allows agencies to generate License Reference Numbers according to their preferred format.
Home Study Recommendation Approval	curam.provider.impl.ProviderSecurity	This interface allows agencies to designate a specific user or a group of users (an organization unit, users in a particular position or with a particular job, etc.) who can approve or reject a home study recommendation.
Provider Offering Approval Criteria	curam.providerservice.impl.ProviderOfferingApprovalCriteria	This interface allows agencies to specify criteria which need to be met in order to approve a service offered by a provider.

Functionality	Interface	Description
Service Offering Validation	curam.serviceoffering.impl.ServiceOfferingValidation	A ServiceOfferingValidation class is used for managing the validations for a service. The default implementation of this interface is provided by ServiceOfferingValidationImpl. A new implementation of this interface is required to change the mechanism used to manage the validations for a service. This interface allows agencies to backdate the start date of a service offering. This may be useful when an agency is unable to add all provider services at the time of enrollment. This interface allows the agency to add a service at a later stage, and indicate that it has always been offered by the provider. The default date can be overridden on case-by-case basis.
External User Password	curam.externaluseraccess.impl.ExternalUserPasswordStrategy	This interface allows agencies to implement a particular strategy for allocating passwords, at the point at which they generate the initial password for a new external user account, or generate a replacement password for a user who has forgotten password and needs to re-establish credentials with the agency.
Provider Member Offering Training Criteria	curam.provider.impl.ProviderMemberOfferingTrainingCriteria	This interface allows agencies to change the default functionality when a provider offering with training requirements is approved. For example, the agency may wish to prevent the approval of a provider offering, if the training requirements for the service are neither 'Complete' nor 'Waived', rather than sending a notification.

Placement and Contract Implementations

Table 2: Placement and Contract Implementations

This table describes customizable Placement and Contract implementations.

Functionality	Interface	Description
Placement Payment	curam.place.impl.PlacementPaymentStrategy	A PlacementPaymentStrategy class is used for determining if a placement is paid on the basis of an invoice or placement. The default implementation of this interface is provided by PlacementPaymentStrategyImpl. A new implementation of this interface is required to change the mechanism used to determine if a placement is paid on the basis of an invoice or placement. For example, an agency may indicate that all placement services should be paid through the receipt of invoices, or it may indicate that only services in a specific service group or services provided by specific providers can be paid through the receipt of invoices.
Flat Rate Contract Cover Pattern	curam.contracts.impl.FlatRateContractCoverPatternStrategy	A FlatRateContractCoverPatternStrategy class is used for determining the cover period pattern for a provider flat rate contract payment. The default implementation of this interface is provided by FlatRateContractCoverPatternStrategyImpl. A new implementation of this interface is required to change the mechanism used to determine the cover period pattern for a provider flat rate contract payment. A cover period pattern specifies how payments or bills are issued, e.g., in advance, in arrears, once-off, etc.
Contract Reference Number Generation	curam.contracts.impl.ReferenceNumberGenerator	A ReferenceNumberGenerator class is used for generating a reference number for a contract. The default implementation of this interface is provided by UniqueNumberGeneratorImpl. A new implementation of this interface is required to change the strategy to generate a reference number.

Training Implementations

Table 3: Training Implementations

This table describes customizable Training implement ions

Functionality	Interface	Description
Approve License Based on Training	curam.provider.impl.LicenseApprovalCriteria	This interface allows agencies to change the default functionality for when a license with training requirements is approved. CPM supports notification to a user where training requirements for one or more services are neither 'Complete' nor 'Waived' for provider members. However, some Agencies may wish to prevent license approval if this validation is not satisfied. This interface is useful in such a scenario.
Approve Person Training	curam.training.impl.ApprovePersonTrainingStrategy	This interface allows agencies to define their approval strategy for person training. The purpose of this interface is the same as that for the approval of provider member training, as described above.
Approve Provider Group Member Training	curam.training.impl.ApproveProviderGroupMemberTrainingStrategy	This interface allows agencies to define their approval strategy for provider group member training. The purpose of this interface is the same as that for the approval of provider member training, as described above.
Approve Provider Member Training	curam.training.impl.ApproveProviderMemberTrainingStrategy	This interface allows agencies to define their approval strategy for provider member training. CPM by default allows the resource manager or the resource manager supervisor to approve training.

The curam.training.impl.ApproveProviderMemberTrainingProgramStrategy interface can be used to facilitate functional scenarios such as the following:

- Agencies may choose to have another user or a group of users (an organization unit, users in a particular position or with a particular job) who can approve the training request.
- Agencies may wish to inhibit authorization of training based on some other additional or alternative approval criterion.
- CPM does not send any notification on approval of a training program. However, an agency may want to send a notification to the provider of the training, the provider the provider member works for, or the provider member themselves.

Service Invoice Implementations

Table 4: Service Invoice Implementations

This table describes customizable Service Invoice implementations

Functionality	Interface	Description
Service Invoice Line Item	curam.financial.impl.ServiceInvoiceLineItemValidationStrategy	ServiceInvoiceLineItemValidationStrategy class is used for validating the number of units of a service invoice line item. The default implementation of this interface is provided by ServiceInvoiceLineItemValidationStrategyImpl. A new implementation of this interface is required to change the mechanism used to validate the number of units of a service invoice line item.
Service Invoice Payment	curam.financial.impl.ServiceInvoicePaymentStrategy	ServiceInvoicePaymentStrategy class is used for managing service invoice payment strategy. The default implementation of this interface is provided by ServiceInvoicePaymentStrategyImpl. A new implementation of this interface is required to change the mechanism used to manage service invoice payment strategy. This may be useful where an agency wishes to re-direct these payments to an individual or a group other than the provider. For example, if the provider is specified as the payee on a service invoice line item but is an active member of a provider group, the agency may re-direct payments to the provider group instead.
Service Invoice Line Item Validation	curam.financial.impl.ServiceInvoiceLineItemValidator	ServiceInvoiceLineItemValidator class is used for validating the service invoice line item. The default implementation of this interface is provided by ServiceInvoiceLineItemValidatorImpl. A new implementation of this interface is required to change the mechanism used to validate the service invoice line item. For example, some agencies may not want to allow a service invoice line item to be added to a service invoice if the status of the service invoice is 'In Progress'. This interface will allow them to implement this validation.

Functionality	Interface	Description
Determine Service Invoice Line Item Payment Amount	curam.financial.impl.DeterminePaymentAmount	This business interface provides a mechanism for determining the amount to be paid for Service Invoice Line Item. If an Agency has a specific way in which they will want to calculate the payment amount, the customized implementation can be provided for this interface.

Custom Rates and Reassessment

Table 5: Custom Rates and Reassessment Implementations

This table describes custom rates and reassessment implementations

Functionality	Interface	Description
Applicable Rate Listener	curam.financial.impl.ApplicableRateListener	This business interface is used for re-assessment of payments for a given period for a service authorization line item/ placement/service invoice line item/provider roster line item. There are two APIs present in ApplicableRateListener having same name as "reAssess" but with different input types. Reassess API having inputs as Service Authorization Line Item and date range is used when no detailed product information is available and only Service Authorization Line Item is known. It searches to retrieve matching Service Invoice Line items, Placements and Provider Roster Line Items for the given Service Authorization Line Item and then it calls the suitable API present in ApplicableRateProcessor API to process the change of rate for any given input(placement /SILI/PRLI) and reassess the payment. Reassess API having inputs as Delivery Evidence Information of the product and date range is used when more product level information is available and the type of service is known. Depending on the product type it calls the suitable API present in ApplicableRateProcessor API to process the change of rate for any given input(placement /SILI/PRLI) and reassess the payment.

Functionality	Interface	Description
Applicable Rate Processor	curam.financial.impl.ApplicableRateProcessor	An ApplicableRateProcessor class is used for reassessment of payments triggered by the change in rates. The default implementation of this interface is provided by ApplicableRateProcessorImpl. A new implementation of this interface is required to change the mechanism used to calculate the reassessed payment amount, due to the change in rates for the reassessment period. This interface allows agencies to process the change of rate for any given input (placement/SILI/ PRLI) and reassess the payment. There are three APIs present in ApplicableRateProcessor named processRateChangeForPlacement, processRateChangeForPRLI and processRateChangeForSILI respectively. All these APIs are having inputs as type of service (Placement, SILI, PRLI) and the reassessment period. It processes the change of rate for any given input(placement /SILI/PRLI) and reassess the payment.
Service Delivery Rate Determination	curam.financial.impl.RateDetermination	A RateDetermination class is used for retrieving the rates for the given period and product delivery. The default implementation of this interface is provided by RateDeterminationImpl. A new implementation of this interface is required to change the strategy to determine the rates for a given delivery type (placement, invoice, or attendance) for a given period of time. For example, the applicable rates for a service can be determined using a custom rate calculation logic which may reference variables that do not reside within CPM, such as the number of children in a family.

Roster Implementations

Table 6: Roster Implementations

This table describes customizable Roster implementations

Functionality	Interface	Description
Generate Rosters	curam.attendance.impl.DetermineRosterSubmissionDueDate	A DetermineRosterSubmissionDueDate class is used for determination of submission due date for a roster. The default implementation of this interface is provided by DetermineRosterSubmissionDueDateImpl. A new implementation of this interface is required to change the way the grace period is used to determine the submission due date. For instance, an agency may wish to consider only the business days to calculate a submission due date.
Match Provider Roster Line Item	curam.attendance.impl.MatchProviderRosterLineItem	A MatchProviderRosterLineItem class is used for performing validations during matching a provider roster line item details with the existing details. The default implementation of this interface is provided by MatchProviderRosterLineItemImpl. A new implementation of this interface is required to change the mechanism used to match the details of a provider roster line item with the existing details. It is used for performing an agency's own program-specific validations during matching a provider roster line item.
Match Provider Roster Line Item	curam.attendance.impl.VoucherValidator	A VoucherValidator class is used for matching and validating the voucher details. The default implementation of this interface is provided by VoucherValidatorImpl. A new implementation of this interface is required to change the mechanism used to match and validate the voucher details of the provider roster line item. For example, the agency might have its own program-specific validations to match and validate the voucher details.

Functionality	Interface	Description
Determine Attendance Based Payment Amount	curam.attendance.impl.AttendancePay	mentDeterminationProcessing AttendancePaymentDeterminationProcessing class is used for the determination of an attendance-based payment amount. The default implementation of this interface is provided by AttendancePaymentDeterminationProcessingImpl A new implementation of this interface is required to change the mechanism used to calculate the attendance-based payment rate. For example, the provider service rate valid either on the end date of the roster line item or the end date of the matching service authorization line item could be used to determine the attendance-based payment amount.
Allocate Units	curam.attendance.impl.PRLIUnitsAllocationProcessing	PRLIUnitsAllocationProcessing class is used for managing the allocation of units from roster line items to matching service authorization line items. The default implementation of this interface is provided by PRLIUnitsAllocationProcessingImpl. A new implementation of this interface is required to change the mechanism used to allocate units to the matching service authorization line items. For example, units could be allocated evenly to all service authorization line items rather than starting with the earliest one.

1.2 Using Events to Add Custom Processing to CPM

Developers can add custom functionality to the events that are raised by CPM. Business events are raised at all extension points. These events can be used by agencies to add functionality before the action is executed, after the action is executed, or both.

Provider Customization Points

The following sections list the available customization points for Providers.

(deprecated) Provider Events

The following events are located in the curam.provider.impl.Provider interface.

Table 7: Provider Event Details

This table describes Provider Events

Event Class	Description	Event is raised before and after
ProviderSuspendEvents	Raised when a Provider is suspended.	curam.provider.impl.Provider.suspend()
ProviderCloseEvents	Raised when a Provider is closed.	curam.provider.impl.Provider.close()
ProviderRejectEvents	Raised when a Provider seeking approval is rejected.	curam.provider.impl.Provider.reject()
ProviderApproveEvents	Raised when a Provider is approved.	curam.provider.impl.Provider.approve()
ProviderReopenEvents	Raised when a closed Provider is reopened.	curam.provider.impl.Provider.activate()
ProviderEnrollEvents	Raised when a Provider is enrolled.	curam.provider.impl.Provider.enroll()
ProviderGetAvailablePlacesInDateRangeEvents	Raised when available Places in the given date range are retrieved.	curam.provider.impl.Provider.getAvailablePlacesIn
ProviderGetServiceOfferingsEvents	Raised when Service Offerings for a Provider are retrieved.	curam.provider.impl.Provider.getServiceOfferings(
ProviderGetCommonApprovedProviderServiceOfferingsEvents	Raised when approved Service Offerings for a Provider are retrieved.	curam.provider.impl.Provider.getCommonApprove

The following events are located in the curam.provider.impl.ProviderApprovalCheck interface.

Table 8: Provider Event Details

This table describes Provider Events

Event Class	Description	Event is raised before and after
ProviderApprovalCheckCreateProviderApprovalCheckEvents	Raised when an approval check for the Provider is created.	curam.provider.impl.ProviderApprovalCheck.creat
ProviderApprovalCheckModifyProviderApprovalCheckEvents	Raised when an approval check for the Provider is modified.	curam.provider.impl.ProviderApprovalCheck.modi
ProviderApprovalCheckCancelProviderApprovalCheckEvents	Raised when an approval check for the Provider is canceled.	curam.provider.impl.ProviderApprovalCheck.canc

Provider Enquiry Events

The following events are located in the curam.provider.impl.ProviderEnquiry interface.

Table 9: Provider Enquiry Event Details

This table describes Provider Enquiry Events

Event Class	Description	Event is raised before and after
ProviderEnquiryCloseEvents	Raised when a Provider Enquiry is closed.	curam.provider.impl.ProviderEnquiry.close()
ProviderEnquiryTransferEnquiryToProviderEvents	Raised when a Provider is enrolled from an enquiry.	curam.provider.impl.ProviderEnquiry.transferEnqu
ProviderEnquirySetProviderEnquiryDetailsEvents	Raised when an enquiry is created.	curam.provider.impl.ProviderEnquiry.setProviderE

Event Class	Description	Event is raised before and after
ProviderEnquirySetProviderEnquiryUpdatedDetailsEvents	Raised when a Provider Enquiry is updated.	curam.provider.impl.ProviderEnquiry.setProviderE

License Events

The following Events are located in the curam.provider.impl.License interface.

Table 10: License Event Details

This table describes License Events

Event Class	Description	Event is raised before and after
LicenseSuspendEvents	Raised when a License is suspended.	curam.provider.impl.License.suspend()
LicenseRejectEvents	Raised when a License is rejected.	curam.provider.impl.License.reject()
LicenseApproveEvents	Raised when a License approved.	curam.provider.impl.License.approve()

Home Study Events

The following Events are located in the curam.homestudy.impl.HomeStudy interface.

Table 11: Home Study Event Details

This table describes Home Study Events

Event Class	Description	Event is raised before and after
HomeStudyApproveEvents	Raised when a Home Study recommendation for a provider is approved.	curam.homestudy.impl.HomeStudy.approve()
HomeStudySubmitEvents	Raised when a Home Study is submitted.	curam.homestudy.impl.HomeStudy.submit()
HomeStudyRejectEvents	Raised when a Home Study recommendation is rejected.	curam.homestudy.impl.HomeStudy.reject()

Compartment Events

The following Events are located in the curam.place.impl.Compartment interface.

Table 12: Compartment Event Details

This table describes Compartment Events

Event Class	Description	Event is raised before and after
CompartmentCloseEvents	Raised when a Compartment is closed.	curam.place.impl.Compartment.close()

Place Events

The following events are located in the curam.place.impl.Place interface.

Table 13: Place Event Details

This table describes Place Events

Event Class	Description	Event is raised before and after
PlaceCloseEvents	Raised when a Place is closed.	curam.place.impl.Place.activate()
PlaceMarkOutOfUseEvents	Raised when a Place is marked out of use.	curam.place.impl.Place.markOutOfUse()
PlaceOccupiedEvents	Raised when a Place is occupied.	curam.place.impl.Place.occupied()
PlaceMarkInUseEvents	Raised when a Place is marked in use.	curam.place.impl.Place.markInUse()
PlaceGetLocationForPlaceEvents	Raised when the location of a Place is retrieved.	uram.place.impl.Place.getLocationForPlace()

Request Events

The following events are located in the curam.externaluseraccess.impl.Request interface.

Table 14: Request Event Details

This table describes Request Events

Event Class	Description	Event is raised before and after
RequestAcceptEvents	Raised when a Request created by an external provider is accepted.	curam.externaluseraccess.impl.Request.accept()
RequestSubmitEvents	Raised when Request created by an external provider is submitted.	curam.externaluseraccess.impl.Request.submit()
RequestRejectEvents	Raised when Request created by an external provider is rejected.	curam.externaluseraccess.impl.Request.reject()

Member Certification Events

The following events are in the `curam.provider.impl.MemberCertification` interface.

Table 15: Member Certification Event Details

This table describes Member Certification Events

Event Class	Description	Event is raised before and after
MemberCertificationModifyCertificationEvents	Raised when a provider member Certification is updated.	curam.provider.impl.MemberCertification.modifyC
MemberCertificationGetDerivedStsEvents	Raised when the status of a provider member Certification is retrieved.	curam.provider.impl.MemberCertification.getDeriv

Provider Deduction Events

The following events are located in the curam.provider.impl.ProviderDeduction interface.

Table 16: Provider Deduction Event Details

This table describes Provider Deduction Events

Event Class	Description	Event is raised before and after
ProviderDeductionActivateProviderDeductionEvents	Raised when Deductions associated to a Provider are activated.	curam.provider.impl.ProviderDeduction.activatePr
ProviderDeductionDeactivateProviderDeductionEvents	Raised when Deductions associated to a Provider are deactivated.	curam.provider.impl.ProviderDeduction.deactivate
ProviderDeductionCreateDeductionForExistingCasesEvents	Raised when a Deduction is created for existing cases.	curam.provider.impl.ProviderDeduction.createDed
ProviderDeductionCreateVariableDeductionForModifiedPaymentTypeEvents	Raised when Variable Deductions are created based on Payment Type.	curam.provider.impl.ProviderDeduction.createVar
ProviderDeductionCancelVariableDeductionForModifiedPaymentTypeEvents	Raised when Variable Deductions are Cancelled.	curam.provider.impl.ProviderDeduction.cancelVar

Provider Offering Events

The following events are located in the curam.providerservice.impl.ProviderOffering interface.

Table 17: Provider Offering Event Details

This table describes Provider Offering Events

Event Class	Description	Event is raised before and after
ProviderOfferingApproveEvents	Raised when a Provider Offering is approved.	curam.providerservice.impl.ProviderOffering.appro
ProviderOfferingDenyEvents	Raised when a Provider Offering is denied.	curam.providerservice.impl.ProviderOffering.deny
ProviderOfferingCheckApprovalCriteriaEvents	Raised when approval criteria are checked for a Provider Offering.	curam.providerservice.impl.ProviderOffering.chec
ProviderOfferingGetContractsEvents	Raised when Contracts are retrieved for a Provider Offering.	curam.providerservice.impl.ProviderOffering.getC

The following events are located in the curam.citizenactivity.impl.ProviderOfferingUtil interface.

Table 18: Provider Offering Event Details

This table describes Provider Offering Events

Event Class	Description	Event is raised before and after
ProviderOfferingUtilGetByServiceOfferingAndProviderEvents	Raised when a Provider Offering is retrieved based on the Service Offering and Provider.	curam.citizenactivity.impl.ProviderOfferingUtil.getB

Provider Offering Rate Events

The following events are located in the curam.providerservice.impl.ProviderOfferingRate interface.

Table 19: Provider Offering Rate Event Details

This table describes Provider Offering Rate Events

Event Class	Description	Event is raised before and after
ProviderOfferingRateModifyForContractEvents	Raised when a Provider Offering Rate is modified.	curam.providerservice.impl.ProviderOfferingRate.

Provider Group Customization Points

The following sections list the available customization points for Providers Groups.

Provider Group Events

The following events are located in the curam.provider.impl.ProviderGroup interface.

Table 20: Provider Group Event Details

This table describes Provider Group Events

Event Class	Description	Event is raised before and after
ProviderGroupCloseEvents	Raised when a Provider Group is closed.	curam.provider.impl.ProviderGroup.close()
ProviderGroupEnrollEvents	Raised when a Provider Group is enrolled.	curam.provider.impl.ProviderGroup.enroll()
ProviderGroupReopenEvents	Raised when a closed Provider Group is reopened.	curam.provider.impl.ProviderGroup.reopen()
ProviderGroupGetCommonApprovedProviderServiceOfferingsEvents	Raised when Service Offerings of a Provider Group are retrieved.	curam.provider.impl.ProviderGroup.getCommonA

Provider Group Associate Events

The following events are located in the curam.provider.impl.ProviderGroupAssociate interface.

Table 21: Provider Group Associate Event Details

This table describes Provider Group Associate Events

Event Class	Description	Event is raised before and after
ProviderGroupAssociateRemoveProviderEvents	Raised when a Provider is removed from a Provider Group.	curam.provider.impl.ProviderGroupAssociate.rem

Service Offering Customization Points

The following sections list the available customization points for Service Offerings.

Service Offering Events

The following events are located in the curam.serviceoffering.impl.ServiceOffering interface.

Table 22: Service Offering Event Details

This table describes Service Offering Events

Event Class	Description	Event is raised before and after
ServiceOfferingGetServiceRatesForPeriodEvents	Raised when Service rates are retrieved for a particular period.	curam.serviceoffering.impl.ServiceOffering.getSer()
ServiceOfferingModifyDescriptionTextTranslationEvents	Raised when the text translation details for the Service Offering description attribute is modified.	curam.serviceoffering.impl.ServiceOffering.modify
ServiceOfferingModifyNameTextTranslationEvents	Raised when the text translation details for the Service Offering name attribute is modified.	curam.serviceoffering.impl.ServiceOffering.modify
ServiceOfferingAddNameTextTranslationEvents	Raised when the text translation is created for the Service Offering name attribute.	curam.serviceoffering.impl.ServiceOffering.addNa
ServiceOfferingAddDescriptionTextTranslationEvents	Raised when the text translation is created for the Service Offering description attribute.	curam.serviceoffering.impl.ServiceOffering.addDe

Service Group Events

The following events are located in the curam.serviceoffering.impl.ServiceGroup interface.

Table 23: Service Group Event Details

This table describes Service Group Events

Event Class	Description	Event is raised before and after
ServiceGroupAddServiceOfferingEvents	Raised when a Service Offering is added to a Service Group.	curam.serviceoffering.impl.ServiceGroup.addServ()
ServiceGroupRemoveServiceOfferingEvents	Raised when a Service Offering is removed from a Service Group.	curam.serviceoffering.impl.ServiceGroup.removeS()
ServiceGroupGetServiceOfferingsEvents	Raised when Service Offerings from a Service Group are retrieved.	curam.serviceoffering.impl.ServiceGroup.getServi()
ServiceGroupRetrieveServiceGroupByReferenceEvents	Raised when the details of a Service Group for a specified reference is retrieved.	curam.serviceoffering.impl.ServiceGroup.retrieveS

Service Authorization Customization Points

The following sections list the available customization points for Service Authorizations.

Service Authorization Events

The following events are located in the curam.serviceauthorization.impl.ServiceAuthorization interface.

Table 24: Service Authorization Event Details

This table describes Service Authorization Customization Points

Event Class	Description	Event is raised before and after
ServiceAuthorizationFindLineItemByServiceProvisionDetailsEvents	Raised when Service Authorization Line Items for a particular Service are retrieved.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationAddLineItemEvents	Raised when a line item is added to a Service Authorization.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationInsertServiceAuthorizationEvents	Raised when a Service Authorization is created.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationAddVoucherToServiceAuthorizationEvents	Raised when a voucher is associated to a Service Authorization.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationDeleteVoucherFromServiceAuthorizationEvents	Raised when a voucher is disassociated with a Service Authorization.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationGetDerivedStatusEvents	Raised when the status of a Service Authorization is retrieved.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizatnAddLineItemEvents	Raised when a specified line item is added to the Service Authorization.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationAddLineItemEvents	Raised when a specified line item is added to the Service Authorization.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationAddSALIToSAUsingFrequencyAndAnchorDateEvents	Raised when Service Authorization Line Items are generated and added to a Service Authorization based on the frequency pattern and date.	curam.serviceauthorization.impl.ServiceAuthorizat

Service Authorization Line Item Events

The following events are located in the curam.serviceauthorization.impl.ServiceAuthorizationLineItem interface.

Table 25: Service Authorization Line Item Event Details

This table describes Service Authorization Line Item Events

Event Class	Description	Event is raised before and after
ServiceAuthorizationLineItemCloseEvents	Raised when a Service Authorization Line Item is closed.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationLineItemInsertEvents	Raised when a Service Authorization Line Item is inserted.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationLineItemModifyEvents	Raised when a Service Authorization Line Item is updated.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationLineItemCancelEvents	Raised when a Service Authorization Line Item is cancelled.	curam.serviceauthorization.impl.ServiceAuthorizat
ServiceAuthorizationLineItemGetDerivedStatusEvents	Raised when the status of a Service Authorization is retrieved.	curam.serviceauthorization.impl.ServiceAuthorizat

Event Class	Description	Event is raised before and after
ServiceAuthorizationLineItemGetRelatedProviderLineItemEvents	Raised when Provider Line Items related to a Service Authorization are retrieved.	curam.serviceauthorization.impl.ServiceAuthorizat

The following events are located in the curam.financial.impl.ProcessReassessmentForSALI interface.

Table 26: Service Authorization Line Item Event Details

This table describes Service Authorization Line Item Events

Event Class	Description	Event is raised before and after
ProcessReassessmentForSALIReAssessOnCancellationEvents	Raised when the payment for the Service Invoice Line Items or Provider Roster Line Items associated with the Service Authorization Line Item on cancellation of Service Authorization Line Item is processed.	curam.financial.impl.ProcessReassessmentForSA
ProcessReassessmentForSALIReAssessOnCloseEvents	Raised when the payment on closing the Service Authorization Line Item is processed.	curam.financial.impl.ProcessReassessmentForSA
ProcessReassessmentForSALIReAssessOnCreationEvents	Raised when reassessment on creation of new Service Authorization Line Item is triggered.	curam.financial.impl.ProcessReassessmentForSA
ProcessReassessmentForSALIReAssessOnModificationEvents	Raised when reassessment on modification of new Service Authorization Line Item is triggered.	curam.financial.impl.ProcessReassessmentForSA

Placement Customization Points

The following sections list the available customization points for Placements.

Placement Events

The following events are located in the curam.place.impl.Placement interface.

Table 27: Placement Event Details

This table describes Placement Events

Event Class	Description	Event is raised before and after
PlacementTransferClientEvents	Raised when a client is transferred to a new Place.	curam.place.impl.Placement.transferClient()
PlacementTransferClientToReservationEvents	Raised when a client is transferred to a new Place and a reservation is created for the new Place.	curam.place.impl.Placement.transferClientToRese

Event Class	Description	Event is raised before and after
PlacementGetOverlappingPlacementsForClientEvents	Raised when overlapping Placement details for a client are retrieved.	curam.place.impl.Placement.getOverlappingPlace

The following events are located in the curam.place.impl.FacilityInformation interface.

Table 28: Placement Event Details

This table describes Placement Events

Event Class	Description	Event is raised before and after
FacilityInformationRetrieveFacilityInformationEvents	Raised when the list of facility information for a Provider, service (or) provider type is retrieved.	curam.place.impl.FacilityInformation.retrieveFacili

The following events are located in the curam.place.impl.PlaceSearch interface.

Table 29: Placement Event Details

This table describes Placement Events

Event Class	Description	Event is raised before and after
PlaceSearchSearchAvailablePlacesEvents	Raised when an available Places in a Provider facility is searched.	curam.place.impl.PlaceSearch.searchAvailablePla

Reservation Events

The following events are located in the curam.reservation.impl.Reservation interface.

Table 30: Reservation Event Details

This table describes Reservation Events

Event Class	Description	Event is raised before and after
ReservationExpireEvents	Raised when a reservation is expired.	curam.reservation.impl.Reservation.expire()
ReservationCreateReservationEvents	Raised when a reservation is created.	curam.reservation.impl.Reservation.createReserv ()
ReservationConfirmPlacementEvents	Raised when a placement is created from a reservation.	curam.reservation.impl.Reservation.confirmPlacer ()
ReservationUpdateReservationEvents	Raised when a reservation is updated.	curam.reservation.impl.Reservation.updateReserv ()
ReservationCancelOverlappingActiveReservationsEvents	Raised when overlapping active reservations are cancelled.	curam.reservation.impl.Reservation.cancelOverlap ()
ReservationGetPlaceAvailableInDateRangeEvents	Raised when available placements in the given date range are retrieved.	curam.reservation.impl.Reservation.getPlaceAvail ()

Contract Customization Points

The following sections list the available customization points for Contracts.

Contract Version Events

The following events are located in the curam.contracts.impl.ContractVersion interface.

Table 31: Contract Version Event Details

This table describes Contract Version Events

Event Class	Description	Event is raised before and after
ContractVersionPrintContractEvents	Raised when a Contract Version is printed.	curam.contracts.impl.ContractVersion.printContra
ContractVersionPreviewContractEvents	Raised when a Contract Version is previewed.	curam.contracts.impl.ContractVersion.previewCon
ContractVersionValidateContractedProviderOfferingRateEvents	Raised when contracted provider Offering rates are validated.	curam.contracts.impl.ContractVersion.validateCon
ContractVersionValidateContractedProviderOfferingPlaceLimitEvents	Raised when contracted provider Offering Place Limits are validated.	curam.contracts.impl.ContractVersion.validateCon

The following events are located in the curam.contracts.impl.ContractVersionProviderOffering interface.

Table 32: Contract Version Event Details

This table describes Contract Version Events

Event Class	Description	Event is raised before and after
ContractVerProvOfferCopyNonContractedPORatesEvents	Raised when contracted provider offering rates are copied to contract.	curam.contracts.impl.ContractVersionProviderOffe
ContractVerProvOfferCreateDefaultRateEvents	Raised when a default Provider Offering Rate is created for the Provider.	curam.contracts.impl.ContractVersionProviderOffe
ContractVerPOCheckForDuplicatePOOnLiveContractEvents	Raised when duplicate Provider Offering on live contract is checked.	curam.contracts.impl.ContractVersionProviderOffe
ContractVerPOCreateContractedPORateEvents	Raised when contracted Provider Offering Rate is created if the non contracted Provider Offering Rate does not exist.	curam.contracts.impl.ContractVersionProviderOffe

Flat Rate Contract Events

The following events are located in the curam.contracts.impl.FlatRateContract interface.

Table 33: Flat Rate Contract Event Details

This table describes curam.contracts.impl.FlatRateContract

Event Class	Description	Event is raised before and after
FlatRateContractActivateEvents	This event is raised during activation of a flat rate contract.	curam.contracts.impl.FlatRateContract.activate()
curam.contracts.impl.FlatRateContractReEditEvents	Raised when a Flat Rate Contract is edited.	curam.contracts.impl.FlatRateContract.reEdit()

Event Class	Description	Event is raised before and after
curam.contracts.impl.FlatRateContractGenerateEvents	Raised when a Flat Rate Contract is generated.	curam.contracts.impl.FlatRateContract.generate()
curam.contracts.impl.FlatRateContractTerminateEvents	Raised when a Flat Rate Contract is terminated.	curam.contracts.impl.FlatRateContract.terminate()
curam.contracts.impl.FlatRateContractRenewEvents	Raised when a Flat Rate Contract is renewed.	curam.contracts.impl.FlatRateContract.renew()
curam.contracts.impl.FlatRateContractCloneFlatRateContractEvents	Raised when a Flat Rate Contract is cloned.	curam.contracts.impl.FlatRateContract.cloneFlatR

Utilization Contract Events

The following events are located in the curam.contracts.impl.UtilizationContract interface.

Table 34: Utilization Contract Event Details

This table describes Utilization Contract Events

Event Class	Description	Event is raised before and after
UtilizationContractDeleteEvents	Raised when a Utilization Contract is deleted.	curam.contracts.impl.UtilizationContract.delete()
UtilizationContractGenerateEvents	Raised when a Utilization Contract is generated.	curam.contracts.impl.UtilizationContract.generate
UtilizationContractActivateEvents	Raised when a Utilization Contract is activated	curam.contracts.impl.UtilizationContract.activate()
UtilizationContractTerminateEvents	Raised when a Utilization Contract is terminated.	curam.contracts.impl.UtilizationContract.terminate
UtilizationContractRenewEvents	Raised when a Utilization Contract is renewed.	curam.contracts.impl.UtilizationContract.renew()
UtilizationContractReEditEvents	Raised when a Utilization Contract is edited.	curam.contracts.impl.UtilizationContract.reEdit()
UtilizationContractCloneUtilizationContractEvents	Raised when a Utilization Contract is cloned.	curam.contracts.impl.UtilizationContract.cloneUtili()
UtilizationContractCloneUtilizationContractForRenewalEvents	Raised when a Utilization Contract is cloned for renewal.	curam.contracts.impl.UtilizationContract.cloneUtili()
UtilizationContractAmendEvents	Raised when a Utilization Contract is amended	curam.contracts.impl.UtilizationContract.amend()

Service Invoice Customization Points

The following sections list the available customization points for Service Invoices.

Service Invoice Events

The following events are located in the curam.financial.impl.ServiceInvoice interface.

Table 35: Service Invoice Event Details

This table describes Service Invoice Events

Event Class	Description	Event is raised before and after
ServiceInvoiceAddLineItemEvents	Raised when a Service Invoice Line Item is added to a Service Invoice.	curam.financial.impl.ServiceInvoice.addLineItem()
ServiceInvoiceBulkApproveEvents	Raised when many Service Invoice Line Items are approved together in bulk.	curam.financial.impl.ServiceInvoice.bulkApprove()
ServiceInvoiceGetServiceInvoiceDetailsStatusEvents	Raised when a Service Invoice status is retrieved.	curam.financial.impl.ServiceInvoice.getServiceInv

Service Invoice Line Item Events

The following events are located in the curam.financial.impl.ServiceInvoiceLineItem interface.

Table 36: Service Invoice Line Item Event Details

This table describes Service Invoice Line Item Events

Event Class	Description	Event is raised before and after
ServiceInvoiceLineItemApproveEvents	Raised when a Service Invoice Line Item is approved.	curam.financial.impl.ServiceInvoiceLineItem.appro
ServiceInvoiceLineItemDenyEvents	Raised when a Service Invoice Line Item is denied.	curam.financial.impl.ServiceInvoiceLineItem.deny
ServiceInvoiceLineItemSubmitEvents	Raised when a Service Invoice Line Item is submitted.	curam.financial.impl.ServiceInvoiceLineItem.subm
ServiceInvoiceLineItemMatchCaseEvents	Raised when a Case Reference in a Service Invoice Line Item is matched with a case participant.	curam.financial.impl.ServiceInvoiceLineItem.matc
ServiceInvoiceLineItemMatchPayeeEvents	Raised when payee details on a Service Invoice Line Item are matched with a provider/provider group.	curam.financial.impl.ServiceInvoiceLineItem.matc
ServiceInvoiceLineItemMatchProviderEvents	Raised when the details of the provider who is providing the service as taken from the Service Invoice Line Item, are matched with a registered provider.	curam.financial.impl.ServiceInvoiceLineItem.matc
ServiceInvoiceLineItemMatchClientEvents	Raised when client details are matched with the client who received the service.	curam.financial.impl.ServiceInvoiceLineItem.matc
ServiceInvoiceLineItemResolveServiceAuthorizationLineItemFromKeyIdentifierEvents	Raised when a Service Authorization Line Item is matched to a Service Invoice Line Item.	curam.financial.impl.ServiceInvoiceLineItem.resol
ServiceInvoiceLineItemValidateLineItemAgainstAuthorizationEvents	Raised when Service Authorization Line Item details are validated against Service Invoice Line Item details.	curam.financial.impl.ServiceInvoiceLineItem.valida
ServiceInvoiceLineItemGeneratePaymentEvents	Raised when a payment is processed for a Service Invoice Line Item.	curam.financial.impl.ServiceInvoiceLineItem.gene

Event Class	Description	Event is raised before and after
ServiceInvoiceLineItemDeterminePaymentAmountEvents	Raised when the payment amount is determined from the established rates for the period specified in the Service Invoice Line Item.	curam.financial.impl.ServiceInvoiceLineItem.deter
ServiceInvoiceLineItemMatchIdentifiersEvents	Raised when Case, Provider, Client details on a Service Invoice Line Item are matched.	curam.financial.impl.ServiceInvoiceLineItem.matc
ServiceInvoiceLineItemDeterminePaymentAmountFromReassessmentEvents	Raised when a payment amount is determined from the established rates to reassess the payment made for Service Invoice Line Item.	curam.financial.impl.ServiceInvoiceLineItem.deter
ServiceInvoiceLineItemMatchAgainstFlatRateContractEvents	Raised when Service Invoice Line Item details are matched with an existing Flat Rate Contract.	curam.financial.impl.ServiceInvoiceLineItem.matc
ServiceInvoiceLineItemRetrieveServiceAuthorizationEvents	Raised when Authorization details related to a Service Invoice Line Item are retrieved.	curam.financial.impl.ServiceInvoiceLineItem.retrie
ServiceInvoiceLineItemSubmitAndApproveSILIForCorrectionEvents	Raised when a Service Invoice Line Item Correction is submitted and approved.	curam.financial.impl.ServiceInvoiceLineItem.subm
ServiceInvoiceLineItemRetrieveSILIAmountPaidEvents	Raised when the amount paid against a Service Invoice Line Item is retrieved.	curam.financial.impl.ServiceInvoiceLineItem.retrie
ServiceInvoiceLineItemListSOAttendanceConfigurationsForSILIEvents	Raised when the Service Offering Attendance Configuration for the Service Offering related to a Service Invoice Line Item is retrieved.	curam.financial.impl.ServiceInvoiceLineItem.listSO
ServiceInvoiceLineItemGetAmountPaidEvents	Raised when the amount paid/ payable against a Service Invoice Line Item is retrieved.	curam.financial.impl.ServiceInvoiceLineItem.getAm

The following events are located in the curam.financial.impl.DeterminePaymentAmount interface.

Table 37: Service Invoice Line Item Event Details

This table describes Service Invoice Line Item Events

Event Class	Description	Event is raised before and after
DeterminePaymentAmountDeterminePaymentAmountForSILIEvents	Raised when the amount to be paid for the Service Invoice Line Item is determined.	curam.financial.impl.DeterminePaymentAmount.d

The following events are located in the curam.financial.impl.PaymentOptionProcessor interface.

Table 38: Service Invoice Line Item Event Details

This table describes Service Invoice Line Item Events

Event Class	Description	Event is raised before and after
PaymentOptionProcessorProcessServiceRaised when the payment option changeForSILEvents	Raised when the payment option change for a Service Offering made through invoices for the specified reassessment period is processed.	curam.financial.impl.PaymentOptionProcessor.pro
PaymentOptionProcessorProcessProviderRaised when the payment option change for a Provider made through invoices for the specified reassessment period is processed.		curam.financial.impl.PaymentOptionProcessor.pro

The following events are located in the curam.financial.impl.ServiceInvoiceLineItemHelper interface.

Table 39: Service Invoice Line Item Event Details

This table describes Service Invoice Line Item Events

Event Class	Description	Event is raised before and after
ServiceInvoiceLineItemHelperMatchClientEvents	Raised when the client details with the client who received the service is matched.	curam.financial.impl.ServiceInvoiceLineItemHelpe
ServiceInvoiceLineItemHelperMatchProviderEvents	Raised when the Provider details with Provider/Provider Group who provided the service is matched.	curam.financial.impl.ServiceInvoiceLineItemHelpe
ServiceInvoiceLineItemHelperMatchCaseEvents	Raised when the case reference in Service Invoice Line Item to the participant case is matched.	curam.financial.impl.ServiceInvoiceLineItemHelpe
ServiceInvoiceLineItemHelperMatchPayeeEvents	Raised when the payee details with Provider/Provider Group is matched.	curam.financial.impl.ServiceInvoiceLineItemHelpe

The following events are located in the curam.financial.impl.ServiceInvoiceLineItemTransactionHelper interface.

Table 40: Service Invoice Line Item Event Details

This table describes Service Invoice Line Item Events

Event Class	Description	Event is raised before and after
ServiceInvoiceLineItemTransactionHelperRaised when a Service Invoice Line Item transaction of type canceled is created.		curam.financial.impl.ServiceInvoiceLineItemTrans
ServiceInvoiceLineItemTransactionHelperRaised when a Service Invoice Line Item transaction of type denied is created.		curam.financial.impl.ServiceInvoiceLineItemTrans
ServiceInvoiceLineItemTransactionHelperRaised when a new transaction with the type as Invoiced for a Service Invoice Line Item is created.		curam.financial.impl.ServiceInvoiceLineItemTrans

Event Class	Description	Event is raised before and after
ServiceInvoiceLineItemTransactionPaymentEvents	Raised when a new transaction with the type as Payment for a Service Invoice Line Item is created.	curam.financial.impl.ServiceInvoiceLineItemTrans

Service Invoice Line Item Correction Events

The following events are located in the curam.financial.impl.ServiceInvoiceLineItemCorrection interface.

Table 41: Service Invoice Line Item Correction Event Details

This table describes Service Invoice Line Item Correction Events

Event Class	Description	Event is raised before and after
ServiceInvoiceLineItemCorrectionApprovedEvents	Raised when a Service Invoice Line Item Correction is approved.	curam.financial.impl.ServiceInvoiceLineItemCorre
ServiceInvoiceLineItemCorrectionDeniedEvents	Raised when a Service Invoice Line Item Correction is denied.	curam.financial.impl.ServiceInvoiceLineItemCorre
ServiceInvoiceLineItemCorrectionSubmittedEvents	Raised when a Service Invoice Line Item Correction is submitted.	curam.financial.impl.ServiceInvoiceLineItemCorre
ServiceInvoiceLineItemCorrectionValidateLineItemAgainstAuthorizationEvents	Raised when the details specified in Service Invoice Line Item is validated against the Service Authorization Line Item details.	curam.financial.impl.ServiceInvoiceLineItemCorre

Attendance Customization Points

The following sections list the available customization points for Attendance.

Provider Roster Line Item Events

The following events are located in the curam.attendance.impl.ProviderRosterLineItem interface.

Table 42: Provider Roster Line Item Event Details

This table describes Provider Roster Line Item Events

Event Class	Description	Event is raised before and after
ProviderRosterLineItemModifyRosterLineItemOnModificationOfSALIEvents	Raised when a Roster Line Item is modified on modification of a service authorization line item.	curam.attendance.impl.ProviderRosterLineItem.m ()
ProviderRosterLineItemModifyRosterLineItemEvents	Raised when a Roster Line Item is modified.	curam.attendance.impl.ProviderRosterLineItem.m ()
ProviderRosterLineItemModifyForDailyAttendanceEvents	Raised when a Roster Line Item is modified based on daily attendance.	curam.attendance.impl.ProviderRosterLineItem.m
ProviderRosterLineItemApproveEvents	Raised when a Roster Line Item is approved.	curam.attendance.impl.ProviderRosterLineItem.ap
ProviderRosterLineItemAddClientEvents	Raised when a Roster Line Item is created for a new client.	curam.attendance.impl.ProviderRosterLineItem.ac

Event Class	Description	Event is raised before and after
ProviderRosterLineItemAddAbsenceEvents	Raised when an absence period is added to a Roster Line Item.	curam.attendance.impl.ProviderRosterLineItem.ad
ProviderRosterLineItemSubmitEvents	Raised when a Roster Line Item is submitted.	curam.attendance.impl.ProviderRosterLineItem.su
ProviderRosterLineItemSubmitRosterLineItemEvents	Raised when a Roster Line Item from a Roster is submitted.	curam.attendance.impl.ProviderRosterLineItem.su
ProviderRosterLineItemDenyEvents	Raised when a Roster Line Item is denied.	curam.attendance.impl.ProviderRosterLineItem.de
ProviderRosterLineItemSubmitAndApproveRosterLineItemCorrectionEvents	Raised when a Roster Line Item correction is submitted for approval.	curam.attendance.impl.ProviderRosterLineItem.su
ProviderRosterLineItemAccommodateOnExistingRosterEvents	Raised when a Roster Line Item is accommodated on an existing Roster.	curam.attendance.impl.ProviderRosterLineItem.ac
ProviderRosterLineItemCalculateExpectedUnitsEvents	Raised when expected units on a Provider Roster Line Item are calculated.	curam.attendance.impl.ProviderRosterLineItem.ca
ProviderRosterLineItemUpdateExpectedUnitsOnNextRosterEvents	Raised when expected units for Roster Line Items on a Roster for a particular client are updated.	curam.attendance.impl.ProviderRosterLineItem.up
ProviderRosterLineItemListSOAttendanceConfigurationEvents	Raised when the Service Delivery Attendance Configuration list for a Roster Line Item is retrieved.	curam.attendance.impl.ProviderRosterLineItem.lis
ProviderRosterLineItemViewExceptionTaskEvents	Raised when an exception task is viewed for a Provider Roster Line Item.	curam.attendance.impl.ProviderRosterLineItem.vi
ProviderRosterLineItemGetCorrectionIndicatorEvents	Raised when the correction indicator for a Provider Roster Line Item is retrieved.	curam.attendance.impl.ProviderRosterLineItem.ge
ProviderRosterLineItemGetCaseHeaderDetailsEvents	Raised when Case Header Details are retrieved.	curam.attendance.impl.ProviderRosterLineItem.ge
ProviderRosterLineItemGetPayBasedOnAttendanceDetailsEvents	Raised when the details for Pay Based on Attendance is retrieved.	curam.attendance.impl.ProviderRosterLineItem.ge
ProviderRosterLineItemGetAbsenceEvents	Raised when the Absence period on a Provider Roster Line Item is retrieved.	curam.attendance.impl.ProviderRosterLineItem.ge
ProviderRosterLineItemGetDailyAttendanceEvents	Raised when Daily attendance is retrieved from a Provider Roster Line Item.	curam.attendance.impl.ProviderRosterLineItem.ge
ProviderRosterLineItemGetOriginalDetailsEvents	Raised when Provider Roster Line Item details are retrieved.	curam.attendance.impl.ProviderRosterLineItem.ge

The following events are located in the curam.attendance.impl.AttendanceInformationProcessing interface.

Table 43: Provider Roster Line Item Event Details

This table describes Provider Roster Line Item Events

Event Class	Description	Event is raised before and after
AttendanceInfoProcessGetRosterLineItemsForCaseEvents	Raised when Case Roster Line Items for a case is retrieved.	curam.attendance.impl.AttendanceInformationPro
AttendanceInfoProcessGetRosterLineItemsForClientEvents	Raised when Client Roster Line Items for a client is retrieved.	curam.attendance.impl.AttendanceInformationPro

The following events are located in the curam.attendance.impl.ProviderRosterLineItemHelper interface.

Table 44: Provider Roster Line Item Event Details

This table describes Provider Roster Line Item Events

Event Class	Description	Event is raised before and after
ProviderRosterLineItemHelperMatchClientEvents	Raised when the client is matched based on client reference number, first name and last name and address.	curam.attendance.impl.ProviderRosterLineItemHe
ProviderRosterLineItemHelperMatchCaseEvents	Raised when the case is matched by the case reference number.	curam.attendance.impl.ProviderRosterLineItemHe
ProviderRosterLineItemHelperMatchVoucherEvents	Raised when the voucher is matched by number assigned to the voucher that has been issued to the client.	curam.attendance.impl.ProviderRosterLineItemHe

The following events are located in the curam.attendance.impl.ProviderRosterLineItemTransactionHelper interface.

Table 45: Provider Roster Line Item Event Details

This table describes Provider Roster Line Item Events

Event Class	Description	Event is raised before and after
ProviderRosterLineItemTransactionHelperCreateDeniedTransactionEvents	Raised when a Provider Roster Line Item transaction of type denied is created.	curam.attendance.impl.ProviderRosterLineItemTra
ProviderRosterLineItemTransactionHelperCreateCanceledTransactionEvents	Raised when a Provider Roster Line Item transaction of type canceled is created.	curam.attendance.impl.ProviderRosterLineItemTra
ProviderRosterLineItemTransactionHelperCreateProviderRosterLineItemTransactionEvents	Raised when Provider Roster Line Item transactions are created.	curam.attendance.impl.ProviderRosterLineItemTra

Provider Roster Line Item Correction (PRLI Correction) Events

The following events are located in the curam.attendance.impl.PRLICorrection interface.

Table 46: Provider Roster Line Item Correction (PRLI Correction) Event Details

This table describes Provider Roster Line Item Correction (PRLI Correction) Events

Event Class	Description	Event is raised before and after
PRLICorrectionApproveEvents	Raised when a Provider Roster Line Item Correction is approved.	curam.attendance.impl.PRLICorrection.approve()
PRLICorrectionDenyEvents	Raised when a Provider Roster Line Item Correction is denied.	curam.attendance.impl.PRLICorrection.deny()
PRLICorrectionSubmitEvents	Raised when a Provider Roster Line Item Correction is submitted.	curam.attendance.impl.PRLICorrection.submit()

Roster Events

The following events are located in the curam.attendance.impl.RosterProcessing interface.

Table 47: Roster Event Details

This table describes Roster Events

Event Class	Description	Event is raised before and after
RosterProcessingGenerateRosterManuallyEvents	Raised when a blank roster is generated manually.	curam.attendance.impl.RosterProcessing.generat
RosterProcessingGetApplicableRosterRangeEvents	Raised when the applicable roster range is retrieved.	curam.attendance.impl.RosterProcessing.getAppl
RosterProcessingCreateRosterOverlappingDateEvents	Raised when roster for Service Authorization Line Item overlapping date is created.	curam.attendance.impl.RosterProcessing.createR
RosterProcessingCreateRosterEvents	Raised when roster for a Service Authorization Line Item is created.	curam.attendance.impl.RosterProcessing.createR

Attendance Payment Frequency Events

The following events are located in the curam.attendance.impl.AttendancePaymentFrequency interface.

Table 48: Attendance Payment Frequency Event Details

This table describes Attendance Payment Frequency Events

Event Class	Description	Event is raised before and after
AttendancePaymentFrequencyGetDetailsEvents	Raised when the status of an attendance payment configuration entry is retrieved.	curam.attendance.impl.AttendancePaymentFrequ()

Service Offering Attendance Configuration Events

The following events are located in the curam.attendance.impl.SOAttendanceConfiguration interface.

Table 49: Service Offering Attendance Configuration Event Details

This table describes Service Offering Attendance Configuration Events

Event Class	Description	Event is raised before and after
SOAttendanceConfigurationGetDerivedStatusEvents	Raised when the status of a Service Offering Attendance Configuration is retrieved.	curam.attendance.impl.SOAttendanceConfiguratio

Service Offering Attendance Payment Events

The following events are located in the interface.

Table 50: Service Offering Attendance Event Details

This table describes Service Offering Attendance Payment

Event Class	Description	Event is raised before and after
SOAttendancePaymentGetDerivedStatusEvents	Raised when the status of a Service Offering Attendance Payment is retrieved.	curam.attendance.impl.SOAttendancePayment.ge

Financial Customization Points

The following sections list the available customization points for Financials.

Financial Events

The following events are located in the curam.financial.impl.FinancialAPI interface.

Table 51: Financial Event Details

This table describes Financial Events

Event Class	Description	Event is raised before and after
FinancialAPIRetrieveServiceDeliverySummaryInformationEvents	Raised when the service delivery summary information is retrieved for a a case, client and service.	curam.financial.impl.FinancialAPI.retrieveServiceD
FinancialAPIRetrieveServiceDelivrySummaryInformationEvents	Raised when the service delivery summary information is retrieved for a service and case participant role.	curam.financial.impl.FinancialAPI.retrieveServiceD
FinancialAPIListReassessmentResultsEvents	Raised when the reassessment results for all the product deliveries created to deliver the services for the given service authorization is retrieved.	curam.financial.impl.FinancialAPI.listReassessme

The following events are located in the curam.financial.impl.GenerateOverUnderPayment interface.

Table 52: Financial Event Details

This table describes Financial Events

Event Class	Description	Event is raised before and after
GenerateOverUnderPaymentGenerateOverPaymentForPRLEvents	RaiseOverPaymentForPRLEvents for Provider Roster Line Item is generated.	curam.financial.impl.GenerateOverUnderPayment
GenerateOverUnderPaymentGenerateOverPaymentForSILEvents	RaiseOverPaymentForSILEvents for Service Invoice Line Item is generated.	curam.financial.impl.GenerateOverUnderPayment
GenerateOverUnderPaymentGenerateUnderPaymentForPRLEvents	RaiseUnderPaymentForPRLEvents for Provider Roster Line Item is generated.	curam.financial.impl.GenerateOverUnderPayment
GenerateOverUnderPaymentGenerateUnderPaymentForSILEvents	RaiseUnderPaymentForSILEvents for Service Invoice Line Item is generated.	curam.financial.impl.GenerateOverUnderPayment

The following events are located in the curam.financial.impl.PaymentProcessing interface.

Table 53: Financial Event Details

This table describes Financial Events

Event Class	Description	Event is raised before and after
PaymentProcessingProcessPaymentRaisedEvents	RaisedEvents the payment for the Service Invoice Line Item is processed.	curam.financial.impl.PaymentProcessing.processI
PaymentProcessingProcessPaymentRaisedEvents	RaisedEvents for reassessment is processed.	curam.financial.impl.PaymentProcessing.processI
PaymentProcessingApproveAndActivateRaisedEvents	RaisedEvents for Provider is approved and activated.	curam.financial.impl.PaymentProcessing.approveA
PaymentProcessingSubmitForApprovalEvents	RaisedEvents when the case is submitted and approved.	curam.financial.impl.PaymentProcessing.submitFo
PaymentProcessingDeterminePayeeDetailsEvents	RaisedEvents the payee for a given Provider and given period is determined.	curam.financial.impl.PaymentProcessing.determin
PaymentProcessingDeterminePayeeDetailsEvents	RaisedEvents the payee for a given Provider and given period is determined.	curam.financial.impl.PaymentProcessing.determin

The following events are located in the curam.financial.impl.ProcessCaseNominee interface.

Table 54: Financial Event Details

This table describes Financial Events

Event Class	Description	Event is raised before and after
ProcessCaseNomineeCreateCaseNomineeEvents	RaisedEvents the Provider Group is created as the case nominee for the given product delivery case of the Provider.	curam.financial.impl.ProcessCaseNominee.create
ProcessCaseNomineeCreateCaseNomineeEvents	RaisedEvents the payee is created as the case nominee for the given product delivery case of the Provider.	curam.financial.impl.ProcessCaseNominee.create

Event Class	Description	Event is raised before and after
ProcessCaseNomineeCreateCaseNomineeEvents	Raised when the Provider Group is created as the case nominee for the given product delivery case of the Provider and contract frequency.	curam.financial.impl.ProcessCaseNominee.create
ProcessCaseNomineeCreateNomineeForAllCaseEvents	Raised when case nominees are created for all existing cases associated with the Provider for whom the Provider Group Associate Payment Configuration is created.	curam.financial.impl.ProcessCaseNominee.create
ProcessCaseNomineeReassignCaseNomineeObjectiveOnCancellationEvents	Raised when Objective Case nominee objectives associated with the Provider Group Associate Payment Configuration is reassigned on cancellation of payment configuration.	curam.financial.impl.ProcessCaseNominee.reassi
ProcessCaseNomineeReassignCaseNomineeObjectiveOnModificationEvents	Raised when Objective Case nominee objectives associated with the Provider Group Associate Payment Configuration is reassigned on modification of Provider Group Associate.	curam.financial.impl.ProcessCaseNominee.reassi
ProcessCaseNomineeReassignCaseNomineeObjectiveOnPayeeModificationEvents	Raised when Objective Case nominee objectives associated with the Provider Group Associate Payment Configuration is reassigned on modification of payment configuration.	curam.financial.impl.ProcessCaseNominee.reassi
ProcessCaseNomineeReAssignCaseNomineeObjectiveOnCasePayeeModificationEvents	Raised when Objective Case nominee objectives associated with the old payee to the new payee for the given product delivery case is reassigned.	curam.financial.impl.ProcessCaseNominee.reAss

The following events are located in the curam.financial.impl.RateValidator interface.

Table 55: Financial Event Details

This table describes Financial Events

Event Class	Description	Event is raised before and after
RateValidatorValidateRatesEvents	Raised when there is any gap or overlapping in the period of the set of rates provided are validated.	curam.financial.impl.RateValidator.validateRates()

Referral Customization Points

The following sections list the available customization points for Referrals.

Referral Events

The following events are located in the curam.referral.impl.Referral interface.

Table 56: Referral Event Details

This table describes Referral Events

Event Class	Description	Event is raised before and after
ReferralSendNotificationEvents	Raised when a notification letter to the Concern Role is sent.	curam.referral.impl.Referral.sendNotification()
ReferralCreateReferralRoleEvents	Raised when a referral role record for a referral is created.	curam.referral.impl.Referral.createReferralRole()

The following events are located in the curam.referral.impl.ReferralNotification interface.

Table 57: Referral Event Details

This table describes Referral Events

Event Class	Description	Event is raised before and after
ReferralNotificationGenerateNotificationDocumentEvents	Raised when a notification document is generated.	curam.referral.impl.ReferralNotification.generateN
ReferralNotificationSendNotificationEvents	Raised when a notification is sent to the Concern Role.	curam.referral.impl.ReferralNotification.sendNotifi

Service Delivery Customization Points

The following sections list the available customization points for Service Deliveries.

Service Delivery Events

The following events are located in the curam.servicedelivery.impl.ServiceDeliveryEstimatedCost interface.

Table 58: Service Delivery Event Details

This table describes Service Delivery Events

Event Class	Description	Event is raised before and after
ServiceDeliveryEstimatedCostDeterminationEvents	Raised when the rate for the Service Offering is determined.	curam.servicedelivery.impl.ServiceDeliveryEstima
ServiceDeliveryEstimatedCostDeterminationWithEffectiveDateEvents	Raised when the rate for the Service Offering is determined for each service occurrence date.	curam.servicedelivery.impl.ServiceDeliveryEstima

The following events are located in the curam.servicedelivery.impl.ServiceDelivery interface.

Table 59: Service Delivery Event Details

This table describes Service Delivery Events

Event Class	Description	Event is raised before and after
ServiceDeliverySubmitEvents	Raised when the Service Delivery is submitted.	curam.servicedelivery.impl.ServiceDelivery.submi

The following events are located in the curam.servicedeliveryevaluation.impl.ServiceDeliveryEvaluation interface.

Table 60: Service Delivery Event Details

This table describes Service Delivery Events

Event Class	Description	Event is raised before and after
ServiceDeliveryEvalCalculateOutcomeEvent	Raised when the Service Delivery Evaluation is calculated.	curam.servicedeliveryevaluation.impl.ServiceDeliv

1.3 CPM Workflow Process Definitions

Merative ™ SPM Provider Management includes a number of workflow process definitions. Agencies can copy any of these workflow process definitions to a custom workflow directory and modify them. The workflows that are included in the initial installation of CPM are described in the following links.

Introduction

Merative ™ SPM Provider Management includes a number of workflow process definitions. Agencies can copy any of these workflow process definitions to a custom workflow directory and modify them.

Note: Custom versions of workflows always take precedence over workflows included in the initial installation.

External Enquiry Workflow

Use the links in this section to learn about the External Enquiry Workflow.

Enacted from

This workflow is enacted when an external party uses Merative ™ SPM Provider Management (CPM) to inquire about the possibility of registering as a provider.

For example, Mr. and Mrs. Smith use an external-facing system to inquire about fostering children. This workflow is enacted by `curam.cpm.eua.facade.impl.ExternalProviderEnquiry.createEnquiry`.

Source Location

This link provides the location of the External Inquiry workflow .xml file.

`EJBServer/components/CPM/workflow/EXTERNALENQUIRYWORKFLOW_v1.xml`

Default Behavior

The workflow that is included with Merative ™ SPM Provider Management creates a manual activity to assign the inquiry to a user for converting a provider inquiry into an enrolled provider.

This manual activity is allocated by using a function allocation strategy. The default implementation of this operation allocates the activity to the provider inquiry work queue. The reviewer can choose to either transfer the inquiry to an enrolled provider or close the inquiry. After this activity is completed, the workflow also gets completed.

Event Details

This link provides the notation of the event details.

The notation of the following event details is as follows:

Table 61: External Enquiry Event Details

Event Raised	Primary Event Data	Raised From
PROVIDERENQUIRY.TRANSFERENQUIRYTOPROVIDER	providerEnquiryID	curam.cpm.facade.impl.ProviderEnquiry.closeProv
PROVIDERENQUIRY.CLOSEENQUIRY	providerEnquiryID	curam.cpm.facade.impl.ProviderEnquiry.transferE

Home Study Approval Workflow

Use the links in this section to learn about the Home Study Approval workflow.

Enacted from

This workflow is enacted whenever a user submits a home study for approval.

This workflow is enacted from `curam.cpm.workflowprocesses.homestudy.impl.HomeStudyImpl.submit`.

Source Location

This link provides the location of the Home Study Approval workflow .xml file.

`EJBServer/components/CPM/workflow/HOMESTUDYAPPROVAL_v1.xml`

Default Behavior

This workflow automatically creates a manual activity to assign a home study recommendation to a user for approval.

The default implementation of this operation submits the home study recommendation to the supervisor of the user who submitted the request. Agencies might want to alter this default behavior.

For example, an agency might want to route the approval request to a user other than the supervisor or to a group of users. The manual activity is allocated by using a function allocation strategy. The reviewer can choose to either approve or reject the approval request. After this activity is completed, the workflow also gets completed.

Event Details

This link provides the notation of the event details.

The notation of the following event details is as follows:

Table 62: Home Study Approval Event Details

Event Raised	Primary Event Data	Raised From
PROVIDERMANAGEMENT.HOMESTUDYAPPROVED	homeStudyID	curam.homestudy.impl.approve
PROVIDERMANAGEMENT.HOMESTUDYRETURNED	homeStudyID	curam.homestudy.impl.reject

New Invoice Created Workflow

Use the links in this section to learn about the New Invoice Created workflow.

Enacted from

This workflow is enacted whenever an external user submits an invoice for processing.

This workflow is enacted from `curam.cpm.facade.impl.Request.createFinancialsTask`.

Source Location

This link provides the location of the New Invoice Created workflow .xml file.

`EJBServer/components/CPM/workflow/NEWINVOICECREATED_v1.xml`

Default Behavior

This workflow creates a manual activity to assign an invoice that was submitted by an external user to another user for processing

. The default implementation of this operation submits the invoice to a financial user. Agencies might want to alter this default behavior.

For example, an agency might want to submit the invoice to a different user for processing. The manual activity is allocated by using a function allocation strategy.

Event Details

This link provides the notation of the event details for the New Invoice Created workflow.

The notation of the event details is as follows:

Table 63: New Invoice Created Event Details

Event Raised	Primary Event Data	Raised From
NEWINVOICECREATED.INVOICECANCELLED	serviceInvoiceID	curam.cpm.facade.impl.cancelServiceInvoic
NEWINVOICECREATED.INVOICESUBMITTED	serviceInvoiceID	curam.cpm.facade.impl.submitSILIForProces

Service Invoice Exception Processing Workflow

Use the links in this section to learn about the Service Invoice Exception Processing workflow.

Enacted from

This workflow is enacted when there is insufficient correct data to match a service invoice line item against its corresponding service authorization.

This workflow is called from `curam.financial.impl.ServiceInvoiceLineItemImpl.processInvoiceLineItem`.

Source Location

This link provides the location of the Service Invoice Exception Processing workflow .xml file.

`EJBServer/components/CPM/workflow/SERVICEINVOICEEXCEPTIONPROCESSING_v1.xml`.

Default Behavior

This workflow creates a manual activity for a user to review service invoice details that do not correspond with the service authorization associated with the invoice.

During invoice processing, certain details on a service invoice line item (SILI) must correspond to the details on the service authorization that is associated with the invoice otherwise the invoice is not paid. The default implementation of this operation allocates the activity to the invoice exception processing work queue for a financial user to review.

Agencies might want to alter this default behavior, for example, by routing the activity to a different work queue. The reviewer can choose to changes the service invoice line item and submit for reevaluation or deny/cancel the SILI. After this activity is completed, the workflow is also completed. This manual activity is allocated by using a function allocation strategy.

The modeled operation for this workflow is: `curam.cpm.workflowprocesses.impl.WorkflowAllocationFunction.siliExceptionProcessin`

Event Details

This link provides the notation of the event details for the Service Invoice Exception Processing workflow.

The notation of the event details is as follows:

Table 64: Service Invoice Exception Processing Event Details

Event Raised	Primary Event Data	Raised From
`PROVIDERMANAGEMENT.SILIPROCESSED`	`serviceInvoiceLineItemID`	`curam.financial.impl.ServiceInvoiceLineIt`
`PROVIDERMANAGEMENT.SILICANCELED`	`serviceInvoiceLineItemID`	`curam.financial.impl.ServiceInvoiceLineIt`
`PROVIDERMANAGEMENT.SILIDENIED`	`serviceInvoiceLineItemID`	`curam.financial.impl.ServiceInvoiceLineIt`

Service Invoice Line Item Approval Workflow

Use the links in this section to learn about the Service Invoice Line Item Approval workflow.

Enacted from

This workflow is enacted when a service invoice line item requires manual approval and has reached the "Pending Approval" status, after successful processing. This workflow is called from curam.financial.impl.ServiceInvoiceLineItemImpl.enactSILIApprovalWorkflow.

Source Location

This link provides the location of the Service Invoice Line Item Approval workflow .xml file.

EJBServer/components/CPM/workflow/SERVICEINVOICELINEITEMAPPROVAL_v1.xml.

Default Behavior

The workflow creates a manual activity to review a service invoice line item and approve or deny it.

The default implementation of this operation allocates the activity to the invoice exception processing work queue for a financial user to approve. Agencies might want to alter this default behavior.

For example, by routing the activity to a different work queue or to a different user. This manual activity is allocated by using a function allocation strategy. The modeled operation for this workflow is `curam.cpm.workflowprocesses.impl.WorkflowAllocationFunction.siliExceptionProcessin`

The reviewer can choose to either approve or deny the service invoice line item. After this activity is completed, the workflow also gets completed.

Event Details

This link provides the notation of the event details for the Service Invoice Line Item Approval workflow.

The notation of the following event details is as follows:

Table 65: Service Invoice Line Item Event Details

Event Raised	Primary Event Data	Raised From
PROVIDERMANAGEMENT.SILIAPPROVED	serviceInvoiceLineItemID	curam.financial.impl.ServiceInvoiceLineItemImpl
PROVIDERMANAGEMENT.SILIDENIED	serviceInvoiceLineItemID	curam.financial.impl.ServiceInvoiceLineItemImpl

Service Invoice Line Item Correction Approval Workflow

Enacted from

This workflow is enacted when a service invoice line item correction is submitted for approval.

This workflow is called from `curam.financial.impl.ServiceInvoiceLineItemCorrectionImpl.enactCorrectionApprovalW`

Source Location

This link provides the location of the Service Invoice Line Item Approval workflow .xml file.

EJBServer/components/CPM/workflow/SERVICEINVOICELINEITEMCORRECTIONAPPROVAL_v1.xml

Default Behavior

This workflow automatically creates a manual activity to assign a service invoice line item correction to a user for approval.

This manual activity is allocated by using a function allocation strategy. The modeled operation for this `curam.cpm.workflowprocesses.impl.WorkflowAllocationFunction.siliExceptionProcessing`. The default implementation of this operation submits the service invoice line item correction to the supervisor of the user who submitted the request. Agencies might want to alter this default behavior.

For example, an agency might want to route the approval request to a user other than the supervisor or to a group of users. The reviewer can choose to either approve or deny the service invoice line item correction. After this activity is completed, the workflow is also completed.

Event Details

This link provides the notation of the event details for the Service Invoice Line Item Approval workflow.

The notation of the following event details is as follows:

Table 66: Service Invoice Line Item Correction Approval Event Details

Event Raised	Primary Event Data	Raised From
PROVIDERMANAGEMENT.SILICORRECTIONAPPROVED	ServiceInvoiceLineItemID	curam.cpm.financial.impl.ServiceInvoiceLineItemCorr
PROVIDERMANAGEMENT.SILICORRECTIONDENIED	ServiceInvoiceLineItemID	curam.cpm.financial.impl.ServiceInvoiceLineItemCorr

Supervisor Request Decision Workflow

Enacted From

This workflow is enacted when a user submits a request to be set up with an external user account.

This workflow is called from `curam.cpm.eua.facade.impl.ExternalRequests.submitRequest`.

Source Location

This link provides the location of the Supervisor Request Decision workflow .xml file.

`EJBServer/components/CPM/workflow/SUPERVISORREQUESTDECISION_v1.xml`

Default Behavior

This workflow creates a manual activity that submits a request of an external user to an administrator user for the external user to be included in the system.

The request is for the external user to be set up in one of the following categories:

- A provider member
- A provider participant
- A provider group member

- A provider group associate

The default implementation submits the request to the external request work queue for an administrator to consider for approval. Agencies might want to alter this default behavior.

For example, by routing the activity to a different work queue. The administrator can approve or reject the request of the external user.

After this activity is completed, the workflow also is completed.

Event Details

This link provides the notation of the event details for the Supervisor Request Decision workflow.

The notation of the following event details is as follows:

Table 67: Supervisor Request Decision Event Details

Event Raised	Primary Event Data	Raised From
REQUESTDECISION.REQUESTACCEPTED	requestID	curam.cpm.facade.impl.Request.raiseAcceptReques
REQUESTDECISION.REQUESTREJECTED	requestID	curam.cpm.facade.impl.Request.rejectRequest

Supervisor View New External User Task Notification Workflow

Use the links in this section to learn about the Supervisor View New External User Task Notification workflow.

Enacted from

This workflow is enacted when an administrator user creates an external user account.

This workflow is called from `curam.cpm.eua.facade.impl.ExternalUser.createExternalUser`.

Source Location

This link provides the location of the Supervisor View New External User Task Notification workflow .xml file.

`EJBServer/components/CPM/workflow/SUPERVISORVIEWNEWEXTERNALUSERTASKNOTIFICATION_v1.xml`

Default Behavior

This workflow creates a route activity to send a notification to the owner of an external user.

When an administrator user sets up a new external user account, the owner of the new external user is sent a notification to inform them that the account was created successfully. The default implementation sends a notification to the resource manager who enrolled the external user. Agencies might want to alter this default.

For example, an agency might want to send the notification to a different user. After this activity is completed, the workflow is also completed.

Events Details

No Events are raised for this workflow.

Roster Exception Processing Workflow

Use the links in this section to learn about the Roster Exception Processing workflow.

Enacted From

This workflow is enacted when there is insufficient correct data to match a roster line item against its corresponding service authorization.

This workflow is called from `curam.attendance.impl.ProviderRosterLineItemImpl.processRosterLineItem`.

Source Location

This link provides the location of the Roster Exception Processing workflow .xml file.

`EJBServer/components/CPM/workflow/ROSTEREXCEPTIONPROCESSING_v1.xml`

Default Behavior

This workflow creates a manual activity for a user to review details of a provider roster line item that does not correspond to its associated service authorization.

Certain details on a roster line item must match the details on the service authorization that is associated with the roster line item otherwise any attendance-based payments that are related to the roster are not paid.

The default implementation of this operation allocates this activity to the roster exception processing work queue for a user to review. Agencies might want to alter this default behavior, for example, by routing the activity to a different work queue. This workflow creates a manual activity to review the provider roster line item in question. This manual activity is allocated by using a function allocation strategy. The modeled operation for this activity is curam.cpm.workflowprocesses.impl.WorkflowAllocationFunction.prliExceptionProcessingAllocationStrategy. The reviewer can choose to change the provider roster line item and submit it for reevaluation or deny/cancel the provider roster line item. After this activity is completed, the workflow is also completed.

Event Details

This link provides the notation of the event details for the Roster Exception Processing workflow.

The notation of the following event details is as follows:

Table 68: Roster Exception Processing Event Details

Event Raised	Primary Event Data	Raised From
`ROSTER.PRLI_PROCESSED`	`providerRosterLineItemID`	`curam.attendance.impl.ProviderRosterLineI`
`ROSTER.PRLI_CANCELED`	`providerRosterLineItemID`	`curam.financial.impl.ServiceInvoiceLineIt`
`ROSTER.PRLI_DENIED`	`providerRosterLineItemID`	`curam.attendance.impl.ProviderRosterLineI`

New Client Added to Roster Workflow

Use the links in this section to learn about the New Client Added to Roster workflow.

Enacted From

This workflow is enacted whenever a provider roster line item is created during creation or modification of a service authorization line item.

This workflow is called from `curam.serviceauthorization.impl.ServiceAuthorizationLineItemImpl.generateTaskForNewClientAdded`.

Source Location

This link provides the location of the New Client Added to Roster workflow .xml file.

EJBServer/components/CPM/workflow/NEWCLIENTADDEDTOROSTER_v1.xml

Default Behavior

This workflow creates an activity to send a notification to the owner of a provider roster line item when a client is added to a roster.

This notification is sent only during the creation or modification of a service authorization line item that leads to creation of a roster line item.

If the roster line item is submitted or canceled or denied, the corresponding generated notification is removed from the user's task inbox.

Event Details

This link provides the notation of the event details for the New Client Added to Roster workflow.

The notation of the following event details is as follows:

Table 69: New Client Added to Roster Event Details

Event Raised	Primary Event Data	Raised From
ROSTER.PRLI_PROCESSED	providerRosterLineItemID	curam.attendance.impl.ProviderRosterLineItemImp
ROSTER.PRLI_CANCELED	providerRosterLineItemID	curam.financial.impl.ServiceInvoiceLineItemImpl
ROSTER.PRLI_DENIED	providerRosterLineItemID	curam.attendance.impl.ProviderRosterLineItemImp

Roster Line Item Approval Workflow

Use the links in this section to learn about the Roster Line Item Approval workflow.

Enacted From

This workflow is enacted when a provider roster line item requires manual approval and has reached the **Pending Approval** status.

This workflow is called from `curam.attendance.impl.ProviderRosterLineItemImpl.approve`.

Source Location

This link provides the location of the Roster Line Item Approval workflow .xml file.

EJBServer/components/CPM/workflow/ROSTERLINEITEMAPPROVAL_v1.xml

Default Behavior

This workflow creates a manual activity to review a provider roster line item and approve or deny it.

The default implementation of this operation allocates the activity to the roster exception processing work queue. Agencies might want to alter this default behavior, for example, by routing the activity to a different work queue. This manual activity is allocated by using a function allocation strategy. The modeled operation for this workflow is `curam.cpm.workflowprocesses.intf.WorkflowAllocationFunction.prliExceptionProcessin`

The reviewer can choose to either approve or deny the provider roster line item. After this activity is completed, the workflow also gets completed.

Event Details

This link provides the notation of the event details for the Roster Line Item Approval workflow.

The notation of the following event details is as follows:

Table 70: Roster Line Item Approval Event Wait Activities Details

Event Raised	Primary Event Data	Raised From
`ROSTER.PRLI_APPROVED`	`providerRosterLineItemID`	`curam.attendance.impl.ProviderRosterLineItemImp`
`ROSTER.PRLI_DENIED`	`providerRosterLineItemID`	`curam.attendance.impl.ProviderRosterLineItemImp`

Roster Line Item Correction Approval Workflow

Use the links in this section to learn about the Roster Line Item Correction Approval workflow.

Enacted From

The workflow is enacted whenever a user approves a provider roster line item correction.

This workflow is called from `curam.attendance.impl.PRLICorrectionImpl.approve`.

Source Location

This link provides the location of the Roster Line Item Correction Approval workflow .xml file.

`EJBServer/components/CPM/workflow/ROSTERLINEITEMCORRECTIONAPPROVAL_v1.xml`

Default Behavior

This workflow contains the processing that is involved in approving a correction made to a provider roster line item.

This workflow creates a manual activity to review a provider roster line item correction and approve or deny it. The default implementation of this operation allocates the activity to the roster exception processing work queue.

This manual activity is allocated by using a function allocation strategy. The modeled operation for this workflow is `curam.cpm.workflowprocesses.intf.WorkflowAllocationFunction.prliExceptionProcessin`
The reviewer can choose to either approve or deny the provider roster line item correction. After this activity is completed, the workflow is also completed.

Event Details

This link provides the notation of the event details for the Roster Line Item Correction Approval workflow.

The notation of the following event details is as follows:

Table 71: Roster Line Item Correction Approval Event Details

Event Raised	Primary Event Data	Raised From
ROSTER.PRLIC_APPROVED	prliCorrectionID	curam.attendance.impl.PRLICorrectionImpl.approv
ROSTER.PRLIC_DENIED	prliCorrectionID	curam.attendance.impl.PRLICorrectionImpl.deny

1.4 CPM Products and Rule Sets

New financial processes have been built for CPM to enable payments to be made to providers. These new processes integrate with existing Merative ™ SPM Platform financial processes. CPM uses Cúram Products and Rule sets for generating the payments for a service provider.

Products

Merative ™ SPM Provider Management consists of four products. You can customize the default implementation.

The following list outlines the four Provider Management products:

- **Provider Invoice**
  You can use the product to generate the payments for the invoices that are furnished by the providers.
- **Provider Placement**
  You can use the product to generate the payments that are related to the placement services that are offered by the provider.
- **Provider Contract**
  You can use the product to generate the payments that are not dependent on the service use.
- **Provider Attendance**
  You can use the product to generate the payments that are based on the client attendance artifacts that are provided by the provider for a particular service.

Rather than being real benefit products with which a user can interact, the products are used as a way of getting to Cúram financials. All the case processing for the products happens in the background on Provider Management events. For example, invoice approval, placement of a client, making a contract live, or provider roster line item approval. The Provider Management products have the designated extension point interfaces as the customization points that are available to an agency.

The following list outlines the five DMX files that are used for the Provider Management products:

- *PRODUCT.dmx*

- *EVIDENCEMETADATA.dmx*
- *PRODUCTEVIDENCELINK.dmx*
- *PRODUCTRULESLINK.dmx*
- *TEMPORALEVIDENCEAPPROVALCHECK.dmx*

You cannot change the DMX files because generating financials depends on the product and evidence approval configurations.

What can I configure or customize?

In Social Program Management, all notifications are generated through the execution of workflow activities. The invocation of the allocation strategy that is associated with that activity dictates to which users that notification is sent.

When the Provider Invoice, Provider Placement, Provider Contract, or Provider Attendance product is created, unlike product delivery cases, notifications are not delivered to any users. The reason is that the products are created for generating financials and it is not intended that a caseworker interacts with the products.

A default implementation of the interface `NotificationExcludedCaseTypes` is used to exclude the four products when product delivery case notifications are generated. If you want to change the list of provider products to exclude, you must programmatically extend the default provider management implementation `NotificationExcludedProviderCaseTypesImpl`.

The next two sections describe how you can customize the default implementation.

Default implementation customization

Organizations can customize the default implementation when provider management is installed. The default implementation lists the following product deliveries types for exclusion when you generate notifications:

- PROVIDERPLACEMENT
- PROVIDERINVOICE
- PROVIDERATTENDANCE
- PROVIDERCONTRACT

Add a class that extends `NotificationExcludedProviderCaseTypesImpl`:

```
public class CUSTOM_NotificationExcludeCaseTypeImpl
  extends NotificationExcludedProviderCaseTypesImpl
```

The class implements the interface `NotificationExcludedCaseTypes`. The interface `NotificationExcludedCaseTypes` defines the following methods that are called when you generate notifications:

- **getCaseTypesToExclude()**
  The method is used to exclude case types.
- **getProductDeliveryTypesToExclude()**
  The method is used to exclude product delivery types.

To maintain the default functionality, custom code must include a call to the appropriate overridden function from the parent class as shown in the examples. If none of the default functionality is required, organizations must not call the parent class and organizations can define different behavior instead.

Adding to the exclusion list

When organizations are generating notifications, you can add any product delivery types or case types to the list of items to exclude. The next section indicates the steps that are required to exclude a product delivery or case type from the notification generation process.

Adding product delivery types to the exclusion list

To add a custom product delivery type to the exclusion list, you create an overriding version of `getProductDeliveryTypesToExclude()`.

The method might contain code like the example that follows. The example assumes that a product delivery type called EXAMPLE_PRODUCT_TYPE exists and the example uses the EXAMPLE_PRODUCT_TYPE to demonstrate how to add a product delivery type to the set of items that are returned by the default implementation:

```
@Override
public Set<PRODUCTTYPEEntry> getProductDeliveryTypesToExclude() {

  final Set<PRODUCTTYPEEntry> customPDExcludeList =
      super.getProductDeliveryTypesToExclude();

  // add product delivery types to the exclusion list here…
  customPDExcludeList.add(PRODUCTTYPEEntry.EXAMPLE_PRODUCT_TYPE);

  return customPDExcludeList;
```

Adding case types to the exclusion list

You can also customize the notification filter to exclude case types when you are generating notifications.

You must use an overriding version of `getCaseTypesToExclude()`. Otherwise, it is like the preceding example where the case type is added to a list of case types that must be excluded when you are generating notifications.

The example assumes that an `EXAMPLE_CASE_TYPE` exists and the example uses that type to demonstrate how you can stop generating notifications for it.

```
@Override
public Set<CASETYPECODEEntry> getCaseTypesToExclude() {
  final Set<CASETYPECODEEntry> customExcludeList =
      super. getCaseTypesToExclude();

  // add case types to the exclusion list here…
  customExcludeList.add(CASETYPECODEEntry.EXAMPLE_CASE_TYPE);

  return customExcludeList;
```

Creating a binding for the custom implementation

You must bind the custom implementation to the default provider management implementation in a new module class. The module class must extend AbstractModule and you must add a configuration for the module class to *MODULECLASSNAME.dmx*:

```
public class Module extends AbstractModule {
  @Override
  public void configure() {

    bind(NotificationExcludedProviderCaseTypesImpl.class)
      .to(CUSTOM_NotificationExcludeCaseTypeImpl.class);
  }
}
```

Relationship between the NotificationExcludedCaseTypes interface and workflow

8.0.2.0

The implementation of the interface `NotificationExcludedCaseTypes` determines whether certain case types are prevented from generating notifications during the enactment of the following two workflows: `CASEREASSESSMENTNOTIFICATION` and `DEFAULTCASENOTIFICATION`.

Each of these workflows has an allocation strategy, `curam.core.sl.intf.NotificationAllocationFunction.userAndCaseTypeStrategy`, which calls the `NotificationExcludedCaseTypes` implementation. If the implementation determines that no notification is to be delivered to any users for a given case type, the allocation strategy function returns an empty list. The empty list prevents the notification from being delivered to any users. Otherwise, the function returns a list `curam.util.workflow.struct.AllocationTargetList` that contains one struct `curam.core.sl.struct.AllocationTargetDetails` that represents the user to target the notification to. This allows the notification to be generated.

If you want to prevent notifications that are raised from other workflows, you must change the allocation function for that workflow to call the `NotificationExcludedCaseTypes` interface.

Note: At a minimum, the workflow requires the following two parameters:

- The username to deliver the notification to.
- The case ID of the associated case.

The following code sample illustrates how you can implement the allocation function:

```
/**
   * The allocation strategy referenced from the workflow definition.
   * This strategy uses the caseID to check the case type and/or product type
   * to determine if the NotificationExcludedCaseTypes specifies that it
   * should be excluded from notification.
   * If it is to be excluded then we return an empty allocation list, otherwise
   * we delegate to the default allocation strategy.
   */
  @Override
  public AllocationTargetList userAndCaseTypeStrategy(final String userName,
    final long caseID) throws AppException, InformationalException {

    final AllocationTargetList result;
    if (shouldExcludeCaseFromNotification(caseID)) {
      // Should exclude. Return an empty list.
      result = new AllocationTargetList();
    } else {
      // Should NOT exclude. Delegate to the default strategy.
      result = defaultStrategy(userName);
    }
    return result;
  }

  /**
   * Indicates whether the specified case should be excluded from notification
   * based on the NotificationExcludedCaseTypes implementation.
   *
   * @param caseID The case ID to consider.
   * @return true if it should be excluded.
   *
   * @throws AppException Standard signature.
   * @throws InformationalException Standard signature.
   */
  private boolean shouldExcludeCaseFromNotification(final long caseID)
  throws AppException, InformationalException {

    boolean result = false;
    // Filter according to case type.
    final CaseHeader caseHeader = CaseHeaderFactory.newInstance();
    final CaseKey caseKey = new CaseKey();
    caseKey.caseID = caseID;
    final CaseTypeCode caseTypeCode = caseHeader.readCaseTypeCode(caseKey);

    // if the case type is a product delivery, get the list of product
    // deliveries to exclude
    if (caseTypeCode.caseTypeCode.equals(CASETYPECODE.PRODUCTDELIVERY)) {

      // It is a Product Delivery, now get the Product Delivery type.

      final curam.core.intf.ProductDelivery productDeliveryObj =
        ProductDeliveryFactory.newInstance();
      final ProductDeliveryKey productDeliveryKey = new ProductDeliveryKey();
      productDeliveryKey.caseID = caseID;
      final ProductDeliveryTypeDetails productDeliveryType =
        productDeliveryObj.readProductType(productDeliveryKey);

      final Set<PRODUCTTYPEEntry> excludedProductDeliveries =
        notificationExcludedCaseTypes.getProductDeliveryTypesToExclude();

      final PRODUCTTYPEEntry thisProductType =
        PRODUCTTYPEEntry.get(productDeliveryType.productType);

      if (excludedProductDeliveries.contains(thisProductType)) {
        result = true;
      }

    } else {
      // Not a Product Delivery case, see if it is one of the case types to
      // be excluded
      final Set<CASETYPECODEEntry> excludedCaseTypes =
        notificationExcludedCaseTypes.getCaseTypesToExclude();

      if (excludedCaseTypes
        .contains(CASETYPECODEEntry.get(caseTypeCode.caseTypeCode))) {
        result = true;
      }
    }

    return result;
  }

  @Override
  public AllocationTargetList defaultStrategy(final String userName)
    throws AppException, InformationalException {
```

Rule Sets

The list of Rule Sets in CPM is described below. The rule sets can be customized, as long as the customized rule set does not depend on new types of evidence.

Table 72: Payment Type and Rule Set Details

Payment Type	Rule Set Source Location
Provider Invoice	EJBServer\components\CPM\rulesets \Product_51.xml
Provider Placement	EJBServer\components\CPM\rulesets \Product_52.xml
Flat-Rate Contract Payments	EJBServer\components\CPM\rulesets \Product_53.xml
Provider Attendance Payments	EJBServer\components\CPM\rulesets \Product_304.xml

1.5 CPM Financials

CPM financials are developed using the Classic Assessment/reassessment framework, evidence functionality and the Classic Rules Engine.

CPM financials include tasks as the following five tasks:

- Maintenance (creation, approval and activation) of the Product Delivery cases (SILI, Attendance, Placement, Contract) for different types of payments.
- Management of evidence using the Evidence functionality.
- Execution of Classic Rule Sets.
- Assessment, reassessment, or both of financials.
- Generation of payments, and so on.

CPM financials leverage Merative ™ SPM Platform financial processing for assessments and payments.

CPM financial processing is responsible for the following three tasks:

- Creating and maintaining Evidence for different types of cases;
- Creating and maintaining Financial Schedules and transactions associated with different case types;
- Processing financial transactions associated with a participant or a case, or both.

Payment Types

There are 4 types of Products configured for different payment types in CPM financials

- Service Invoice;
- Placement;
- Flat Rate Contract;
- Attendance

Service Invoice

Service Invoice processing relies on the creation of a Service Authorization when services are allocated to the clients. The individual line items within a Service Authorization can be for a number of different services allocated to that client, which can be provided by different providers.

After providing a service, the Provider submits an invoice to the SEM agency. The Provider gets paid once Service Invoice is approved. Service Invoice Line Item payment amounts are treated as evidence for the Service Invoice financial processing.

A product delivery case of type Provider Invoice is created the first time a service invoice line item for a provider is approved. Evidence is created on the case to correspond to the payment amount determined by CPM. The frequency of payment is set based on the established payment frequency for the provider, and leverages Merative ™ SPM Platform functionality around due dates for financial components.

If there is a change in payee, a separate PD case will be created for the payee.

All payments due for the provider for the period will be rolled up and paid as a single payment.

Placement

Placement is a type of service, in which a client is physically placed with the Provider for a period of time. Once a placement service is authorized, a client can be placed with the provider and financials will be started from day one. The unit of measure for the placement will be always a number of days. These placement details will be considered as evidence for processing the placement related financials.

A product delivery case of type Provider Placement is created the first time a placement is made with a provider. The system creates one PD case for each Placement. When a client is transferred within a Provider facility (i.e. form one place to another), this also creates a new product delivery case.

The delivery pattern on the product delivery case is set to a value specified in the property administration section of CPM administration.

For example, if a placement is made for a provider for the first time on June 15th, for the period from June 1st till June 30th, and the frequency is set to the first day of every month, the product delivery case is created on June 15th and the evidence data is set to June 1st till June 30th. The first payment due date is set to July 1st.

Flat rate contract

A Flat Rate Contract is a formal agreement between a Provider and the SEM agency which establishes terms under which services will be delivered. Each contract can cover single or multiple services. All the Contract details are treated as evidence for Flat Rate Contract financial processing.

A product delivery case of type Provider Contract is created the first time a flat rate contract is activated. The system creates one PD case for each Contract per Provider. It also creates a new PD case whenever an existing Contract is renewed.

The information specified in the contract is used to establish a payment schedule for the provider.

Attendance

Attendance rosters are used when services are delivered to the client which require that client attendance be tracked and reported through Attendance Tracking. Attendance is tracked either

through a roster submitted by the provider and entered on to the system by an internal user, or by the provider accessing the system externally.

Attendance Rosters can be generated automatically based on a configured frequency for a service. Rosters are submitted to the agency after capturing all attendance details. These attendance details are used as evidence for processing the financial details.

A product delivery case of type Provider Attendance is created the first time a roster is approved for a provider. Evidence is created on the case to correspond to the payment amount determined by CPM.

The frequency of payment is set based on the established payment frequency for roster based payments. If set, this frequency applies across all providers on the system. If this frequency is not set, the frequency of payment is set based on the established payment frequency for the provider, and leverages Merative ™ SPM Platform functionality around due dates for financial components.

All payments due for the provider for the period will be rolled up and paid as a single payment.

1.6 Service Deliveries

A service delivery is a type of service delivered to a client, which can be created and managed within an integrated case or an outcome plan. These services can be configured to use product delivery processing, Provider Management (CPM) processing, or a combination, depending on how the agency wishes the service to be delivered to the client.

Services which use product delivery processing can use standard product delivery functionality, for example, eligibility determination for a service and the calculation of payments based on custom rates (a rate which can change over time and can change based on circumstances). Services which use CPM processing can use CPM's financial processing and rate hierarchy. For example, invoices submitted by a provider are matched to a service authorization, and payments are generated based on the provider offering rate, using an out of the box Provider Invoice product delivery case (one per provider). Services which use a combination of both CPM processing and product delivery processing can utilize some or all of the standard features of a product delivery while fully integrating with CPM's service authorization and invoice processing.

If a service offering is configured to use product delivery processing for any aspect of service delivery, a corresponding product must be configured. This chapter outlines the actions and extension points available in CPM to utilize these product delivery features. For more information, see the *Configuring Integrated Case Management* and the *Provider Management Guide* related links.

Product Design and Configuration

Where service deliveries are configured to use product delivery processing to determine eligibility or payment amounts, the underlying product needs to be associated with a CER rule set and rate tables appropriate to the SEM agency's requirements. For detailed instructions on configuring products and rule sets see the *Cúram How To Build a Product Guide*.

Rule Set Creation

The rule set used to determine eligibility and calculate payment amounts in respect of the service must be configured to use a combination of client, case, service and invoice or attendance evidence values, depending on the requirements of the agency and the service delivery type. If product delivery processing is being used to determine both eligibility and the payment amount then the recommended approach is to use a separate objective to calculate each of these as follows:-

- The eligibility objective must be configured such that entitlement is determined by checking the value of the relevant attributes, for example, the client's date of birth or employment status. The valueType of the Objective Tag Type for this Objective must be a non money type such as Double to ensure an eligible decision does not result in the generation of financial components, as this is a non-financial objective.
- Entitlement to the payment objective should check for entitlement to the eligibility Objective as well as checking the value of attributes related to custom rates, invoice or attendance evidence. The valueType of the Objective Tag Type for this objective must be Money to ensure the generation of financial components, as this is a financial objective.

Evidence and Evidence Maintenance

The evidence entities used in the rule set calculations must be configured to use the appropriate propagator type. For example, InvoicePaymentEvidence must be configured to use the `ActiveEvidenceRowRuleObjectPropagator`, ServiceInvoiceLineItem should use the `RuleObjectPropagator`. For detailed instructions on how to configure propagation of different evidence types, see *`The Inside Cúram Eligibility and Entitlement Using Cúram Express Rules Guide`*.

Evidence types that are used to determine eligibility and calculating payments in respect of services must be configured and associated with the product underlying the service during administration. Shared evidence is maintained at the integrated case level can also be used in rule set calculations.

Changes in evidence values used by the rule set will trigger the assessment engine to run the calculations again resulting in updated decision and payment information.

For detailed information on designing evidence, see the *`Cúram Dynamic Evidence Configuration Guide`*.

Custom Rates

If custom rates are to be used to calculate payment amounts in respect of a service, then a rate table must be created and associated with the rule set. For more information on creating and associating rate tables with CER rule sets, see the *`Inside Cúram Eligibility and Entitlement Using Cúram Express Rules Guide`*.

The value attribute of the Case Decision Objective must be populated in the rule set using values read from your rate tables. Otherwise an appropriate value from CPM such as the amount from an invoice or roster can be used. An attribute to calculate the Estimated Cost can also be included in the rule set where custom rates are used instead of using the default CPM calculation for this value.

Service Delivery Creation

On creation of a service delivery of type 'Product Delivery', 'Product Delivery with Invoicing' or 'Service Delivery with Eligibility', a product delivery case will be created by the system. This case is an instance of the product type that was configured on the underlying Service Offering. This product delivery is not visible to the user. The caseID of this product delivery is set as the deliveryTypeRelatedID on the service delivery record, and will also be associated with any invoice or attendance payment evidence records associated with the case (i.e., it will be set as the caseID on the associated Evidence Descriptor record). Service deliveries of type Service Delivery will continue to use the caseID of the associated integrated case or outcome plan to populate these fields.

For service deliveries that use product delivery processing to determine eligibility, a hook has been provided to listen for events raised by the Assessment Engine. A default implementation for the `postInsertExamineDecisions` method has been added in `curam.cpm.sl.impl.CPMAssessmentEngineEventListener`, which listens for the creation of new decisions. Where the new decision relates to a service delivery of type 'Product Delivery with Invoicing' or 'Service Delivery with Eligibility' and the decision result is 'Eligible', then a service authorization and any service authorization line items are automatically created for the service delivery. This default behaviour can be altered or enhanced as per agencies own requirements.

A standalone public API 'curam.servicedelivery.facade.impl.CreateServiceDeliveryWizard.createServiceDelivery' is also available that creates a service delivery and returns the ID of it. This API forwards the call to the service delivery handler on the basis of the delivery mechanism configured. The delivery mechanism is retrieved based on the deliverytype(SODELIVERYTYPEEntry) configured for the service offering.

Display of Product Delivery Information

Any product delivery functionality that is related to eligibility and financial processing such as financial transactions, determinations, and evidence is automatically displayed at the service delivery level and can be viewed by a case worker in the context of that service delivery. Other product delivery functionality can also be configured for display if required. For example, certification and appeal details. However, some development effort is required to display this information. The display of this information must be configured through the use of client navigation files.

1.7 Compliancy for Provider Manager

CPM contains a sample component to help the test team to test CPM APIs and the development team to test CPM extension points. Use or customization of the CPM Sample component is not supported.

The CPM Sample component has 3 packages that must not be used or customized.

- curam.cpmsample.changecases.impl: This package is mainly used for testing extension points in CPM.
- curam.cpmsample.facade.impl: This package has façade classes and the associated client directory is *`components/CPMSample`*.

- curam.cpmsample.impl: This has a Module class. It is also used for testing extension points in CPM.

Miscellaneous Entities

CPM created new entities in CPM component to add a new feature which supports multiple clients for provider roster line item. As this feature is not supported by the application currently, these entities may change in the future. It is highly recommended that these entities are not used.

- PRLIClient
- PRLIClientHistory
- PRLICorrectionClient

1.8 Appendix A

The structure of the xml is based on the Castor v0.9.5.4 Mapping xml schema.

```
<?xml version="1.0" encoding="UTF-8"?>
            <mapping>
                <class auto-complete="false"
                name="curam.taxonomy.util.impl.Taxonomy">
                    <map-to xml="taxonomy" />
                    <field collection="arraylist"
 name="taxonomyTerms"
                    type="curam.taxonomy.util.impl.TaxonomyTerm">
                        <bind-xml name="record" />
                    </field>
                </class>
                <class auto-complete="false"
                name="curam.taxonomy.util.impl.TaxonomyTerm">
                    <map-to xml="record" />
                    <field name="name" type="java.lang.String">
                        <bind-xml name="name" node="element" />
                    </field>
                    <field name="code" type="java.lang.String">
                        <bind-xml name="code" node="attribute" />
                    </field>
                    <field name="definition"
 type="java.lang.String">
                        <bind-xml name="definition"
 node="element" />
                    </field>
                    <field name="facet" type="java.lang.String">
                        <bind-xml name="facet" node="element" />
                    </field>
                    <field name="comments"
 type="java.lang.String">
                        <bind-xml name="comments"
 node="element" />
                    </field>
                    <field name="bibliographicReference"
 type="java.lang.String">
                        <bind-xml name="bibliographicReference"
 node="element" />
```

```
                    </field>
                    <field name="createdDate"
type="java.lang.String">
                        <bind-xml name="createdDate"
node="element" />
                    </field>
                    <field name="lastModifiedDate"
type="java.lang.String">
                        <bind-xml name="lastModifiedDate"
node="element" />
                    </field>
                    <field collection="arraylist"
name="taxonomyTerms"
                    type="curam.taxonomy.util.impl.TaxonomyTerm">
                        <bind-xml name="record" />
                    </field>
                    <field collection="arraylist"
name="externalTerms"
                    type="curam.taxonomy.util.impl.ExternalTerm">
                        <bind-xml name="externalTerm" />
                    </field>
                    <field collection="arraylist"
name="relatedConcepts"

type="curam.taxonomy.util.impl.RelatedConcept">
                        <bind-xml name="relatedConcept" />
                    </field>
                    <field collection="arraylist"
name="useReferences"
                    type="java.lang.String">
                        <bind-xml name="useReference" />
                    </field>
                    <field collection="arraylist"
name="relatedTerms"
                    type="java.lang.String">
                        <bind-xml name="see Also" />
                    </field>
                    <field collection="arraylist"
name="oldCodes"
                    type="java.lang.String">
                        <bind-xml name="oldCode" />
                    </field>
                </class>
                <class auto-complete="false"
                name="curam.taxonomy.util.impl.RelatedConcept">
                    <map-to xml="relatedConcept" />
                    <field name="code" type="java.lang.String">
                        <bind-xml name="code" node="attribute" />
                    </field>
                    <field name="name" type="java.lang.String">
                        <bind-xml node="text" />
                    </field>
                </class>
                <class auto-complete="false"
                name="curam.taxonomy.util.impl.ExternalTerm">
                    <map-to xml="externalTerm" />
                    <field name="externalCode"
type="java.lang.String">
```

```
                        <bind-xml name="externalCode"
node="element" />
                    </field>
                    <field name="name" type="java.lang.String">
                        <bind-xml name="name" node="element" />
                    </field>
                    <field name="system" type="java.lang.String">
                        <bind-xml name="system" node="element" />
                    </field>
                </class>
            </mapping>
```

1.9 Appendix B: Schema definitions of the XML fragments created by Import process.

The *`UseReference.xsd`* file

```
<?xml version="1.0" encoding="UTF-8"?>

                <xs:schema xmlns:xs="http://www.w3.org/2001/
XMLSchema"
                elementFormDefault="qualified">
                    <xs:element name="useReferences">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="useReference" />
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="useReference">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="text" />
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="text">
                        <xs:complexType>
                            <xs:simple Content>
                                <xs:extension base="xs:string">
                                    <xs:attribute name="locale"
use="required"
                                        type="xs:string" />
                                </xs:extension>
                            </xs:simple Content>
                        </xs:complexType>
                    </xs:element>
                </xs:schema>
```

The *`RelatedConcept.xsd`* file

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
                    <xs:schema xmlns:xs="http://www.w3.org/2001/
XMLSchema"
                    elementFormDefault="qualified">
                        <xs:element name="relatedConcept">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element ref="name" />
                                </xs:sequence>
                                <xs:attribute name="code"
 use="required"
                                    type="xs:NCName" />
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="name">
                            <xs:complexType mixed="true">
                                <xs:attribute name="locale"
 use="required"
                                    type="xs:string" />
                            </xs:complexType>
                        </xs:element>
                    </xs:schema>
```

The *ExternalTerm.XSD* file

```
<?xml version="1.0" encoding="UTF-8"?>

                    <xs:schema xmlns:xs="http://www.w3.org/2001/
XMLSchema"
                    elementFormDefault="qualified">
                        <xs:element name="externalTerm">
                            <xs:complexType>
                                <xs:sequence>
                                    <xs:element ref="name" />
                                    <xs:element ref="system" />
                                </xs:sequence>
                                <xs:attribute name="code"
 use="required"
                                    type="xs:string" />
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="name">
                            <xs:complexType mixed="true">
                                <xs:attribute name="locale"
 use="required"
                                    type="xs:string" />
                            </xs:complexType>
                        </xs:element>
                        <xs:element name="system" type="xs:string" />
                    </xs:schema>
```

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.