IBM Cúram Social Program Management
Version 7.0.0

*Cúram Dynamic Evidence
Configuration Guide*

IBM

Contents

Figures

Tables

Configuring Dynamic Evidence

You define and maintain dynamic evidence types by using the Dynamic Evidence Editor in the Cúram administration application. You can follow the simple example to create a dynamic evidence type, and then read further to familiarize yourself with more advanced dynamic evidence features.

Creating a Simple Dynamic Evidence Type

Use this information to understand the process of defining a simple Dynamic Evidence Type, and key dynamic evidence concepts and components. It assumes that you have a run-time installation of the Cúram Platform.

Prior to Cúram 6.0, the definition and maintenance of Evidence Types was a development-time activity. In order to define or modify an Evidence Type, a number of artifacts had to be defined in the Cúram Development Environment, Java code had to be written, screens had to be designed and specified, and the entire application had to be built, deployed and tested before the new Evidence Type could be used in practice.

In Cúram 6.0, an alternative to this traditional development-time version of Evidence (hereafter termed 'Non-Dynamic Evidence') has been provided called Dynamic Evidence. Rather than involving Development-time activity, creation and maintenance of Dynamic Evidence Types is a purely administrative exercise. Using the Cúram Administration Suite and the Dynamic Evidence Editor, administrators can dynamically define equivalent artifacts to those specified by a Cúram developer of Non-Dynamic Evidence.

Creating a Dynamic Evidence Type

A Dynamic Evidence Type is the administrative equivalent to an Non-Dynamic Evidence Type - a logical grouping of related attributes about which an organization wants to record information in respect of a Case (e.g. Income Evidence, Medical Expense Evidence, Residency Evidence, etc.).

Note: In Non-Dynamic Evidence, Evidence Types are ultimately represented as individual database tables; in Dynamic Evidence, all Dynamic Evidence Types are represented behind the scenes using a standard set of generic tables.

For the purposes of example, let's say that we have an Evidence Type called 'Sample Income' that we want to define, and it consists of two attributes: Income Amount (a money value), and Income Type (a dropdown containing different types of income)

To begin creating this as a Dynamic Evidence Type, perform the following actions:

- Log in as an Administrator (e.g. Admin).
- Open the **Administration Workspace**
- In the Shortcuts panel, select **Rules and Evidence**
- Click on **Dynamic Evidence** - this will show a list of all Dynamic Evidence Types currently defined in the system
- Click on **New...**, enter the following information, and then hit '**Save**':
  Evidence Type: Sample Income

Logical Name: SampleIncome

Effective From: 1/1/2011

Security Group: SAMPLE_INCOME_GROUP

This will create the overall structure of our 'Sample Income' Dynamic Evidence Type. Next, this Dynamic Evidence Type will be fleshed out in a bit more detail.

Editing the Dynamic Evidence Type Version

Dynamic Evidence Types, unlike Non-Dynamic Evidence Types, can have multiple versions which vary over time. With Non-Dynamic Evidence, a single Evidence Type has a fixed set of attributes and relationships for the duration of its existence, and these can only be changed in tandem with a re-development and data migration exercise.

Dynamic Evidence, however, allows for information recorded in respect of an Evidence Type to evolve in response to the evolution of legislative and administrative Evidence requirements. Each Version of a Dynamic Evidence Type is effective from a particular date, and remains effective until the next version.

As part of creating a Dynamic Evidence Type, the system also creates a default Dynamic Evidence Type Version (as in order to be useful, a Dynamic Evidence Type must have at least one Evidence Type Version associated with it), and it is at the Evidence Type Version level that we define attributes and other details for the Evidence Type.

To define such details, we must launch the Dynamic Evidence Editor. To do this:

* Expand the flipper on the newly created Dynamic Evidence Type
* In the Actions button in the list item, select '**Edit Metadata**'

In the next sections we will define two key aspects of a Dynamic Evidence Type - the Model and the User Interface.

Defining the Model

The first key dimension to a Dynamic Evidence Type Version is its Model, which describes its structure and behavioral characteristics. The Model defines its attributes and their datatypes, its relationships to other Dynamic Evidence Types, its validations, how a Case Evidence record of this Type is to be summarized in descriptions, and various other functional aspects.

After launching the Editor in the previous step, the **Model** tab will be opened by default. Here we will create the two attributes that were specified in the requirements above: Income Amount and Income Type.

* Add a Data Attribute by clicking on the icon. A Data Attribute is the Dynamic equivalent of a database column on a Non-Dynamic Evidence Entity: a single piece of information we want to record for an Evidence Type
* In the **Data Attribute Properties** panel, enter the following information:
  - **Attribute Name:** incomeAmount
  - **Attribute Type:** Money
  - **Mandatory:** <checked>

* Add another Data Attribute by clicking on the icon.
* In the **Data Attribute Properties** panel, enter the following information:

– **Attribute Name:** incomeType
– **Attribute Type:** Codetable
– Expand the **Codetable Options** flipper, and enter the following information:
  Codetable Name: IncomeTypeCode

Finally, save your work by clicking on the ⊞ icon. There are many other options available in the Model section of the Dynamic Evidence Editor, and these are described in detail in "Defining a model for a client data type version" on page 23. Next, we turn our attention to defining the User Interface for this Dynamic Evidence Type Version.

Defining the User Interface

Ultimately, case workers interact with Evidence Types when they try to enter information in respect of those types for Cases. Dynamic Evidence is supported for use in Integrated Cases and both Benefit and Liability Product Deliveries.

With Non-Dynamic Evidence, Cúram Developers provide the screen definitions for each Evidence Type as input to the build process; with Dynamic Evidence, administrators specify this screen definition information using the Dynamic Evidence Editor. Ultimately, this screen definition information will be used to generate the screens for Creation, Modification and Viewing of Case Evidence at runtime (via the Case Evidence Dashboard).

For our `Sample Income` Dynamic Evidence Type Version, we shall define a simple user interface to be used on all maintenance screens - create, modify and view. We shall place both of our newly added Data Attributes adjacent to each other on a Cluster, and the infrastructure will deal with everything else that needs to happen to display these screens at runtime.

To achieve this, perform the following steps:
- Click on the **User Interface** tab of the Dynamic Evidence Editor

- Add a Data Attribute Cluster by clicking on the following icon: ▦

- In the **Attribute Cluster Properties** panel, enter the following information:
  Title: Income Details
  All other properties can use the default settings.

- Click on the first of the Data Attribute Fields in the list, called `Income Amount` and, keeping the mouse button down, drag it until it is over the central blue area of the newly added `Income Details` cluster; then release the mouse button (the blue area will change color when you are over a valid drop zone). This will get dropped as a new Field onto the Cluster.

  Note: After dropping the Field onto the Cluster, the Data Attribute which was dragged will disappear from the Data Attribute Fields Panel (as it is not valid to drop the same field onto a screen twice)

- Click on the newly dropped Field (it will be called **incomeAmount Label 1** or something similar), and in the **Field Properties** panel, enter the following:
  Label: Income Amount
  All other properties on this panel can use the default settings.

- Similarly, click on the only remaining Data Attribute Field in the list, called `IncomeType`and, keeping the mouse button down, drag it until it is over the central blue area of the newly added `Income Details` cluster; then release the mouse button. This will also get dropped as a new Field onto the Cluster.

> **Note:** The dragged field will be represented as dropdown list, as the attribute that has been dropped is of type `CodeTable` - similarly, all other Attribute types will be represented with the appropriate User Interface controls

- Click on the newly dropped Field (it will be called **incomeAmount Label 1** or something similar), and in the **Field Properties** panel, enter the following:

  Label: Income Type

  All other properties on this panel can use the default settings.

As when working with the Modelling section, save your work by clicking on the following icon: 💾 , and as we are finished our work in the Dynamic Evidence Editor, you can close the Editor tab. Again, many other options are available in the User Interface section of the Dynamic Evidence Editor, and these are described in detail in "Eligibility and Entitlement Rule Sets" on page 85.

Activating the Dynamic Evidence Type Version

Up to this point, we have been dealing with a Dynamic Evidence Type Version which is in a status of `In Edit`. This means that it has not yet been approved for use in a Product or Case, and indicates that an administrator is still working on its definition. Note that this is another difference from Non-Dynamic Evidence Types; as the latter are developed rather than configured, they are assumed to be available for use in a Case once they are deployed on a system i.e. they have no `In Edit` status.

For our Sample Income Dynamic Evidence Type to be available for use in a Case, it must have at least one `Active` Dynamic Evidence Type Version.

To do this, do the following:

- Make sure that the **Dynamic Evidence** tab is in the foreground (it should be after closing the Dynamic Evidence Editor tab in the previous step, but if not then follow the instructions under Creating a Dynamic Evidence Type section)
- From the Action menu at the end of the Evidence Type Version row, select **Activate...**.
- On the **Activate Dynamic Evidence Type Version** dialog box that the appears, select **Yes**.

You will notice that the **Status** column of the Dynamic Evidence Type Version is now `Pending Activation`. After some time, if you refresh this page, it will change to `Active`; the Dynamic Evidence Type is now available for use in a Case. The next logical step, then, is to link this Dynamic Evidence Type with a Case Type.

Associating the Dynamic Evidence Type with a Case Type

It is possible that you already have at least one Case Type registered on your system with which you could associate your new Dynamic Evidence Type, but this example assumes that you want to create a new Case Type from scratch.

Note: It is Dynamic Evidence Types, not the finer-grained Dynamic Evidence Type Versions, which are associated with Case Types. If a Dynamic Evidence Type has more than one Version, then all of the Versions are effectively associated with the Case Type; in this respect, the evolution of a Dynamic Evidence Type is linked to the evolution of any Case Types with which it is associated.

To create a new Case Type, in this example an Integrated Case Type, perform the following:

- In the Administration Workspace Shortcuts panel, click on **Case**, followed by **Integrated Cases**. This will open the **Integrated Cases** list page, a listing of all currently defined Integrated Case Types on the system.
- Click on the **New..** button.
- On the **New Integrated Case Type** dialog which appears, enter the following information and then click **Save**:

  Integrated Case Type: Sample Integrated Case

  All other fields in this dialog can use the default values.
- Click on the **Sample Integrated Case** link in the **Type**column of the Integrated Cases list page

This will open up the Sample Integrated Case homepage. Finally, to associate our Sample Income Dynamic Evidence Type with our new Integrated Case Type, perform the following:

- Click on the **Evidence Types** content tab
- Click on the **Add Evidence..** button.
- On the **Add Existing Evidence Type** dialog which appears, enter the following information and then click **Save**:

  Evidence Type: Sample Income

  Category: Income

  All other fields in this dialog can use the default values.

At this point, we have done all of the necessary administration, and all that remains is to test our new Dynamic Evidence Type by creating Case Evidence in respect of it as a case worker. This is accomplished in the final step in this simple example.

Creating Case Evidence for this Dynamic Evidence Type

In order to create Case Evidence in respect of our **Sample Income** Dynamic Evidence Type, we must first create an Integrated Case in respect of our new **Sample Integrated Case** Integrated Case Type. To do this, perform the following steps:

- Log into the caseworker application as a case worker (e.g. Using the Caseworker username).
- On the Home tab, click on the **Search for a Person...** link, and in the resultant **Person Search** Reference field, enter 24684 and click on **Search**.
- Click on the resultant hyperlink for **James Smith - 24684** the Person Tab for James Smith will open.
- On the **Actions** menu for the Person Tab, select the **New Case...** menu item.
- On the **New Case** dialog, select **Sample Integrated Case** for the Type field and click on **Save**. The Integrated Case Tab for our newly created Case will now open.

Now that we have a Case against which we can record Case Evidence, we can now enter some Sample Income Evidence.

Note: From a case worker perspective, there should be no difference between Dynamic Evidence Types and Non-Dynamic Evidence Types - they are all just Evidence that a case worker can enter, and a case worker should neither have to know nor care how a particular Evidence Type was defined.

- On the Integrated Case Tab, click on the **Evidence** content tab.

- By default, the Evidence Dashboard shows all Evidence Types in respect of which Case Evidence has been recorded. To show all Evidence Types, click on the **All** button in the **Income** bar, and our recently defined `Sample Income` Dynamic Evidence Type will appear.
- Click on the **Sample Income** link to open the Sample Evidence Workspace Tab, which by default will show the Sample Evidence List page.
- On the **Actions** menu for the Person Tab, select the **New** menu item.
- The screen that we defined using the Dynamic Evidence Editor for our Sample Income Dynamic Evidence Type will now appear.

    Note: The Dynamic Evidence infrastructure behind the scenes has added in a **Received Date** field (which is common to all Evidence Types), in addition to populating the appropriate Codetables with Codetable Descriptions and showing mandatory fields with an asterisk.

- Enter any valid Money amount in the **Income Amount** field, and any selection in the **Income Type** dropdown, and click **Save**.
- The information that you have entered has been written to the database as Case Evidence.

You have now successfully created and tested a new Dynamic Evidence Type.

Summary

In this chapter we have:
- Created a new Dynamic Evidence Type and Dynamic Evidence Type Version
- Using the Dynamic Evidence Editor, defined a Model, consisting of two Data attributes for this Dynamic Evidence Type Version
- Also using the Dynamic Evidence Editor, defined a User Interface, consisting of a Cluster and two Data Attribute Fields for this Dynamic Evidence Type Version
- Saved and Activated this Dynamic Evidence Type Version
- Created a new Integrated Case Type
- Associated our new Dynamic Evidence Type with our new Integrated Case Type
- Tested our new Dynamic Evidence Type by creating a Case in respect of the new Integrated Case Type, and creating Evidence in respect of the new Dynamic Evidence Type

In subsequent chapters we explore each of the preceding administration steps in more detail, starting with Chapter 3: Administration, which covers the creation and maintenance of Dynamic Evidence Types and Dynamic Evidence Type Versions in the Administration Suite.

Administering dynamic evidence

Use this information to understand the administration functions that you can use to create and maintain dynamic evidence in the Cúram Administration application. You can create dynamic evidence types and dynamic evidence type versions, and configure their lifecycles, and their association with case types.

Administration screens for dynamic evidence are in the **Rules and Evidence** section of the Curam Administration Application. Click the **Dynamic Evidence** link in this section to open a tab that shows a paginated list of all (non-Cancelled)

dynamic evidence types on the system. Each of these can be further expanded to show a list of dynamic evidence type versions in respect of the selected dynamic evidence type.

Why Dynamic Evidence Types and Versions

Evidence is typically gathered in support of program rules, and such rules tend to evolve over time as policies are adjusted and refined. As a result, in many cases the Evidence requirements for such Programs can also evolve, as the requirement to collect more or less information about a Case or Person changes.

As mentioned earlier, for Non-Dynamic Evidence, such changes need to be made using the Curam Development Environment, tested and eventually deployed into production. If Evidence data structures have changed in the newer version, then some amount of Data Migration may be required.

Dynamic Evidence, however, supports this requirement for Evidence definitions to evolve over time without the need for redeployment or data migration.

Note: The Dynamic Evidence infrastructure does however impose restrictions on how metadata can change over time (see "Evolution of Dynamic Evidence Type Metadata" on page 14).

To achieve this behavior, Dynamic Evidence is designed to have two main elements: Dynamic Evidence Types, and Dynamic Evidence Type Versions.

Dynamic Evidence Types

The Dynamic Evidence Type is the Dynamic equivalent of an Evidence Type in pre-6.0 versions of Cúram i.e. a grouping of related data elements to be recorded against a Case. Dynamic Evidence Types can be linked to Case Types to indicate that information of this type is relevant and maintainable for a particular program. In essence, they act as the container or 'header record' for all other elements and definitions necessary for Dynamic Evidence to function properly at runtime.

A Dynamic Evidence Type must have one or more Dynamic Evidence Type Versions.

Dynamic Evidence Type Versions

The Dynamic Evidence Type Version describes the structure and behaviour of a Dynamic Evidence Type at a particular point in time. Dynamic Evidence Type Versions have effective dates to support the evolution of Evidence Types over time. A Dynamic Evidence Type Version applies from its effective date, and will be used to maintain Case Evidence data from this date onwards. However, unlike Non-Dynamic Evidence, it is possible for multiple Versions of Dynamic Evidence to exist in the system, as long as all of these Versions have different effective dates; there can only ever be a single Dynamic Evidence Type Version active for a particular effective date.

Note: If a case worker attempts to enter Case Evidence in respect of a Dynamic Evidence Type for a date in the past, the system will use the Dynamic Evidence Type Version effective at that date. This may be the latest Version, or may be one that was defined and in use some time ago. In this way, the system will only ask for information required at each point in the history of the Evidence Type, and thus users never have to wonder whether or not a field should be filled in - if it isn't relevant, then it isn't displayed.

The next two sections explore the details of Dynamic Evidence Types and Dynamic Evidence Type Versions in greater depth.

Dynamic Evidence Types

Dynamic Evidence Types are the fundamental elements of Dynamic Evidence. They are stored in the EvidenceTypeDef entity and link together all components involved in the administration of Dynamic Evidence. They interact with the Cúram Evidence infrastructure in the same way as Non-Dynamic Evidence Types do, and are largely functionally equivalent (any differences in behavior are described at the appropriate points in this document).

Dynamic Evidence Types are accessed and managed via the Dynamic Evidence List Page, which is located in the **Rules and Evidence** section of the Cúram Administration Application.

Create new dynamic evidence type

To create a new Dynamic Evidence Type, the user must click the **New...** button on the Dynamic Evidence List Page. This will open the New Dynamic Evidence Type page in a modal window.

When creating a new Dynamic Evidence Type, to save the administrator some time the system will automatically create both a Dynamic Evidence Type and a Dynamic Evidence Type Version in the background - this is because valid Dynamic Evidence Types need to have at least one Dynamic Evidence Type Version.

This page has the following properties:

Evidence Type
> This is the display name of the Dynamic Evidence Type. Unlike other properties, the name is not stored in the EvidenceTypeDef entity. Instead, it is stored in the EvidenceType Codetable and can be localized using the standard localization process for Codetables (note that the Evidence Type code for this Codetable is generated by the Dynamic Evidence infrastructure).

> The Evidence Type name is mandatory and must be unique across all Dynamic and Non-Dynamic Evidence Types. The system enforces a check for uniqueness verifying that the supplied name is not in use by other Evidence Types, and this check is made for all supported locales.

Logical Name
> The Logical Name is a unique identifier for the Dynamic Evidence Type on the system. It is used to reference the Dynamic Evidence Type from a number of other configuration components e.g. Rule classes, localizable resources, etc. The Logical Name follows a specific naming format: it must start with a lowercase or uppercase English alphabetic character, and can contain lowercase and uppercase English alphabetic characters, numeric characters and underscores.

> The Logical Name is a non-modifiable property which once entered cannot be changed on the Modify page.

Effective From Date
> This is the effective date used for the initial Dynamic Evidence Type Version created from this page.

Security Group
> The Security Group property is used to create a new Cúram Security Group for the Dynamic Evidence Type. Dynamic Evidence Type Security

Identifiers for Case Evidence maintenance operations (generated by the Dynamic Evidence infrastructure) will be added to this group. The Security Group name is additionally used to construct the names of generated Security Identifiers (see "Security Identifiers and Security Groups" on page 103).

Note that there is a single group for the Dynamic Evidence Type, rather than there being separate groups for each Dynamic Evidence Type Version. In practice, this means that if a case worker has permission to access one Dynamic Evidence Type Version, then they have permission to access all Dynamic Evidence Type Versions for a Dynamic Evidence Type.

The Security Group name is a non-modifiable property and must be unique across Dynamic Evidence Types.

Definition

The Definition field represents a short piece of descriptive text explaining the Dynamic Evidence Type. The Definition field is non-localizable, optional, and is currently used for annotation purposes only - this Definition is not referred to from the Case Evidence screens, for example.

Description

The Description field is a localizable text property displayed on the New Evidence Type search page in the case worker workspace (i.e. the page that allows users to select an Evidence Type to create a new Case Evidence record). If no value is provided for this property, the system will default to displaying the Evidence Type Name in the description column of this page.

The Description field is localized using Evidence Type properties resources (see "Evidence Type Properties Resources" on page 87). Each Evidence Type has an associated localizable properties resource. The naming convention for these properties resources is "DynEvd_EvidenceType_" followed by the logical name of the Dynamic Evidence Type and ended with ".properties" (e.g. `DynEvd_EvidenceType_<logicalName>.properties`). The Description text entered on the Create Dynamic Evidence Type page is inserted in `DynEvd_EvidenceType_<logicalName>.properties` for the default locale. The property key name is made up of the Evidence Type logical name followed by ".description" (e.g. <logicalName>.description).

The creation of a new Dynamic Evidence Type generates several additional artifacts:

- A new Evidence Type code is generated and inserted into the EvidenceType Codetable using the Evidence Type name entered on the Create page as a Codetable item description. The new Codetable entry is recorded against the server locale. See "EvidenceType Codetable Entries" on page 103 for more info.

  warning: Generated Dynamic Evidence Type Codetable codes should never be manually modified or removed via the System Administration application! Proper functioning of the C⌑ram system cannot be guaranteed if generated Dynamic Evidence Type codes are manually modified.

  warning: Auto-generated Codetable entries are generated in respect of the server locale, but displayed for the locale of the currently logged in user. It is highly recommended that Administrator users configuring Dynamic Evidence operate in the same locale as the server. In multi locale deployments this will prevent localization issues that may occur before auto-generated Codetable items are translated for all supported locales.

- As mentioned earlier, an initial Dynamic Evidence Type Version for the new Dynamic Evidence Type is generated. This Version has blank metadata and its effective from date is set to the Effective From attribute entered on the Create Dynamic Evidence Type page.

- Security Group and Security Identifiers for security administration are also generated (see "Security Identifiers and Security Groups" on page 103). A new Security Group is created using the name specified on the Create Dynamic Evidence Type page. In addition, operation Security Identifiers are created and added to the new Security Group.

  If a Security Group with the name "EVIDENCEGROUP" is present in the system, the new Security Identifiers will be added to this Group too. This Group can be created for convenience by administrators, and allows administrators to set the security rights for all Dynamic Evidence Types at once by simply adding this Group to the appropriate Security Roles.

  warning: The Dynamic Evidence Type Security Group is a non-modifiable property and cannot be changed via the Dynamic Evidence Type Modify page. However, it is physically possible to change the name of a generated security group in the System Administration application. This is not supported and should never be done. Proper functioning of the Cúram system cannot be guaranteed if names of generated Security Groups are manually modified!

Modify Dynamic Evidence Type

Dynamic Evidence Type details can be modified via the **Edit** action available in the Dynamic Evidence Type list item action menu. Selecting the **Edit** action will open the Modify page in a modal dialog. The following Dynamic Evidence Type properties can be modified:

- Evidence Type Name
- Definition
- Description

The preceding rules apply to these properties for the Create process also apply to the Modify process.

View Dynamic Evidence Type Details

The details of a Dynamic Evidence Type can be viewed by expanding the Evidence Type list item and selecting the ' **Details** ' tab. This tab displays a read only view of the Evidence Type details.

Delete Evidence Type

Deleting a Dynamic Evidence Type is performed using the ' **Delete** ' action available in the Dynamic Evidence Type list item action menu. Deleting a Dynamic Evidence Type removes all generated artifacts from the system, including Codetable entries for the Evidence Type code, security information, localized text etc.

If the Dynamic Evidence Type is linked to Products or Integrated Cases, these links are also removed. A Dynamic Evidence Type can only be deleted if it has no active or in-edit Dynamic Evidence Type Versions.

New Dynamic Evidence Type Version

A new Dynamic Evidence Type Version can be created using the ' **New Version** ' action from the Dynamic Evidence Type list item action menu. This action is only enabled if the Dynamic Evidence Type has no Active or In Edit Dynamic Evidence Type Versions.

Selecting this action opens a new modal window in which the user is required to enter the Effective From date for the new Dynamic Evidence Type Version. The new Version is created with blank metadata.

Dynamic Evidence Type Versions

As mentioned earlier, the Dynamic Evidence Type Version contains much of the structural and behavioral details for a Dynamic Evidence Type from a particular point in time. Dynamic Evidence Type Versions store such information in the form of metadata, and in doing so provide the details necessary to generate Case Evidence pages for collecting and recording Evidence data. Each Case Evidence data record in respect of a Dynamic Evidence Type is linked to the Dynamic Evidence Type Version used to create it.

Versions are recorded in the EvidenceTypeVersionDef entity where metadata is stored as a blob in XML format. Editing metadata is only supported via the Dynamic Evidence Editor, provided with the Dynamic Evidence administration component. Modifying this XML metadata directly is not supported and may result in incorrect system behavior; in addition there is no guarantee that the structure of this XML metadata will not change between releases of Cúram.

Dynamic Evidence Type Versions are linked to Dynamic Evidence Types in a many to one relationship; each Version belongs to one Dynamic Evidence Type, and a Dynamic Evidence Type can have many Versions, each with different Effective From dates.

Versions can be accessed by expanding individual Dynamic Evidence Type list items on the Dynamic Evidence Type List page. This shows a list of Versions for the selected Dynamic Evidence Types sorted by Effective From date in descending order. If there is Version of a Dynamic Evidence Type in an In Edit status, it is placed at the start of the list (see "Dynamic Evidence Lifecycle" on page 13 for more information on the status of Dynamic Evidence Type Versions). A number of actions can be performed on Dynamic Evidence Type Versions:

View Metadata
This action launches the Dynamic Evidence Editor in a new tab to view the metadata for the selected Version. Users may explore the metadata but cannot save any changes, i.e. The **Save** button and various other Palette buttons will be disabled.

Edit Metadata
Launches the Dynamic Evidence Editor in a new tab to edit the metadata. In this mode, users can save changes they make. This action is available only for Dynamic Evidence Type Versions with a status of In Edit.

Note: While using the Editor, users can access the same Dynamic Evidence Type Version record from two places: the Dynamic Evidence Editor and the Dynamic Evidence Type List Page. If the status of the edited Dynamic Evidence Type Version is changed via the administration page (for example, by activating it), subsequent attempts to save metadata from the Editor will result in an error.

Edit Effective From
Allows the Effective From date of a Dynamic Evidence Type Version to be modified. This date is a mandatory property and cannot remain blank. The action is available only for Dynamic Evidence Type Versions with a status of In Edit.

New InEdit Copy

This action copies the selected Dynamic Evidence Type Version and creates a new version of it with a status of In Edit. This action is only available for the latest Dynamic Evidence Type Version with a status of Active in the list, and only if that Dynamic Evidence Type has no Versions already with a status of In Edit i.e. Only the latest Active Dynamic Evidence Type Version can be copied and extended (although it is of course possible to change the structure of the new Dynamic Evidence Type Version metadata using the Dynamic Evidence Editor, subject to the normal restrictions on the evolution of Dynamic Evidence Type Versions).

Activate

As mentioned earlier, only Dynamic Evidence Types with at least one Active Version can be used as Case Evidence in a Program. The **Activate** action marks a Dynamic Evidence Type Version as Active, and this action is available for all versions with a status of In Edit. When activated, a Dynamic Evidence Type Version can be linked to a Product or Integrated case, and thus can be used in the case worker workspace for recording Case Evidence data.

Behind the scenes, Dynamic Evidence Type Version activation is complex, and goes through two stages.

In the first, the selected Dynamic Evidence Type Version changes its state to PendingActivation to allow Rulesets to be generated and published. Once this is completed, the Dynamic Evidence Type Version status changes automatically to Active.

Note: Because of its intrinsic complexity, the Activation process for Dynamic Evidence Type Versions uses Cúram Deferred Processing, which is asynchronous in nature. As a result, the user may have to use the Refresh Button on the Dynamic Evidence Type List Page to see the relevant status turn from PendingActivation to Active.

A number of additional artifacts are generated on activation of a Dynamic Evidence Type Version, such as user interface Tab configurations, CER Rulesets, etc. (see "Generated Artefacts" on page 102 for more information).

Activating a Dynamic Evidence Type Version is the last step before the Version becomes live and Case Evidence data can be recorded in respect of it. To ensure the validity of the Version and metadata, a set of validations is automatically performed upon activation. Validation problems are displayed in the confirmation dialog presented to the administrator on activation. All validation problems must be fixed before the Dynamic Evidence Type Version can be activated.

The following validations are performed:

- A new Dynamic Evidence Type Version must have an Effective Date later than the Effective Date of the latest Active Version in the same Dynamic Evidence Type, i.e. The Effective Date cannot overlap with previous Active Versions.

- A new Dynamic Evidence Type Version must have an Effective Date that is later than the latest recorded Case Evidence record in respect of previous Active Versions in the same Dynamic Evidence Type, i.e. The Effective Date cannot overlap with previously recorded Case Evidence data.

- The XML metadata is structurally validated against a predefined schema.

- Additional validations on the XML metadata that cannot be expressed in an XML schema (including cross version validations to enforce constraints on metadata evolution over time) are also performed.

Delete This action deletes the selected Dynamic Evidence Type Version and all related artifacts generated upon activation, e.g. Dynamic UIM pages and localizable resources, CER Rulesets, Tab configurations etc. A Dynamic Evidence Type Version cannot be deleted if it has associated `In Edit` or `Active` Case Evidence records. Such Case Evidence records must first be deleted in order to delete the Dynamic Evidence Type Version.

Note: This is of particular interest when testing new Dynamic Evidence Type Versions in a test or staging environment before putting them live in a production environment; when testing new versions (which will generally be revisions to previous versions), all test Case Evidence records in respect to the previous Versions must be deleted before their corresponding Dynamic Evidence Type Versions can be deleted.

Dynamic Evidence Lifecycle

Dynamic Evidence, unlike Non-Dynamic Evidence, has an associated lifecycle. Whereas Non-Dynamic Evidence Types are always in effect Active, Dynamic Evidence can be in an number of other states. This section discusses the Dynamic Evidence Lifecycle from two perspectives; the Dynamic Evidence Type lifecycle, and the Dynamic Evidence Type Versions lifecycle.

The Dynamic Evidence Type lifecycle is simple and has only two possible statuses internally: `Active` and `Canceled`. The management of these is handled by the system, and is effectively hidden from the user. When a Dynamic Evidence Type is created it is automatically given a status of `Active`, and only Active Dynamic Evidence Types are available for use as Case Evidence.

Deleting a Dynamic Evidence Type changes its state to `Canceled`, i.e. The Dynamic Evidence Type is not physically removed from the EvidenceTypeDef entity. However, all related generated artifacts *are* physically deleted from the system. Canceled Dynamic Evidence Types cannot be reactivated, and are filtered out by the administration and case worker workspaces.

Dynamic Evidence Type Versions have a different lifecycle which includes more states, and at all times the state is visible to users and clearly identifies what actions can be performed on each Dynamic Evidence Type Version:

InEdit The initial status assigned to a Dynamic Evidence Type Version when it is created is that of `In Edit`. This status indicates that the Version has not yet been completely specified, and that it is still available for modification.

No Case Evidence records in respect of `In Edit` Versions can be created, and indeed Versions with this status have no impact on the runtime case worker application. A Dynamic Evidence Type is restricted to having zero or one `In Edit` Version at any point in time.

Finally, an `In Edit` Version can only be deleted or activated.

PendingActivation
Activating a Dynamic Evidence Type Version changes its state to `Pending Activation`. This is a temporary state to allow time for generation and publishing of Rulesets. At the end of this process the Dynamic Evidence Type Version status is automatically changed to `Active`.

After activating a Dynamic Evidence Type Version, users may or may not see the `Pending Activation` status, i.e. It is possible to immediately see the Dynamic Evidence Type Version status go to `Active`. This depends on the time taken to generate and publish CER Rulesets, and also the time to refresh the administration page after activation. If a status of `Pending Activation` is displayed, the administration page will not refresh automatically when the state is internally changed to `Active`. In this case, users will need to manually refresh the administration page to reflect the change of status; this can be done using the **Refresh** button on the Dynamic Evidence Type List Page.

Note: The Activation process for Dynamic Evidence Type Versions uses Cúram Deferred Processing, which is asynchronous in nature. In case of failure, this deferred process changes the status of the Dynamic Evidence Type Version from `Pending Activation` back to `In Edit`. In this case, the administrator will be shown a red exclamation mark on the Dynamic Evidence list page item which will link to a dialog box showing the reasons for the failure during Activation. Administrators should ensure that prior to activation of a new Dynamic Evidence Type Version, CER rules are validated and published to avoid activation errors.

Active The `Active` state indicates that a Dynamic Evidence Type version is now live, and that it can be used to maintain Case Evidence in the case worker workspace. As mentioned earlier, for a Dynamic Evidence Type to be available to be linked to a Case Type, it must itself have a status of `Active` and also have at least one Dynamic Evidence Type Version which has a status of `Active`.

Once in this status, Dynamic Evidence Type Versions can only transition out of this state by being deleted.

Canceled

When a Dynamic Evidence Type Version is deleted in the Dynamic Evidence Type List Page, its status is changed to `Canceled`. This status marks the end of life of the Version. Like Dynamic Evidence Types, Versions are not physically deleted from the EvidenceTypeVersionDef entity. Canceled Versions can no longer be used in either in the administration or case worker workspace, nor can they be recovered.

Evolution of Dynamic Evidence Type Metadata

As discussed previously, Dynamic Evidence Types can evolve over time. For example, a change in legislation may require that a new Evidence attribute must now be recorded, starting from a specified date. Dynamic Evidence supports this requirement by using Dynamic Evidence Type Versions to record modifications to metadata over time.

Metadata changes are made by copying the latest Active version (using the Dynamic Evidence Type Version **New InEdit Copy** action) to create a new Dynamic Evidence Type Version with a status of `In Edit`. The effective date of the new Version can be set by the administrator to the date when the new change is required to take effect as specified by the legislation. Users may edit the new `In Edit` Version metadata to make the appropriate chance to the Evidence structure.

Note: It is only possible to create a copy of the latest Active Version. Previous Versions cannot be copied. This mechanism is designed to support the natural evolution of metadata over time, and to implement additional restrictions on modifications to metadata elements between Versions

As metadata evolves, some metadata elements cannot be freely modified between Versions, and certain limitations have been put in place by the Dynamic Evidence infrastructure. These limitations are not enforced by the Dynamic Evidence Editor, but a set of validations is performed upon activation of subsequent Dynamic Evidence Type Versions. The first Dynamic Evidence Type Version for a Dynamic Evidence Type, while in an `In Edit` state, allows for the modification of all metadata elements.

Once activated, however, the following restrictions are applied to subsequent Dynamic Evidence Type Versions:

Parents
> Once added in the initial Dynamic Evidence Type Version, Optional and/or Mandatory Parents cannot later be removed. New Mandatory and/or Optional Parents can be added in later Versions.

Attributes
> Once defined in the initial Active Dynamic Evidence Type Version, the following Attribute properties cannot be changed in later Versions:
> - Data Type
> - Volatile
> - The class of an Attribute e.g. If an Attribute is defined as a Data Attribute in the initial Version, it cannot be changed to be a Calculated Attribute in later Versions.
>
> Finally, note that Attributes defined in the initial Version can be deleted in later Versions and re-introduced in subsequent Versions, but they can only be re-introduced with the same attribute category, data type and volatility as defined in the initial Version.

Business Start and End Date
> The Evidence Business Start and End Dates can be set or be blank in the initial Version, but these cannot be changed in later Versions.

New Dynamic Evidence Type Versions
> When a new Dynamic Evidence Type is created, the system automatically creates a new Version for it with blank metadata. Subsequently, the following restrictions apply:
> - While the Dynamic Evidence Type has an `In Edit` Version, no new `In Edit` Versions can be added to it.
> - If the Dynamic Evidence Type only has `Active` Versions, it is possible to create a new `In Edit` Version by copying the latest `Active` Version using its **New InEdit Copy** action. Metadata from the latest `Active` Version is copied into the new Version.
> - If all `Active` and `In Edit` Versions are deleted from the Dynamic Evidence Type it is possible to create a new `In Edit` Version with blank metadata using the Dynamic Evidence Type **New Version** action.

It should be noted that the restrictions to modify Attributes, Parents and Business Start and End Dates are only applied if there is at least one `Active` Version in the Dynamic Evidence Type. If at any time the Dynamic Evidence Type is in an `In Edit` Version only (e.g. By deletion of all `Active` Versions), then all such restrictions are dropped and the `In Edit` Version is considered as an initial Version.

Associating Dynamic Evidence Types with Product Delivery or Integrated Cases

Dynamic Evidence Types have to be associated with Product Delivery or Integrated Cases in order to be usable in the case worker workspace to capture Case Evidence data. This is accomplished by using the same administration interface that is used to associate Non-Dynamic Evidence Types to Case Types (see the *Cúram Evidence Guide* for more information). There is no distinction between Dynamic and Non-Dynamic Evidence in this process. All Evidence Types in the system are presented in one selection list to users.

A Dynamic Evidence Type becomes available for association with a Product Delivery or Integrated Case if it has at least one `Active` Version. Dynamic Evidence Types with only an `In Edit` Version are filtered out from the selection list.

Note: It is possible for an administrator to create a Dynamic Evidence Type and an associated Dynamic Evidence Type Version, create metadata, activate the Dynamic Evidence Type Version and link the Dynamic Evidence Type to a Product Delivery or an Integrated Case Type.

The administrator can subsequently go back and cancel the `Active` Dynamic Evidence Type Version. This condition could potentially cause errors in the case worker workspace, as there would be no metadata available to generate the Case Evidence maintenance pages for such a Dynamic Evidence Type (note that the link to the Case Type would still exist).

To avoid such problems, Dynamic Evidence Types with no `Active` Versions are filtered out from the case worker workspace and cannot be selected by case workers.

Links between a Dynamic Evidence Type and Product Deliveries or Integrated Cases are automatically removed when the Dynamic Evidence Type is deleted.

Security Administration

Dynamic Evidence security administration is relatively straightforward, and follows the standard Cúram security administration process (with a few caveats). Dynamic Evidence supports operation-level security; note that field-level security is currently not supported and is intended for a future major release.

Security Groups and operation Security Identifiers are generated when a Dynamic Evidence Type is created (see "Create new dynamic evidence type" on page 8). The Security Group Name specified on the Dynamic Evidence Type Create page is used to create a new Security Group specific to the new Dynamic Evidence Type. The generated Dynamic Evidence Type Security Identifiers are added to this Security Group.

One technical point of note: unlike other Cúram operation Security Identifiers (which are generated for modeled facade operations), Dynamic Evidence operation Security Identifiers are slightly different. There are no facade operations specific to each Dynamic Evidence Type in Dynamic Evidence (as Dynamic Evidence Types are defined at administration time, not development time). As such, all Case Evidence maintenance operations in respect of Dynamic Evidence Types are funneled through a single generic facade where the operation-level security is managed.

Because there are no real facade operations, the operation Security Identifier names are generated based on information from the Dynamic Evidence Type definition (see "Security Identifiers and Security Groups" on page 103). Three Security Identifiers are generated for each Dynamic Evidence Type: one each for Create, Modify and View operations. These Security Identifiers are added to the Security Group created in respect of the Dynamic Evidence Type. The Dynamic Evidence Type Security Group and Security Identifiers can be managed via the Cúram administration application (see the *Cúram Administration Guide* for more information). Security Groups can be added to User Roles to give access rights for the maintenance of individual Dynamic Evidence Types.

A special security group named "EVIDENCEGROUP" can also be used to administer security for Dynamic Evidence Types. If this security group exists, all generated Dynamic Evidence Type Security Identifiers will be added to it i.e. This group is a placeholder for all Dynamic Evidence Security Identifiers. It is intended as a convenience group that can be used to grant access to all Dynamic Evidence operations at once (which is typically of great use in demo scenarios).

warning: The names of the Security Groups and the names of generated Security Identifiers for Dynamic Evidence Types should never be manually modified in the System Administration application. This will result in undefined system behavior.

Note: The "EVIDENCEGROUP" Security Group is not added automatically by the system if it does not exist. Administrators can create this Group if they so wish. If any Dynamic Evidence Types were created before creating this Security Group, they will not be automatically added to it. However, users can manually add the other Dynamic Evidence Types to this group without risk.

Dynamic Evidence Editor overview

The Dynamic Evidence Editor consists of two tabs: **Model** and **User Interface**. In the Editor, you must define the **Model** (attributes and relationships) before you define the **User Interface** (clusters and fields).

When you first open the Dynamic Evidence Editor, the **Model** tab has focus. Both tabs are structured similarly as shown in the following figure:
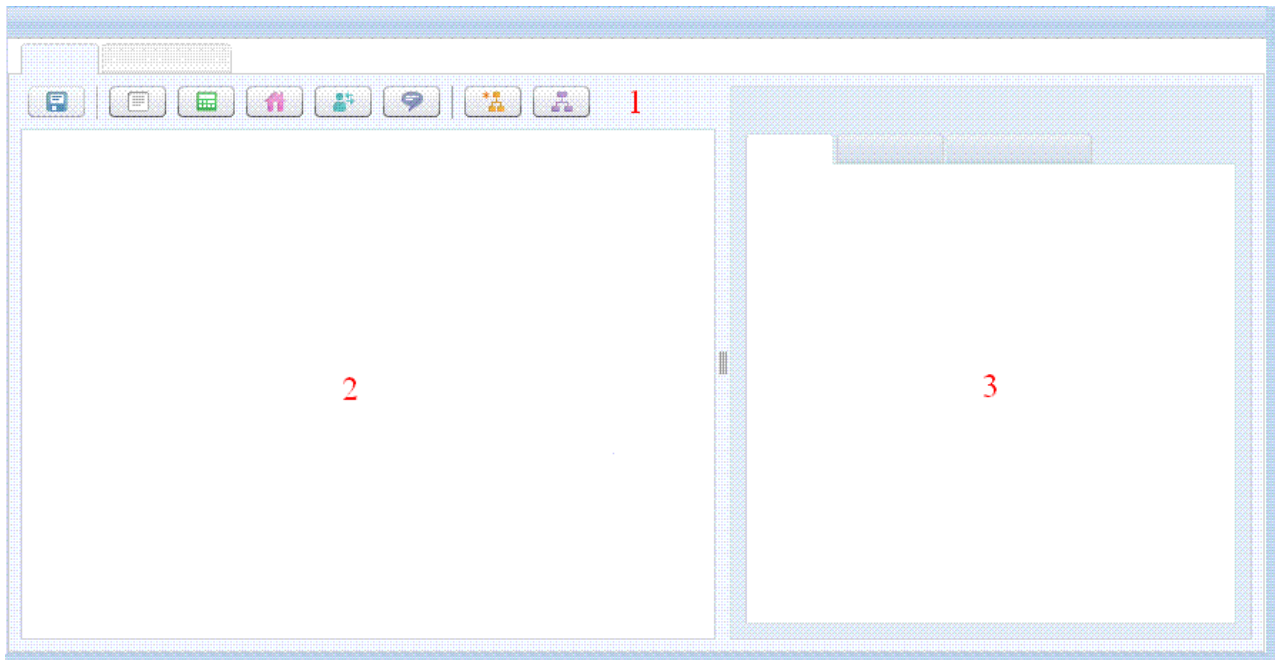
Figure 1. Dynamic Evidence Editor Structure

- **Palette (1)** :

  The Palette is where you can create artifacts.

  - For the **Model** tab, the Palette contains buttons to create Data Attributes, Calculated Attributes, Address Attributes, Related Case Participant Attributes, Related Employment Attributes, Comments Attribute, Optional Parents and Mandatory Parents.
  - For the **User Interface** tab, the Palette is in the left frame of the Editor. The Palette contains a button to create a new Data Attribute Cluster and an accordion control that contains all of the attributes in the Model, categorized by type. You can drag attributes to the Canvas to create fields and clusters.

    Tip: You cannot directly add a field to the user interface. Fields must relate to an existing attribute because the Dynamic Evidence Editor infrastructure uses the attribute definition to determine the associated screen widgets to display.

  For both the **Model** and **User Interface** tabs, the Palette contains a **Save** button that you can use to save the current client data type version to the Cúram database.

- **Canvas (2)**

  The Canvas shows the artifacts that you are modeling.

  - For the **Model** tab, the Canvas graphically represents the client data type version that you are modeling, its underlying attributes, and parent and child relationships with other client data types.

    The client data type version is represented as a shape or a class on the Canvas that you cannot remove. Attributes are represented in a list within this class, and relationships are represented as separate classes that are linked to the client data type version via lines.
  - In the **User Interface** tab, the **Content** tab is to the right of the Palette and above the Properties Panel.

The Canvas is a view of how the Create, Modify and View screens for the client data type version might look in the user interface, and contains the clusters and fields that are currently defined.

Note: The user interface view is not an exact representation of runtime screens; it is intended for layout and planning purposes so that you can envisage what the Client Data screens might look like. For example, when generating screens for each client data type version, the client data infrastructure adds a range of additional fields and features such as Received Date for Create pages, Effective Date of Change and Change Reason for Modify Pages, Mandatory indicators for attributes marked as mandatory in the Model, and Save and Cancel buttons.

- **Properties Panel (3)**

  The Properties Panel contains screens that you can use to add or change properties of a selected artifact. The properties that display in the Properties Panel depend on the type of artifact that you select. For example, the client data type version Properties Panel contains three more sub-panels of information.

  – In the **Model** tab, the Properties panel is to the right of the Canvas. Properties screens exist for the client data type version, attribute types, and relationship types.

  – For the **User Interface** tab, the Properties panel is under the Canvas. Properties screens exist for all cluster types and all field types.

The Dynamic Evidence Editor

The Dynamic Evidence Editor is a graphical editing environment for Dynamic Evidence Type Version Metadata. It allows administrators to define the storage, behavioral and visual characteristics of Dynamic Evidence Type Versions.

These include:
- Model definitions and options for the Dynamic Evidence Type Version, including:
  – Attributes of the Dynamic Evidence Type Version, such as Data Attributes, Calculated Attributes, Address Attributes, Related Case Participant Attributes, Related Employment Attributes and Comments Attributes
  – Mandatory and Optional Parent Relationships between a Dynamic Evidence Type Version and other Dynamic Evidence Types
  – Standard Validations to be invoked at runtime when case workers enter data
  – Summary Details definitions which govern how a Case Evidence record is described in the Evidence Workspace pages at runtime
  – Various behavioral characteristics of a Dynamic Evidence Type Version, such as the width of its create and modify pages at runtime, and Business Start and End dates for the Evidence Type
- User Interface definitions and options for the Dynamic Evidence Type Version, including
  – Clusters of Fields, such as Data Attribute Clusters, Address Clusters, Related Participant Clusters, Related Employment Clusters and Comments Clusters
  – Fields and Skip Fields

Editor Structure

There is a logical order involved in defining the various components of Dynamic Evidence: it is always first necessary to define the Model (Attributes and Relationships) before defining the User Interface (Clusters and Fields). This is

because it is not possible to directly add a Field to the User Interface - the Field has to be in respect of an existing Attribute, as the Dynamic Evidence infrastructure will use the Attribute definition to work out what screen widgets to show in respect of each.

As such, the Dynamic Evidence Editor consists of two Tabs - Model and User Interface - and when first opened, the Model tab has focus.

Both tabs consist of three main structures (in slightly different configurations, but with the same basic meaning):
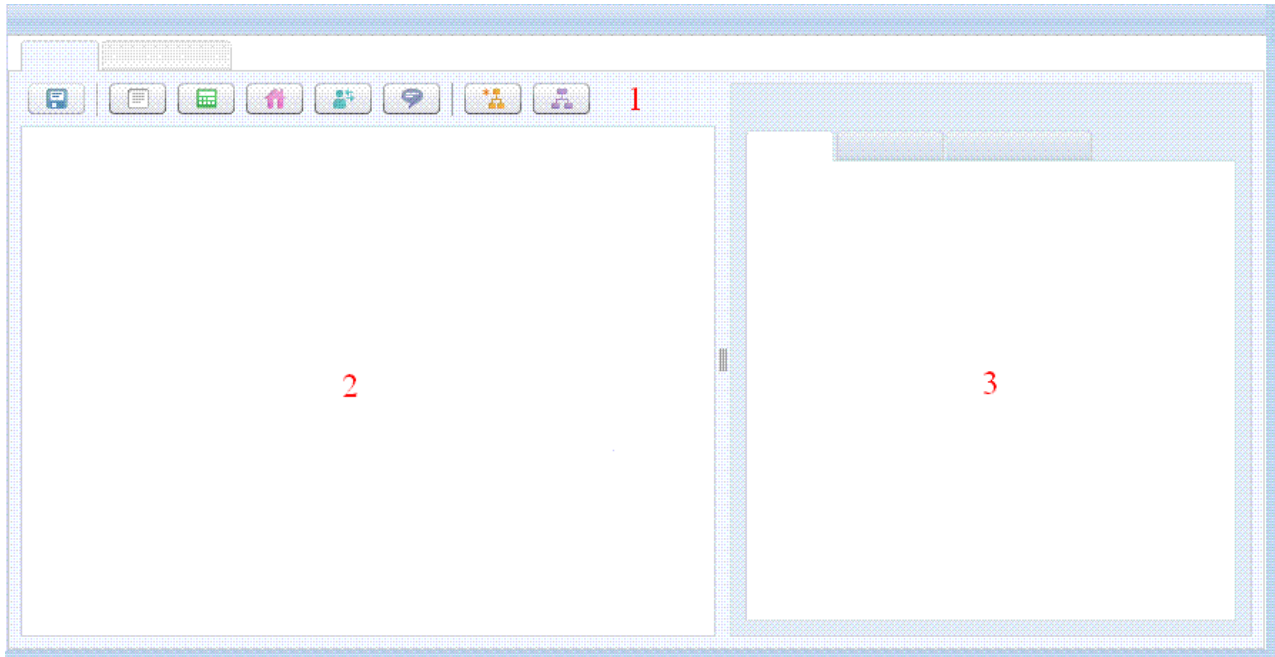
Figure 2. Dynamic Evidence Editor Structure

- **Palette (1)** :

  The Palette is used to create artifacts.

  The Palette for both the Model and User Interface Tabs each has a **Save** button, used to save the current Editor version of the Dynamic Evidence Type Version to the Cúram database.

  - For the Model Tab, the Palette contains buttons to create Data Attributes, Calculated Attributes, Address Attributes, Related Case Participant Attributes, Related Employment Attributes, Comments Attribute, Optional Parents and Mandatory Parents.
  - For the User Interface Tab, the Palette is more complex, having a button to create a new Data Attribute Cluster, but also containing an Accordion control containing all of the Attributes in the Model, categorized by type; these attributes can be dragged onto the Canvas to create Fields and Clusters. In the User Interface Tab, the Palette is located to left of the Editor.

- **Canvas (2)**

  The Canvas contains a graphical representation of the artifacts being modeled.

– For the Model Tab, the Canvas contains a representation of the Dynamic Evidence Type Version being modeled, together with representations of all Attributes in respect of it, and all Parent/Child Relationships it has to other Dynamic Evidence Types.

The Dynamic Evidence Type Version is represented as a shape or a class on the Canvas which is always present - it is not possible to remove this. Attributes are represented in a list within this class, and Relationships are represented as separate classes linked to the Dynamic Evidence Type Version via lines.

– In the User Interface Tab, the Content tab is located to the right of the Palette, and above the Properties Panel.

Here, the Canvas contains a graphical representation of what the User Interface for the Create, Modify and View screens for this Dynamic Evidence Type Version are going to look like at Case Evidence maintenance time, containing all currently defined Clusters and Fields.

Note: The User Interface view is not an exact WYSIWYG representation of runtime screens; rather it is used for layout and planning purposes, to get a feel for what the Evidence screens will look like. In particular, the Dynamic Evidence infrastructure, when generating screens for each Dynamic Evidence Type Version, will add in a range of additional fields and features such as Received Date for Create pages, Effective Date of Change and Change Reason for Modify Pages, Mandatory indicators for attributes marked as mandatory in the Model, Save and Cancel Buttons, etc.

• **Properties Panel (3)**

The Properties Panel is used to house screens that allow an administrator to add or change various properties of the currently selected artifact (see the next section for more information about selections). The Properties Panel will change its available properties dependent on the type of artifact currently selected (for example, the Dynamic Evidence Type Version Properties panel is relatively complex, having a further three sub-panels of information which can be maintained).

– For the Model Tab, Properties screens exist for the Dynamic Evidence Type Version, all Attribute types and all Relationship Types, and the Properties Panel is located to the right of the Canvas.

– For the User Interface Tab, Properties screens exist for all Cluster Types and all Field types, and the Properties Panel sits below the Canvas.

Properties data display

When you select an item in the Dynamic Evidence Editor, your item selection governs the pre-populated data that displays in the Properties Panel.

You can select the following artifacts:

• All Model Attributes
• All Model Parents
• Client Data Type Version
• All Clusters
• All Data and Calculated Attribute Fields

You can select an artifact by clicking on it once. Typically, when you hover the mouse over an item without clicking it, the items that you can select are outlined in the Editor. When you select an item, the artifact reflects a different color.

Tip: An item that is you select is shared between the **Model** and **User Interface** tabs. Therefore, if you select an item in the **Model** tab and subsequently select an item in the **User Interface** tab, the original item is no longer selected when you return to the **Model** tab.

In some cases, you might select an artifact at the same time as you create a new artifact to immediately change its name or some other property.

Once you select an artifact, the Properties Panel updates to reflect the configurable properties for the artifact type with pre-populated values.

Note: You might select an artifact that does not have any configurable properties. For example, if you select attributes that can be dragged in the User Interface palette, the Properties panel contains no properties.

Changing Properties

By selecting the client data type version class in the Model Canvas...

In the main, Properties screens consist of a set of property names and property values, the values generally being maintainable by an administrator; where a value cannot be set by an administrator, it will not even be editable in the Editor.

In order to change the value of a property, simply type in the new value and either hit the 'Enter' or 'Tab' keys. Hitting the 'Enter' key will in most cases update the value and keep the focus in the current property value field (the exception being multi-line fields like 'Description' properties where 'Enter' will add a carriage return to the property value), whereas hitting the 'Tab' key will update the value and move the focus on to the next property value field, if any.

Note: Switching the currently selected item to a different artifact while changing a field will attempt to commit the value before switching to the new Properties screen.

Note: Remember to hit the 'Save' button to persist your Editor work to the Cúram database!

Validation Problems

Every property value is validated upon commit. Different properties have different associated naming and formatting rules, but in each case the administrator will be informed if a proposed property value fails a validation. In the case of validation failures:

- The property will not be updated to the new value. The property field will still contain the (invalid) value entered by the administrator to give them a chance to update it, but behind the scenes the Dynamic Evidence Type Version Metadata will not be updated.
- The property value field will be surrounded with a red border (to indicate invalid content) until the invalid value is cleared or updated to be a valid value.
- Putting the mouse over the red field results in the validation error message appearing in a red popup area. This message also contains information on the format to use to get the property value to pass validation.

Editor Functionality

The next two chapters describe the definition of both Model and User Interface aspects for Dynamic Evidence Type Versions.

Defining a model for a client data type version

To define metadata for a client data type version, you must define the model in the Dynamic Evidence Editor as a first step. A model is the set of attributes and relationships that governs the structure of the Case Evidence that you want to store at run time.

In the **Model** tab of the Dynamic Evidence Editor, you can add a new artifact and add it to the client data type version class in the Canvas area of the Editor. You can model and maintain the following artifacts:

* Properties and information that is specific to the client data type version, in the Properties Panel
* Attributes of the client data type version and its relationships to other client data types, in the Palette area of the Editor

The following topics address how to configure properties, attributes, and relationships from your perspective as an administrator. The definitions that you specify in the Editor determines how the end user will interact with the software application.

Dynamic Evidence Model Definition

Client Data Type Version Properties

In the Properties Panel of the Dynamic Evidence Editor, you can configure the properties of a client data type version, define the property validations to be invoked at runtime, and define the summary details that display at runtime.

You can complete your configuration tasks on the following tabs in the Properties Panel:

* General Properties
* Validations
* Summary Details.

General Properties

To configure properties that determine how the end user will interact with the software application, use the **General Properties** tab in the Properties Panel of the Dynamic Evidence Editor.

The following table lists the properties that you can configure.

Table 1. General Properties

Property name	Description
Read only	Configures a client data type to be read-only. This property disables create and modify operations for an evidence record that is associated with a read-only evidence type. The worker can manually delete read-only evidence records.

Table 1. General Properties (continued)

Property name	Description
Correction Only	Configures updates to an evidence record to allow only corrections. If you set this property to true, a case worker can correct the evidence record but cannot create a succession to an associated evidence record. In this case, The **Effective Date of Change** field is removed from the modify page for the evidence record.
Save and New	Adds a **Save** and **New** button to a Case Evidence Create page for the client data type. This property configures functions at runtime that allows the case worker to complete these tasks: • Save data that is currently entered to the database but not dismiss the page, and • Empty fields of data to allow data entry for another Case Evidence record
Related CP Attribute	Optional: Sets the participant attribute on the Evidence Descriptor record to be the selected Related Case Participant at runtime. If you do not set this property, and the client data type has no Mandatory or Optional Parent relationships, the participant field on the Evidence Descriptor record is the Primary Client of the Case. If you do not set this property, and the Dynamic Evidence Type has at least one Mandatory or Optional Parent relationship, the client data infrastructure iterates through the parent hierarchy until a suitable participant is identified, for example, parent, grandparent, and so on. In order to avail of the Multiple Participant Evidence Update, this field must be set.
Business Start Date	Optional: Sets the business start date to be one of the data attributes that is defined for the current client data type version, with a data type of `Date`. When CER rules process data propagation, this property determines the start date from which to start propagating data for this client data type. For more information about rules data propagation, see "Eligibility and Entitlement Rule Sets" on page 85.
Business End Date	Optional: Sets the business end date to be one of the data attributes that is defined for the current client data type version, with a data type of `Date`. When CER rules process data propagation, this property determines the end date after which data should not be propagated for this client data type. For more information about rules data propagation, see "Eligibility and Entitlement Rule Sets" on page 85.

Table 1. General Properties (continued)

Property name	Description
Create Dialog Width	Optional: Configures the display width, in pixels, of the Create page for the client data type version. If you do not set this property, the default value is 600 pixels.
Modify Dialog Width	Optional: Configures the display width, in pixels, of the Modify page for the client data type version. If you do not set this property, the default value is 600 pixels.
Calculated Attributes Rule Set Name	Optional: Sets the name of the CER Rule Set to be used in the runtime evaluation of Calculated Attribute values. The Dynamic Evidence Editor treats this property as an optional property. However, if one or more Calculated Attributes are defined for the client data type version, you must set this property to be able to activate the client data type version. If you set the property, the property must be the name of a valid CER rule set.
Description	Optional: Sets a description for this client data type version that cannot be localized. This property is for administrative annotation purposes only. The client data infrastructure does not use the property and the Evidence screens in the case worker workspace do not display it at runtime..
Online Help	Optional: Displays the online help text for the Case Evidence Create, Modify and View Pages at run time. The online help text can be localized.

Validations

Validations are data validity checks on Case Evidence records that must be implemented before the records can be saved or activated. To configure validations, use the **Validations** tab in the Properties Panel of the Dynamic Evidence Editor.

In Non-Dynamic Evidence, validations are implemented at development time by using Java code. In Dynamic Evidence, however, you must configure validations to occur at runtime assuming the client data type version has defined validations for the following scenarios:.

- On saving data on a Case Evidence Create page
- On saving data on a Case Evidence Modify page
- On selecting **Apply Changes** or **Validate Changes** from the Evidence Workspace page

Separately or in combination, you can define the following client data validations for a client data type version:

- **Standard Validations**

  By default, the Dynamic Evidence Editor provides a set of configurable Standard Validations. Based on an analysis of evidence validations over time, these Standard Validations are found to be the validations that are frequently used in evidence processing.

- **Additional Validations**

  To specify a validation that cannot be expressed by using a Standard Validation, you can define a CER rule set to process validations. If you define additional validations, the CER validation rule set runs in addition to Standard Validations.

Standard Validations:

The Validations tab of the Evidence Properties Panel, when opened, shows a list of all currently defined Standard Validations in respect of the current Dynamic Evidence Type Version. In this list is a column containing the type of the Validation, plus a column which provides a narrative of the Validation that will be executed at runtime. Standard Validations are typically defined in terms of Model Attributes in the current Dynamic Evidence Type Version.

From the Validations tab, two buttons control the creation and deletion of Standard Validations:

- **Add**

  Clicking this button brings up the Add Validation dialog. This dialog allows for the creation of a Standard Validation for this Dynamic Evidence Type Version. The functionality provided by this dialog is described in the following section.

- **Delete**

  Only enabled when a Validation is selected in the Standard Validations list on the Validations tab, clicking this button removes the currently selected Validation from the Dynamic Evidence Type Version.

Also on the Validations Tab is a panel for Additional Validations. This panel allows the administrator to provide the name of a valid CER Validations Rule Set which will be executed at Case Evidence creation and modification time in addition to the list of Standard Validations. By default the radio button which indicates that there are no additional Validations defined for this Dynamic Evidence Type Version (labeled **None**) is selected; when this is selected, the Dynamic Evidence infrastructure will not look for a CER Validation Rule Set to execute.

Selecting the other radio button (**Use Rule Set**) enables the **Rule Set Name** field on the Additional Validations panel. This free form text field allows administrators to specify a CER Rule Set to be used as a Validation Rule Set for this Dynamic Evidence Type Version; for more information, see "Validation Rule Sets" on page 79. If this field is filled in, it must contain the name of a valid CER Rule Set before this Dynamic Evidence Type Version can be activated.

Standard Validation Types:

Standard validations that you can configure include comparison validations, dependency validations, date of birth validations, and duplicate validations.

The following sections describe each validation type in detail.

Comparison Validation

In a Comparison Validation, a data attribute is compared with another data attribute or a literal by comparing operators and attribute values.

Comparison Validation Property	Description
Source Field	The attribute whose value is to be compared. The Source Field can be a Data Attribute, Calculated Attribute or a Related Case Participant Attribute.
Comparison	The operator to use in the comparison; for more information, see the table that describes available operators.
Target Field	The attribute whose value is to be compared. As a rule, the Target Field must be of the same type (either Attribute Type or Attribute Data Type) to be comparable, and the Target Field list is filtered to only display attributes that are valid for comparison with the selected Source Field. One exception to this rule is for data attributes with a data type of Integer, Money or Float; these numeric types are mutually comparable. The Target Field must not point to the same attribute as the Source Field **Restriction:** You cannot use Related Employment Attributes, Address Attributes and Comments Attributes in Comparison Validations.

The following table describes valid combinations of operators and data types for Data Attributes and Calculated Attributes. Because the behavior of Related Case Participant Attributes differs, the detail of those attributes are described in the next information table.

Table 2. Supported operators and applicable Data/Calculated Attribute data types in Comparison Validations

Operator	Applicable Data Types	Description
==	Boolean, String, Integer, Float, Money, Codetable and Date.	The *equal to* operator checks that Source and Target Fields have exactly the same value; if values differ, the validation check fails. More information follows about using this operator for Date fields.
<>	Boolean, String, Integer, Float, Money, Codetable, Date and Date Time.	The *not equal to* operator checks that Source and Destination Fields do not have exactly the same value; if values are the same, the validation check fails. More information follows about using this operator for Date fields.

Table 2. Supported operators and applicable Data/Calculated Attribute data types in Comparison Validations (continued)

Operator	Applicable Data Types	Description
<	Integer, Float, Money, Date and Date Time.	The *less than* operator checks that the Source Field value is less than the Destination Field value. If the Source Field value is greater than or equal to the Destination Field value, the validation check fails.
<=	Integer, Float, Money, Date and Date Time.	The *less than or equal to* operator checks that the Source Field value is less or equal to than the Destination Field value . If the Source Field value is greater than the Destination Field value, the validation check fails.
>	Integer, Float, Money, Date and Date Time	The *greater than* operator checks that the Source Field value is greater than the Destination Field value. If the Source Field value is less than or equal to the Destination Field value, the validation check fails.
>=	Integer, Float, Money, Date and Date Time	The *greater than or equal to* operator checks that the Source Field value is greater than or equal to the Destination Field value. If the Source Field value is less than the Destination Field value, the validation check fails.
before	Date and Date Time.	The *before* operator checks that the Source Field value is before the Destination Field value. If the Source Field value is on or after the Destination Field value, the validation check fails.
on or before	Date and Date Time.	The *on or before* operator checks that the Source Field value is on or before to the Destination Field value. If the Source Field value is after the Destination Field value, the validation check fails.

Table 2. Supported operators and applicable Data/Calculated Attribute data types in Comparison Validations (continued)

Operator	Applicable Data Types	Description
after	Date and Date Time	The *after* operator checks that the Source Field value is after the Destination Field value. If the Source Field value is on or before to the Destination Field value, the validation check fails.
on or after	Date and Date Time	The *on or after* operator checks that the Source Field value is after or equal to the Destination Field value. If the Source Field value is before the Destination Field value, the validation check fails.

Note: When the Source Field is populated with a data attribute that has a data type of `Date`, two additional attributes that do not exist in the client data type version metadata are added to the Target Field list:

- **evidenceReceivedDate**

  This attribute represents the date on which an agency receives a piece of Evidence into the organization. The Received Date is stored on the Case Evidence Descriptor at runtime and is frequently used in evidence comparison validations. The client data infrastructure automatically adds this field to every Create and Modify evidence page that relates to client data type version.

- **evidenceEffectiveDateOfChange**

  This attribute represents the effective date of change for case evidence record. The Effective Date of Change is stored on the Case Evidence Descriptor at runtime and is frequently used in evidence comparison validations. The client data infrastructure automatically adds this field to every Modify evidence page that relates to a client data type version.

The following table describes the operators that apply to Related Case Participant Attributes in Comparison Validations.

Table 3. Supported operators for Related Case Participant Attributes in Comparison Validations

Operator	Description
==	The *Equal To* operator checks that Source and Target Fields represent the same participant; if field values do not equate to the same participant, the validation fails. An additional boolean attribute, `shallow` is provided for this validation but the client data infrastructure ignores it when the operator is ==. If the Related Case Participant ID is the same, the underlying Concern Role ID must also be the same.

Table 3. Supported operators for Related Case Participant Attributes in Comparison Validations (continued)

Operator	Description
<>	The *Not Equal To* operator checks that Source and Destination Fields do not have exactly the same value; if field values are the same, the validation check fails. An additional boolean attribute called shallow is provided for this Validation. If shallow is checked in the Create Validation dialog, only the Related Case Participant IDs on the Evidence Record are compared. If shallow is not checked in the Create Validation dialog, the underlying Concern Role IDs are also checked for equality.

The following table describes more options for Comparison Validation.

Table 4. Additional options for Comparison Validation

Options	Description
Literals	A source attribute (Data Attributes or Calculated Attributes) can be compared against literals. As a rule, the literal value must be the same data type as the selected source attribute to be comparable. To compare a source attribute against a literal, select the **Use Literal** check box; subsequently you can type the literal value in the Target Fidel or select the value in the case of data types such as Codetable or Boolean or Date. **Tip:** You might need to select a code table item as a literal value when the data type of the source attributes is Codetable. If the data type of the first attribute or second attribute is Boolean, specify values of true or false. If the data type of the attribute is Date, specify the date value or select the date by using the Date Picker. For numeric data types such as Integer, Float and Money, you can enter locale-specific format for the literal value. For the Money attribute, you can also type the currency symbol. **Restriction:** Specify literal values only for data attributes.

Table 4. Additional options for Comparison Validation (continued)

Options	Description

Multiple Clauses	By selecting the **Multiple Clause** check box, you can specify multiple clauses in a Comparison Validation. Each clause must succeed for the overall validation check to succeed.
	
	To control the creation and deletion of Multiple Clauses, use these functions:
	• **Add**
	    By clicking this button, you can add a clause to the current validation based on the source, target and operator fields that are selected.
	• **Delete**
	    Only enabled when a clause is selected in the Clause list, clicking this button removes the clause/comparison validation that is selected.

Message ID	By clicking the **Add Validation Message** dialog adjacent to the Message property, you can set a custom validation message. More information follows about the Custom Validation Message for Comparison Validations. **Restriction:** If a validation has Multiple Clauses, this property is mandatory.

The following table describes the mandatory properties for Multiple Clauses in Comparison Validations.

Table 5. Multiple Clause Properties

Multiple Clause Properties	Description
Conjunctions	Controls whether any clause or all the clauses in a group are validated at run time. • If you select the **Any Clause** radio button, and if any one of the clauses passes at runtime, the entire validation check succeeds. • If you select the **All Clauses** radio button, all clauses must succeed for the entire validation check to succeed.

Dependency Validation

Use the Dependency Validation to enforce a dependency of a particular type between two attributes.

Restriction: You cannot use Calculated Attributes in Dependency Validations.

Dependency Validation Property	Description
First Attribute	The Data Attribute, Address Attribute, Related Case Participant Attribute or Comments Attribute on which the Second Attribute depends.
Second Attribute	The Data Attribute, Address Attribute, Related Case Participant Attribute or Comments Attribute that depends on the First Attribute.
Dependency	The nature of the dependency that can include one of the following values: • **Must enter second attribute** If you select this value, and subsequently the case worker enters field value for the attribute pointed to by the First Attribute, they must also enter a field value for the attribute pointed to by the Second Attribute. If the case worker enters a value in the first field but not in the second field, this validation check fails. • **Must not enter second attribute** If you select this value, and subsequently the case worker enters a field value for the attribute pointed to by the First Attribute, they must not enter a field value for the attribute pointed to by the Second Attribute. If the case worker enters values in both fields, this validation check fails. • **At least one attribute** If you select this value, subsequently the case worker must enter a value into either or both fields pointed to by First Attribute and Second Attribute. If the case worker leaves both fields empty, the validation check fails. • **Only one attribute** If you select this value, subsequently the case worker must enter a value into one or other fields pointed to by First Attribute and Second Attribute. If the case worker enters values in both fields or does not enter values in either field, this validation check fails.
Bidirectional	This boolean property applies to Dependency Validations with a defined dependency of **Must enter second attribute** and **Must not enter second attribute**only. Selecting the Bidirectional property causes the words **...And Vice Versa** to be added to the descriptions under **Dependency**. **Restriction:** You cannot use this property when the other dependency values are selected.

The following table describes more options in Dependency Validation.

Table 6. Additional options in Dependency Validation

Options	Description
Literals	You can specify literal values for both the First Attribute and the Second Attribute. As a rule, the literal value must be the same data type as the selected first attribute or second attribute. To specify literals either to an First Attribute or Second Attribute, select the **Use Literal** checkbox; subsequently you can type the literal value in the Source or Target Literal Field A source attribute (Data Attributes or Calculated Attributes) can be compared against literals. As a rule, the literal value must be the same data type as the selected source attribute to be comparable. **Tip:** You might need to select a code table item as a literal value when the data type of the source attributes is `Codetable`. If the data type of the first attribute or second attribute is `Boolean`, specify values of `true` or `false`. If the data type of the attribute is `Date`, specify the date value or select the date by using the Date Picker. For numeric data types such as Integer, Float and Money, you can enter locale-specific format for the literal value. For the Money attribute, you can also type the currency symbol. **Restriction:** Specify literal values only for data attributes.
Message ID	By clicking the **Add Validation Message** dialog adjacent to the Message property, you can set a custom validation message. More information follows about the Custom Validation Message for Comparison Validations.

Date of Birth Validation

In Date of Birth Validations, the date of birth of the Participant that is pointed to by a Related Case Participant Attribute in the client data type version is verified to be on or before a specific date.

Restriction: You cannot use Calculated Attributes in Date of Birth Validations.

Date of Birth Validation Property	Description
Related Participant	The Related Case Participant whose date of birth value is to be used in the comparison.

The choicelist for this property is pre-populated with all Related Case Participant Attributes that are currently defined for the client data type version. At runtime, the date of birth for the Person pointed to by the Related Case Participant is compared against the date that is specified in the **Input Date** field. If the **Input Date** is before the date of birth, the validation check fails.

Restriction: Only Related Case Participants of type `Person` are valid for use in Date of Birth validations, even though the Dynamic Evidence Editor does not enforce this restriction. |
| Input Date | Data attributes with a data type of `Date` to be used in the comparison. Two additional attributes that do not exist in the client data type version metadata are added to the Input Date field list:

- **evidenceReceivedDate**

  This attribute represents the date on which an agency receives a piece of Evidence into the organization. The Received Date is stored on the Case Evidence Descriptor at runtime and is frequently used in evidence comparison validations. The client data infrastructure automatically adds this field to every Create and Modify evidence page that relates to client data type version.

- **evidenceEffectiveDateOfChange**

  This attribute represents the effective date of change for a case evidence record. The Effective Date of Change is stored on the Case Evidence Descriptor at runtime and is frequently used in evidence comparison validations. The client data infrastructure automatically adds this field to every Modify evidence page that relates to a client data type version. |

The following table describes more options in Date of Birth Validation.

Table 7. Additional options in Date of Birth Validation

Options	Description
Message ID	By clicking the **Add Validation Message** dialog adjacent to the Message property, you can set a custom validation message. More information follows about the Custom Validation Message for Comparison Validations.

Duplicate Validation

In Duplicate Validations, Case Evidence records that are verified, according to specified criteria, as duplicates are prevented from being recorded on the system.

To identify duplicates, the Duplication Validation can check different sets of evidence records. For example, if the client data type version has one or more parent client data types, the Duplicate Validation only checks child records of parent records at runtime, that is, the sibling records of the current record..

However, if the client data type version has no parent relationships, the Duplicate Validation checks all evidence records for duplicates at run time.

Restriction: You cannot use Calculated Attributes in Duplicate Validations.

Duplicate Validation Property	Description
Use Date Range, Start Date, End Date	If you select the **Use Date Range** check box, two mandatory properties display on the Create Validation dialog: Start Date and End Date. These properties must point to a data attribute with data type of `Date`.
	At runtime, the Duplicate Validation identifies any records in the selection set that have attribute values pointed to by the Start and End Dates, and are equal to the values that are specified in the Create or Modify evidence screens. If duplicate records are returned, the validation check fails.
Other Attributes to Check	Optional: a list of other attributes whose values are to be compared to check for duplicates (in tandem with any Date Range provided)
	If records in the selection set have attribute values that are equal to the values for the attributes in the **Other Attributes to Check** list (specified in the Create or Modify evidence screens), the validation check fails. **Note:** If multiple attributes are in the list, can take the following actions: • To check the attributes individually, select **Check each attributes individually**. • To check attributes in combination, that is, by checking that all the attributes in the selection list are unique, select **Check attributes together**.

Duplicate Validation Property	Description
Validate Date Range and Other Attributes together	If you select this property, at runtime, the Duplicate Validation checks for evidence records that have overlapping Start and End dates and attribute values that are equal to the values for attributes in the **Other Attributes to Check** list (specified in the Create or Modify evidence screens). If duplicate evidence records are returned, the validation check fails.

If you do not select this property, at run-time, the Duplicate Validation either checks for evidence records that have overlapping Start and End dates or attribute values that are equal to the values for attributes in the **Other Attributes to Check** list (specified in the Create or Modify evidence screens). If duplicate evidence records are returned, the validation check fails. |

The following table describes some additional available options in Duplicate Validation.

Table 8. Additional options in Duplicate Validation

Options	Description
Message ID	By clicking the **Add Validation Message** dialog adjacent to the Message property, you can set a custom validation message for both Date Range attributes and for other attributes. More information follows about the Custom Validation Message for Comparison Validations.

Custom Validation Message

To set a custom validation message for an validation type, specify the following properties:

Table 9. Custom validation message properties

Validation Message Mapping Properties	Description
Message	The text to use for the validation message.

In the **Messages Parameters** property list, you can specify parameters with attribute names in this format: opening curly brace, parameter number, closing curly brace, for example {0}, {1}, and so on . If a validation check fails at runtime, the parameters that you specify are substituted in the validation message that displays.

For example, suppose that the Message property is set as follows: {0} must not be equal to true and {1} must not be equal to SX2. The Message Parameters are set as follows: isPregnant, gender.

If the comparison validation fails at run time, after a user creates an evidence record, the message that displays to the user is similar to the following: isPregnant must be equal to true and gender must be equal to female. |

Table 9. Custom validation message properties (continued)

Validation Message Mapping Properties	Description
Message ID	A mandatory string to be used as the key to the Message property value. The string can be any valid identifier, for example, `My EvidenceTypeVersion.ComparisonValidation.Message`
Message Parameters	An ordered list of Data Attributes, Related Case Participant Attributes, Address Attributes, Comments Attributes, or Calculated Attributes to be used in the Message. For Comparison Validations or Date of Birth Validations, two additional attributes that do not exist in the client data type version metadata are added to the field choicelists: `evidenceReceivedDate` and `evidenceEffectiveDateOfChange`. See the preceding sections for more details about these attributes.

Summary Details

As mentioned earlier, the Summary Details (also known as Summary Information) for a Dynamic Evidence Type Version specifies the value to be used in the Description column of the Dynamic Evidence Type Workspace page for each Case Evidence record in respect of this Dynamic Evidence Type Version. As a reminder, the Dynamic Evidence Type Workspace page is the tab opened by clicking on an Evidence Type name on the Evidence Dashboard. The Description appears on the Dynamic Evidence Type Workspace page as a hyperlink that, when clicked, opens up the Business Object Tab for the selected Case Evidence record.

Dynamic Evidence provides a number of different ways to specify Summary Details for a Dynamic Evidence Type Version, controlled by the radio buttons on the Summary Details Tab of the Evidence Properties Panel.

The following options are provided:

- **None**

  No Summary Details have been provided for the Dynamic Evidence Type Version. At runtime, the Description for Case Evidence records in respect of this Dynamic Evidence Type Version will be represented as an asterisk (to allow the hyperlink to be clicked). However, it is strongly recommended that Summary Details be provided for every Dynamic Evidence Type Version, as it would otherwise be impossible for case workers to differentiate Case Evidence records for such Dynamic Evidence Type Versions.

- **Use Rule Set**

  As with Validations, it is possible to use a CER Rule Set to calculate the Summary Details description at runtime, and this is typically used in the case where Summary Mappings cannot be used to achieve the desired outcome.

  If this option is selected, the **Rule Set Name** field gets enabled for editing. As the name suggests, the Rule Set Name should be populated with the name of a valid CER Summary Rule Set if the Summary Details for the Dynamic Evidence Type Version is to be calculated using CER Rules. Activating the Dynamic Evidence Type Version will validate that any value provided for this property points to a valid CER Rule Set.

  Information on defining Summary Details using CER can by found in the chapter on CER Rule Sets; see "Summary Information Rule Sets" on page 77 for more details.

- **Use Mapping**

  The most usual (and straightforward) way of defining Summary Details for a Dynamic Evidence Type Version is by defining Summary Mapping information. Using Summary Mapping allows the administrator to specify the description to

be used in the case worker workspace as a Resource Message, and also allows the administrator to specify a number of attributes to be used in the display of the Dynamic Evidence Type Workspace pages. Summary Mapping is described in detail in the next section.

Note: The options available for specifying Validations are not mutually exclusive; it is possible to define both Standard Validations and CER Rule Set Validations for a single Dynamic Evidence Type Version.

With Summary Details, however, these options are mutually exclusive - it is only possible to select one of the preceding options.

Summary Mapping: If **Summary Mapping** is selected as the means of providing Summary Details for the Dynamic Evidence Type Version, administrators can set the following optional properties:

Summary Mapping Property	Description
Start Date	If this property is set in the Editor, the value of the Attribute pointed to by the Start Date property will be used in the Dynamic Evidence Type Workspace page as the first date in the **Period** column of the list for each Case Evidence record. If this property is not set in the Editor, the system will attempt to use the value for the Evidence Business Start Date property as the first date in the **Period** column. If, in turn, this date is also not set, then the Case Start Date will be used as the first date in the **Period** column at runtime. In the Editor, this property can be set to any Data Attribute or Calculated Attribute with a data type of `Date`.
End Date	If this property is set in the Editor, the value of the Attribute pointed to by the End Date property will be used in the Dynamic Evidence Type Workspace page as the second date in the **Period** column of the list for each Case Evidence record. If this property is not set in the Editor, the system will attempt to use the value of the Evidence Business Start Date property as the second date in the**Period** column. If, in turn, this date is also not set, the Period will be displayed as open-ended i.e. Without a second date displayed in the **Period** column of the list for each Case Evidence record (e.g. `2/2/2001 -`) In the Editor, this property can be set to any Data Attribute or Calculated Attribute with a data type of `Date`

Summary Mapping Property	Description
Participant	If this property is set in the Editor, the name of the participant pointed to by the Participant property will be used to populate the Participant column of the Dynamic Evidence Type Workspace page at runtime. If this property is not set in the Editor, the name of the Primary Client will be populated instead.
Message ID	To set the text to be used for the Description column of the Dynamic Evidence Type Workspace page, administrators must set the `Message ID` property. To set this property, the search icon adjacent to the `Message ID` property should be clicked; this opens the Edit Summary Message dialog.

Edit Summary Message Dialog

To set or modify the Message for a Summary Mapping, the following properties must be specified

Summary Mapping Property	Description
Message	The text to use for the Summary Details Message. This Message can be parameterized with Attribute values, placeholders for which are specified in the Message with the format of: opening curly brace, parameter number, closing curly brace - e.g. {0}. At runtime, the parameters specified in the following `Message Parameters` list will be substituted into the Message to construct the Evidence Description to display. See the `Message Parameters` property for more information on parameterization of messages.
Message ID	Mandatory string to be used as the key to the Message property value; can be any valid identifier (e.g.`MyEvidenceTypeVersion.SummaryMessage`)

Summary Mapping Property	Description
Message Parameters	An ordered list of Data Attribute and/or Calculated Attributes or Address Attribute to be used in the Message. Placeholders are put in the Message to indicate that an Attribute value should be substituted into the Message at runtime, and these placeholders follow the following pattern: {0}, {1}, {2}, etc. For example, say the Message property is set to the following: Earns ${0} {1} working for {2} And the Message Parameters are set the following: incomeAmount, incomeFrequency, employerName Then the message (given the appropriate data in the Case Evidence Record) could be displayed as something like: `Earns $123.50 weekly working for Midway Services Inc.`

Adding New Attributes

Evidence Attributes of various types are added to a Dynamic Evidence Type Version using the following buttons on the Model Palette:

- Add Data Attribute

- Add Calculated Attributes

- Add Address Attribute

- Add Related Case Participant Attribute

- Add Related Employment Attribute

- Add Comments Attribute

The following sections describe each of these in detail.

Add Data Attribute

The New Data Attribute button creates a new Data Attribute for the Dynamic Evidence Type Version, displays it in the Model Canvas, selects it, and opens the Properties Panel for Data Attributes. This panel allows the administrator to set the following properties:

Table 10. Data Attribute Properties

Property name	Description
Attribute Name	This mandatory property sets the Model name for the selected Data Attribute. The value of this property is treated as the internal identifier for the Data Attribute across the entire Dynamic Evidence Type Version meta data, and as such must be unique across all Attributes in the Dynamic Evidence Type Version. The Attribute Name is never used in the generation of screens for Case Evidence pages, and so the case worker will never see its value. Attribute Names must follow a specific naming format: • They must start with a lowercase English alphabetic character • They can only contain lowercase or uppercase English alphabetic characters, numeric characters and underscores. • They must not contain any reserved words such 'relatedEmployment' or 'comments' as these are reserved identifier in the Dynamic Evidence Editor.

Table 10. Data Attribute Properties (continued)

Property name	Description
Attribute Type	This property sets the data type for the selected Data Attribute. The following Data Types are available: • String (default) - any character value • Integer - whole numbers • Date - a calendar date • DateTime - a calendar date and time • Money - a currency value • Boolean - true or false • Float - a floating-point number • Codetable - a Cúram codetable code The Attribute Type fundamentally affects many aspects of the Data Attributes administrative and runtime behavior: • It governs what Type Safety Validations will get executed for values entered into a Field in respect of this Attribute; all Case Evidence Attribute values are validated by the Dynamic Evidence infrastructure before they can be saved to the database to ensure that they are valid values for the specified Type. • It governs what User Interface widget is used to display the Attribute Value, both at administration time (in the User Interface tab of the Editor) and at runtime (on Case Evidence create, modify and view screens in respect of this Dynamic Evidence Type Version). • It controls the Attributes available for selection in Standard Validations, Summary Details, General Evidence Properties, etc. • It determines the data type options available in the relevant Attribute Properties panels • And many others - where the behavior of Dynamic Evidence is affected by the Attribute Type, this is documented separately in this guide Attempting to change the Attribute Type of a Data Attribute which is already referenced somewhere in the Dynamic Evidence Type Version (e.g. It is on the User Interface, or referenced in Validations, or referenced in Summary Information, etc.) will cause the Editor to confirm with the administrator whether or not they want to proceed. If they choose to proceed, the Editor will remove all references to it (e.g. In Business Start and End Dates, Validations, Summary Information, etc.), and will update the User Interface based on the new Type.

Table 10. Data Attribute Properties (continued)

Property name	Description
Domain Name	Domain Name can be created with custom widget implementation and it should be added to the Curam Domain list. For more details on developing Domain and its custom widget, please refer to the Cúram Custom Widget Development Guide and Appendix A **Note:** • The domain (custom widget) should be configured with edit-renderer, view-renderer and converter • The domain converter (custom widget) should be derived from the Standard Domain like `SvrStringConverter`, `SvrInt32Converter`, etc. • The target value on the edit-renderer (widget implementation) should be in basic data type format • Custom widget with multiple hierarchies is not supported
Default Value	This optional property sets the default value to be pre-populated on Case Evidence create pages for the selected Data Attribute. Default values must be valid for the selected Attribute Type, and the Editor will not allow default values with incorrect formats for the selected Attribute Type to be set by the administrator. **Note:** In most cases, the Default Value is a free from text field, but this changed for the following attributes. drop down containing 'true' and 'false' where the Attribute Type is set to 'Boolean'. drop down containing code table items where the Attribute Type is set to 'Codetable'. text field with date picker where the Attribute Type is set to 'Date'. Locale specific format can be typed in for the data types such as Date, Integer, Float and Money and typing locale specific Currency Symbol can be typed in case of Money attribute.

Table 10. Data Attribute Properties (continued)

Property name	Description
Volatile	This property setting indicates whether values for this Data Attribute can logically change over time, and is currently solely used in CER Ruleset processing; this property has no effect on the runtime Case Evidence screens.

For example, a 'Vehicle' Evidence Type may have two attributes - vehiclePurchasePrice and vehicleFairMarketValue. In this example, vehiclePurchasePrice is non-volatile (it has a value at Case Evidence record creation time, and this value cannot ever change), whereas vehicleFairMarketValue is volatile (the fair market value will likely decrease over time).

If set to True, the Volatile Property will cause this Attribute to be generated as a Time line Attribute in generated Dynamic Evidence Data Rule Sets, allowing its value to change across succession sets. If set to False, the Attribute will be generated as a non-time line Attribute which can have a single value over time.

The Volatile property cannot be changed between Dynamic Evidence Type Versions for a Dynamic Evidence Type; once this property is set in a Dynamic Evidence Type Version and that Version is activated, it must continue to have the same value for all subsequent Dynamic Evidence Type Versions. |
Mandatory	This property setting indicates that the selected Attribute should be considered Mandatory in Case Evidence create and modify pages. This property does not apply to Boolean or Codetable Data Attributes, Calculated Attributes or Related Case Participant Attributes. If set, the caseworker must specify a value for Fields in respect of this Data Attribute, and such Fields will be displayed with an asterisk to indicate to the case worker that they are required. Note that all Mandatory Attribute must appear in the User Interface!
Calculate on create If Blank	This property is used to specify a calculated attribute to be used to populate this data attribute if it is blank on create. A drop-down box enables linking the Data Attribute to a Calculated Attribute. The drop-down lists Calculated Attributes in the Evidence Type that are of the same data type as the Data Attribute. Once a Calculated Attribute is selected it will be used to calculate the value of the Data Attribute if it is blank on the Create page.
<<Data Type>> Options	Data Attributes have a number of options that can be applied to them that govern their behavior. These vary for each Attribute Type. The options are described in "Data Attribute and Calculated Attribute Type Options" on page 50 .

Table 10. Data Attribute Properties (continued)

Property name	Description
Description	This property configures a non-localizable Model description value for the selected Attribute. This is for annotative purposes only, and is never displayed to the case worker at runtime.

Add Calculated Attribute

Calculated Attributes are Dynamic Evidence Type Version Attributes which, rather than being provided by the case worker, are calculated by the system at runtime (and are as such read-only). Values for Calculated Attributes are evaluated via the execution of a CER Calculated Attributes Rule Set.

Clicking on the **Add Calculated Attribute** button in the Model Palette creates a new Calculated Attribute for the Dynamic Evidence Type Version, displays it in the Model Canvas, selects it, and opens the Properties Panel for Calculated Attributes.

All Attribute property settings relevant to Data Attributes are also relevant to Calculated Attributes with the exception of Default Value, Volatile and Mandatory settings (in general, any properties which affect the editing of the Attribute do not apply to Calculated Attributes as they are read-only in nature). Please refer to the Data Attributes property settings for more details.

As mentioned in the section on Evidence Properties, the Rule Set Name for Calculated Attributes is configured on the General tab of the Dynamic Evidence Type Properties Panel. For more information on the use of CER Rule Sets for Calculated Attributes, please refer to "Calculated Attributes Rule Sets" on page 83.

Add Address Attribute

An Address Attribute is a single field which represents a complete address which can be maintained by case workers at runtime; Address Attributes can be dragged as User Interface Clusters, where they expand into a complete set of address fields such as Address Line 1, Country, etc.

Clicking on the **Add Address Attribute** button in the Model Palette creates a new Address Attribute for the Dynamic Evidence Type Version, displays it in the Model Canvas, selects it, and opens the Properties Panel for Address Attributes. This panel contains the following properties:

Table 11. Address Attribute Properties

Property name	Description
Attribute Name	This property sets the Model name for the selected Address Attribute. The value of this property is treated as the internal identifier for the Address Attribute across the entire Dynamic Evidence Type Version meta data, and as such must be unique across all Attributes in the Dynamic Evidence Type Version. The Attribute Name is never used in the generation of screens for Case Evidence pages, and so the case worker will never see its value.

Attribute Names must follow a specific naming format:

• They must start with a lowercase English alphabetic character

• They can only contain lowercase or uppercase English alphabetic characters, numeric characters and underscores.

• They must not contain any reserved words such as `relatedEmployment` or `comments` as these are reserved identifier in the Dynamic Evidence Editor. |
| Volatile | This property setting indicates whether values for this Address Attribute can logically change over time, and is currently solely used in CER Ruleset processing; this property has no effect on the runtime Case Evidence screens.

For example, a `Vehicle` Evidence Type may have two attributes - `vehiclePurchasePrice` and `vehicleFairMarketValue`. In this example, `vehiclePurchasePrice` is non-volatile (it has a value at Case Evidence record creation time, and this value cannot ever change), whereas `vehicleFairMarketValue` is volatile (the fair market value will likely decrease over time).

If set to True, the Volatile Property will cause this Attribute to be generated as a Time line Attribute in generated Dynamic Evidence Data Rule Sets, allowing its value to change across succession sets. If set to False, the Attribute will be generated as a non-time line Attribute which can have a single value over time.

The Volatile property cannot be changed between Dynamic Evidence Type Versions for a Dynamic Evidence Type; once this property is set in a Dynamic Evidence Type Version and that Version is activated, it must continue to have the same value for all subsequent Dynamic Evidence Type Versions. |

Table 11. Address Attribute Properties (continued)

Property name	Description
Mandatory	This property setting indicates if the selected Attribute should be considered Mandatory in Case Evidence create and modify pages. If set, the caseworker must specify enter information into the Address Field during creation or modification of Case Evidence records in respect of this Dynamic Evidence Type Version. Note that all Mandatory Attribute must appear in the User Interface! **Note:** In Cúram 6.0, Mandatory Address Attributes will not be displayed with an asterisk next to them at runtime in the case worker Case Evidence maintenance pages
Description	This property configures a non-localizable Model description value for the selected Attribute. This is for annotative purposes only, and is never displayed to the case worker at runtime.

Add Related Case Participant

Related Case Participants are Participants other than the Primary Client to be associated with a Case Evidence record. Related Case Participants allow case workers to do one of three things:

- Select an existing Case Participant (i.e. a Participant who has already been added to the Case) to associate with the current Case Evidence record
- Search for a Participant (i.e. a Participant who has not already been added to the Case) to associate with the current Case Evidence record (and also with the Case, creating a new Case Participant record at the same time)
- Register a new Case Participant of type Representative and associate them with the current Case Evidence record, creating a new Case Participant record at the same time

In many cases it is sufficient that the Case Evidence record be associated with an existing Case Participant, and so it is possible for the administrator to configure Related Case Participants so that only the first of these options is available to case workers.

Clicking on the **Add Related Case Participant Attribute** button in the Model Palette creates a new Related Case Participant Attribute for the Dynamic Evidence Type Version, displays it in the Model Canvas, selects it, and opens the Properties Panel for Related Case Participant Attributes.

This panel contains the following properties:

Table 12. Related Case Participant Attribute Properties

Property name	Description
Attribute Name	This property sets the Model name for the selected Related Case Participant Attribute. The value of this property is treated as the internal identifier for the Related Case Participant Attribute across the entire Dynamic Evidence Type Version metadata, and as such must be unique across all Attributes in the Dynamic Evidence Type Version. The Attribute Name is never used in the generation of screens for Case Evidence pages, and so the case worker will never see its value. Attribute Names must follow a specific naming format: • They must start with a lowercase English alphabetic character • They can only contain lowercase or uppercase English alphabetic characters, numeric characters and underscores. • They must not contain any reserved words such `relatedEmployment` or `comments` as these are reserved identifier in the Dynamic Evidence Editor.
Participant Type	The case worker may want to register a new participant on the case for a Related Case Participant attribute value rather than selecting or searching for one that is currently registered on the System. When a new related case participant is required, basic participant details are entered and the System will create a Representative for this Participant. The Participant Type property is used to indicate the type of participant a Representative will represent. The setting of this property will not result in the creation of a Participant of that specified. For Example, A Person is not created if Participant Type = Person, however, a Representative is created to represent that person. The Participant Type setting specifies the type of participant that this representative will represent. The default value for representative `Type = CONTACT` for all participant type settings when creating the Representative. For further information on Representative types please refer to the Cúram Participant Guide. The following values can be selected for Participant Type: • Person • Employer • Service Provider • Unknown
Participant Role Type	This property, possible values for which are taken from the CaseParticipantRoleType codetable, specifies what type of Case Participant Role record will be created when the Related Case Participant Attribute is entered in the Case Evidence create or modify screens as runtime.

Table 12. Related Case Participant Attribute Properties (continued)

Property name	Description
Participant Type Options	This property allows administrators to display only a subset of Case Participant Role Types in the second panel of the Related Case Participant Cluster (i.e. The panel which allows case workers to search for a Participant to associate with this Case Evidence record). Possible values are taken from the CaseParticipantRoleType codetable.

Add Related Employment Attribute

Related Employment Attributes represent employment records to be associated with Case Evidence records. Related Employment Attributes operate similarly to Mandatory and Optional Parent Relationships, in that they result in caseworkers being presented with a wizard for Evidence records creation. At run time, case workers will be presented with a list of employments from which they will select one to relate to the Evidence record; the final page in the wizard will then show the create page for the Evidence Type. On View and Modify Pages, the Related Employment Attribute will detail the employer and case participant for the related employment record.

Clicking on the **Add Related Employment** button in the Model Palette creates a new Related Employment Attribute for the current Dynamic Evidence Type Version, displays it in the Model Canvas, selects it and opens the Properties Panel for Related Employment Attribute. Only one Related Employment Attribute can exist in respect of a Dynamic Evidence Type Version.

Table 13. Related Employment Attribute Properties

Property name	Description
Attribute Name	For Related Employment Attributes, the Attribute name is always defaulted to `relatedEmployment`, which is a reserved identifier in the Dynamic Evidence Editor. This Attribute Name cannot be changed.
Description	This property configures a non-localizable Model description value for a selected Related Employment Attribute. This is for annotative purpose only, and is never displayed to the Case worker at run time.
Participant Type Options	At run time, Related Employment Attributes result in a Select Employment page in the Case Evidence creation wizard for case workers. This page shows a list of employment records, one of which has to be selected by the case worker before progressing to create an Evidence Record for the associated Evidence Type Version. By default, this list of employment records will consist of those for both the Primary Client and any Member Participant Role Types for Case Participants on the related case. It is however possible to refine this list by specifying a list of Case Participant Role Types (from the Case Participant Role Type code table) to be used to retrieve Employment records. Note that if any item is added to this list, the above defaults do not apply - i.e. it will be up to the administrator to manually add Primary Client and Member if this is desired.

Add Comments Attribute

Clicking on the **Add Comments Attribute** button will add a Comments Attribute to the Dynamic Evidence Type Version. At runtime, for Comments Clusters in respect of such attributes, the case worker will see a multi-line free text comments field on the Case Evidence create and modify screens. In situations where this Attribute is being viewed, the text will be displayed as a label. Only one Comments Attribute can exist in respect of a Dynamic Evidence Type Version.

The following properties can be administered for Comments Attributes:

Table 14. Comments Attribute Properties

Property name	Description
Attribute Name	For Comments Attributes, the Attribute name is always defaulted to `comments`, which is a reserved identifier in the Dynamic Evidence Editor. This Attribute Name cannot be changed.
Description	This property configures a non-localizable Model description value for the selected Comments Attribute. This is for annotative purposes only, and is never displayed to the case worker at runtime.

Data Attribute and Calculated Attribute Type Options

For Data Attributes and Calculated Attributes, the Attribute Type controls which additional options are available for that Attribute. Changing the Attribute Type for a Data Attribute or Calculated Attribute will remove any previous options which had been set in respect of that Attribute, and cause the relevant new Options Property Panel to be displayed.

The following sections describe the Options properties available for each Attribute Data Type.

The Attribute Type fundamentally affects many aspects of the Data Attributes administrative and runtime behavior:

* It governs what Type Safety Validations will get executed for values entered into a Field in respect of this Attribute; all Case Evidence Attribute values are validated by the Dynamic Evidence infrastructure before they can be saved to the database to ensure that they are valid values for the specified Type.
* It governs what User Interface widget is used to display the Attribute Value, both at administration time (in the User Interface tab of the Editor) and at runtime (on Case Evidence create, modify and view screens in respect of this Dynamic Evidence Type Version).
* It controls the Attributes available for selection in Standard Validations, Summary Details, General Evidence Properties, etc.
* It determines the data type options available in the relevant Attribute Properties panels
* And many others - where the behavior of Dynamic Evidence is affected by the Attribute Type, this is documented separately in this guide

String: The following properties can be set in respect of **Data Attributes** with a data type of 'String':

Table 15. String Type Properties

Property name	Description
Minimum Length	This property sets the minimum character length for the Attribute when entered by the case worker. Entering a value with a lesser number of characters than the value of this property will result in a validation error at Case Evidence record save time. Locale specific format can be typed in for Minimum Length.
Maximum Length	This property sets the maximum character length for the Attribute when entered by the caseworker. Note that at runtime, case workers will not be able to physically enter more characters into a Field in respect of this Attribute than is specified in this property value.Locale specific format can be typed in for Maximum Length.
Compress Embedded Spaces	This property indicates that any extra whitespace embedded in the Attribute string, and that all leading and trailing whitespace, be removed from the value entered by the case worker before storing it. For more details please refer to *Cúram Server Modelling Guide* , in the section on Domain Definitions Options.
Remove Leading Spaces	This property indicates that any leading spaces should be stripped off the Attribute value entered by the case worker.
Remove Trailing Spaces	This property indicates that any trailing spaces should be stripped from the Attribute value entered by the case worker before storing it.
Convert to Uppercase	This property indicates that the contents of this Attribute value entered by the case worker should be converted to uppercase before storage.

Integer, Float or Money: The following properties can be set in respect of Data Attributes with a data type of `Integer`, `Float` or `Money`:

Table 16. Numeric Type Properties

Property name	Description
Maximum Value	This property sets the maximum value for the Integer, Float or Money Attribute entered by the case worker. Locale specific format can be typed in for Maximum Value (Currency Symbol can also be typed along with the Maximum Value in case of Money attribute).
Minimum Value	This property sets the minimum value for the Integer, Float or Money Attribute entered by the case worker. Locale specific format can be typed in for Minimum Value (Currency Symbol can also be typed along with the Maximum Value in case of Money attribute).

Date: The following properties can be set in respect of Data Attributes with a data type of `Date`:

Table 17. Date Type Properties

Property name	Description
Current Date Comparison	This property setting will perform a validation against the the value specified by the case worker for this Date, to ensure that it is before,after, equal to, not equal to, on or before, on or after to the current date.
Custom Message	To set a custom validation message, the administrator must set the `Custom Message` property. To set this property, select the radio button **Current Date Comparison** and the search icon adjacent to the `Custom Message` property should be clicked: this brings up the Add Validation Message dialog. For more details on Custom Validation Message, see the Custom Validation Message section.

DateTime: The following properties can be set in respect of Data Attributes with a data type of `DateTime`:

Table 18. DateTime Type Properties

Property name	Description
Current Date Comparison	This property setting will perform a validation against the the value specified by the case worker for this Date, to ensure that it is before, after, on or before, on or after to the current date time (note that it is not possible or sensible to check a DateTime for equality against the Current Date, and so `equal to` has been removed from the list of available operators).
Custom Message	To set a custom validation message, the administrator must set the `Custom Message` property. To set this property, select the radio button **Current Date Comparison** and the search icon adjacent to the`Custom Message` property should be clicked: this brings up the Add Validation Message dialog. For more details on Custom Validation Message, see the Custom Validation Message section.

Codetable: The following properties can be set in respect of **Data Attributes or Calculated Attributes** with a data type of 'Codetable':

Table 19. Codetable Type Properties

Property name	Description
Codetable Name	This property sets the Codetable name for the Data Attribute or Calculated Attribute. At runtime, the case worker will be able to select a value for Fields in respect of this Attribute from a dropdown list of Codetable items for the Codetable whose name is specified in this property value.

Adding Relationships

Relationships for a Dynamic Evidence Type Version define an association between the Version and other Dynamic Evidence Types. Two such Relationships are

currently supported, Mandatory Parents and Optional Parents. It is not possible to create a Relationship between a Dynamic Evidence Type Version and a Non-Dynamic Evidence Type.

Note that such Relationships are always made between a Dynamic Evidence Type Version and a Dynamic Evidence Type - structuring the relationships in this way allows relationships for a Dynamic Evidence Type to evolve over time (e.g. It is possible to add a relationship in subsequent Dynamic Evidence Type Versions).

When creating a Case Evidence record in respect of a Dynamic Evidence Type Version which has one or more Parent relationships, it is necessary to select a parent record before the child record can be created. For example, say we have the following scenario:

We have two Dynamic Evidence Types `Income` and `IncomeAllocation`; the Dynamic Evidence Type Version for `IncomeAllocation`as a Mandatory Parent Relationship with Income; in Evidence terms, `IncomeAllocation` is a child Evidence Type of Income. When a case worker tries to create a Case Evidence record of type `IncomeAllocation`, they will be presented with a wizard, the first page of which will contain a list of potential Parent records of type `Income`. The case worker will need to select one of these before being able to create the `IncomeAllocation`Case Evidence record.

Note: The use of the terms *Mandatory* and *Optional* in relation to Dynamic Evidence Type Parent relationships is historical in nature, and slightly misleading. In actual fact BOTH of these Parent relationship types result in the case worker having to select a parent record at run time, so neither are really optional; the optionality refers to their having an option of which parent Dynamic Evidence Types to associate with the Case Evidence record. See the following sections for more information

Note: A Dynamic Evidence Type Version can have both Mandatory and Optional Parents

Parent Relationships are added to the Dynamic Evidence Type Version Model definition using the following buttons from the Model Palette:

- Add Mandatory Parent

- Add Optional Parent

Add Mandatory Parent
The **New Mandatory Parent Relationship** button adds a new Mandatory Parent Relationship to a Dynamic Evidence Type Version Model definition, creates a shape in the Model Canvas representing the new Relationship, selects the Relationship, and opens up the Mandatory Parent Properties Panel. The Dynamic Evidence Type Version then, in essence, becomes a Child Evidence Type of the Parent Evidence Type.

Dynamic Evidence Type Versions can have multiple Mandatory Parent Relationships. Create pages for Evidence Types that define Mandatory Parents will be wizard based. Each Mandatory Parent will result in a new wizard page containing a selection list for that parent on the Case Evidence create page. The case workers will have to select one record as the parent record from each wizard page list (i.e. It is mandatory to select a parent record for each Mandatory Parent type).

It is not valid to have multiple Mandatory Parent Relationships to the same Dynamic Evidence Type for the same Dynamic Evidence Type Version, and the Evidence Editor will enforce this restriction.

The following properties can be set in respect of a Mandatory Parent Relationship:

Table 20. Mandatory Parent Properties

Property name	Description
Relationship Name	This property sets the Model Relationship Name for the selected Mandatory Parent Relationship. The value of this property is treated as the internal identifier for the relationship and must be unique across all Relationships in the Dynamic Evidence Type Version.The Relationship Name is never used in the generation of screens for Case Evidence pages, and so the worker will never see its value. They are however used in the generation of CER Data and Processing Rule Sets for this Dynamic Evidence Type Version - see "Eligibility and Entitlement Rule Sets" on page 85 for more details. Relationship Names must follow a specific naming format: • They must start with a lowercase English alphabetic character • They can only contain lowercase or uppercase English alphabetic characters, numeric characters and underscores • They must not contain any reserved words such as `relatedEmployment` or `comments` as these are reserved identifier in the Dynamic Evidence Editor.
Parent Type Code	This property setting specifies the Evidence Type Code for the Mandatory Parent. At runtime, the case worker must select one instance of a Case Evidence record for the Parent Type Code during the creation of the child Case Evidence record.
Filter Attribute Name	The attributes of Evidence Type Code for Mandatory Parent whose attribute value is to be filtered. The Field can be a Data Attribute of type String, Boolean, Integer, Money, Float and Date.

Table 20. Mandatory Parent Properties (continued)

Property name	Description
Filter Attribute Literal Value	A Filter Attribute can be compared against literals. As a general rule, the literal value must be of the same data type as that of the selected attribute in order to be comparable. To filter an attribute against a literal, select the filter attribute name from Filter Attribute Details; the literal value can then be typed (or selected in case of data types such as Codetable, Boolean or Date) in to the Filter Attribute Value field. **Note:** Literal Value can only be specified for Data Attributes. Administrators may need to select a code table item as a literal value when the data type of the selected filter attribute is "codetable". Where the data type of filter attribute is `Boolean`, the values of `true` and `false` can be provided. In case where the data type is Date, date value can either be typed or can be selected through a Date Picker. Locale specific format can be typed for the literal value for the numeric data types such as Integer, Float and Money attribute.
Description	This property configures a non-localizable Model description value for the selected Mandatory Parent Relationship. This is for annotative purposes only, and is never displayed to the case worker at runtime.

Add Optional Parent

The New Optional Parent Relationship button adds a new Optional Parent Relationship to a Dynamic Evidence Type Version Model definition, creates a class in the Model Canvas representing the new Relationship, selects the Relationship, and opens up the Optional Parent Properties Panel.

Dynamic Evidence Type Versions can have multiple Optional Parent Relationships. Create pages for Evidence Types that define Optional Parents will be wizard based. Case Evidence records for all of these Dynamic Evidence Types will be listed in a single list on a single wizard page that is part of the Case Evidence wizard for this Dynamic Evidence Type Version, and case workers will have to select one record from the list as the parent record (i.e. Even though it is 'mandatory' to select one record in the Case Evidence maintenance pages, case workers have an option in which Case Evidence Type they choose as the Evidence Type of the parent record).

It is not valid to have multiple Optional Parent Relationships of the same Evidence Type for the same Dynamic Evidence Type Version, and the Evidence Editor will enforce this restriction.

The following properties can be set in respect of an Optional Parent Relationship:

Table 21. Optional Parent Properties

Property name	Description
Relationship Name	This property sets the Model Relationship Name for the selected Optional Parent Relationship. The value of this property is treated as the internal identifier for the relationship and must be unique across all Relationships in the Dynamic Evidence Type Version.The Relationship Name is never used in the generation of screens for Case Evidence pages, and so the case worker will never see its value. They are however used in the generation of CER Data and Processing Rule Sets for this Dynamic Evidence Type Version - see "Eligibility and Entitlement Rule Sets" on page 85 for more details.
	Relationship Names must follow a specific naming format:
	• They must start with a lowercase English alphabetic character
	• They can only contain lowercase or uppercase English alphabetic characters, numeric characters and underscores
	• They must not contain any reserved words such as 'relatedEmployment' or 'comments' as these are reserved identifier in the Dynamic Evidence Editor.
Parent Type Code	This property setting specifies the Evidence Type Code for the Optional Parent. At runtime, the case worker must select one instance of a Case Evidence record for the Parent Type Code during the creation of the child Case Evidence record.
Description	This property configures a non-localizable Model description value for the selected Optional Parent Relationship. This is for annotative purposes only, and is never displayed to the case worker at runtime.

Deleting Attributes

Administrators can remove Attributes of any Type from the Dynamic Evidence Type Version.

This is achieved by clicking the 'x' button in the Attribute in the Evidence class on the Model Canvas.

Note: Similar to changing the Type of a Data Attribute or Calculated Attribute, if the Attribute being deleted is currently referenced in Validations, Summary Information, Business Start and End Dates, Related CP Properties, or has a Field in respect of it in the User Interface, the Editor will confirm if the administrator still wants to delete them. If the administrator confirms the delete, the Editor will proceed to remove all traces of the deleted Attribute from the Dynamic Evidence Type Version (e.g. by blanking out Business Start and End Dates, by removing Validations which reference it, by removing User Interface components which reference it, etc.)

Deleting Relationships

Administrators can remove Relationships of any type from the Dynamic Evidence Type Version. Note that it is not valid to remove a Relationships if the Dynamic Evidence Type in question has an Active Version with this relationship present (the Editor will allow it, but this will fail Activation for the modified Dynamic Evidence Type Version).

This is achieved by clicking the 'x' button in the Mandatory or Optional Parent class on the Model Canvas. Deleting a Relationship will remove the class and its line to the Dynamic Evidence Type Version class from the Model Canvas, trigger all other Relationship classes to reposition themselves on the Model Canvas, and if the last Relationship was deleted, trigger the Dynamic Evidence Type Version class to reposition itself.

Save Dynamic Evidence Type Version Updates

Clicking on the Save button will save the current state of the Dynamic Evidence Type Version (for both Model and User Interface Tabs) to the database.

Defining the user interface

You use the User Interface tab of the Dynamic Evidence Editor to graphically specify the contents and layout of Case Evidence screen Fields and Clusters.

The Canvas

The Canvas Panel is used as the drawing surface for the Case Evidence screens for the Dynamic Evidence Type Version. On this canvas, administrators can create, drag and drop, and maintain the following artifacts:

- Attribute Clusters
- Data Attribute Fields
- Calculated Attribute Fields
- Address Clusters
- Related Participant Clusters
- Related Employment Clusters
- Comments Clusters
- Utility Fields

The Canvas allows the administrator to see the layout of the Case Evidence screen layout more or less as it will be viewed by the case worker when creating, modifying and reading Case Evidence records in respect of this Dynamic Evidence Type Version.

Note: The User Interface Canvas is not a 100% WYSIWYG Editor; at runtime, the Dynamic Evidence infrastructure will generate Case Evidence screens which are similar to, but not absolutely identical to the contents of the User Interface Canvas. For example, generated Modify pages for a Dynamic Evidence Type Version will contain the following items which are not drawn in the User Interface Canvas in the Editor:

- Save Button
- Cancel Button
- Asterisks for Mandatory Fields
- Evidence Received Date
- Effective Date of Change

- Change Reason
- Codetable Items for Codetable Data Attributes
- etc.

However, these discrepancies are relatively minor in visual impact, and in general it is easy for the administrator to see how the Case Evidence screens will look at runtime when using the User Interface Canvas.

Properties for each artifact in the preceding list can be maintained by selecting that component on the Canvas and editing the associated properties in the Properties Panel (which on the User Interface tab is located below the Canvas). The following sections describe each User Interface artifact in detail, describing what the artifact is used for, its properties, and the impact these properties have on the Case Evidence screens at runtime.

Mapping Model Attributes to User Interface Artifacts

There are two basic kinds of artifact which can exist on the User Interface Canvas, namely **Fields** and **Clusters**.

Fields

Fields contain values to be displayed on runtime Case Evidence screens in respect of Dynamic Evidence Type Versions, and essentially consist of a Label and a Value.

Fields are defined in terms of Data Attributes or Calculated Attributes - a Field is basically the Screen representation of such an Attribute, with a number of additional properties which can be set. The Attribute's data type will determine what widget the Field will use at runtime in order to display the Attribute value. The following table shows a mapping between data types and Field widgets used to render these types:

Table 22. Attribute Data Types and their Corresponding Field Renderers

Attribute Data Type	Field Renders Using:
String, Integer, Money, Float	Text Box
Codetable	Dropdown
Boolean	Checkbox
Date	Cúram Date widget
DateTime	Cúram DateTime widget

It is not necessary for the administrator to specify which widget to use; these are automatically inferred by the Dynamic Evidence infrastructure.

Reordering Fields: Fields, once on an Attribute Cluster on the User Interface Canvas, can be reordered at any time, and can be moved both within Attribute Clusters and between Attribute Clusters.

In the first case, it is possible to move a Field from one position in an Attribute Cluster to another position in the same Attribute Cluster. To do this, click on the Label of the Field in question, and, by holding the mouse button down, drag it until it is directly over the Label of the Field with which it is to be positionally swapped. Finally, release the mouse button, and the Field will move to its new position.

Note: Clusters cannot be dropped onto the position they current occupy

In the second case, it is possible to move a Field from one Attribute Cluster to another Attribute Cluster. This will have the effect of removing it completely from the first Cluster, and adding it as the last Field in the second Cluster. To do this, click on the Label of the Field in question, and, by holding the mouse button down, drag it until it is directly over the middle of the Attribute Cluster to which it is to be moved. Finally, release the mouse button, and the Field will move to its new position.

Deleting Fields: It is possible for administrators to delete Fields from an Attribute Cluster at any time. To do this, they click on the button marked 'x' which appears when the mouse is over the Field in question. The Field is then immediately removed from the screen, and the Cluster is redrawn.

Clusters

Clusters are containers of information on Case Evidence screens, and they come in a number of different varieties.

- Attribute Clusters, which are the containers for Fields. Data Attributes and Calculated Attributes can both be dragged from the accordion view in the Palette and dropped as Fields onto Attribute Clusters
- Address Clusters, which are the User Interface representations of Model Address Attributes. Address Attributes can be dragged from the accordion view in the Palette and dropped as Clusters onto the area over an existing Cluster.
- Related Case Participant Clusters, which are the User Interface representations of Related Case Participant Attributes. Like Address Attributes, Related Case Participant Attributes can be dragged from the accordion view in the Palette and dropped as Clusters onto the area over an existing Cluster.
- Related Employment Clusters, which are the User Interface representations of Related Employment Attributes. Related Employment Attributes can be dragged from the accordion view in the Palette and dropped as Clusters onto the area over an existing Cluster.
- Comments Clusters, the User Interface representations of Model Comments Attributes. Comments Attributes can also be dragged from the accordion view in the Palette and dropped as Clusters onto the area over an existing Cluster.

Reordering Clusters: Clusters, once on the User Interface Canvas, can be reordered at any time; this is especially important for Attribute Clusters, in that these are always added as the last Cluster on a Page. Clusters of any type can be reordered in exactly the same manner - this method applies equally to Attribute Clusters, Address Clusters, Related Case Participant Clusters, Related Employment Clusters and Comments Clusters.

To reorder a Cluster, click on a white-colored area of the Cluster to be moved and, by holding the mouse button down, drag it to the area directly over another Cluster. When the Cluster is over an area onto which it can be dropped, the drop area will change color to light blue. Finally, release the mouse button, and the Cluster will move to its new position.

Note: Clusters cannot be dropped into the area directly over themselves, as this essentially attempts to move the Cluster to the location that it already occupies.

Deleting Clusters: It is possible for administrators to delete Clusters from the User Interface Canvas at any time. To do this, they click on the button marked 'x' which appears when the mouse is over the Cluster in question. The Cluster is then immediately removed from the screen, and the page is redrawn.

Attribute Clusters

Attribute Clusters can only be created by clicking on the **Add Attribute Cluster** button on the User Interface Palette (Note that they are the only User Interface artifact created by clicking a button): [icon] . Note also that the Attribute Cluster will always be added as the last Cluster in the User Interface Canvas; if desired, administrators can subsequently move the newly added Attribute Cluster to another location on the Canvas, as described earlier in this chapter.

At Activate time, Dynamic Evidence Type Versions will be checked to ensure that all Attribute Clusters contain at least one Field.

The following properties can be set in respect of a Data Attribute Cluster:

Table 23. Standard Cluster Properties

Property name	Description
Title	An optional Cluster Title for the default locale. Note that most Clusters have Titles, but they are not in fact enforced as mandatory by the Dynamic Evidence infrastructure.
Title ID	A Resource property identifier for the Title property. Mandatory if a Title is provided (this property is only displayed in the Editor if the Title has one or more characters).
Description	An optional Cluster Description for the default locale. Cluster Descriptions, if provided, appear after the Title at runtime.
Description ID	A Resource property identifier for the Description property. Mandatory if a Description is provided (this property is only displayed in the Editor if the Title has one or more characters).
Number of Columns	A mandatory numeric drop down containing values from 1 to 4, the Number of Columns dictates how many Columns will be used to lay out all contained Fields in this Attribute Cluster in generated create, modify and view pages in respect of this Dynamic Evidence Type Version. Typically, this value is set to 2 (and occasionally 1) for most Dynamic Evidence Types.
Label Width	The percentage of the width of an Attribute Cluster column that the label should occupy. The specified width will be applied to every other column. In most cases this should be set to 40 which is a default value.
On Create Page	Possible values are true or false. If true, the Attribute Cluster is shown on the Case Evidence create page; if false, the Cluster is not displayed on the Create page. In most cases this should be set to true (i.e. The check box should be checked).
On Modify Page	Possible values are true or false. If true, the Attribute Cluster is shown on the Case Evidence modify page; if false, the Cluster is not displayed on the Modify page. In most cases this should be set to true (i.e. The check box should be checked).

Table 23. Standard Cluster Properties (continued)

Property name	Description
On View Page	Possible values are true or false. If true, the Attribute Cluster is shown on the Case Evidence view page; if false, the Cluster is not displayed on the View page. In most cases this should be set to true (i.e. The check box should be checked).
Online Help	An optional property containing Cluster information for Online Help on Case Evidence screens.

warning: Care must be taken with the use of the On Create page and On Modify page properties to ensure that Clusters which do not appear on Case Evidence maintenance screens do not contain Fields in respect of Data Attributes which are marked as Mandatory. This is a logical error (which is not currently enforced by the Editor), in that it would not be possible for case workers to provide values for mandatory Attributes, and hence should be avoided.

Data Attribute Fields

Data Attributes can be dragged from the accordion control on the User Interface Palette, and dropped as Fields onto Attribute Clusters.

The following Field properties can be set for Data Attributes:

Table 24. Data Attribute Field Properties

Property name	Description
Label	Field Label for the default locale is enforced as mandatory by the Dynamic Evidence infrastructure.
Label ID	A Resource property identifier for the Label property. Mandatory if a Label is provided (this property is only displayed in the Editor if the Label has one or more characters).
Source Attribute	The name of the Model Data Attribute that this Field populates.
Modifiable	Possible values are true or false. This property determines whether or not the Field is read-only on the Case Evidence modify page. If false (i.e. The Field is read-only after first creation), then the Field Value is rendered as a Label. Otherwise an editable widget appropriate to the associated Data Attribute Type will be used.
Use Default	Possible values are true or false. A true setting indicates that the field should use the default value specified for the `Default Value` property for the related Source Attribute. The case worker user will see the Default Value populated in this field on a Case Evidence create operation. If no Default Value has been provided in the Data Attribute Model Properties, but Use Default is true, then the Field will be populated with appropriate default values for the data type in question (e.g. the current date for Data Attributes of type `Date,` 0 for Data Attributes of type `Integer,` etc).

Table 24. Data Attribute Field Properties (continued)

Property name	Description
Use Blank	This property is only applicable if the Source Attribute has a data type of `Codetable`. This property indicates the dropdown for the Source Field should have a selectable blank value in it (in effect, making the Codetable mandatory or non-mandatory). It has a setting of true or false. If true the caseworker will see a blank value added to the list of possible selectable Codetable values in the dropdown for this Field. If it is set to false, only the list of Codetable values will be displayed, thus making the Field mandatory.
Online Help	An optional localizable text containing field or attribute information for Online Help on Case Evidence screens.

Calculated Attribute Fields

Calculated Attributes can also be dragged from the accordion control on the User Interface Palette, and dropped as Fields onto Attribute Clusters.

The following Field properties can be set for Calculated Attributes:

Table 25. Calculated Attribute Field Properties

Property name	Description
Label	An optional Field Label for the default locale. Note that most Fields have Labels, but they are not in fact enforced as mandatory by the Dynamic Evidence infrastructure.
Label ID	A Resource property identifier for the Label property. Mandatory if a Label is provided (this property is only displayed in the Editor if the Label has one or more characters)
Source Attribute	The name of the Model Calculated Attribute that this Field references.
Online Help	An optional localizable text containing calculated attribute information for Online Help on Case Evidence screens.

Address Clusters

Address Attributes can be dragged from the accordion control on the User Interface Palette, and dropped over existing Clusters as Address Clusters.

The property settings for address clusters are the same as those for Attribute clusters; please refer to "Attribute Clusters" on page 60 for more details.

Related Case Participant Clusters

To configure properties that determine how case workers will interact with the Related Case Participants are Participants.

As discussed in the Model Definition chapter, Related Case Participants are Participants other than the Primary Client to be associated with a Case Evidence record. Related Case Participants allow case workers to do one of three things:

- Select an existing Case Participant (i.e. a Participant who has already been added to the Case) to associate with the current Case Evidence record
- Search for a Participant (i.e. a Participant who has not already been added to the Case) to associate with the current Case Evidence record (and also with the Case, creating a new Case Participant record at the same time)
- Register a new Case Participant of type Representative and associate them with the current Case Evidence record, creating a new Case Participant record at the same time

Related Case Participant Clusters are the User Interface representation of Related Case Participant Attributes.

The following properties can be set for a Related Case Participant Cluster

Table 26. Related Case Participant Cluster Properties

Property name	Description
Name for Cluster	An optional (but practically essential) localizable partial name for the Related Case Participant Cluster (partial, as whatever the administrator adds as text in this field will be suffixed by Details e.g. **Income Details**).
Name for Cluster ID	A Resource property identifier for the Name for Cluster property. Mandatory if a Name for Cluster is provided (this property is only displayed in the Editor if the Name for Cluster has one or more characters).
Name for Labels	An optional (but again practically essential) localizable partial name to use for a number of Labels on the 3-Panel Related Case Participant Cluster. These are: • For the first Panel, the field label will be set to **Name for Labels Value Participant** • For the second Panel, the field label will be set to **Name for Labels Value** • If the **One Name** field is set to true, the field label for the first field in the third Panel will be set to **Name for Labels Name** • If the **One Name** field is set to false, the field label of the first field in the third Panel will be set to**Name for Labels First Name**, and the field label for the second field in the third Panel will be set to**Name for Labels Surname**.
Name for Labels ID	A Resource property identifier for the Name for Labels property. Mandatory if a Name for Labels is provided (this property is only displayed in the Editor if the Name for Labels has one or more characters).
Name for Descriptions	An optional (but again practically essential) localizable Label to be used in all descriptions on the Related Case Participant Panel (e.g. If the **Name for Descriptions** is a case participant, please select from the following list of case participant).

Table 26. Related Case Participant Cluster Properties (continued)

Property name	Description
Name for Descriptions ID	A Resource property identifier for the Name for Descriptions property. Mandatory if a Name for Descriptions is provided (this property is only displayed in the Editor if the Name for Descriptions has one or more characters).
Case Participant Descriptor	The setting of this value will be used in the label of the Case Participant Panel(the first panel) as a suffix to the in the For Example **Name for Labels Value Case Participant Descriptor** The Default value is Participant.
Case Participant Descriptor ID	A Resource property identifier for the Case Participant Descriptor property. Mandatory if a Case Participant Descriptor is provided (this property is only displayed in the Editor if the Case Participant Descriptor has one or more characters).
First Name Label	This value setting is used as an alternative label setting for the First Name field label of the Related Case Participant widget Third Panel. The First Name field value will will be set to this property setting if it has 1 or more characters .
First Name Label ID	A Resource property identifier for the First Name Label property. Mandatory if a First Name Label is provided (this property is only displayed in the Editor if the First Name Label has one or more characters).
Second Name Label	This value setting is used as an alternative label setting for the Second Name field label of the Related Case Participant widget Third Panel. The Second Name field value will will be set to this property setting if it has 1 or more characters. This property is not displayed if the **One Name** field setting is true.
Second Name Label ID	A Resource property identifier for the Second Name Label property. Mandatory if a Second Name Label is provided (this property is only displayed in the Editor if the Second Name Label has one or more characters). This property is not displayed if the **One Name** field setting is true.
Allow Modification	Possible values are **no**, **single**, and **multiple**. The functionality of this setting is described in conjunction with the **Show all Panels** property. Please refer to "Show All Panels and Allow Modification" on page 65 for more information.
Search Type	The **Search Type** property determines the search popup page that the case worker will see on create and modify pages. This property have such values as Person, Employer, Product Provider, and Service Supplier, its default setting being blank. E.g. If **Search Type** is **Person**, the caseworker will see a Person search popup when searching for related participants. If blank, the search widget seen on the create/modify pages will be a multi popup search type, where the type of search result can be specified by the case worker at runtime.

Table 26. Related Case Participant Cluster Properties (continued)

Property name	Description
Show All Panels	Possible values are true or false. Please refer to "Show All Panels and Allow Modification" .
Field	Possible values are true or false. If this option is set to true, a single name field (called **Name**) will be presented for any new Related Case Participant created. If this value is false then both First and Second Name can be specified.
On Create Page	Possible values are true or false. If true, the Related Case Participant Cluster is shown on the Case Evidence create page; if false, the Cluster is not displayed on the Create page. In most cases this should be set to true (i.e. The checkbox should be checked).
On Modify Page	Possible values are true or false. If true, the Related Case Participant Cluster is shown on the Case Evidence Modify page; if false, the Cluster is not displayed on the Modify page. In most cases this should be set to true (i.e. The checkbox should be checked).
On View Page	Possible values are true or false. If true, the Related Case Participant Cluster is shown on the Case Evidence View page; if false, the Cluster is not displayed on the View page. In most cases this should be set to true (i.e. The checkbox should be checked).
Default To Blank	Possible values are true or false. If true, the Participant drop-down defaults to blank option on Related Case Participant cluster when evidence is being added. If false, the Participant drop-down defaults to the primary case participant if this participant is available in the drop-down. This option has no effect on what is displayed on evidence modify screen, as when evidence is modified the entry displayed represents existing case participant value. It may be empty if no case participant was specified upon evidence creation.
Online Help	An optional localizable text containing Cluster information for Online Help on Case Evidence screens.
Enable Multi Case Member Updates	This attribute, when set will allow updates to evidence to be applied across active case participants. This property is only available in the participant that is selected in the **Related CP Attribute** in the **General Properties**. To enable this feature deselect the **Show All Panels**. Once selected **Allow Modification** is set to No.

Show All Panels and Allow Modification

If **Show All Panels** is set to true for a Related Case Participant Cluster, then three panels are displayed in the Case Evidence Create page in respect of the Related Case Participant. If this property is set to false, then only the first panel is displayed.

Panel one allows the user to select an existing Case Participant, to associate it with the Case Evidence record currently being recorded.

Panel two provides the user with the ability to search for a Participant on the System.

Panel three presents the user with name, address and phone details fields to register a new participant (of type Representative).

The **Allow Modification** property value governs whether or not a Related Case Participant can be modified.

If **Allow Modification** is set to **multiple**, the case worker can always update the Related Case Participant Reference on the Case Evidence modify page.

If **Allow Modification** is set to **single**, if the Related Case Participant was not entered during the initial Case Evidence record creation, then it will be modifiable on the Case Evidence modify page. However, once the Related Case Participant has been set for the first time and saved (either as a result of a create or modify action), the current Related Case Participant name and age will subsequently be displayed in a label, and no further modification will be possible.

If **Allow Modification** is set to **no**, then the Related Case Participant value is mandatory on create and it cannot be modified subsequently. In this case a read-only **name and age** label for the Related Case Participant is always displayed during in the modify page.

If **Show All Panels** is set to false and **Allow Modification** is set to **no**, the mandatory marker is displayed on the create page

Related Employment Clusters

The Related Employment Cluster provides a container for Related Employment Attributes. A Related Employment, when dragged on to the evidence canvas, adds a cluster containing fields such as Participant and Employer. At run time, to create or modify an evidence record, the case worker will be shown a wizard, which mandates that the case worker select an employment record from a list before being able to create the Evidence record. Based on the employment record selection in the wizard, the Related Employment cluster displays its associated Participant Name and Employer Name in the second page of the wizard.

Table 27. Related Employment Cluster Properties

Property name	Description
Title	An optional Cluster Title for the default locale. Note that most Clusters have Titles, but they are not in fact enforced to be mandatory by the Dynamic Evidence infrastructure.
Title ID	A Resource property identifier for the Title property. Mandatory if a Title is provided (this property is only displayed in the Editor if the Title has one or more characters).
Description	An optional Cluster Description for the default locale. Cluster Descriptions, if provided, appear after the Title at runtime.

Table 27. Related Employment Cluster Properties (continued)

Property name	Description
Description ID	A Resource property identifier for the Description property. Mandatory if a Description is provided (this property is only displayed in the Editor if the Title has one or more characters).
Number of Columns	A mandatory numeric dropdown containing values from 1 to 4, the Number of Columns dictates how Columns will be used to lay out all contained Fields in this Related Employment Cluster in generated create, modify and view pages in respect of this Dynamic Evidence Type Version. Typically, this value is set to 2 (and sometimes 1) for most Dynamic Evidence Types.
Related Employment Attribute	The name of the Model Related Employment Attribute that this Field populates.
On Create Page	Non Editable property which defaults to true, which displays the Related Employment Cluster on the Case Evidence Create Page.
On Modify Page	Possible values are true or false. If true, the Related Employment Cluster is shown on the Case Evidence Modify page; if false, the Cluster is not displayed on the Modify page. In most cases this should be set to true (i.e. The checkbox should be checked).
On View Page	Possible values are true or false. If true, the Related Employment Cluster is shown on the Case Evidence View page; if false, the Cluster is not displayed on the View page. In most cases this should be set to true (i.e. The checkbox should be checked).
Online Help	An optional localizable text containing Cluster information for Online Help on Case Evidence screens.

Comments Clusters

Comments Attributes can be dragged from the accordion control on the User Interface Palette, and dropped over existing Clusters as Comments Clusters. Comments Clusters will be drawn as a cluster with a single standard multi-line text box.

Table 28. Comments Cluster Properties

Property name	Description
Title	An optional Cluster Title for the default locale. Note that most Clusters have Titles, but they are not in fact enforced to be mandatory by the Dynamic Evidence infrastructure.
Title ID	A Resource property identifier for the Title property. Mandatory if a Title is provided (this property is only displayed in the Editor if the Title has one or more characters).
Description	An optional Cluster Description for the default locale. Cluster Descriptions, if provided, appear after the Title at runtime.

Table 28. Comments Cluster Properties (continued)

Property name	Description
Description ID	A Resource property identifier for the Description property. Mandatory if a Description is provided (this property is only displayed in the Editor if the Title has one or more characters).
Number of Columns	A mandatory numeric dropdown containing values from 1 to 4, the Number of Columns dictates how Columns will be used to lay out all contained Fields in this Comments Cluster in generated create, modify and view pages in respect of this Dynamic Evidence Type Version. Typically, this value is set to 2 (and sometimes 1) for most Dynamic Evidence Types.
On Create Page	Possible values are true or false. If true the Comments Cluster is shown on the Case Evidence create page; if false, the Cluster is not displayed on the Create page. In most cases this should be set to true (i.e. The checkbox should be checked).
On Modify Page	Possible values are true or false. If true the Comments Cluster is shown on the Case Evidence modify page; if false, the Cluster is not displayed on the Modify page. In most cases this should be set to true (i.e. The checkbox should be checked).
On View Page	Possible values are true or false. If true the Comments Cluster is shown on the Case Evidence view page; if false, the Cluster is not displayed on the View page. In most cases this should be set to true (i.e. The checkbox should be checked).
Online Help	An optional localizable text containing Comments Cluster information for Online Help on Case Evidence screens.

Utility Fields

'Skip Field' and 'Label Field' is the only available Utility Field for administrators; these can be dragged from the User Interface Palette and dropped onto Attribute Clusters.

Skip Fields

The Skip Field causes a blank field to be drawn in the position specified in its location in the Attribute Cluster. Skip Fields allow for greater control over the layout of the fields in the containing Cluster. Skip Fields have no properties.

Label Fields

The Label Field causes text information to be drawn in the position specified in its location in the Attribute Cluster. Label Fields allows adding text information to dynamic evidence screens. It has the ability to show or hide the text information on create,modify or view dynamic evidence screens.

Following properties can be set in respect of Label Field.

Table 29. Label Field Properties

Property name	Description
Label	Field Label for the default locale is enforced as mandatory by the Dynamic Evidence infrastructure.

Table 29. Label Field Properties (continued)

Property name	Description
Label ID	A Resource property identifier for the Label Field property. Mandatory if a Label is provided (this property is only displayed in the Editor if the Label has one or more characters).
On Create Page	Possible values are true or false. If true, the Label Field in an Attribute Cluster is shown on the Case Evidence create page; if false, the Label Field is not displayed on the Create page. In most cases this should be set to true (i.e. The check box should be checked).
On Modify Page	Possible values are true or false. If true, the Label Field in a Attribute Cluster is shown on the Case Evidence modify page; if false, the Label Field is not displayed on the Modify page. In most cases this should be set to true (i.e. The check box should be checked).
On View Page	Possible values are true or false. If true, the Label Field in an Attribute Cluster is shown on the Case Evidence view page; if false, the Label Field is not displayed on the View page. In most cases this should be set to true (i.e. The check box should be checked).
Online Help	An optional localizable text containing label field information for Online Help on Case Evidence screens.

Configuring Multiple Participant Evidence Update

Administrators can configure the multiple participant evidence update via the Dynamic Evidence Editor .

The Dynamic Evidence Editor is used to enable the multiple participant evidence update. To configure the multiple participant evidence update, the developer must perform the following steps;

1. Set the participant in the Related CP Attribute drop down in the **General Properties** in the Dynamic Evidence Editor. For more information, refer to the **Related CP Attribute** entry in the "General Properties" on page 23 section

2. Deselect the **Show All Panels** in the related participant properties panel and select he **Enable Multi Case Member Updates** in the related participant properties panel. For more information refer to the **Enable Multi Case Member Updates** entry in the "Related Case Participant Clusters" on page 62 section.

Dynamic Evidence Rule Sets

You can use CER Rule Sets with Dynamic Evidence for a number of purposes.

Generated Rule Sets

Rule Sets written for Calculated Attributes, Summary Information and Validation are collectively referred to as containing Evidence Processing logic. These Rule Sets

need to access data pertaining to the relevant Dynamic Evidence Type as part of their logic. To facilitate this, for every Dynamic Evidence Type a Processing Rule Set is generated.

Rule Sets written for determining Eligibility and Entitlement for a particular Program are called Eligibility and Entitlement Determination Rule Sets. These Rule Classes need to access data from Dynamic Evidence Types which are involved in the determination. To facilitate Dynamic Evidence Types participating in Eligibility and Entitlement determination, a Rule Set called a Data Rule Set is also generated for each Dynamic Evidence Type

These Rule Sets are generated whenever changes to an "In-Edit" Evidence Type Version are saved (note that these Rule Sets are generated per-Dynamic Evidence Type, not per-Dynamic Evidence Type Version).

The following sections discuss the structure of these Generated Rule Sets in detail.

Processing Rule Sets

Processing Rule Sets are generated with a Rule Set name of `<Dynamic Evidence Type Logical Name>RuleSet`. The Rule Set contains a Rule Class whose name is `<Dynamic Evidence Type Logical Name>`, and is associated with the Dynamic Evidence: Processing category. An instance of this Rule Class represents a Case Evidence record of this Dynamic Evidence Type.

Data Attributes: A Rule Attribute is generated for each Data Attribute in the corresponding Dynamic Evidence Type Version. The name of the Rule Attribute is the same as that of the Data Attribute. The type of the Rule Attribute is dependent on that of the Data Attribute, but this is not a one to one Mapping. The following table summarizes the mapping from Data Attribute Types to Rule Attribute Types.

Table 30. Data Attribute Type to Rule Attribute Type Mapping

Data Attribute Type	Rule Attribute Type
Boolean	`java.lang.Boolean`
Date	`curam.util.type.Date`
Time	`curam.util.type.DateTime`
Integer	`java.lang.Number`
Float	`java.lang.Number`
Money	`java.lang.Number`
String	`java.lang.String`
Codetable	Inbuilt Codetable type in CER

Related Case Participant Attributes: Each Related Case Participant Attribute defined for a Dynamic Evidence Type Version results in the generation of two Rule Attributes in the Processing Rule Class. One Attribute is of type `java.lang.Number` and is named after the Related Case Participant Attribute.

The other Rule Attribute is generated to represent the Case Participant Role Object corresponding to the Related Case Participant Attribute. This Attribute is named `related_<Related Case Participant Attribute Name>`, and the type of this Attribute is specified as the `CaseParticipantRole` Rule Class in `CaseEntitiesRuleSet.`

Related Employment Attributes: Any Related Employment Attribute defined for a Dynamic Evidence Type Version results in the generation of two Rule Attributes in the Processing Rule Class. One Attribute is of type `java.lang.Number` and is named after the Related Employment Attribute.

The other Rule Attribute is generated to represent the Employment Object corresponding to the Related Employment Attribute. This Attribute is named `related_<Related Employment Attribute Name>`, and the type of this Attribute is specified as the `Employment` Rule Class in `ParticipantEntitiesRuleSet`.

Parent/Child Relationships: The Dynamic Evidence Editor supports the definition of Mandatory and Optional Parents in the Modeling section. As such, Relationships are defined in one of the Dynamic Evidence Type Versions of the Child Evidence Type. As soon as the metadata of the Child Evidence Type Version is saved, a Rule Attribute will be generated. This Rule Attribute can be used to navigate to the Parent Evidence records for a given Child Evidence Record. The name of this Rule Attribute will be the same as that of the Parent Relationship and the type will be `java.util.List<Parent Rule Class>` . In the case where a Dynamic Evidence Type has multiple Parent Types, an Attribute will be generated for each Parent Type.

The Rule Set of the Parent Evidence Type will be updated to contain a Rule Attribute to navigate to the Child Records for a given Parent Record. One such Attribute will be generated for each Child Evidence Type. The Attribute for a particular Child Evidence Type will be generated only when the Child Evidence Type Version containing the Parent definition is activated.

When parent and child relationship is established and both parent and child Evidence Type Versions are activated the system generates a relationship attribute in the parent processing rule class. This attribute is named **'related_<child Evidence Type logical name>'** and may be used in customized rulesets to navigate from a parent evidence record to its children. This attribute should not be referenced in customized rulesets before both parent and child Evidence Type Versions are activated. If used earlier it will cause a CER validation error when activating the parent or child version. This is not a defect but a recommended approach to developing rulesets for parent child Dynamic Evidence Types.

Address Attributes: Each Address attribute defined for a Dynamic Evidence Type will result in the generation of two Rule Attributes in the Processing Rule Class. One Attribute will be of type `java.lang.Number` and will be named after the Address attribute (this represents the ID of the Address).

Another Rule Attribute will be generated to represent the Address Object corresponding to the Address Attribute. This Attribute will be named `related_<Address Attribute Name`. The type will be specified as the Address Rule Class in `ParticipantEntitiesRuleSet`

Calculated Attributes: No Rule Attributes are generated for Calculated Attributes; Rule Attributes corresponding to Calculated Attributes will have to be defined in the Calculated Attributes Rule Set for this Dynamic Evidence Type Version.

Data Rule Sets

Data Rule Sets are generated with a Rule Set Name of `<DynamicEvidence Type Logical Name>DataRuleSet`. The Rule Set will contain a Rule Class whose name is `<Dynamic Evidence Type Logical Name>`, and will be associated with the Dynamic

Evidence: Data category. An instance of this Rule Class represents an Active Succession Set of Case Evidence Records corresponding to this Dynamic Evidence Type.

Business Dates: Business Start and End Dates are defined in the Modeling section of a Dynamic Evidence Type. These dates are used to define the temporal boundaries of a Succession Set. The period covered by these dates is called the *Evidence Business Object Lifetime.* The Editor allows for the mapping of Data Attributes of type `Date` to the Business Start and End Date Attributes of the Dynamic Evidence Type Version.

The Cúram Rule Object Propagation mechanism requires all Data Rule Sets to specify the Evidence Object Lifetime. For Dynamic Evidence Types, the generated Data Rule Set will contain the necessary elements to map the Business Start and End Dates defined in the Editor to the Evidence Lifetime.

Data Attributes: A Rule Attribute is generated for each Data Attribute in the corresponding Dynamic Evidence Type Version. The name of the Rule Attribute is the same as that of the Data Attribute. The type of the Rule Attribute is dependent on that of the Data Attribute, but this is not a one to one mapping. The table below summarizes the mapping from Data Attribute Types to Rule Attribute Types.

Table 31. Data Attribute Type to Rule Attribute Type Mapping

Data Attribute Type	Rule Attribute Type
Boolean	`java.lang.Boolean`
Date	`curam.util.type.Date`
Time	`curam.util.type.DateTime`
Integer	`java.lang.Number`
Float	`java.lang.Number`
Money	`java.lang.Number`
String	`java.lang.String`
Codetable	Inbuilt Codetable type in CER

Also, for Data Rule Sets the "Volatile" setting of the Data Attributes are also considered when determining the Data Type of a Rule Attribute. A Data Attribute which is marked as Volatile would have the type determined as described above, but inside a Timeline. This is to facilitate the Rule Object Propagation mechanism to create a Timeline of values while populating each Attribute.

Parent/Child Relationship: Similar to the Processing Rule Set, the Data Rule Set will also contain Attributes to navigate Parent/Child Relationships. However, this differs from the Processing Rule Set in two ways:
- The data type of the parent navigation Attributes will be set to the Data Rule Class for the Parent Evidence Type, rather than a list of Parent Data Rule Class Objects. This is because the Parent Data Rule Class represents a Succession Set and not a single record, and there can only be one Succession Set for each Parent Evidence Type.
- The data type of the child navigation Attribute will be a `java.util.List` of Data Rule Class objects for the Child Evidence Type, as multiple Succession Sets of Child Evidence Types can be related to one Parent Succession Set.
- When parent and child relationship is established and both parent and child Evidence Type Versions are activated the system generates a relationship

attribute in the parent processing rule class. This attribute is named `related_<child Evidence Type logical name>` and may be used in customized rulesets (e.g. eligibility and entitlement ruleset) to navigate from a parent evidence record to its children. This attribute should not be referenced in customized rulesets before both parent and child Evidence Type Versions are activated. If used earlier it will cause a CER validation error when activating the parent or child version. This is not a defect but a recommended approach to developing rulesets for parent child Dynamic Evidence Types.

Related Employment Attributes: The generation logic for Related Employment Attributes is similar to that for Processing Rule Sets.

Related Case Participant Attributes: The generation logic for Related Case Participant Attributes is similar to that for Processing Rule Sets, except that for Volatile Related Case Participant Attributes, the data type of both the Case Participant Role ID Attribute and the Case Participant Role Object Attribute will be wrapped inside a timeline.

Address Attributes: Address Attributes are again treated similar to how they are handled for Processing Rule Sets, with the difference that for Volatile Address Attributes, the generated Rule Attributes will have their data types wrapped inside a Timeline.

Calculated Attributes: Calculated Attributes do not have any corresponding elements generated in the Data Rule Set.

Propagator Configurations

The generated Data Rule Sets will be used to access Case Evidence data from Eligibility and Entitlement Determination Rule Sets. As such, the Data Rule Sets are useful only if Case Evidence data is propagated to them at runtime.

Rule Object Propagators are configured to specify the Rule Sets to which data in respect of each Evidence Type should be propagated. As part of activating a Dynamic Evidence Type, a Propagator Configuration is also generated in addition to the Rule Sets. This Propagator Configuration will be named, `<Data Rule Set Name> - Active Succession Set Propagator Configuration`, and will specify that Case Evidence records of this Dynamic Evidence Type should be propagated to instances of the generated Data Rule Class.

Support for Multiple Dynamic Evidence Type Versions

A Dynamic Evidence Type can have multiple Dynamic Evidence Type Versions over a period of time. In each Dynamic Evidence Type Version, new Attributes and Relationships could have been added. Also, existing Attributes and Relationships could have been dropped. When it comes to representing these variations in the generated Rule Sets, there are two possibilities:

- Generate one Rule Set that represents all Dynamic Evidence Type Versions (i.e. one Rule Set for each Dynamic Evidence Type)
- Generate one Rule Set for each Dynamic Evidence Type Version.

For a variety of reasons, the first option is being used for generating Rule Sets for Dynamic Evidence. As such, the generated Rule Set for a Dynamic Evidence Type will contain Rule Attributes corresponding to the sum total of all Attributes and Relationships in respect of all Dynamic Evidence Type Versions. For this reason, logic in Handcrafted Rule Sets for Summary Information/Validation/Calculated Attributes may need to branch based on the availability of a value for an Attribute

on the Evidence Record (Rule Object) being processed; this is because an Attribute may have been added from a particular point in time, and before this time it has no value.

Loading of Dynamic Evidence Rule Objects

Handcrafted Rule Sets for Evidence Processing are called from appropriate points during the maintenance of Case Evidence records in respect of Dynamic Evidence Types. For example, the Rule Set for Summary Information is called every time a Case Evidence record for a Dynamic Evidence Type is viewed. These Processing Rule Sets act on a particular Case Evidence record. Dynamic Evidence maintenance creates and populates a Rule Object with the details of the Case Evidence record for which the processing Rule Set is invoked.

When the Processing Rule set is invoked as part of reading a Case Evidence record, the Rule Object is populated with the data stored in the Database. When the Processing Rule set is invoked during creation or modification of a Case Evidence record, the corresponding Rule Object created is populated with the new or modified data.

The steps involved in loading a Dynamic Evidence Rule Object are as follows:
- The generated Rule Set or Rule Class for the Dynamic Evidence Type is identified. A Rule Object is created for this Rule Class.
- "evidenceID", "correctionSetID", "successionID", "caseID", "type", "receivedDate", "effectiveFrom" and "status" are attributes common to all Dynamic Evidence Types. These are populated from the Evidence Descriptor.
- Each Rule Attribute corresponding to the Data Attributes are populated with values from the database, or the value specified by the user if the Case Evidence record is being created or modified
- If the Case Evidence record is being created:
  - "evidenceID", "correctionSetID" and "successionID" will all be set to zero
  - Rule Attributes for navigating to Parent Records will be populated with the Parent Records selected as part of the create process. This is because the normal derivation of these Rule Attributes would try to read the Parent records from the database and the Relationship would not have been established yet, as the Case Evidence record is yet to be created. Also, Rule Attributes corresponding to Child Evidence Types would be populated with an empty List for this Case Evidence instance.
  - The Rule Attribute for Address attributes will have a value of zero. The Rule Attribute corresponding to the Address object will be populated with the Address Details entered during creation.
  - The Rule Attribute for Related Case Participant Attributes will have a value of zero, if an existing Case Participant is not chosen by the case worker. The Rule Attribute corresponding to the Related Case Participant object would be populated with the Details entered during creation.

Utility Rule Classes and Functions

Some utility java static operations are provided to help with the writing of Dynamic Evidence Processing Rule Sets. These Java operations are contained in the `curam.dynamicevidence.cer.impl.DynamicEvidenceStatics` class. The following section provides a high level description of these operations. Please refer the Javadoc for Dynamic Evidence to obtain more detailed information such as parameters and return types of each operation.

Listing Parent Rule Objects

For each Parent Dynamic Evidence Type, a Rule Attribute will be available in the generated Processing Rule Class. This attribute will provide a list of all Parent Records. The operations provided in the `DynamicEvidenceStatics` class to get a filtered list of Parent Evidence records are:

- `getActiveParentList()` - This operation returns a list of Parent Records which are Active.

- `getActiveandPendingChangesParentList()` - This Operation returns a list of In-Edit and Active Parent Records which do not have any pending update. For example, if there are three records R1, R2 and R3 where R1 is active, R2 is active and R3 is an In-Edit correction on R2, this operation would return R1 and R3.

- `getParent()` - This operation returns a Parent Record for the given Dynamic Evidence. Consider the following rules snippet which illustrates the usage:

```
<Attribute name="relatedMember">
    <type>
      <ruleclass
        name="Enrollment"
        ruleset="EnrollmentRuleSet"
      />
    </type>
    <derivation>
      <call
        class="curam.dynamicevidence.cer.impl.DynamicEvidenceStatics"
        method="getParent"
      >
        <type>
          <ruleclass
            name="Enrollment"
            ruleset="EnrollmentRuleSet"
          />
        </type>
        <arguments>
          <this/>
          <String value="Enrollment"/>
        </arguments>
      </call>
    </derivation>
</Attribute>
```

In this example, the getParent method retrieves the rule Object representation of Enrollment evidence type which is used as a value to the relatedMember attribute.

Listing Child Rule Objects

For each child Dynamic Evidence Type, a Rule Attribute will be available in the generated Processing Rule Class. This Attribute will provide a list of all Child Records, irrespective of their state. The operations provided in the `DynamicEvidenceStatics` class to get a filtered list of Child Evidence records are:

- `getActiveChildList()` - This operation returns a list of those Child Records which are Active.

- `getActiveandPendingChangesChildList()` - This Operation returns a list of In-Edit and active Child Records which do not have any pending update.

- `getChildList()` - This operation returns a list of Child Records for the given Dynamic Evidence. Consider the following rules snippet which illustrates the usage:

```
<Attribute name="relatedEmployerSponsoredCoverage">
    <type>
      <javaclass name="java.util.List">
        <ruleclass
```

```
                name="EmployerSponsoredCoverage"
                ruleset="EmployerSponsoredCoverageRuleSet"
             />
           </javaclass>
        </type>
        <derivation>
          <call
            class="curam.dynamicevidence.cer.impl.DynamicEvidenceStatics"
            method="getChildList"
          >
            <type>
              <javaclass name="java.util.List">
                <ruleclass
                  name="EmployerSponsoredCoverage"
                  ruleset="EmployerSponsoredCoverageRuleSet"
                />
              </javaclass>
            </type>
            <arguments>
              <this/>
              <String value="EmployerSponsoredCoverage"/>
            </arguments>
          </call>
        </derivation>
    </Attribute>
```

In this example, the getChildList method retrieves the list of
EmployerSponsoredCoverage evidence type which is used as a value to the
relatedEmployerSponsoredCoverage attribute.

Listing Rule Objects for a Particular Dynamic Evidence Type

Sometimes, it is necessary to obtain a list of Rule Objects for a Dynamic Evidence
type for a particular Case. For example, a Validation Rule Set might get a list of
Rule Objects of a particular Dynamic Evidence Type on a particular Case, and
check them against the Case Evidence being validated to perform duplicate checks.
Three relevant operations are provided for this:

- getEvidenceListForCase() - Returns a list of Rule Objects pertaining to Case
  Evidence Records in respect of a particular Dynamic Evidence Type for a
  particular Case.
- etActiveEvidenceListForCase() - Returns a list of Rule Objects pertaining to
  Active Case Evidence Records in respect of a particular Dynamic Evidence Type
  for a particular Case.
- getActiveAndPendingChangesEvidenceListForCase() - Returns a list of Rule
  Objects pertaining to Active and In-Edit Case Evidence records with no pending
  updates for a particular Dynamic Evidence Type for a particular Case.

Attribute Availability

It is possible for a Processing Rule Set to be used for more than one Dynamic
Evidence Type Versions. For example, the same Validation Rule Set can be used for
many Versions of an Evidence Type. In such a case, the Validation Rule Set will
have to factor the structural changes across different Versions into its logic.

For example, Version 1 of an Evidence Type might have "authorizedExpense" and
"actualExpense" as two Attributes. In version 2, the "actualExpense" Attribute
might have been replaced by a Child Evidence Type called "Expense". In this case,
the total actual expense should be calculated by summing the expense value from
all Child Records of Type "Expense". If there is a validation which checks the
actual expense amount against the authorized expense amount, and the same
Validation Rule set is being used for both Version 1 and 2, the validation logic

should first check for the existence of the "actualExpense" Attribute. This is required because the generated Rule Set contains Rule Attributes corresponding to both versions 1 and 2..

The operation `isAttributeAvailable()` can be used to check if a particular Attribute is available in a particular Evidence record. So for this example, the Validation Rule Set can check if the "actualExpense" attribute is available in the record being validated. If not, the logic can calculate the actual expense by adding up the expense from "Expense" child records.

Specific Dynamic Evidence Rule Set Types

This section discusses each of the usages of CER Rule Sets in Dynamic Evidence in turn, namely:

- Summary Information Rule Sets
- Validation Rule Sets
- Calculated Attributes Rule Sets
- Eligibility and Entitlement Rule Sets

For each of these types of Dynamic Evidence Rule Set, we discuss the contract expected of the Rule Set and how to author them.

Summary Information Rule Sets

Most of the time it should be possible to use Summary Mapping functionality for defining Summary Information for a Dynamic Evidence Type Version. However, sometimes the Summary Information may not be a straight forward mapping to Attributes of the Dynamic Evidence Type Version. For example. the Summary Information might need data from a Parent or Child Evidence Record. In such cases, CER Rule Sets can be used to define the Summary Information.

An administrator can specify the Rule Set to be used by enabling the **Use Rule Set** radio button, and then specifying the Rule Set using the **Rule Set Name** option.

Contract: Dynamic Evidence has certain expectations for the required structure of Summary Rule Sets in terms of the Rule Classes and Attributes that they should contain. Having said this, the generic CER infrastructure does not have any Evidence Processing concepts. So in order to make sure handcrafted Summary Rule Sets meet these expectations, the following restrictions are enforced on them:

- Summary Rule Sets must contain one Concrete Rule Class which extends from `DefaultEvidenceSummary` Rule Class in the EvidenceSummaryRuleSet.
- This Rule Class must contain a Rule Attribute named "evidence". The type of this Rule Attribute should be the generated Processing Rule Class, and the derivation of this Attribute should use the specified expression. The specified expression is the default derivation when an Attribute is defined in the CER Editor. So, for example, if the logical name of a Dynamic Evidence Type is "Alien", a Rule Set named "AlienRuleSet" with a Rule Class named "Alien" would be generated. In this case, the concrete Rule Class in the Summary Rule Set for this Dynamic Evidence Type must contain an Attribute named "evidence" whose type is the "Alien" Rule Class in the "AlienRuleSet".

Authoring: The `DefaultEvidenceSummary` Rule Class has the following Attributes corresponding to the different elements of Summary Information.

Table 32. DefaultEvidenceSummary Attributes

Rule Attribute	Type	Value
startDate	Date	null
endDate	Date	null
isStartDateAvailable	Boolean	true
isEndDateAvailable	Boolean	true
summary	String	null
participantDetails	String	null

The idea behind defining these Attributes in the DefaultEvidenceSummary Rule Class is twofold:

- In their Summary Rule Sets, administrators need to define only those Attributes which need a different value from those defined in DefaultEvidenceSummary.
- In a future major release of Cúram, if a new element is added to the Summary Information infrastructure, a corresponding Attribute will be added to DefaultEvidenceSummary . So any existing hand-crafted Summary Rule Sets would not need to be changed to include this new Attribute, as long as the default value provided in DefaultEvidenceSummary is applicable.

Authoring a Summary Rule Set thus involves re-defining (overriding in Object-Oriented terminology) only those Attributes whose values should be different from those defined in DefaultEvidenceSummary.

The values from the startDate and endDate Attributes are used in the "Period" column of the Dynamic Evidence Type Workspace page only if isStartDateAvailable and isEndDateAvailable have a value of true . Otherwise, the Business Start Date and Business End Date options defined in the *Modeling* section is used to calculate the Evidence Workspace "Period" column. As such, if the intention is not to set startDate and endDate as part of a Summary Information Rule Set, then the Rule Set should s et isStartDateAvailable and isEndDateAvailable to false.

Typically, the logic to derive different elements of Summary Information is based on the data from the corresponding Case Evidence instance. This is why the contract for Summary Rule Sets specify that they should have an Attribute named "evidence" -this Attribute is populated with data from the corresponding Case Evidence instance whenever the Rule Set is invoked.

Dynamic Evidence Administration provides support for authoring Summary Rule Sets by generating a "starter" Rule Set, if the Summary Information Rule Set specified does not exist already. The following sections talk about the process of authoring a Summary Rule Set by using such a "starter" Rule Set, or by reusing an existing Summary Rule Set.

Using a Starter Rule Set

If the Summary Rule Set specified for a Dynamic Evidence Type Version does not exist, a "starter" Rule Set with the given name is generated. The generated starter Rule Set will have a class called "SummaryInformation" with the base Rule Class and the "evidence" Attribute mentioned in the previous section. This Rule Set will be associated with the Dynamic Evidence Summary Information category.

The Rule Set should be further edited by the administrator to define an Attribute corresponding to each element of Summary Information for which the value defined in `DefaultEvidenceSummary` should be overridden. When the Dynamic Evidence Type Version is activated, the Summary Rule Set is also activated.

Using an Existing Rule Set

Sometimes, an existing Rule Set is specified as the Rule Set to be used for Summary Information. Typically, this happens when a new Dynamic Evidence Type Version is created and the previous Dynamic Evidence Type Version already had a Rule Set specified for Summary Information.

In this case, the existing Rule Set is not modified automatically. This Rule Set needs to be modified only if the logic to compute the Summary elements needs to change. A point to note is that if an existing Summary Rule Set is modified, the changes will be visible to all the Dynamic Evidence Type Versions which have been using this Rule Set. So, if the changes to the Summary Information derivation is required for only the new Dynamic Evidence Type Version, a new Summary Rule Set should be used instead of modifying an existing Rule Set.

Validation Rule Sets

Most of the time it should be possible to use Standard Validations for defining Validations for an Dynamic Evidence Type Version. But, sometimes the validations may be a complex combination of many conditions or might involve calculations based on data from related Parent or Child Case Evidence instances. In such cases, CER Rule Sets can be used to define the Validations.

An administrator may specify the Rule Set to be used by enabling the **Use Rule Set** radio button under **Additional Validations** in the **Validations** Tab of the Evidence Properties, and then specifying the Rule Set using the **Rule Set Name** option.

Contract: As with Summary Information, Dynamic Evidence has certain expectations for the required structure of Validation Rule Sets in terms of the Rule Classes and Attributes that they should contain. Having said this, the generic CER infrastructure does not have any Evidence Processing concepts. So in order to make sure handcrafted Validation Rule Sets meet these expectations, the following restrictions are enforced on them:

- Validation Rule Sets should contain one concrete Rule Class which extends from the `DefaultEvidenceValidationResult` Rule Class in the EvidenceValidationRuleSet.
- This Rule Class must contain a Rule Attribute named "evidence". The type of this Rule Attribute should be the generated Processing Rule Class, and the derivation of this Attribute should use the "specified" expression. The "specified" expression is the default derivation when an Attribute is defined in the CER Editor.

  So, for example, if the logical name of a Dynamic Evidence Type is "Alien", a Rule Set named "AlienRuleSet" with a Rule Class named "Alien" will be generated. In this case, the concrete Rule Class in the Validation Rule Set for this Evidence Type must contain an Attribute named "evidence" whose type is the "Alien" Rule Class in the "AlienRuleSet".

Infrastructure Rule Classes: The EvidenceValidationRuleSet provides several Rule Classes which are to be used when authoring Validation Rule Sets:

Validation

The `Validation` Rule Class represents a particular Validation. It contains the following Attributes:

Table 33. `Validation` Rule Class - Attributes

Name	Type	Description
isFailure	Boolean	Indicates if the Validation has failed for the given Case Evidence record.
failureMessage	curam.creole.value.Message	The message to be shown to the user if the Validation fails.
informationalType	curam.dynamicevidence .validation. impl.InformationalType	Specifies if a Validation failure should be reported as a warning, error or a fatal error. By default, this is set to `error`. A warning will not stop the user action from getting completed. For example, if a Validation fails during creation of a Case Evidence record, but the informational type is `warning`, the Case Evidence record will get created in the database and a warning message will be displayed to the user. However, if the informational type is `error` or `fatal error`, any database changes as part of the user action will be rolled back. In addition, a fatal error will stop the validation process immediately. The informational type of a Validation should be specified as `fatal error`, if it is considered that proceeding with subsequent Validations is not useful if this Validation fails.

ValidationMode

The `ValidationMode` Rule Class represents the operation as part of which the Validation is being invoked. This can be used to determine the set of Validations to be applied and the informational type to be used for a particular Validation.

This Rule Class has a single Attribute, namely `mode` of type `curam.dynamicevidence.validation.impl.ValidationMode`.

DefaultEvidenceValidationResult

As explained in the contract, Validation Rule Sets must have a Rule Class that extends from `DefaultEvidenceValidationResult` . This Rule Class has the following Attributes; all of these Rule Attributes are of type List<Validation>

Table 34. `DefaultEvidenceValidationResult` Attributes

Rule Attribute	Purpose	Value
detailsValidations	These Validations are called before writing a new or modified Case Evidence record to the database. When these Validations are called as part of creating a new Case Evidence record, Parent/Child Relationship will not have been established. As such, any Validation that involves navigating to Parent Case Evidence records should not be included as part of detailsValidations . Typically these Validations contain Single Field or Cross Field validations.	null
standardValidations	These Validations are called after a Case Evidence record is created, modified, activated or as part of validating a Case Evidence Record. Typically this contains Validations which involve navigating to other Case Evidence records such as Parent/Child records or Case Evidence instances of the same Type on the Case.	null
preCreateValidations	This set of Validations is called before a Case Evidence record and any entity related to that Case Evidence record are created on the Database. The related entity could be an Address or a Case Participant Role.	null
preModifyValidations	This set of Validations is called before a Case Evidence record and any entity related to that Case Evidence record are modified on the Database. The related entity could be an Address or a Case Participant Role.	null
postCreateValidations	As the name suggests, this set of Validations is called after a Case Evidence record and any entity related to that Case Evidence record are created on the Database. The related entity could be an Address or a Case Participant Role.	null

Table 34. `DefaultEvidenceValidationResult` Attributes (continued)

Rule Attribute	Purpose	Value
postModifyValidations	As the name suggests, this set of Validations is called after a Case Evidence record and any entity related to that Case Evidence record are modified on the Database. The related entity could be an Address or a Case Participant Role.	null

In addition to the preceding Attributes, this Rule Class also contains a `validationMode` Attribute.

The idea behind defining these Attributes in the `DefaultEvidenceValidationResult` Rule Class is twofold:

- In their Validation Rule Sets, administrators need to define only those Attributes which correspond to the sets of Validations required for the particular Dynamic Evidence Type Version.
- In a future major release of Cúram, if a new set of Validations is added, a corresponding Attribute would be added to `DefaultEvidenceValidationResult`. As such, any existing hand-crafted Validation Rule Sets would not need to be changed to include this new Attribute, as long as the new set of Validations is not required.

Authoring: Authoring a Validation Rule Set involves the following steps:

Defining the Rule Set

If the Rule Set specified in the "Additional Validations" section does not exist, the Dynamic Evidence infrastructure will generate a "starter" Rule Set. The generated starter Rule Set will have a class called "ValidationResult" with the base Rule Class and an "evidence" Attribute as mentioned in the *Validation Rule Sets Contract* section, and will be associated with the Dynamic Evidence Validation category.

Sometimes, an existing Rule Set is specified as the Rule Set to be used for Validation. Typically, this happens when a new Dynamic Evidence Type Version is created and the previous Dynamic Evidence Type Version already had a Rule Set specified for Validation. If an existing Rule Set is used, it should adhere to the Validation Rule Sets contract. A point to note is that if an existing Validation Rule Set is modified, the changes would be visible to all Dynamic Evidence Type Versions which have been using this Rule Set.

Defining Validation Sets

As described in the section about `DefaultEvidenceValidationResult` Rule Class, this Rule Class has Attributes corresponding to different sets of Validations. All these Attributes have a value of null , which means by default there are no Validations defined in any of the Validation sets. For a particular Dynamic Evidence Type, all or only some of the Validation Sets might be required. For example. a Dynamic Evidence Type Version might require only "detailsValidations" and "standardValidations". As such, the Attributes corresponding to Validation sets should be redefined. Typically, a CER Fixed List is used to construct the list of Validations in a set.

Defining a Validation

Typically, a Validation will refer to data from the Case Evidence record or related Case Evidence records - this is why the contract for Validation Rule Sets specify that they should have an Attribute named "evidence". This Attribute is populated with data from the corresponding Case Evidence instance whenever the Rule Set is invoked.

A Validation can be defined by using the CER create expression. The create expression when called for the `Validation` Rule Class requires that a derivation be specified for the isFailure Attribute and failureMessage Attribute.

The derivation for the isFailure Attribute must produce a boolean value. For example, let us say there are two Attributes, "startDate" and "endDate" in a Dynamic Evidence Type, and the Validation required is that the "startDate" should not be the same as the "endDate". The derivation for the isFailure Attribute of this Validation can use a CER compare expression which compares the "startDate" and "endDate" Attributes. The compare expression will obtain the value of these two Attributes by using a reference expression on the "evidence" Attribute of the Validation Rule Class.

The derivation for the failureMessage Attribute should produce a value of type `curam.creole.value.Message` . For this, the derivation can use an XML Message or a Resource Message expression. Again, if the message is to be parameterized with data from the Case Evidence record, the required Attribute can be accessed through the "evidence" Attribute of the Validation Rule Class using the reference expression.

Optionally, a value can be specified for the "informationalType" of the Validation. If no value is specified, the informational type will be considered in error. However, the static operations (error() , warning() and fatalError()) in `curam.dynamicevidence.validation.impl.InformationalType` can be used to specify a different informational type.

Typically, the informational type is dependent on the Validation Mode. The Validation mode for a particular Validation session can be accessed through the "validationMode" Attribute of the Validation Result Rule Class. The validation mode can then be compared with one or more Validation Modes using the static operations (applyChanges() , approve() , insert() , modify() , validateChanges()) defined in `curam.dynamicevidence.validation.impl.ValidationMode` to determine the informational type to be used. For example, the Validation Mode for a Validation Session can be compared with "Apply Changes" and "Validate Changes" and the informational type can be specified as error for the former and warning for the latter case.

Calculated Attributes Rule Sets

Calculated Attributes are Attributes whose value is derived from manipulating values of other Attributes of the same Dynamic Evidence Type or related Dynamic Evidence Types. These Calculated Attributes are typically used in the Case Evidence View pages in respect of a Dynamic Evidence Type, but they could be used for other purposes as well such as in Validations. For example, a Dynamic Evidence Type for Adoption may have Case Participant Role Ids for the Parent and Child; Calculated Attributes could be defined to calculate the names of Parent and Child from the respective Case Participant Role IDs.

For each Dynamic Evidence Type Version, a Calculated Attributes Rule Set must be defined if the Dynamic Evidence Type Version has Calculated Attributes in its model. The following section describes the process of writing a CER Rule Set for Calculated Attributes.

Contract: As with Other Dynamic Evidence Rule Set types, Dynamic Evidence has certain expectations for the required structure of Calculated Attributes Rule Sets in terms of the Rule Classes and Attributes that they should contain. As such, the following restrictions are enforced on such Rule Sets:

- The Calculated Attributes Rule Set must contain one Concrete Rule Class which extends from the DefaultCalculatedAttributes Rule Class in the EvidenceCalculatedAttributesRuleSet Rule Set.
- This Rule Class must contain a Rule Attribute named "evidence". The type of this Rule Attribute must be the generated Processing Rule Class and the derivation of this Attribute should use the "specified" expression. The "specified" expression is the default derivation when an Attribute is defined in the CER Editor. So, for example, if the logical name of a Dynamic Evidence Type is "Alien", a Rule Set named "AlienRuleSet" with a Rule Class named "Alien" will be generated. As such, the concrete Rule Class in the Calculated Rule Set for this Dynamic Evidence Type must contain an Attribute named "evidence" whose type is the "Alien" Rule Class in the "AlienRuleSet".
- This Rule Class must contain a Rule Attribute corresponding to each Calculated Attribute defined in the Dynamic Evidence Type Version. The name and type of these Rule Attributes must match those of the corresponding Calculated Attribute.

Authoring: While editing the metadata for a Dynamic Evidence Type Version, a Calculated Attributes Rule Set should be specified using the **Calculated Attributes Rule Set Name** option in the Evidence Properties Panel in the Evidence Editor.

The logic to compute the value of a Calculated Attribute is typically based on the data from the relevant Case Evidence record or from Case Evidence records related to it, and this is why there is a requirement to have an evidence Rule Attribute in Calculated Attribute Rule Sets. When a Calculated Attribute Rule Set is invoked during Case Evidence maintenance, this Attribute will be populated with a Rule Object containing the data for the Case Evidence record for which the Calculated Attributes have to be computed.

Using a Starter Rule Set

If the specified Rule Set does not exist, a starter Rule Set with the given name will be generated. The generated starter Rule Set will have a class called `CalculatedAttributes` which has the base Rule Class and the evidence Attribute mentioned in the previous section, and will be associated with the Dynamic Evidence Calculated Attributes category.

This Rule Set should be further edited by the administrator to define Attributes corresponding to each Calculated Attribute. When the Dynamic Evidence Type Version is activated, the Calculated Attributes Rule Set is also activated.

Using an Existing Rule Set

Sometimes, an existing Rule Set is specified as the Rule Set to be used for Calculated Attributes. Typically, this happens when a new Dynamic Evidence Type

Version is created and the previous Dynamic Evidence Type Version already had a Rule Set specified for Calculated Attributes.

In this case, the existing Rule Set is not modified automatically. If no new Calculated Attributes Rule Set has been added and the logic to compute the Calculated Attributes does not need to change, then the existing Rule Set can be used. If an additional Calculated Attribute is added to the new Dynamic Evidence Type Version, a corresponding Rule Attribute would have to defined in the existing Calculated Attributes Rule Set.

Eligibility and Entitlement Rule Sets

The Eligibility and Entitlement Determination Rule Sets are typically architected in three layers.

- `Data Rule Classes`

  Data Rule Classes are the closest to Case Evidence Data. They mirror the structure of the Case Evidence Data i.e. they contain Rule Attributes corresponding to Attributes and relationships of a Dynamic Evidence Type. Rule Objects are created for these Rule Classes whenever Case Evidence records in respect of the corresponding Dynamic Evidence Types are created or modified.

- `Calculator Rule Classes`

  Calculator Rule Classes contain calculations which represent more coarse-grained business concepts than those represented by the data elements for Dynamic Evidence Types. For example, Dynamic Evidence Types representing the different kinds of income and the composition of a Household may exist; A Calculator Rule Class might calculate the Net Household Income by summing up the Income of all the Household members, after applying legislation to exclude those incomes that should not be counted.

  Generally, such business concepts are factored in to separate Calculator Rule Classes, so that they can be reused across many Programs. Irrespective of the reuse, factoring big chunks of Rules in to many Calculator classes make the Rule Sets modular and readable.

- `Program Rule Classes`

  Program Rule Classes use Data and Calculator Rule Classes to determine the eligibility for a particular Program. These Rule Classes need to exhibit certain characteristics in terms of their structure, so that the Eligibility and Entitlement Engine can work with them to create Determinations. For more details on developing Eligibility and Entitlement Rules, please refer to the *Working with Cúram Express Rules* guide.

Of these three, Data Rule Classes are generated for all Dynamic Evidence Types. So, for custom Programs, Eligibility and Entitlements Rules can be written by developing Program Rules that make use of these generated Dynamic Evidence Data Rule Classes. Alternatively, Program Rules can be written to make use of a layer of Calculator Rule Classes which in turn make use of the generated Data Rule Classes..

Localizing dynamic evidence

A description of the artifacts that are involved in the localization process for Dynamic Evidence, and the steps that are necessary to localize them.

Localizing process overview

The process of localizing Dynamic Evidence Types is not dissimilar to that of localizing other parts of Cúram; the artefacts and mechanisms used are broadly the

same, and it is only the resource locations and naming conventions that are specific to Dynamic Evidence. The reader should therefore be familiar with the localization process described in the *Cúram Web Client Reference Manual*.

Most of the following steps involve localizing properties resources in the Application Resource Store. The administration pages for the Resource Store can be accessed in the following manner:

- Log in as an administrative user (e.g. Admin.
- In the Shortcuts panel, expand the **Intelligent Evidence Gathering** section.
- In this section, click on **Application Resources**.

Localizing the Dynamic Evidence Editor Properties Resource in Administration Suite

The Dynamic Evidence Editor uses a properties resource called "DynEvd_EvidenceEditor.properties" to specify values such as labels, text, tool tip text, labels on button, items in dropdowns, messages etc. in the Dynamic Evidence Editor. This properties file contains key/value pairs which are in UTF-8 format. To support localization, locale specific values have to be provided for the keys.

- Using the Application Resource Store administration screens, download the version of the properties resource (`DynEvd_EvidenceEditor.properties`) for the default locale.
- Using a text editor, change the value of each property to the appropriate localized equivalent.
- Re-upload the modified properties and publish the changes.
- To view the changes made to properties , launch the Dynamic Evidence Editor. To do this:
  – In the Actions button in the list item, select 'Edit Metadata'

Localizing the Runtime Dynamic Evidence User Interface

This section discusses artefacts to be used when localizing case worker runtime Case Evidence screens in respect of Dynamic Evidence Types (i.e. create, modify and view screens).

Static Properties Resource

Dynamic Evidence uses a properties resource called `DynEvdStaticProperties.properties` to store various strings common to all Dynamic Evidence Types - such as the labels for 'Save' and 'Cancel' buttons, and standard messages used in Related Case Participant Clusters. This file must be localized for each desired locale.

To achieve this:

- Using the Application Resource Store administration screens, download the version of `DynEvdStaticProperties.properties` for the default locale.
- Using a text editor, change the value of each property to the appropriate localized equivalent.
- Back in the Administration Suite, create a new Application Resource called `DynEvdStaticProperties.properties` for the desired locale with the newly localized file as the content.

This localization step needs to be performed only once.

Evidence Type Properties Resources

Dynamic Evidence also uses individual properties resources to store Evidence Type specific localizable text. There is one such resource per Dynamic Evidence Type and it currently stores only the Evidence Type localizable runtime description. The description property is used in Evidence Maintenance to provide a description for the Dynamic Evidence Type (for example, on the 'New Evidence' screen accessed from the Evidence Dashboard). The naming convention for these properties resources is "DynEvd_EvidenceType_" followed by the logical name of the Dynamic Evidence Type and ended with ".properties" . They must be localized for each Evidence Type and each supported user locale.

To achieve this:
- Using the Application Resource Store administration screens, download the version of the properties resource (
  `DynEvd_EvidenceType_<logicalName>.properties`) for the default locale.
- Using a text editor, change the value of each property to the appropriate localized equivalent.
- Back in the Administration Suite, create a new Application Resource for the desired locale with the newly localized file as the content.

A new Evidence Type properties resource is created when a new Dynamic Evidence Type is created and it is deleted when the Evidence Type is deleted.

Evidence Type Version Properties Resources

When editing a Dynamic Evidence Type Version using the Dynamic Evidence Editor, a number of localizable strings are specified in the form of labels and titles for User Interface elements. When the Evidence Type Version is activated, these localizable strings are also stored in properties resources in the Application Resource Store. The naming convention for these properties resources is `DynEvd` followed by the logical name of the Dynamic Evidence Type and then a numeric form of the Evidence Type Version effective date.

The strings provided via the Dynamic Evidence Editor are used to populate the properties resource for the default locale. This means that if a localized version is not available in a locale-specific properties resource, then the string from the default locale resource will be used via the standard fallback mechanism.

This properties resource must be localized for each Evidence Type Version and locale you wish to support. The localization process is as described in "Static Properties Resource" on page 86.

Generation of the Localized User Interface

It is important to point out that the properties resources mentioned in "Evidence Type Version Properties Resources" are not directly referred to by the Dynamic Evidence Case Management screens at runtime. Rather, they function as the input to other properties files which are generated by the Dynamic Evidence infrastructure.

The actual properties files used at runtime are generated the first time a user accesses the page for a particular locale. Therefore it is important to note that changing the source properties files after this point will not result in any change to the user interface. Ideally, all localization activity should be completed before case workers actually start accessing the Dynamic Evidence screens at runtime.

Message Files

Cúram message files must be localized for each locale to be supported; this process does not differ for Dynamic Evidence.

Codetable Items

Each Dynamic Evidence Type has a codetable entry generated on the **EvidenceType** codetable. Its description must be correctly localized, as it is used to generate various parts of the User Interface (such as titles for evidence maintenance pages).

Customizing dynamic evidence

Use the following information for the steps to use when customizing default Dynamic Evidence Types. You can also use this information when upgrading to new versions of Dynamic Evidence Types where they were customized as part of a project implementation.

This product contains a number of components which ship with predefined Dynamic Evidence Types. Typically, customers will want to modify either the structure or behavior of some of these types to fit their specific business requirements, before deploying them into a production environment.

Customization Configuration Prerequisite

To enable Dynamic Evidence customization the following property must be enabled from the System Administration application:

`curam.dynamicevidence.evidencetype.customisation.enabled`

This property, when set, will cause a 'customized Indicator' to be set for Dynamic Evidence Type Versions created or modified by customers; this indicator will be shown in the Dynamic Evidence Type Versions list page in the Administration Suite. Cúram-shipped Dynamic Evidence Type Versions will not have this flag set. As such, it will be possible for customers to see at a glance which Dynamic Evidence has been customized and which has not.

Customization Process

The following sequence of steps is the recommended process for developing Dynamic Evidence.

1. Evidence Gap Analysis.
2. Evidence Definition
3. Extract Evidence
4. Source Control Management
5. Pre-Production system
6. Runtime Activation

Step 1 Evidence Gap Analysis

The first step is to work out the Evidence requirements for a project. Describing this exercise is beyond the scope of this document, but it will be based upon program requirements, administrative and legislative policy, and other business needs. Once the evidence requirements have been identified, these must be mapped onto Cúram-shipped Dynamic Evidence Types.

In some cases, this will result in a requirement to create completely new, project-specific Dynamic Evidence Types to implement the data requirements of their programs. In other cases, customers will want to change or augment Cúram-supplied Dynamic Evidence Types.

In either case, Dynamic Evidence Type specifications must be defined using the Dynamic Evidence Administration pages and the Dynamic Evidence Editor; See "Dynamic Evidence Types" on page 8 and "Dynamic Evidence Type Versions" on page 11 for more details.

Step 2 Evidence Definition

During this step, the Dynamic Evidence Administration pages are used to either create new Dynamic Evidence Types, or to customize existing Cúram-supplied ones.

Create new, project-specific Dynamic Evidence Types and Versions

In this situation, a new Evidence Type is created and, using the Dynamic Evidence Editor, metadata for its Evidence Type Version is created as outlined in the section on "Dynamic Evidence Type Versions" on page 11.

Note that it is important that appropriate `curam.dynamicevidence.type.code.`prefix and DYNEVDCODE settings are selected for the new Dynamic Evidence Types. This will ensure that there will be no conflict in the Evidence Type Codetable entries for the newly defined Dynamic Evidence Types. These properties are described in more detail in the section on the "Dynamic Evidence Configuration Extractor" on page 93

Reuse or Modify Cúram-shipped Dynamic Evidence Type Versions

In some situations, a Cúram-shipped Dynamic Evidence Type Version (for a particular Dynamic Evidence Type) might match customer requirements exactly, or need a small modification to meet the requirements. In such cases, you must clone the Cúram-shipped Active Dynamic Evidence Type Version with the **New InEdit Copy** action, even if no change is required. This process might seem counter-intuitive for situations where no changes are needed, but it is important to ensure that your usage of these Dynamic Evidence Type Versions are unaffected by future updates.

The effective date of Cúram-shipped Dynamic Evidence Types is normally set to a date in the distant past, for example 1st January 1900. You must assign an effective date that is appropriate for the business requirements of the program that is being implemented to the newly cloned Dynamic Evidence Type Version.

If a Cúram-shipped Dynamic Evidence Type is updated by IBM, it is always done with the release of a new Dynamic Evidence Type Version. That is, previously released Dynamic Evidence Type Versions are not changed. The Cúram-shipped Dynamic Evidence Type Version effective date is incremented by one day from the previous version. The new effective date ensures that customers can always analyze the changes and decide whether to apply them to their customized versions.

Where a Cúram-supplied Dynamic Evidence Type is used in a customer program, such as a Product Delivery or Integrated Case, you must cancel all Dynamic Evidence Type Versions that are not required. This cancellation ensures that no Evidence Records are inadvertently recorded against them in a runtime

environment. While they exist on a product system, a case worker can potentially try to enter an Evidence Record for a received date that overlaps with the Cúram-shipped Dynamic Evidence Type Version, resulting in undesired behavior.

Step 3 Extract Evidence

When all Dynamic Evidence Types have been defined, most customers will at this stage want to preserve the state of these Dynamic Evidence Types in their Software Configuration Management system. The Dynamic Evidence Extractor provides this functionality (see "Dynamic Evidence Configuration Extractor" on page 93 for more details).

Note: That it is not a completely mandatory requirement – it is possible to create Dynamic Evidence Types in a Testing or Staging environment and transport them into a production environment without this step – but for the majority of customers, this will be required.

If a customer does extract their Dynamic Evidence Types, they have to manage the database primary keys and use keys from designated key ranges. If not configured, the data base will just create arbitrary Primary Keys which could result in Primary Key conflicts.

The recommended approach to manage PKs for Dynamic Evidence is to use the Range Aware Key Server (RAKS). This mechanism was developed to support Cúram Configuration Transport Manager and is documented in the Cúram Business Object Module Development Guide. It involves enabling all entities being extracted by the extractor to use RAKS and administration configuration of the RAK server. The benefit of RAKS is that once the system is configured it is guaranteed to generate the correct PKs for newly created records. Dynamic Evidence Type artifacts can safely be extracted as is, without the risk of Primary Key conflicts.

Step 4 Source Control Management

At this point the customer will have a component folder containing all new and customized Dynamic Evidence Types. Once the custom component containing the extracted artifacts has been included in the server component order (Please see the IBM Cúram Server Developer's Guide for more details about component orders), rebuilding the Cúram database will import all extracted artifacts in the Cúram system. The contents of the custom component folder can now be placed under source code control.

Step 5 Pre-Production system

All Dynamic Evidence Types can now be transported from the Source system to the production system using the Cúram Configuration Transport Manager.

Attempting to transport modified Dynamic Evidence Type Versions where data already exists in respect of them in the target system is not supported. Such modifications would require a new Evidence Type Version where there is no existing live Evidence Record data. Note that all Dynamic Evidence type versions must be in an Active state prior to transportation.

For more details please see the IBM Cúram Configuration Transport Manager Guide.

Step 6 Runtime Activation

The final step in the process is to activate all transported Dynamic Evidence Types in the production system so that they are available for use.

Dynamic Evidence Timelines

A lot of Dynamic Evidence processing hinges on key Evidence dates in ways that are not immediately obvious. Use this information to understand the logic behind how Dynamic Evidence works with dates: how dates entered by administrators affect case workers maintaining Evidence and vice versa

Note in particular the Evidence framework concept that a piece of Case Evidence may change over time, giving rise to a succession of different versions of the Case Evidence record that each apply to a particular period of time. Understanding the difference between correcting a Case Evidence record and recording a change in circumstances is essential.

Dates in Dynamic Evidence Administration

Non-Dynamic Evidence does not have the concept of Evidence Type Versioning present in Dynamic Evidence. With Non-Dynamic Evidence, every piece of Case Evidence recorded over time has the same data structure and the same user interface. This can cause problems if, for instance, legislation changes to require an additional piece of Evidence Data to be recorded. An entirely new Evidence Type would need to be created, and code and user interface modified to take this into account; alternatively, the existing Evidence Type would need to be modified and either data migration would be required to populate the new field for existing Case Evidence records, or code would need to take into account the fact that the attribute would not be present for some Case Evidence records. Either way, it is awkward.

Dynamic Evidence, however, includes the concept of Dynamic Evidence Type Versions, allowing the structure of Case Evidence to change over time. A Dynamic Evidence Type might have only one Dynamic Evidence Type Version or it might have several, forming a timeline of modifications to the structure of the Dynamic Evidence Type.

The date on which this hinges is the Effective Date field of the Dynamic Evidence Type Version. The timeline begins with the Dynamic Evidence Type Version with the earliest Effective Date and extends to infinity in the future (there is no end date). The period of time during which each Dynamic Evidence Type Version is active extends to the day before the effective date of the next Dynamic Evidence Type Version in the timeline.

There are restrictions on what you can and cannot do when administering Dynamic Evidence Type Versions. One restriction is that you can only change the effective date on a Dynamic Evidence Type Version when it is in an 'InEdit' status, not when it is 'Active'.

You cannot create a Dynamic Evidence Type Version with an Effective Date earlier than an existing Dynamic Evidence Type Version, because this could render processing of existing Case Evidence records ambiguous or incorrect. Likewise, if you are creating a new Dynamic Evidence Type Version to supersede an existing one, there should be no Case Evidence records for the Evidence Type for the period that the new Dynamic Evidence Type Version would cover.

Note that there is currently no administrative way to correct a live Dynamic Evidence Type Version that has existing Case Evidence records, and so it is important to ensure that Dynamic Evidence Type Versions are fully tested before live Case Evidence records are entered in respect of them on a production system.

Dates in Runtime Case Evidence Maintenance

This section describes the dates used in runtime Case Evidence maintenance and how they relate to the Dynamic Evidence Type Version Effective Dates. In the maintenance of Case Evidence records in respect of Dynamic Evidence Types (and indeed Non-Dynamic Evidence Types) the crucial dates are the Received Date and the Effective Date of Change.

Creating Case Evidence records

All Evidence create pages (for Dynamic and Non-Dynamic Evidence Types) have a Received Date field which defaults to the current date. The received date marks the start of the period for which the Case Evidence record will be active, and is a mandatory field.

For Non-Dynamic Evidence Types, case workers can (unless custom validations preclude it) enter any date they choose as the received date. Dynamic Evidence however imposes some limitations on this. One such limitation is that the Received Date may not be before the Effective Date of the earliest version of the Dynamic Evidence Type. In effect, a Case Evidence record cannot be valid before the Dynamic Evidence Type itself is valid, which makes sense.

If several Versions of the Dynamic Evidence Type for which the Case Evidence is being created exist, how does the system know which one is being created, and hence which user interface to present to the user? The system will always initially present the user interface for creating the Version of the Dynamic Evidence Type that is Active on the current date.

If the user, while creating the Case Evidence record, modifies the Received Date so that it falls in a time period where a different Dynamic Evidence Type Version is Active, then the system will redirect the user to the appropriate user interface for the correct Dynamic Evidence Type Version. The data that the user has already entered will be pre-populated on the page to which the user is redirected.

Therefore, the Received Date field is effectively the key to determining the Dynamic Evidence Type Version in respect of which the Case Evidence record is created. The caseworker user should not have to concern themselves with this - they simply consider the business meaning of the date and the system will present them with the appropriate user interface for creating the Case Evidence structure that applies at that point in time.

Modifying In-Edit Case Evidence Records

Modifying a Case Evidence record with a status of 'InEdit' is really just an extension of the create process. The record will be changed in-place, but no new member of the Case Evidence record's succession set will be created.

The Received Date is present on the modify page too, and if it is changed it may result in the user being redirected to another modify page to finish editing the record, as described for the Create page. This will happen when the user attempts to save the Case Evidence record. In this way InEdit Case evidence records can be modified to belong to a different Dynamic Evidence Type Version (although in practice, this is an edge case which should happen very rarely).

The modify page header also has a field for another important business date - Effective Date of Change - however this does not become valid until the Case Evidence record has been Activated, and attempting to change it will result in an error being displayed. This behaviour has been retained is identical to that in respect of Non-Dynamic Evidence Types..

Modifying Active Case Evidence Records

As described in the *Cúram Evidence Guide* , there are two ways in which a user can modify an active Case Evidence record: correction of incorrect data, and recording a change in a Participant's circumstances. This mechanism hinges on the Effective Date of Change field.

The Effective Date of Change field, in business terms, records the start of a time period for which a version of the Case Evidence record is valid. For instance, if the piece of Case Evidence is a record of a Participant's employment, what happens if the client gets promoted? It is a continuation of their previous employment, so it does not require an entirely new employment record. The record of the client's new job role succeeds the previous record, which remains valid up until the point when the client got their promotion. That point in time is captured by the Effective Date of Change.

If the Effective Date of Change is entered, then the system treats the modification of the Active evidence as a change in circumstances. A new Case Evidence record is created as part of the same succession set, and its Active period commences at the Effective Date of Change. For Dynamic Evidence Types, If the Dynamic Evidence Type Version that applies for the Effective Date of Change is not the same as that which applied for the previous version of the evidence record then, as before, the user will be redirected to the appropriate user interface for the applicable Dynamic Evidence Type Version.

Conclusion

Caseworker users should be able to use different Dynamic Evidence Type Versions in an intuitive manner based on the same business dates that are used with Non-Dynamic Evidence Types. Administrators simply need to be aware of the impact that the effective dates for the different Versions of Dynamic Evidence Types have on the user interfaces presented to case workers.

Dynamic Evidence Configuration Extractor

The Dynamic Evidence Configuration Extractor tool extracts dynamic evidence configuration information from a runtime Cúram database, and writes it to the file system using standard Cúram development artifacts, such as DMX, CTX, XML blobs and clobs, tab configuration files, and so on.

Overview

One of the primary goals of Dynamic Evidence is to provide an administrative-time alternative to traditionally developed Non-Dynamic Evidence. Within this was another configuration goal, namely that of generating as many Cúram artefacts as possible to reduce the complexity and number of artefacts that Cúram Administrators need to define. Such artefacts include Dynamic UIM Pages (for Create, Read, Modify, etc.), user interface Tab configurations, CER Rule Sets, Property files, Security Identifiers and Groups, CER Propagators, etc.

Initially this was targeted at customers who did not have development installations of Cúram, in that after creation or maintenance of Dynamic Evidence Types, the

database would be the 'system of record' for these administrated and generated artefacts; database backups would ensure that this data was persisted.

However, for customers with Cúram development projects, it was necessary to locate and export all such generated artefacts so that they could be source code controlled. Typically the requirement would be to save such configurable artefacts as DMX, CTX, clobs, blobs, etc. using the Cúram Data Manager. This way once the database was recreated (a typical development-time activity in Cúram), any Dynamic Evidence Types would be persisted and would not have to be recreated every time.

To assist in this effort, a Dynamic Evidence Configuration Extractor tool has been provided in Cúram version 6.0 SP2. This tool extracts Dynamic Evidence configuration information from a runtime Cúram database, and writes it to the file system using standard Cúram development artefacts (DMX, CTX, XML blobs and clobs, Tab configuration files, etc.).

Features

Launched using the Cúram batch launcher, the main features of this tool are as follows:

- The tool extracts Dynamic Evidence configuration information and stores it in standard Cúram development artefacts (e.g. DMX, CTX, XML blob/clob and section configuration files). This is done so that that these artefacts can be automatically recreated as part of a database rebuild using the existing Cúram data manager.
- The tool writes its output to a single directory and it expects this to be a standard Cúram component directory (e.g. `custom`). For example, within this directory it creates subdirectories such as `codetable`, `data`, and `tab`.
- Extracting database records with their generated primary keys poses a potential risk for key clashes when the records are uploaded. The reason for this is that when the database is rebuilt, the key generation mechanism is reset and it is very likely to produce the same keys as those in the extracted Dynamic Evidence Types. To avoid this, the extractor replaces generated database primary keys with new keys from a pre-defined key range. The key range is applied to each extracted table individually rather than being shared across all tables (this way keys are used more efficiently). Only two tables: `CreoleRuleset` and `CreoleRulesetEditAction` share the same key range, because both tables are referenced from the same field in `CreoleRulesetCategoryLink` table.
- Tab configuration files (for generated Evidence tabs) are extracted as blobs, as part of extracting the `AppResource` entity. Section configuration files, however, cannot be extracted as individual blobs. They are extracted as contribution section files (and placed in the `tab` folder), so they can be merged with other component-specific section files on a database build.
- The tool optionally extracts Dynamic Evidence links to Application, Product and Integrated Cases, configurable via the extractor input parameters. Note that the extractor does not extract Product or Integrated Case configuration information, only the links to them.
- The extractor implements three extraction strategies (the one to be used is determined by the input parameters):
  - *Extract all Dynamic Evidence Types*: All active Dynamic Evidence Types on the system are extracted.

- *Extract a list of Dynamic Evidence Types:* This strategy allows users to specify a list of Dynamic Evidence Types to extract (using a list of Evidence Type logical names).
- *Extract a set of Dynamic Evidence Types identified by Evidence Type code prefix:* Dynamic Evidence Type codes are generated using a customizable three character prefix e.g. `DET`. This extraction strategy allows users to only extract Dynamic Evidence Types that use a specific code prefix.

- Dynamic Evidence Types have localizable descriptions. Prior to Cúram V6.0 SP2 the descriptions for all Dynamic Evidence Types on the system were stored in a single properties file (`DynEvd_EvidenceTypeDescriptions.properties`) in the `AppResource` entity. In Cúram V6.0 SP2 this mechanism was changed to store Dynamic Evidence Type descriptions in individual properties resources, one per Evidence Type. If the extractor finds the old evidence descriptions properties file, it will split it into individual Evidence Type specific properties files.

- The extractor extracts the key set (DYNEVDCODE) used to generate Dynamic Evidence Type codes (preserving the Next Unique Block ID for this key set). Dynamic Evidence uses the Cúram key server ability to generate human readable keys. This is used to generate Dynamic Evidence Type codetable codes. When the database is reset, the key generation mechanism is also reset and there is a likelihood of producing keys that clash with previously generated ones. This is best avoided by preserving the state of the Dynamic Evidence key set used to generate Evidence Type codetable codes.

- The extractor extracts both Active and In Edit Evidence Type versions.

- A number of rulesets related to Evidence Type Versions are extracted: generated data and processing rulesets, and custom (calculated attributes, validations, summary information) rulesets. Custom rulesets can be edited by customers. Newly created custom rulesets are extracted by the tool (together with In Edit Evidence Type Versions). However, any changes made to published custom rulesets will not be picked up by the extractor until the changes are published.

- The extractor does not extract localizable resource bundles referenced from rulesets. Rulesets can include localizable resource messages which are stored in resource bundles in the `AppResource` entity. If users choose to use such messages in customized rulesets they have to handle the extraction of containing resource bundles manually.

- The extractor can either be run from the command line or the Eclipse development environment.

- Before uploading extracted artefacts back to the database, codetables have to be re-generated (via the **ctgen** target or server build) to include the extracted Dynamic Evidence Type codetable codes.

Running the Extractor

Before running the extractor users may have to customize the curam.dynamicevidence.generated.business.object.tabs.sections curam.dynamicevidence.generated.business.object.tabs.sections application property. It can be found in the **Dynamic Evidence - configuration** section of the administration application. This property is used to specify a list of sections to which generated Dynamic Evidence tabs are added. The list must contain a comma separated list of section IDs. This list is used to generate section contribution (`.sec`) configuration files (written to the `tab` folder). The default section list is:

DefaultAppSection,SUPERAPPSection,AUDITORAPPSection,AUDITCOAPPSection,
INVESTRAPPSection,FINAPPSection

The extractor can be run in two ways:

- Running the extractor from the command line:
  - Set SERVER_DIR and CURAMSDEJ variables if not already set. Depending on the project setup, PRE_CLASSPATH variable may have to be set to include any referenced libraries.
  - From the EJBServer folder call
    ./components/DynamicEvidence/script/
    extractDynamicEvidenceConfiguration.bat file passing all input parameters using the -D option, for example:
    ./components/DynamicEvidence/script/
    extractDynamicEvidenceConfiguration -Dcomponent=MyComponent
    -DlowerKey=20000 -DupperKey=20999
    -DetExtractionList="etLogicalName1;etLogicalName2;etLogicalName3"
  - Alternatively from the EJBServer folder call
    components/DynamicEvidence/script/
    extractDynamicEvidenceConfiguration.xml ant script using the -D option to pass input parameters, for example:
    ant -f ./components/DynamicEvidence/script/
    extractDynamicEvidenceConfiguration.xml -Dcomponent=MyComponent -
    DlowerKey=20000 -DupperKey=20999 —DextractAll=true
- Running the extractor from the Eclipse development environment:
  - In Eclipse create a new Java Application Run Configuration.
  - Specify the main class to be curam.util.impl.BatchLauncher.
  - Two program arguments must be specified (in the **Arguments** tab of the run configuration set up): the first argument is the batch operation to be run by the batch launcher, the second argument is the list of all input parameters to the extractor:
    curam.dynamicevidence.sl.util.configextractor.intf.
    DynamicEvidenceConfiguration
    Extractor.extractConfigArtefacts
    serverDir=D:\CC\DynamicProductWS\DynamicProductMain
    EJBServer,component=
    MyComponent,lowerKey=20000,upperKey=20999,etCodePrefix=DET

    The class path will have to be set to include all referenced libraries (depending on the project setup).

    Note the *serverDir* parameter. It contains the location of the EJBServer folder. The list of input parameters is comma separated and should not contain any spaces.

Extractor Input Parameters

When running the extractor from the command line, all input parameters should be passed using the -D option, i.e. **-D<parameter name>=<parameter value>.**

Mandatory Parameters:

- *component* - the name of the destination component folder where the extractor output is written (e.g. 'custom'). If this folder does not exist in SERVER_DIR/components the extractor will report an error.
- *serverDir* - this parameter is automatically set (from the SERVER_DIR variable) when running the extractor from the command line. It must be specified if running the extractor from Eclipse, and defines the location of the 'EJBServer' folder.

- *lowerKey* - specifies the end or the starting key for the key range used to generate replacement primary keys for extracted database records.
- *upperKey* - specifies the starting or the end key for the key range used to generate replacement primary keys for extracted database records.

Parameters defining the Evidence Type extraction strategy, i.e. which evidence types to extract. At least one of these parameters has to be specified or the extractor will report an error:

- *extractAll* - if this parameter is provided with a value of 'true', all Active Dynamic Evidence Types on the system (in the `EvidenceTypeDef` table) will be extracted. This parameter overrides the other extraction strategy parameters. If it is absent or its value is not 'true' the subsequent parameters will be considered.
- *etExtractionList* - specifies a list of Dynamic Evidence Types to be extracted. The list must contain one or more Evidence Type logical names separated by semicolons (no white space allowed). This parameter overrides the *etCodePrefix* parameter.
- *etCodePrefix* - specifies a Dynamic Evidence Type code prefix, e.g. 'DET'. Dynamic Evidence Types with codes beginning with the specified prefix will be extracted. The code prefix is a customizable application property (curam.dynamicevidence.type.code.prefix in Dynamic Evidence – located in the configuration section of the administration application).

Optional Parameters:

- *extractProductLinks* - enables extraction of Dynamic Evidence Type links to Products. If set to true the relevant records from `ProductEvidenceTypeDefLink` entity will be extracted.
- *extractICLinks* - enables extraction of Dynamic Evidence Type links to Integrated Cases. If set to true the relevant records from `AdminICEvidenceTypeDefLink` entity will be extracted.
- *extractCaseLinks* - enables extraction of Dynamic Evidence Type links to Application Cases. If set to true the relevant records from `CaseConfigEvidenceLink` entity will be extracted.
- *datamanagerDir* - name of the directory where datamanger (DMX, blob, clob) files are to be written. This directory is located in the destination component directory and it is automatically created if it does not exist. The default directory name is 'data'.
- *dmxDir* - name of the directory where DMX files are written. This directory lives in the *datamanagerDir* directory and it is automatically created if it does not exist. The default directory name is 'initial'. Inside this folder, the tool creates directories for blob and clob files.
- *codetableDir* - name of the directory where codetable (CTX) files are written. This directory is located in the destination component folder and it is created automatically if it does not exist. The default directory name is 'codetable'.
- *tabDir* - name of the directory where section configuration (SEC) files are written. This directory is located in the destination component folder and it is created automatically if it does not exist. The default directory name is 'tab'.
- *preserveRangeKeys* - This parameter specifies a list of Range Keys. Primary keys of extracted database records will remain unchanged if the key falls inside one of the rangeKeys specified in this property. If not withing any range specified the primary key will be generated based on the `upperKey` and `lowerKey` properties specified. Specify as follows:- -DpreserveRangeKeys="20000-20999,23000-23999"

- *java.extra.vmargs* - This parameter helps specify the arguments to the Java virtual machine. This helps resolve any occurrence of Out of memory runtime errors. Example : To set PermGen space while running the extractor, specify as follows:--Djava.extra.vmargs="-XX:PermSize=128m"

Extracted Artefacts

Codetables:

Codetables are extracted and saved in files with `ctx` extensions. The files are written into a folder located in the destination component (see `codetableDir` input parameter). Each codetable is extracted into a separate file. Two codetables are extracted:

- `EvidenceType`
- `TemporalEvTypeApproval`

Entities (Database Tables):

Entities are extracted and saved in files with `dmx` extension. There is one such file per entity. These files are written to the data manager folder located in the destination component (see *datamanagerDir* and *dmxDir* input parameters). Blob and clob records are extracted into external files and referenced from DMX files. Blob and clob files are saved into separate folders (named `blob` and `clob`) located inside the `dmxDir` folder.

- `EvidenceTypeDef`: this entity stores the Dynamic Evidence Types on the system and it is the first to be extracted. The Evidence Type extraction strategies mentioned earlier determine what records to extract from this entity. If no records can be extracted from `EvidenceTypeDef` the extraction process terminates.
- `EvidenceTypeVersionDef`: extraction of this entity is dependent on `EvidenceTypeDef`. It stores multiple metadata Versions for Dynamic Evidence Types. Active and In Edit Versions are extracted for each extracted record (Evidence Type) from `EvidenceTypeDef`.
- `EvidenceTypeDefinition`: maps an Evidence Type to an Evidence Nature. Extraction is dependent on `EvidenceTypeDef` entity.
- *SecurityGroup*: contains security group definitions on the system. There is an auto generated Security Group for each active Dynamic Evidence type, hence this entity is dependent on `EvidenceTypeDef` in the extraction process. Users have access to Security Groups via the System Administration application. If auto-generated Dynamic Evidence Security groups are modified or deleted they will not be picked up by the extractor. Additionally, Security Identifiers and Security Group SID mappings will not be extracted for modified groups.
- `SecurityIdentifier`: this entity depends on `SecurityGroup` and `SecurityGroupSid` in the extraction process. Only Security Identifiers linked to extracted security groups are extracted.
- `SecurityGroupSid`: links Security Identifiers to Security Groups. If any of the extracted Security Identifiers are linked to the EVIDENCEGROUP security group, these links are also extracted.
- `AppResource`: this entity stores miscellaneous information in blob fields. It depends on `EvidenceTypeDef` and `EvidenceTypeVersionDef` in the extraction process. The following artefacts are extracted:
  - Evidence Type localizable descriptions: extracted as properties files in the `blob` folder, one per Dynamic Evidence Type.
  - Evidence Type Version localizable properties: extracted as properties files in the `blob` folder, one per Active Evidence Type Version.

- Tab configuration files: extracted as XML files in the `blob` folder. Up to three files are extracted per Active Evidence Type Version (tab, menu, navigation configuration files).
- The previously used single evidence descriptions properties resource (`DynEvd_EvidenceTypeDescriptions.properties`), if found, is split into individual (one per Evidence Type) properties files and stored in the blob folder.

- `CreoleRuleset`: this entity stores published rulesets and it is dependent on `EvidenceTypeDef` and `EvidenceTypeVersionDef` entities. Data and processing rulesets are determined from extracted `EvidenceTypeDef` records, while custom defined rulesets (calculated attributes, validation and summary information) are determined from extracted active `EvidenceTypeVersionDef` records.
- `CreoleRulesetEditAction`: stores In Edit rulesets and has dependency on `EvidenceTypeDef` and `EvidenceTypeVersionDef` entities. Data and processing rulesets are determined from extracted `EvidenceTypeDef` records, while custom defined rulesets are determined from extracted In Edit `EvidenceTypeVersionDef` records.
- `CreoleRulesetCategoryLink`: links rulesets to categories. It is dependent on both `CreoleRuleset` and `CreouleRulesetEditAction` in the extraction process
- `EvidenceRulesetDef`: this entity maps an Evidence Type to In Edit data and processing rulesets and to a Rule Object Propagator Configuration.
- `RuleObjectPropagatorConfig`: stores Propagator configurations, one for each Active Dynamic Evidence Type. `EvidenceRulesetDef` links this entity to Dynamic Evidence Types (`EvidenceTypeDef`).
- `LocalizableText`: this entity links Propagator configurations to their localizable descriptions. It has a dependency on `RuleObjectPropagatorConfig` for extraction.
- `TextTranslation`: stores Propagator configuration descriptions and depends on `LocalizableText` entity for extraction.
- `ProductEvidenceTypeDefLink`: this entity links Dynamic Evidence Types to Products and it is optionally extracted. It depends on `EvidenceTypeDef` entity in the extraction process.
- `AdminICEvidenceTypeDefLink`: this entity links Dynamic Evidence Types to Integrated Cases and it is optionally extracted. It depends on `EvidenceTypeDef` entity in the extraction process.
- `CaseConfigEvidenceLink`: this entity links Dynamic Evidence Types to Application Cases and it is optionally extracted. It depends on `EvidenceTypeDef` entity in the extraction process.
- `KeyServer`: only a single record is extracted from this entity. This is the key set (DYNEVDCODE) used to generate Dynamic Evidence Type codes.

Section Configuration Files:

Section configurations are stored in the `AppResource` entity. Each section configuration may refer to multiple components which makes extraction via DMX files impossible. Instead, section configurations are extracted into section contribution files (with extension `sec`) and saved in the `tab` folder (see *tabDir* input parameter).

Dynamic Evidence Metadata Loader

The Dynamic Evidence Metadata Loader allows you to import and export Dynamic Evidence configurable metadata.

Overview

One of the configuration goal of Dynamic Evidence is to provide an administrative-time to reduce complexity in developing and maintaining the artefacts that Cúram Administrators need to define. Initially this was targeted at customers who did not have development installations of Cúram, in that after creation or maintenance of Dynamic Evidence Type Versions, the database would be the system of record for these administrated and generated artefacts; database backups would ensure that this data was persisted.

As discussed previously, Dynamic Evidence Type can evolve over time. For example, a change in legislation may require that a new Evidence attribute must now be recorded, starting from a specified date. Dynamic Evidence supports this requirement by using Dynamic Evidence Type Versions to record modifications to metadata over time. As Metadata evolves, It was necessary for customers with Cúram development projects to import and export the configurable Metadata using Cúram Data Manager. This way once the database was recreated (a typical development-time activity in Cúram), any Dynamic Evidence Type Versions would be persisted and would not have to be recreated every time.

To assist in this effort, a Dynamic Evidence Metadata Loader tool has been provided in Cúram version 6.0.5.0. This tool provides two features:

Download: This tool downloads Dynamic Evidence Metadata information of an effective date from a run time Cúram database, and writes Metadata information in an XML file to the file system.

Upload: This tool uploads Dynamic Evidence Metadata information for an effective date from a file system and updated a record to a run time Cúram database.

Features

Launched using the Cúram batch launcher, the main features of this tool are as follows:

- The tool downloads the Metadata information of a Dynamic Evidence Type Version and writes it output as XML file to the file system with a prefix as as "<EVIDENCE_TYPE_LOGICAL_NAME_EFFECTIVE_DATE> .
- The tool uploads the XML file that contains Metadata information from the file system by reading its output as blob and update the Dynamic Evidence Type Version record to a run time Cúram database.
- User are allowed to download and upload Dynamic Evidence Metadata only one at a time.
- The upload and download utility tool implements three strategies (the one to be used is determined by the input parameters):
  - *Location:* Name of the location in the file system.
  - *Evidence Type:* This strategy allows users to specify a logical name of the Dynamic Evidence Type.
  - *Effective Date:* This strategy allows the user to specify the Effective Date.
- The utility downloads both Active, In Edit and Canceled Evidence Type versions.
- The utility uploads the Metadata information only for an In Edit Evidence Type Versions.
- It is the responsible for the user to ensure that the downloaded or imported Metadata do not break the data consistency within the Cúram setup.

- The download and upload process can either be run from the command line or the Eclipse development environment. The rationale is that these downloaded Metadata can be updated and can be imported back with new context in to the database using the upload tool.

Running the Downloader

The Downloader can be run in two ways:

- Running the Downloader from the command line:
  - From the EJBServer folder call **./components/DynamicEvidence/script/ downloadDynamicEvidenceMetadata.bat** file passing all input parameters using the -D option, for example:

    ./components/DynamicEvidence/script/downloadDynamicEvidenceMetadata -Dlocation="DirectoryLocation" -DevidenceType="etLogicalName" -DeffectiveDate="etvEffectiveDate"

  - Alternatively from the 'EJBServer' folder call **components/DynamicEvidence/ script/downloadDynamicEvidenceMetadata.xml** ant script using the -D option to pass input parameters, for example:

    ant -f ./components/DynamicEvidence/script/ downloadDynamicEvidenceMetadata.xml -Dlocation="DirectoryLocation" -DevidenceType="etLogicalName" -DeffectiveDate="etvEffectiveDate"

- Running the Downloader from the Eclipse development environment:
  - In Eclipse create a new Java Application Run Configuration.
  - Specify the main class to be curam.util.impl.BatchLauncher.
  - Two program arguments must be specified (in the **'Arguments'** tab of the run configuration set up): the first argument is the batch operation to be run by the batch launcher, the second argument is the list of all input parameters to the extractor:

    curam.dynamicevidence.sl.util.metadataloader.intf.

    DynamicEvidenceMetadataDownloader.downloadMetadata

    location=DirectoryLocation,evidenceType=etLogicalName,

    effectiveDate=etvEffectiveDate

    The class path will have to be set to include all referenced libraries (depending on the project setup).

Downloader Input Parameters

When running the Downloader from the command line, all input parameters should be passed using the -D option, i.e. **-D<parameter name>=<parameter value>.**

Mandatory Parameters:

- *location* - the name of the destination folder where the Downloader output is written. If this folder does not exist or does not have privilege to create the folder in the file system the extractor will report an error.
- *evidenceType* - specifies the logical name of the Dynamic Evidence Type to be extracted.
- *effectiveDate* - specifies the effective date of a Dynamic Evidence Type Version to be extracted.

Running the Uploader

The Uploader can be run in two ways:

- Running the Uploader from the command line:

- From the EJBServer folder call **./components/DynamicEvidence/script/ uploadDynamicEvidenceMetadata.bat** file passing all input parameters using the -D option, for example:

  ./components/DynamicEvidence/script/uploadDynamicEvidenceMetadata -Dlocation="XML File Location" -DevidenceType="etLogicalName" -DeffectiveDate="etvEffectiveDate"

- Alternatively from the EJBServer folder call **components/DynamicEvidence/ script/uploadDynamicEvidenceMetadata.xml** ant script using the -D option to pass input parameters, for example:

  ant -f ./components/DynamicEvidence/script/ uploadDynamicEvidenceMetadata.xml -Dlocation="XML File Location" -DevidenceType="etLogicalName" -DeffectiveDate="etvEffectiveDate"

- Running the upload from the Eclipse development environment:
  - In Eclipse create a new Java Application Run Configuration.
  - Specify the main class to be curam.util.impl.BatchLauncher.
  - Two program arguments must be specified (in the **Arguments** tab of the run configuration set up): the first argument is the batch operation to be run by the batch launcher, the second argument is the list of all input parameters to the extractor:

    curam.dynamicevidence.sl.util.metadataloader.

    intf.DynamicEvidenceMetadataUploader.uploadMetadata

    location=XML File
    Location,evidenceType=etLogicalName,effectiveDate=etvEffectiveDate

    The class path will have to be set to include all referenced libraries (depending on the project setup).

Uploader Input Parameters

When running the Uploader from the command line, all input parameters should be passed using the -D option, i.e. **-D<parameter name>=<parameter value>.**

Mandatory Parameters:

- *location* - the location of the XML file that contains metadata information in the file system.If this file does not exist or does not have valid contents the Uploader will report an error.
- *evidenceType* - specifies the logical name of the Dynamic Evidence Type to be uploaded.
- *effectiveDate* - specifies the effective date of a Dynamic Evidence Type Version to be uploaded.

Generated Artefacts

Dynamic Evidence generates a number of artifacts automatically as Dynamic Evidence Types are administered. Use this information to understand what these artifacts are, when they are generated, and which actions you can take on them.

Introduction

It is important for administrators to be aware of these artefacts because they appear in the Cúram administration screens and because removing or altering them may result in runtime problems with the operation of Dynamic Evidence Types.

Administrators should also be aware that the deletion of a Dynamic Evidence Type will also remove these generated artefacts. This could have impact for code or configurations which depend on generated Dynamic Evidence artefacts.

EvidenceType Codetable Entries

When a new Dynamic Evidence Type is created, a Codetable entry is automatically added to the **EvidenceType** Codetable. The generated Codetable code will begin with the value of the curam.dynamicevidence.type.code.prefix Cúram environment variable, and then a generated unique number will be appended. The Codetable item description will be the name of the new Dynamic Evidence Type as entered by the administrator on the Dynamic Evidence Type Create page. The locale will be the server locale.

The Codetable code prefix is limited to three characters and its default value is *DET*.

Evidence Type Codetable entries will be removed if the Dynamic Evidence Type is cancelled.

warning: Auto-generated Codetable entries are recorded against the server locale, but displayed for the locale of the currently logged in user. It is highly recommended that Administrator users configuring Dynamic Evidence operate in the same locale as the server. In multi locale deployments this will prevent localization issues that may occur before auto-generated Codetable items are translated for all supported locales.

Properties Resources for Evidence Types

Each Dynamic Evidence Type has a Property Application Resource containing the localizable Evidence Type description property. This resource is created when the Evidence Type is created. The property resource name is made up of "DynEvd_EvidenceType_" followed by the logical name of the Dynamic Evidence Type and ended with ".properties" extension (e.g. `DynEvd_EvidenceType_<logicalName>.properties`).

Security Identifiers and Security Groups

Each Dynamic Evidence Type is associated with a Security Group, the name of which is specified by the administrator when the Dynamic Evidence Type is created. This Security Group will be added automatically.

Security Identifiers (SIDs) are created for Case Evidence maintenance operations when Dynamic Evidence Types are created. Three SIDs are generated for each Dynamic Evidence Type: one each for create, modify and read of Case Evidence in respect of the Dynamic Evidence Type.

These are named "DynEvd.create.*EvidenceTypeGroupName*", "DynEvd.modify.*EvidenceTypeGroupName*", and "DynEvd.read.*EvidenceTypeGroupName*", where *EvidenceTypeGroupName* should be replaced with the name of the Security Group the administrator chose for the Dynamic Evidence Type.

The generated SIDs are automatically added to both the Security Group specified by the administrator, and to the EVIDENCEGROUP Security Group, if it is exists.

Generated SIDs and Security Groups are removed if a Dynamic Evidence Type is cancelled.

CER Rule Sets

On activation of each new Dynamic Evidence Type Version several CER Rule Sets are generated. The generated Rule Sets will be removed if the Dynamic Evidence Type Version is cancelled.

Administrators should refer to "Dynamic Evidence Rule Sets" on page 69 for more information on generated Rule Sets.

Propagator Configuration

Propagator Configurations are generated corresponding to the generated Data Rule Sets. Administrators should see "Dynamic Evidence Rule Sets" on page 69 for more information on dealing with generated Propagator Configurations.

Properties Resources for Evidence Type Versions

A Property Application Resource is generated on activation for each Dynamic Evidence Type Version containing all localizable strings associated with its user interface. The naming convention for these is "DynEvd_" followed by the logical name of the Dynamic Evidence Type and then the Effective Date of the Dynamic Evidence Type Version. This Property Application Resource is generated in the default locale and is created on activation of the Dynamic Evidence Type Version. Administrators should not modify or remove these resources.

These resources are removed when a Dynamic Evidence Type Version is cancelled.

For more information about these resources, please see "Localizing dynamic evidence" on page 85.

Dynamic UIM Resources

Each Dynamic Evidence Type Version has a number of Dynamic UIM pages and corresponding Property Application Resources associated with it. These are generated on the first page access in the case worker application. All such files are prefixed with a "DynEvd_" prefix. Administrators should not edit or remove these Resources.

These Resources are removed when a Dynamic Evidence Type Version is cancelled.

Tab Configuration and Application Section Resources

The Dynamic Evidence runtime user interface requires some tab configuration Resources. These are stored in the Application Resource Store.

Each Dynamic Evidence Type Version will have a corresponding tab resource. The Resource name will be the Dynamic Evidence Type code concatenated with the Effective Date of the Dynamic Evidence Type Version. There will also be a nav configuration Resource, the name of which will be identical to that of the tab configuration Resource, but with the word "Nav" appended.

If the Dynamic Evidence Type is parent to another Dynamic Evidence Type, then a menu configuration Resource will also be generated. The name of this Resource will be identical to that of the tab resource with the word"Menu" appended.

All Section resource files for each Cúram Application View will also be updated to include these generated tab configurations when a new Dynamic Evidence Type Version is activated.

If a Dynamic Evidence Type Version is cancelled, the tab configuration Resources associated with it will be removed, and all references in the Application section files will be cleaned up.

Domain Definitions

Dynamic Evidence generates Domain Definitions dynamically where required. These are stored in the Application Resource store in a Resource called `DynEvdDomains.xml`. This is a single Resource that is shared between all Dynamic Evidence Types. It will be generated on startup when the client application attempts to load it, and will be regenerated each time a new Dynamic Evidence Type Version is Activated or Cancelled. Administrators should not modify or remove this Resource.

Evidence Type Definition

The EvidenceTypeDefinition entity is used to store information about the behaviour of an Evidence Type. For each new Dynamic Evidence type an entry is added to this database table.

Summary of Generated Artefacts

This section is a summary of generated artefacts. It is intended as a useful checklist to accompany the preceding information.

Table 35. Summary of Generated Artefacts

Name	Type of artefact	Multiplicity
curam.dynamicevidence.type .code.prefix plus a unique identifier	Entry in EvidenceType codetable	One codetable entry per evidence type
"DynEvd_EvidenceType_" *ET Logical Name* ".properties"	Properties resource in AppResource store	One per Dynamic Evidence Type
Specified by administrator on evidence type creation	Security Identifiers (SIDs)	Three SIDs per evidence type - create, modify and read
Evidence Type Logical Name and various suffixes	CER Rulesets	Multiple rulesets per evidence type
Corresponding to the ruleset names	Ruleset propagator configurations	One propagator configuration per generated ruleset
"DynEvd_" *ET Logical Name* plus a number	Properties resource in AppResource store	One per evidence type version
"DynEvd_" and various suffiixes	Dynamic UIM and corresponding properties resource in AppResource store	Several pairs per evidence type version
Dynamic Evidence Type code concatenated plus a number	Tab Configuration in AppResource store	One per evidence type version
The same as the tab configuration resource name but with "Nav" appended	Nav Configuration in AppResource store	One corresponding to each tab resource

Table 35. Summary of Generated Artefacts (continued)

Name	Type of artefact	Multiplicity
The same as the tab configuration resource name but with "Menu" appended	Menu Configuration in AppResource store	One for each evidence type that has child evidence types
Named for the Cúram application view	Section resource files in AppResource store	Each Cúram application view has a "Section" resource
DynEvdDomains.xml	Entry AppResource store	One
EvidenceTypeDefinition	Database table	One entry on the table per evidence type

Dynamic Evidence Application Properties

Cúram application properties for Dynamic Evidence.

Cúram application properties are administered through the Cúram system administration application. For more information, see the *Cúram System Configuration Guide*.

Table 36. Dynamic Evidence Application Properties

Property name	Description	Default
curam.dynamicevidence .type.code.prefix	Prefix for auto-generated Dynamic Evidence Type Codetable codes. Prefix length is limited to three characters.	DET
curam.dynamicevidence .inprog.cache.max.size	Dynamic Evidence contains a LRU cache that is used to temporarily store data for in-progress user operations where the user transfers between interfaces for different versions of the same evidence type. This property specifies the maximum size for this cache.	1000

Compliancy and Upgrades

Specific information about dynamic evidence compliancy and upgrades.

Java API

Dynamic Evidence provides public APIs which may be invoked from custom Rule Sets and application code. Nothing will be changed or removed from this public API without following standards for handling customer impact. These APIs can be identified by looking at the Javadoc information shipped with Dynamic Evidence.

Unless explicitly permitted in the Javadoc information, you must not provide your own implementation of any Dynamic Evidence Java interface, nor subclass any Dynamic Evidence implementation Java class.

Infrastructure Rule Sets

Dynamic Evidence ships with the following infrastructure Rule Sets:
- DynamicEvidenceRuleSet

- EvidenceCalculatedAttributesRuleSet
- EvidenceSummaryRuleSet
- EvidenceValidationRuleSet

These Rule Sets should not be modified as part of custom development as these act as the contract between Dynamic Evidence and custom Dynamic Evidence Processing Rule Sets.

Dynamic Evidence Types Prefix

As described in "Generated Artefacts" on page 102, a Codetable entry is generated in the **EvidenceType** Codetable for each Dynamic Evidence Type. By default the generated entries will have a prefix of "DET". As per the general compliancy rules regarding Codetable entries, customers should not use the "DET" prefix for their Dynamic Evidence Types. Administrators should use the application property curam.dynamicevidence.type.code.prefix to define a different prefix for their custom Dynamic Evidence Types. Refer to "Dynamic Evidence Application Properties" on page 106 for more information on this property.

Source Code for the Sample Widgets

Complete the following steps for the sample widget.

Step 1

Configure the Domain name and it's plug-in in the `DomainsConfig.xml` file in `\webclient\components\<component-name>`

```
<dc:domain name="EXAMPLE_DOMAIN_NAME">
 <dc:plug-in name="edit-renderer" class="EditRenderer"/>
 <dc:plug-in name="view-renderer" class="ViewRenderer"/>
 <dc:plug-in name="converter" class="DomainConverterPlugin"/>
</dc:domain>
```

Step 2

Implement the class plug-in in `\webclient\components\<component-name>\javasource`

For examples, see the IBM Curam Custom Widget Development documentation.

Step 3

Once the implementation is done, build the client for the changes to be reflected.

Now "EXAMPLE_DOMAIN_NAME" can be used as domain name in dynamic evidence configuration

For more details on developing Domain and its custom widget, see the Curam Custom Widget Development documentation

Notices

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

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

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

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-17

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

Intellectual Property Licensing Legal and Intellectual Property Law IBM Japan Ltd. 19-21, Nihonbasl

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

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

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

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

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

Privacy Policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings

can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

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

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

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

Trademarks

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

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

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

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

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

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

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

IBM ®

Printed in USA