IBM Cúram Social Program Management

Working with Cúram Express Rules

Version 6.0.4

This edition applies to version 6.0.4 of IBM Cúram Social Program Management and all subsequent releases and modifications unless otherwise indicated in new editions.

Table of Contents

Chapter 1

Introduction

1.1 Overview

The CER Rules Editor is a part of the application which facilitates setting up and maintaining logical conditions (rules) and the results associated with those conditions. This includes creating and maintaining all Cúram Express Rule (CER) Sets.

The purpose of this guide is to cover the fundamentals you need to know when creating rule sets using the CER Rules Editor and adding business and technical logic to rule sets.

1.2 Audience

Writing CER rule sets requires a collaborative approach between business analysts and technical rules developers.

Therefore, this guide is intended for business analysts and developers responsible for designing and implementing rules.

1.2.1 The Role of the Business Analyst

Business Analysts have the knowledge and experience to translate legislative rules into a set of logical rules used as qualifying criteria for a social welfare program or service. These logical rules have to be converted into CER rules that can be understood by the rules engine during the execution of the ruleset. The role of the Business Analyst during the development of a given ruleset is to describe each rule within the editor to ensure it can be understood by both the business and technical community.

1.2.2 The Role of the Technical Rules Developer

The role of a Technical Rules Developer is to take what the business analyst has created and expand on it to implement the complete set of executable rules.

1.3 Related Reading

For detailed information on topics covered in this guide, see the *Cúram Express Rules Reference Manual* .

For information on Eligibility and Entitlement rules using CER Rules, see the *Inside Cúram Eligibility and Entitlement Using Cúram Express Rules* .

Chapter 2

Start with the Basics

2.1 Introduction

The CER language is a means of expressing qualification criteria in terms of logical rules and combined with real world data, allow decisions to be made on what should happen next or what should be displayed. A ruleset contains a combination of rules that can be used anywhere but are designed and implemented to produce a specific outcome.

Each rule set has a unique name and can have a different status depending on where it is the development or maintenance cycle. For example Newly Created, Published or Published - In Edit. This chapter describes the basics of the CER Editor which include the Editors layout and steps to create a rule set through to executing a rule set. Other important basics include creating a rule, adding logic to its diagram, validating your changes and publishing the final rule set.

2.2 CER Editor Layout

The CER Editor provides a user-friendly environment and interface for both technical and business users to create, edit and validate a ruleset and its rules. It primarily consists of the following components:

1. Global Menu.
2. Business and Technical Views.
3. Diagram Canvas.
4. Rule Outline View (Business View) and the Class Outline View (Technical View).
5. Pan and Zoom Controls.
6. Properties and Validations Panel.

7. Tools and Templates Palette.

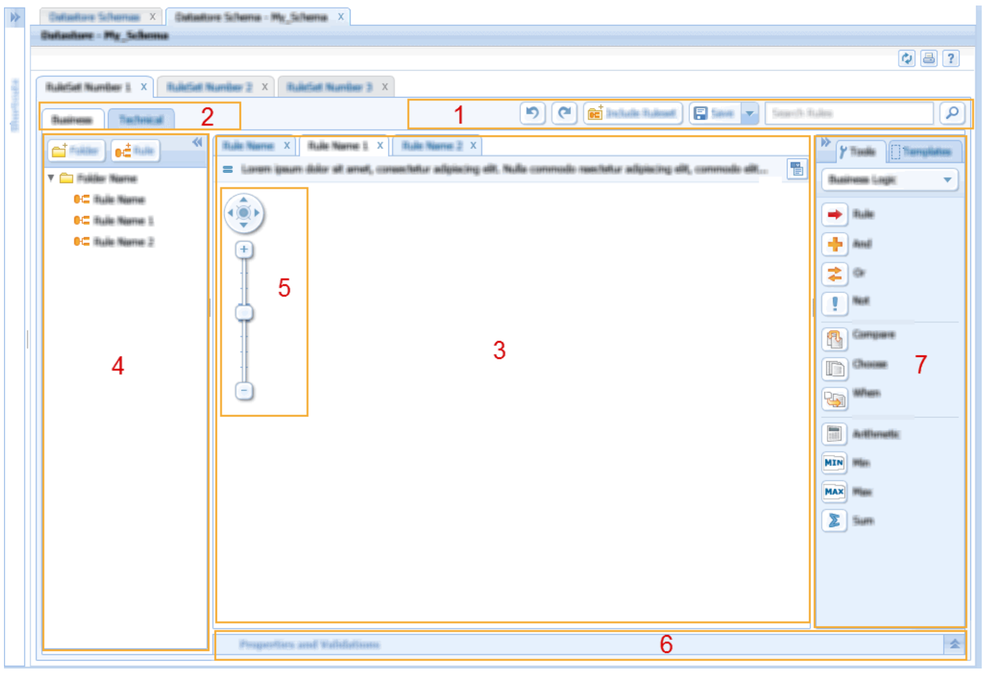

Figure 2.1 CER editor layout

Note

The diagram canvas contains pan and zoom controls to help the user navigate large and complex rules.

2.2.1 CER Editor Business View

This view is designed to assist business users with the creation of business rules. The text, structure and in some cases the data is required to ensure the rule is readable. The Rules structure can be simple or complex, therefore business view only contents information relevant to the business user and displays a business rule centric view of the ruleset.

2.2.2 CER Editor Technical View

This view provides the technical users with information relevant to their role in rules development. The implementation of the rules requires access to all the tools and templates available in the editor. More technical elements of the ruleset are presented such as classes, attributes and their associated properties.

2.3 Creating a Simple Rule Set

A new rule set can be created from the Cúram Express Rule Sets area of the

Cúram Administration Application . The following is an outline of the steps to create a new rule set.

1. Open the *CER Administration Application* and select *Actions*→*New* menu option.
2. Enter a name for your new rule set in the new rule set pop-up and select an appropriate category.

A new rule set can be opened from the *List Actions*→*Continue Editing* menu option.

Tip

The *List Actions* menu option to open the editor will be *Continue Editing* when the status is *Published - In Edit* and *Newly Created* . When a rule set is *Published* the menu option to open the editor will be *Open Editor* .

To increase the readability of the ruleset within the editor, it is recommended that the browser is set to full screen. This can normally be achieved using the F11 key.

2.4 Creating a Folder

A business user creates folders in order to group their rules. For example a business user might create a folder to group all the rules related to the income of the household.

1. Open the “New Folder” dialog by selecting the “Folder” button at the top of the Rule Outline View

Name	Explanation
Name	The name of the folder you wish to create.

Table 2.1 Folder Properties

2.5 Creating a Rule

Typically a business user wants to create rules with text and structure that communicates their meaning. A diagram is the graphical representation of an individual rule. A business user will use this diagram to describe the rule structure and logic. For example, What is the total household income?

1. Open the *CER Editor* Rule dialog by selecting the Rule button in the Rule Outline View.
2. In the Name field enter *Simple Calculation* .

3. From the Type dropdown, select the *Number* entry.

Name	Explanation
Name	The description to associate with the Rule and its Diagram.
Type	This is the type of data this rule will return. For example the rule that verifies the client is alive will return a Boolean value either true or false.

Table 2.2 Rule Properties

2.6 Adding a Simple Calculation

In the diagram for the rule which is opened when a new rule is created, a rule element is displayed. This is the starting point for any rule and elements from the palettes can be dropped onto this element. A description can be added to the rule in the properties panel that will describe this rule from the business perspective.

In the description property add *Simple Calculation* as the description for this rule.

1. From the *Business Logic* palette drag an *Arithmetic* element onto the rule element on the diagram. A *Golden Circle* will appear when the arithmetic element is correctly positioned over the rule element. Releasing the mouse will allow you to add the arithmetic to this rule. The default behavior for an arithmetic expression is to add two elements. Double click on the *Addition* text on the arithmetic element and enter *Adding 2 and 2*

2. From the Tools palette select the dropdown to change from the business Logic palette to the *Data Types* palette. Now drag the *Number* element onto the left *Empty Member* of the arithmetic element. In the property panel set the value to 2. Repeat this for the right side of the arithmetic operation.

The above example simply adds 2 numbers to answer the question “What's 2 + 2” . This is a example of a very basic calculation and rulesets used for eligibility and entitlement or for displaying information to application users will be more complex in structure and how they calculate their results.

2.7 Organizing the Business Rules

Once the Business user has completed their portion of the ruleset where the high level business rules are defined, the technical user can then provide the full implementation for each rule. The business rules from the business view are created on the default class, allowing them business meaning of the rules and not worry about the technicalities of the CER language. The editor provides a technical user with the ability to change the attribute associated with the rule allowing them to spread the rules across multiple classes.

1. From the Rules Outline View, select the appropriate rule and click on the context dropdown menu. From the list of options choose the *Edit Rule* option.
2. The dialog displays the name of the rule, a folder dropdown to move the rule and the attribute associate with the rule. To change the attribute select the change link.

Name	Explanation
Name	Edit the description of a diagram.
Folder	Change the folder a diagram is grouped in.
Attribute	Change the attribute for the rule. The rule diagram can be moved to another attribute using the *Change* link.

Table 2.3 Edit Rule Properties

2.8 Exporting Diagrams

The CER Editor allows users to export business rule diagrams. There are two types of export:

- Single Business Rule Diagram Export - where the current diagram can be exported as a PNG image file.
- All Business Rule Diagrams Export - where all the diagrams listed in the rule outline view are exported to PNG image files, presented in a HTML file and downloaded as a ZIP archive file.

2.8.1 Single Business Rule Diagram Export

The CER Editor allows users to export a single business rule diagram as a PNG image to the local hard disk. This functionality is accessed via the Export button on the top right hand corner of each diagram. The file name and location where the file is to be saved can be chosen by the user.

The image generated will reflect the current state of the diagram, for example if the diagram is in technical mode when the Export button is pressed, then the exported image will capture the technical mode details.

Steps to export a single business rule diagram

1. Open the diagram to be exported.

2. Press the Export button on the top right hand corner of the diagram.
3. A Save dialog box appears allowing the user to enter the file name and location where the PNG file is to be saved.

2.8.2 All Business Rule Diagrams Export

The CER Editor also allows users to export all business rule diagrams in a ZIP archive which can be saved to the user's local hard disk. This functionality is accessed via the Export drop down option of the Actions button on the Global Menu. The ZIP archive contains the following:

- A PNG image file for each diagram listed in the rule outline view. The name of each file will be based on the diagram's class and attribute names.
- A HTML file which is used to view all the diagrams exported. The name of the HTML file will be the rule set name with a ".html" extension. The order the diagrams are listed in this file will exactly reflect the order of the diagrams in the rule outline view in the CER Editor. The following information is also displayed with each image; the diagram name, the diagram description and the diagram's class and attribute names.

Steps to export all business rule diagrams

1. Open the rule set containing the diagrams that are to be exported.
2. Use the *Actions*→*Export* menu option to export the diagrams.
3. A dialog box appears showing the progress of the export process. When the process has been completed the Save button on the dialog is enabled. The user must press this button to save the zipped images to the local hard disk. The file name and location where the ZIP file is to be saved can be chosen by the user.

   Note

   Unlike the Single Diagram Export above, the diagrams do not need to be opened in the editor in order to be exported. All diagrams listed in the rule outline view will be automatically exported.

   The images will be generated in business mode (i.e. the images will not contain information that is displayed when the diagram is in technical mode).

2.9 Validating the Rule Set

CER includes a comprehensive rule set validator which can detect errors in your rule set before allowing your rule set to be published or executed. The CER editor provides functionality to allow a user to validate a rule set from within the editor.

1. In the *CER Editor* select the dropdown menu on the “Save” button.
2. In this dropdown select the “Validate” option.

Validation messages will be displayed in a validations panel. This panel is displayed under the Diagram canvas the first time a rule set is validated.

Tip

Double-clicking a validation error in this rule set will result in the element thats causing the validation being highlighted in red in the main content area.

For a detailed explanation of the CER validator and how it works, see Chapter 3 of the *Cúram Express Rules Reference Manual* .

2.10 Publishing the Rule Set

While a rule set is being modified in the editor it will have a status of *Published - In Edit* or *Newly Created* . Changes made in the editor will not be picked up the next time the rules are executed. When an eligibility check is run on a case it will use the *Published* version of a rule set. A rule set can be published once a user is happy with the changes and any validation errors have been fixed.

1. From the *CER Administration Application* select the *Actions→Publish* menu option.
2. A list of rule sets that have a status of either *Published - In Edit* or *Newly Created* will be displayed. Use the check box to select the rule sets you want to publish.

2.11 Executing the Rule Set

Running a rule set creates a rule session. A rule session controls the executing of rules, e.g. your application might create a rule session to determine John Smith's eligibility for child benefit, by invoking the appropriate rule set and asking eligibility questions regarding John's personal circumstances. See Section 2.1.2, of the *Cúram Express Rules Reference Manual* .

CER supports you executing rule sessions in:

1. production environments, where CER integrates with your application to answer questions when needed; and
2. a stand-alone test environment where you create repeatable automated tests for your rule sets.

Chapter 3

Creating Point-in-time Business Rules

3.1 What Are Point-in-time Rules?

Point-in-time rules are rules that are run to answer questions on a particular day. The purpose of *Point-in-time* rules is to intentionally capture data which applies to a particular date only. For example, is a person eligible on a particular date?

3.2 Defining Business Requirements

An organization decides it needs a new program, for the purpose of this document it is called Sample Benefit. It is to be introduced in order to provide support for families with children and young students. A business user examines the relevant legislation and extracts a set of business requirements from that legislation. Here is an extract from the requirements.

Sample Benefit Requirements

A household is deemed eligible for Sample Benefit if it contains either a child under 16 years of age, or a student between the ages of 16 and 22, and the household's total annual income is between 0 and 20,000 . The household's total income is considered to be the total of the countable amounts for each income received by each member of the household over the age of 18.

3.2.1 Identify High-level Questions

A business user will extract basic questions that need to be answered in order to decide eligibility for Sample Benefit. The following are examples of questions which may need to be answered to determine eligibility.

- Does this household contain an eligible person?

- What age does a student in a household have to be in order to make the household eligible for Sample Benefit?
- What type of income should be counted?

3.2.2 Group Related Questions

Once you have identified the high-level questions that the “Sample Benefit” rule set needs to answer you should group the questions that are related to each other. The business rules that will be created for these questions can then be grouped into folders.

3.3 Creating a Point-in-time Rule Set

In Chapter 2 we described how to create a new rule set (see Section 2.3, *Creating a Simple Rule Set*).

3.4 Creating Folders for Related Questions

A business user will want to group related rules together. The editor provides a mechanism for doing this through the use of folders. To start with create the two folders listed below. More information on this is available in the Section 2.4, *Creating a Folder* .

Example Folders for Sample Benefit

- Sample Benefit Eligibility Rules
- Person Rules

3.5 Creating a Business Rule for Each Question

To begin answering questions a business user will need to start creating rules. A business rule can be easily created without the user having to consider how its implemented or the associated attribute. They only need to consider the type of data this rule will use and calculate. Later in the rule set development life cycle the technical user will organize the rule set into an executable structure. See Section 4.2, *Organizing Rules into Classes and Attributes* .

Steps to create a Business Rule

1. from the Rules Outline view in the Business View click on the context menu on the “Sample Benefit Rules” folder and select the “New Rule” option to create a new Rule.
2. Name the Rule “Eligibility”
3. The default Type is set to Boolean so there is no need to change this.

4. Click on the “Save” button.
5. In the Rule properties panel add a description “Eligibility for Sample Benefit” .

3.6 Adding Business Logic For Rules

This section describes how a business user adds business logic onto the rules diagram. The CER editor allows business users to add business logic to a diagram without having to worry about any of the technical values usually associated with authoring a rule set. A business user provides meaningful descriptions for the rules they add.

Throughout these steps you will be asked to change or add descriptive text to rule elements within the diagram. To make a diagram element text editable double-click on its text. A text area will become highlighted and editable. A description can be split over multiple lines using *Ctrl-ENTER* .

3.6.1 Eligibility

Now that we have a `SampleBenefit` rule set with a Rule called “Eligibility” it's time to add some business logic.

Create Eligibility Rules

1. From the *Business (default)* palette on the right-hand side of the editor, drag an *AND Rule Group* element onto the rule element on the diagram (you will see a Golden circle appear when you move over the rule element). Change the name of the *AND Rule Group* to be “The household contains an eligible person and passes the income test” .
2. Drag a *Rule* element from the *Business (default)* palette onto the “Empty Member” element within the *AND Rule Group* . A dialog will appear where the “Empty Rule” is selected. For now just click the “Save” button to close the dialog. On the empty reference change the name of the to be “The household contains an eligible person” .
3. Drag another *AND Rule Group* element over this empty reference until you see arrows to the left and right of the *reference* . Move the mouse over the right-hand arrow until it turns green and drop it. Change the name of the *AND Rule Group* to be “The income is between 0 and 20000” .
4. Drag a *Compare* element onto the “Empty Member” of the new *AND Rule Group* . Change its name to be “Income is greater than 0” .
5. Drag another *Compare* element to the right of the first compare and change its name to be “Income is less than 20000” .

This rule has now sufficient text and structure that can be understood by both a business user and a technical user. The Rule has an empty reference with two compares with an “AND” conjunction between them. There is also an “AND” conjunction between the reference and the group of compares.

6. Create a new Rule in the *Sample Benefit Rules* folder called “Household Contains Eligible Person” . Again leave the type as Boolean.
7. Add a description to the rule: “Household Contains Eligible Person” .
8. Drag a *Repeating Rule* from the *Business (extended)* palette onto the rule. Change the name of the *Repeating Rule* to be “Any of the people in the household are eligible” .
9. Drag a *Rule* onto the *Empty List* element in the repeating rule. An empty rule is fine for now. Change its name to be “All the people in the household” .
10. Drag another *Rule* onto the *Empty Members* element within the repeating rule. Again an empty rule is fine here. Change its name to be “The current person is eligible” .
11. Using the context menu on the *repeating rule* element (this is available on the right side of the element), choose the “Succeed on Any” option.

This rule has now sufficient text and structure that can be understood by both a business user and a technical user. However if more information is required, add extra context to the description of the Empty rule elements.

12. Create a new rule in the “Person Rules” folder called “Eligible Person” .
13. Drag an *OR Rule Group* element from the *Business (default)* palette onto the rule element on its diagram and change its name to be “The person is either a child or a student” .
14. Drag a *Compare* element onto the *Empty Member* element within the *OR Rule Group* and change its name to be “The person is a child under the age of 16” .
15. Drag an *AND Rule Group* below the compare element and change its name to be “The person is a student in the right age range” .
16. Drag a rule onto the *Empty Member* element within the *AND Rule Group* and change its name to be “The person is a student” .
17. Drag another *AND Rule Group* to the right of this “The person is a student” rule and change its name to be “Their age is between 16 and 22” .
18. Drag a *Compare* element onto the *Empty Member* element within the *AND Rule Group* and change its name to be “Age is greater than 16” .
19. Drag another *Compare* element to the right of this *Compare* and change its name to be “Age is less than 22” .

This rule has now sufficient text and structure that can be understood by both a business user and a technical user. When reading this rule the user should see that this rule states: “ The person is under 16 *OR* is a student *AND* in the age range of 16 to 22 ” . However if more information is required, add extra context to the description of the Empty rule elements.

3.6.2 Household Income

Based on the requirements defined in the Section 3.2, *Defining Business Requirements* we identified high-level questions related to eligibility. We created rules to answer these questions. Next a business user will need to follow the same process to create business rules around a household's income.

Countable Income Rules for household

1. Create a new Rule called “Household Income” in the *Sample Benefit Rules* folder. Set the type as Number.
2. Drag a *Repeating Rule* element onto this rule and change its name to be “The total income for each countable household member” . Remember to change the name double click on the text of the element on the diagram or enter a value in the Name field of the Business Properties panel.
3. Drag a *Rule* onto the *Empty List* element of the *Repeating Rule* . An empty rule is fine for now and change its name to be “All people in the household who are countable from an income perspective” .
4. Drag a *Rule* onto the *Empty Members* element of the *Repeating Rule* . An empty rule is fine for now and change its name to be “The current person's total income” .
5. From the context menu on the right side of the *repeating rule* element choose the *Sum Items* option from the menu which pops up.

This rule has now sufficient text and structure that can be understood by both a business user and a technical user. However if more information is required, add extra context to the description of the Empty rule elements.

6. Create a new Rule in the “Sample Benefit Rules” folder called “Countable Household Members” . Set the type as Number.
7. Drag a *Filter* element onto the rule.
8. Drag a *Rule* onto the *Empty List* element of the filtered list. An empty rule is fine for now and change its name to be “All the people in the household” .
9. Drag a *Rule* onto the *Empty Members* element of the filtered list and again an empty rule is fine for now. Change its name to be “The person is countable” .

This rule has now sufficient text and structure that can be understood by both a business user and a technical user. However if more information is required, add extra context to the description of the Empty rule elements.

10. Create a new Rule in the “Person Rules” folder called “Countable Person” . Leave the type as Boolean.
11. Drag a *Compare* element onto the rule and change its name to be “The person's age is greater than or equal to 18” .

This rule has now sufficient text and structure that can be understood by both a business user and a technical user.

12. Create a new Rule in the “Person Rules” folder called “Person Income” . Set the type as Number.
13. Drag a *Repeating Rule* onto this rule element and change its name to be “The countable amount for each income” .
14. Drag a *Rule* onto the *Empty List* element of the repeating rule. An empty rule is fine for now and change its name to be “All the incomes for this person” .
15. Drag another *Rule* onto the *Empty Members* element of the repeating rule and again an empty rule is fine. Change its name to be “The countable amount for this income” .
16. From the context menu on the right side of the *Repeating Rule* choose the *Sum Items* option.

This rule has now sufficient text and structure that can be understood by both a business user and a technical user.

17. Create a new Rule in the “Person Rules” folder called “Countable Income” .Set the type as Number.
18. In the outline view select the rule and open the menu on its right side by selecting the dropdown icon for the rule. Choose the *Set Rule Type* option from the menu which pops up. Select the *Decision Table* option in the dropdown and click next; choose *Number* as the type and click next; choose the *Create a new rule* option, enter “type” as the name and click *Next* ; choose *String* as the type and click *Save* .
19. Change the name for the table to be “Countable Income” (the box at the top of the table).
20. Change the Condition Description in the left-hand column of the table to be “Income Type” . Remember to double click on the text. To change the right hand column header, change the attribute name to be “Amount” . To do this select the technical view and in the properties panel change the name property to “Amount” .
21. Change the name of the left-hand cell for the first row to be “Wages” and the description of the right-hand cell to be “The full amount” .

22. Open the context menu on the rule (at the top of the table) and from the menu which pops up, choose the *New Row* option. Change the descriptions for the left and right-hand cells to be “Tips” and “Half the amount” respectively.
23. Open the context menu on the rule and from the menu which pops up, choose the *New Row* option. Change the descriptions for the left and right-hand cells to be “Commission” and “Twice the amount” respectively.
24. Add another Row and change the descriptions for the left and right-hand cells to be “Rental Income” and “The full amount, up to a limit of 500” respectively.
25. Add one final Row and change the description of the right-hand cell to be “Zero” .

The decision table has sufficient information for other users to understand its structure and purpose.

3.6.3 Entitlement

Based on the requirements defined in the Section 3.2, *Defining Business Requirements* we identified high-level questions related to eligibility and income. We created rules to answer these questions. Next a business user will need to follow the same process to create business rules around a household's entitlement.

Entitlement Rules

1. Create a new Rule in the “Sample Benefit Rules” folder called “Entitlement” . Set the type to be Boolean
2. Select the context menu on the rule on the outline view and choose the *Set Rule Type* option from the menu which pops up. Select the *Decision Table* option and click *Next* ; choose *Number* as the type and click *Next* ; choose the *Create a new rule* option, enter *numEligibleMembers* as the name and click *Next* ; choose *Number* as the type and click *Save* .
3. Change the description for the table to be “Entitlement” (the box at the top of the table).
4. Change the description for the header of the left-hand column of the table to be “Number of eligible people” .
5. Change the description of the left-hand cell for the first row to be “1” and the description of the right-hand cell to be “100” .
6. Open the context menu on the rule (at the top of the table) and from the menu which pops up, choose the *New Row* option. Change the descriptions for the left and right-hand cells to be “2” and “180” respectively.

7. Add another row to the table and change the descriptions for the left and right-hand cells to be “3” and “250” respectively.
8. Add another row to the table and change the descriptions for the left and right-hand cells to be “4” and “310” respectively.
9. Add another row to the table and change the descriptions for the left and right-hand cells to be “5” and “360” respectively.
10. Add the final row to the table and change the description of the right-hand cell to be “400” .

The decision table has sufficient information for other users to understand its structure and purpose.

3.6.4 Eligible Members

Based on the requirements defined in the Section 3.2, *Defining Business Requirements* we identified high-level questions related to eligibility income and entitlement. Next a business user will need to follow the same process to create business rules around the number of eligible members in a household.

Number of eligible Members

1. Create a new Rule in the “Sample Benefit Rules” folder called “Number of eligible members” . Change the description of the rule on this diagram to be “Number of eligible people in the household” . Set its type to be Number.
2. Drag a *Size* element from the *Business (extended)* palette onto the rule.
3. Drag a *Filter* element onto the *Empty Member* element within the *size* .
4. Drag a *Rule* onto the *Empty List* element of the filter. Use an empty rule for now and change its name to be “All the people in the household” .
5. Drag a *Rule* onto the *Empty Members* element of the filter. Use an empty rule for now and change its name to be “The person is eligible” .

Chapter 4

For the Technical Reader - Developing Point-in-time Rules

4.1 How Do Business and Technical Users Work Together?

In the previous chapter we showed how a business analyst can create the business logic for the product sample benefit. A technical user can take this rule set as input and create a rule set that can be run within the Application. The technical user will add technical logic based on a business users descriptions. In our sample benefit example the technical user would need to update the rule set in the following way.

- Organize the classes and attributes associated with the rules.
- Edit empty rules to point to the correct rules.
- Populate the arguments for *Compare* elements.
- Set the values of data types. For example *Numbers* , *Strings* .
- Update decision tables with values based on row description.
- Validate and test the rule set until it's ready to be published.

4.2 Organizing Rules into Classes and Attributes

Up to this point a business user has been creating diagrams on the default class for generated rules. The first task of a technical user is to organize rules into classes. Rules should be grouped in classes that make sense. For example, all the rules related to income should go in an Income class. Organizing rules can be performed using the *Edit Rule* Wizards.

4.3 Adding Technical Logic to a rule set

This section details the steps involved in updating our `SampleBenefit` rule set so that it can be published. For more details on any of the items mentioned here, see chapter 5 in the *Cúram Express Rules Reference Manual* .

4.3.1 Eligibility

1. In the Technical View of the editor, the list of available classes are displayed. To add more classes use the “Class” button on the Class Outline View.
2. In the Rules Outline View of the Business view of the editor, select the “Eligibility” rule and from the context menu on the right side select *Edit Rule* . In the dialog which pops up, click the *Change* link beside the rule name. Here a new class must be selected. Select the click the *Change* link beside the rule name of the Rule Class and select the appropriate class. Select the radio button beside *Create new rule* option, enter “eligible” as the name and click *Next* ; choose *Boolean* as the type and click *Save* .
3. Edit the “Household Contains Eligible Person” rule in the same way. Change the rule name to be a new rule called *householdContainsAnEligiblePerson* of type Boolean.
4. Within the technical View on the “Eligibility” rule diagram, open the context menu for the “Eligible Person” rule and choose *Edit Rule* from the menu which pops up. Choose the “householdContainsAnEligiblePerson” rule from those listed in the drop-down and click *Save* .
5. Drag a Reference from the *Business (default)* palette over the left side element of the “Income” is greater than “0” comparison. Choose the *Create a new rule* option, enter “householdIncome” as the name and click *Next* ; choose “Number” as the type and click *Save* . Drag a Number from the *Data Type* palette over the right side element of the “Income” is greater than “0” comparison. In the Value field of the properties panel enter the number “0” .
6. Update the compare elements for the “Income is less than 20000” comparison so that it becomes “householdIncome < 20000” .
7. Using the Class Button on the Class Outline view create a new class called “Person” .
8. On the “Household Contains Eligible Person” rule diagram, edit the “All the people in the household” rule. Enter “household” as the name of the new rule and click *Next* ; choose *Rule Class* and check the “List”

checkbox. Click *Next* ; choose “Person” as the type and click *Save* .

9. Edit the “Eligible Person” rule so that its rule name is a new rule called “eligible” in the *Person* class, of type *Boolean* . You can choose the person class by using the *Change* link for the *Current Rule Class* in the edit rule wizard.
10. On the “Household Contains Eligible Person” rule diagram, edit the “current person is eligible” rule to refer to the “eligible” rule in the Person class.
11. In the “Eligible Person” diagram, add the appropriate elements to the comparisons to create “age < 16, age >= 16” and “age < 22” respectively, where “age” is a new rule in the Person class of type Number.
12. Edit the “person is a student rule” on this diagram to refer to a new rule in the Person class called *isStudent* of type Boolean.

4.3.2 Household Income

1. Edit the “Household Income” rule so that it refers to the “householdIncome” rule in the “SampleBenefit” class.
2. On the “Household Income” rule diagram, edit the “All people in the household who are countable from an income perspective” rule to refer to a new rule in the “SampleBenefit” class called “countableMembers” , which is a list of Person objects.
3. Also on the “Household Income” rule diagram, edit the “person total income” rule to refer to a new rule in the Person class called “totalIncome” of type Number.
4. Edit the “Countable Household Members” rule so that it refers to the “countableMembers” rule in the SampleBenefit class.
5. On the “Countable Household Members” rule diagram, edit the “all the people on the household” rule to refer to the “household” rule in the SampleBenefit class.
6. On the same rule diagram, edit the “Person is countable” rule to refer to a new rule in the Person class called “isCountable” of type Boolean.
7. Edit the “Countable Person” rule so that it refers to the “isCountable” rule in the Person class.
8. On the “Countable Person” rule diagram, add the appropriate elements to the comparison so that it becomes *age >= 18* .
9. Create a new rule class called “Income” .
10. Edit the “Person Income” rule so that it refers to the “totalIncome” rule in the Person class.
11. On the Person Income rule diagram, edit the “all the incomes for this

person” rule to refer to a new rule in the Person class called “incomes” which is a list of Income objects.

12. Edit the “countable amount” rule on the same diagram to refer to a new rule in the Income class called “countableAmount” of type Number.
13. Edit the “Countable Income” rule so that it refers to the “countableAmount” rule in the Income class. You should see a dialog which tells you that another new rule will need to be created in the Income class called “type” click *Ok* on this dialog.
14. On the “Countable Income” rule diagram, click on the square icon in the top right-hand corner of the diagram to present the technical view of the decision table. For each row in the decision table, you should now see an extra area underneath the business descriptions containing the technical details.
15. Edit the technical details for the left-hand column so that they all match their descriptions (i.e. Wages, Tips, Commission, Rental Income).
16. Drag a Rule from the *Business (default)* palette over the icon to the left of the “Full Amount” description for the wages row. You should see a golden circle appear,allowing you to drop the rule. Edit the Reference to refer to a new rule in the Income class called “amount” of type Number.
17. Drag an Arithmetic element onto the result for the Tips row. Click on the “operator” dropdown in the middle of the element and change it to “/” operator. Drag a Rule from the *Business (default)* palette to the left of the operator and refer it to the “amount” rule. Now drag a Number from the *Data Type* palette onto the empty element on the right of the operator and set its value to “2” .
18. Open the context menu on the division element and choose the “copy” option from the menu which pops up. Move the mouse over the icon for the Commission row until you see the golden circle and click on it to paste the copied element. Change the operator of this element to be “*” .
19. From the *Business (default)* palette, drag a *Choose* element onto the result for the “Rental Income” row. In the dialog which pops up, select “Number” as the type.
20. Drag a *Compare* element onto the *Empty Condition* of the when. Add the appropriate elements to the comparison so that it becomes “amount < 500” .
21. Drag a *Rule* onto the *Empty Value* element of the when and refer it to the *amount* rule.
22. From the *Data Types* palette, drag a *Number* element onto the *Empty Value* element of the otherwise and change its value in the properties panel to be “500” .

23. Edit the value of the result for the otherwise of the decision table to be “0” .

4.3.3 Entitlement

1. Edit the Entitlement rule so that it refers to a new rule in the “SampleBenefit” class called “entitlement” of type Number.
2. On the “Entitlement” rule diagram, expand out the decision table to show the technical view and edit the values for the “numEligibleMembers” cells to be the same as their descriptions (i.e. 1, 2, 3, etc.).
3. Do the same for each of the results.

4.3.4 Eligible Members

1. Edit the “Number of Eligible Members” rule so that it refers to the “numEligibleMembers” rule in the “SampleBenefit” class.
2. On the “Number of Eligible Members” rule diagram, edit the rule for “all the people in the household” to refer to the “household” rule in the SampleBenefit class.
3. On the same diagram, edit the “person is eligible” rule to refer to the “eligible” rule in the Person class.
4. Find any attributes on classes in the Technical View and delete all that are not required or begin with “newAttribute” .

4.4 Testing and Executing Rules

Once you have created a CER rule set, you will want to execute and test your rules. See Chapter 3 of the *Cúram Express Rules Reference Manual* for a description of how to test your rules.

4.4.1 Validate Sample Benefit

CER rule sets are XML files which adhere to the CER-supplied rule schema. This schema enforces many structural validations. CER also includes a comprehensive rule set validator which can detect errors in your rule set before allowing your rule set to execute.

Before you can execute your `Sample Benefit` program you will need to ensure you are working with a validate rule set. See Section 2.9, *Validating the Rule Set* .

4.4.2 Publishing Sample Benefit

CER provides a sandbox to allow rule set developers makes changes to rule sets without those changes being picked up by the products that are using those rule sets. Modifying a published rule set in the editor results in a published - in-edit record being created for that rule set.

In order for your in-edit changes to be picked up the next time a rule session is kicked off for that rule set you must first publish the rule set. See Section 2.10, *Publishing the Rule Set* .

4.4.3 Extracting Rule Sets from the Database for Testing

Once you have validated a CER rule set, you may want to write some Java® code to execute and test your rules. To do this you need to publish the rule set and extract your newly-created rule set from the database. This section describes how to extract your rule set from the database.

Use the extractdata build target to extract the contents of a table from the database and transform it into a database independent XML file. More detail on this target is provided in Chapter 5 in the *Curam Server Developer Guide* .

Steps to extract rule set

1. Open a command prompt in your `Curam/EJBServer` directory.
2. Run **build extractdata -Dtablename=CREOLERULESET**
3. All rule sets for the table you specified should be extracted to the folder `Curam\EJBServer\build\dataextractor\blob` . Records are extracted to a file based on the tablename. Find the rule set you are working with and rename it based on the rule set name. Give it an XML file extension, e.g. `SampleBenefit.xml` .
4. Copy `SampleBenefit.xml` to the `CREOLE_Rule_Sets` directory of the component you are developing the rule set for. You may need to create this directory.
5. Copy `CREOLERULESET.dmx` from the directory `Curam\EJBServer\build\dataextractor` to `data\initial` directory of the component you are developing the rule set for. You may need to create this directory.
6. Edit the file `CREOLERULESET.dmx` located in the `data\initial` directory of the component and delete all the entries except the entry related to your newly created rule set `SampleBenefit.xml` . This entry can be found by searching for your rule set name in <value> SampleBenefit</value> tag in the `CREOLERULESET.dmx` file.
7. Set the rule set definition path as ./../build/svr/creole.gen/Rules/components/ *`your component/ your rule set name`* in the `CREOLERULESET.dmx` . And also empty the ruleSetVersion value as shown in the sample XML below.

```
<attribute name="ruleSetDefinition">
   <value>./../build/svr/creole.gen/Rules/components/
     custom/SampleBenefit.xml
   </value>
 </attribute>

 <attribute name="ruleSetVersion">
   <value/>
 </attribute>
```

Steps to extract rule set properties

1. Open a command prompt in your `Curam/EJBServer` directory.
2. Run **build extractdata -Dtablename=APPRESOURCE**
3. Copy `APPRESOURCE.dmx` from the directory `Curam\EJBServer\build\dataextractor` to `data\initial` directory of the component you are developing the rule set for. You may need to create this directory.
4. Edit the file `APPRESOURCE.dmx` located in the `data\initial` directory of your component and delete all the entries except the entry related to your newly created rule set `SampleBenefit.xml` . This entry can be found by searching for <value>RULESET- *`your ruleset name and version`* </value> in `APPRESOURCE.dmx` . For example, <value>RULESET-SampleBenefit1</value>
5. Copy the resources files specified in the `APPRESOURCE.dmx` file, from the folder `Curam\EJBServer\build\dataextractor\blob` to `data\initial` directory of your component. The names of the resources files can be found in the tag <attribute name="content"> <value>./blob/ *`resource file name`* </value></attribute> in `APPRESOURCE.dmx` . For example, <attribute name="content"> <value>./blob/Appresource2289</value>lt;/attribute>
6. Set the resources path as ./ *`your component`* /data/initial/blob/ *`resource file name`* in `APPRESOURCE.dmx` file as shown in the sample XML below.

```
<attribute name="content">
  <value>./custom/data/initial/blob/Appresource2289</value>
</attribute>
```

7. Rebuild the database.

4.4.4 Generating Test Classes

This section explains CER's support for generating rule set test classes. We can test our `Sample Benefit` rule set using this support. For this example we will look at the “Eligible Person” Rule. This diagram represents the “eligible” attribute on the `Person` class.

The `Person` class has 2 attributes which need to be specified in order to calculate the eligible attribute for a person.

Specified attributes in **`Person`** class

1. age - a number to hold a person's age.
2. isStudent - a boolean to indicate if the person is a student.

Derived attributes in **`Person`** class

1. eligible - a boolean to indicate if a person is eligible for “Sample Benefit”

Below is the XML for the `Person` class from the `Sample Benefit` rule set.

```
<Class name="Person">
      <Attribute name="eligible">
        <Annotations>
          <Label name="A person is eligible"/>
        </Annotations>
        <type>
           <javaclass name="Boolean"/>
        </type>
        <derivation>
        <any>
          <Annotations>
              <Label
              name="The person is either a child or a student" />
          </Annotations>
          <fixedlist>
              <listof>
                  <javaclass name="Boolean"/>
              </listof>
          <members>
          <compare comparison=">">
              <Annotations>
                 <Label
                 name="The person is a child under the age of 16"/>
              </Annotations>
              <reference attribute="age"/>
               <Number value="16" />
           </compare>
         <all >
          <Annotations>
            <Label
          name="The person is a student in the right age range"/>
            </Annotations>
            <fixedlist >
                  <listof>
             <javaclass name="Boolean"/>
              </listof>
                   <members >
              <reference attribute="isStudent">
                  <Annotations>
              <Label name="The person is a student"
              </Annotations>
            </reference>
            <all>
                <Annotations>
                      <Label
                      name="Their age is between 16 and 22"
                 </Annotations>
              <fixedlist>
              <listof>
               <javaclass name="Boolean"/>
              </listof>
                  <members>
                      <compare comparison=">=">
                            <Annotations>
                      <Label name="Age is greater than 16"
                      </Annotations>
                      <reference attribute="age"/>
                       <Number value="16"/>
                      </compare>
                          <compare comparison="<" >
                      <Annotations>
                            <Label name="Age is less than 22"
                        </Annotations>
                            <reference attribute="age"/>
                            <Number value="22"/>
                        </compare>
                        </members>
                     </fixedlist>
                </all>
                </members>
                      </fixedlist>
                  </all>
                   </members>
              </fixedlist>
            </any>
        </derivation>
     </Attribute>
           <Attribute name="age">
        <type>
             <javaclass name="Number"/>
         </type>
         <derivation>
               <specified/>
          </derivation>
      </Attribute>
           <Attribute name="isStudent">
        <type>
             <javaclass name="Boolean"/>
         </type>
         <derivation>
               <specified/>
             </derivation>
         </Attribute>
     </Class>
```

The next step is to run the *CER Test Code Generator* on the `SampleBenefit.xml` which we extracted in Section 4.4.3, *Extracting Rule Sets from the Database for Testing* .

Steps to run *CER Test Code Generator*

1. Open a command prompt in your `Curam/EJBServer` directory.
2. Run **build creole.generate.test.classes** to generate test classes from your rule set. This target runs on the rules sets in the `CREOLE_Rule_Sets` directory for each component.
3. The CER code generator will place its output in the `EJBServer/build/svr/creole.gen/source` directory.

The generated code is only intended for use in test environments where it is a straightforward matter to recompile changes to code. The generated code is not portable across machines, as it contains absolute paths to the rule sets on the local machine. In particular, you must not use the generated code in any production environment where rule sets may change dynamically.

Tip

You should regenerate your test classes if you make structural changes to your rule sets e.g.

- create a new rule set or remove an existing rule set;
- add a new rule class to a rule set or remove an existing rule class to a rule set;
- add a new rule attribute to a rule class or remove an existing rule attribute from a rule class;
- change the “extends” value for an existing rule class; and/or
- change an attribute's data type.

You do *not* need to regenerate the test classes if your changes are limited to the *implementation* of a rule attribute (i.e. its derivation expressions). The derivations are always processed dynamically from the rule set at run time and are not present in the generated test classes.

4.4.5 Writing and Running Unit Tests

The next step involved in testing a rule set is to write a JUnit test to test the logic in our rule set. Below is an example of a unit test for the `Person` class. This test calculates the `eligible` attribute when the `isStudent` and `age` attributes are specified.

```
import curam.creole.calculator.CREOLETestHelper;
import curam.creole.execution.session.RecalculationsProhibited;
import curam.creole.execution.session.Session;
import curam.creole.execution.session.Session_Factory;
import curam.creole.execution.session.StronglyTypedRuleObjectFactory;
import curam.creole.ruleclass.SampleBenefit.impl.Person;
import curam.creole.ruleclass.SampleBenefit.impl.Person_Factory;
import curam.creole.storage.inmemory.InMemoryDataStorage;
import curam.test.framework.CuramServerTest;


/** Class tests the Person rule class.*/
public class TestPersonEligible extends CuramServerTest {

  private Session session;


  public TestPersonEligible(String arg0) {
    super(arg0);
  }


  /*
   * All tests in this class will use a newly-created session
   * that creates strongly-typed rule objects
   */
  @Override
  protected void setUpCuramServerTest() {
    session = Session_Factory.getFactory().newInstance(
      new RecalculationsProhibited(),
        new InMemoryDataStorage(
          new StronglyTypedRuleObjectFactory()));
  }


  /**
   * Tests that a person's eligibility is correctly calculated
   * when supplied a person's student status and age.
   */
  public void testPersonEligiblity() {

    /* Create a "bootstrap" rule object */
    final Person personObj =
      Person_Factory.getFactory().newInstance(session);

   /**
    * Specify a person with an age of 25.
    */
    personObj.age().specifyValue(25);

    /**
     * Specify a person who is a student.
     */
    personObj.isStudent().specifyValue(true);

    /**
     * Check that the person is eligible
     */
    CREOLETestHelper.assertEquals(
      true, personObj.eligible().getValue());
  }
}
```

This test can be added to the test folder in the component you are working on. You can run the test as a standard JUnit test through the IDE you are using or alternatively you can create an Apache Ant target to run the test for you.

For more information on testing rules see the guide *Cúram Express Rules Reference Manual* .

Chapter 5

Creating Timeline Aware Rules

5.1 What Are Timeline Aware Rules?

Chapter 3 explained what CER point in time rules are. This chapter will explain what CER timeline aware rules are and how to create a timeline aware rule. A CER Timeline is simply a value that varies over time, and it is the simplicity of this concept that allows timelines to be used to great effect in the application.

5.2 Identifying Timeline Aware Rules

Anything that can change over time is potentially a timeline aware rule. To introduce the concept, here is an everyday example of data which can vary over time.

A person's total income will tend to vary over time as the person receives pay rises or moves between employments. Given that a person's income at a point in time can be represented as a Number, then a person's varying income over time can be represented as a timeline of Numbers.

To identify what rules in “Sample Benefit” should be timeline aware you need to decide which rules are required to be calculated on different days. Here is a list of some of the rules in “Sample Benefit” that we could make timeline aware

- Household Income
- Number of eligible members
- Person Income

For more information on identifying timeline aware rules see section 4.4 of the *Cúram Express Rules Reference Manual* .

It is the job of a technical user to make a rule timeline aware. A rule diagram needs to be displayed in the technical view in order for timeline information and menu options to be available in the editor.

5.3 Making Expressions Timeline Aware by Setting Up Timeline Markers

Any rule that changes over time can be made timeline aware. Not every expression used in the derivation of a rule needs to be timeline aware. For example if we were calculating a person's income and we needed to deduct a fixed percentage of that income the fixed percentage would not need to be timeline aware but the person's income would.

In this scenario we can make the rule timeline aware and use timeline markers to mark the values that should be treated as timeline data. In the next example the rule “Total income for a person” is made timeline aware by setting up timeline markers.

Total income for a person

1. Open the rule diagram for the “Person Income” rule.
2. View the diagram in technical view. Remember this is the icon in the top right corner of the diagram.
3. In the Rules Outline View select the rule context menu and select *Make Timeline* from the menu. This rule is now timeline aware.
4. Next we need to mark the expressions in the rule diagram that need to be timeline aware. Open the menu on the rule “All the incomes for this person” and select *Make Timeline Interval* . Do the same for the rule “The countable amount for this income” . Notice the timeline icon appear on the element icon on the right.

5.4 Creating an Existence Timeline

We have seen how anything that can change over time is potentially a timeline aware rule. Existence timelines allow you to specify the dates that a rule can exist as well as what value to use before, during and after these dates. An existence timeline creates a timeline of a specified type from a pair of inclusive start and end dates, either of which is optional.

Create an Existence Timeline

1. Create a new rule called “Student Grants Amount” and set its type to Number. From the *Technical* palette on the right-hand side of the editor, drag an *existencetimeline* element onto the rule element on the diagram. Change the name of the *existencetimeline* to be “Student Grants Amount” .

2. View the diagram in technical view. See the Section 2.2, *CER Editor Layout* .
3. Drag a *Date* from the *Data Types* palette to the start date for the *existencetimeline* . Repeat this step for the end date.
4. Set the dates to be the date a person started college and the date a person finished college.
5. Drag a *Number* from the *Data Types* palette to the before, during and after date ranges.
6. Set the “Value Before Date Range” to be 0.
7. Set the “Value During Date Range” to be 1000.
8. Set the “Value After Date Range” to be 200.

5.5 Testing Timeline Outputs

You should write tests for your timeline attributes.

See Section 4.4, *Testing and Executing Rules* for details on how to write tests.

See `Cúram Express Rules Reference Manual` , *Testing Timeline Outputs* for approaches to testing timeline calculations.

Notices

Programming Interface Information

This publication documents intended programming interfaces that allow the customer to write programs to obtain the services of IBM Cúram Social Pogram Management.

Trademarks