IBM Cúram Social Program Management

Cúram Evidence Generator Modeling Guide

Version 6.0.4

Note

Before using this information and the product it supports, read the information in Notices at the back of this guide.

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 Purpose

The purpose of this document is to outline the entity modelling required when using the Cúram Evidence Generator, as it relies on the existence of certain, attributes, structs and aggregations within the generated code.

Starting with the basic modelling required for all entities, the guide will then outline any additional modelling required for the different metadata patterns available in the generator.

1.2 Prerequisites

In order to model the structs, the reader should

- Be familiar with the Cúram Server Modeling Guide
- Have read the Cúram Temporal Evidence Solutions Guide
- Have read the Cúram Temporal Evidence Developers Guide

1.3 Audience

This guide is targeted at those developers intending to make use of the Cúram Evidence Generator.

Chapter 2

Entity Modeling

2.1 Entity

The entity is the starting point of development for any engineer. Details of how to model an entity are outlined in the Cúram Server Modeling Guide.

In addition to the normal entity modelling, the following settings are required in order that it can work correctly with the generated code.

2.1.1 Code Package

The Code Package for the Entity and all its associated structs must be specified in the model, for example:

```
CODE_PACKAGE=seg.evidence.entity
```

Note

The code package must correspond with the product.ejb.package property (see Cúram Evidence Generator Cookbook, section 8.2)

2.1.2 Optimistic Locking

Optimistic locking must be switched on at the entity level. This is because the temporal evidence solution, which will be interacting with the entity, relies on database controlled versioning.

2.1.3 Required Attributes

The evidence generator relies on the existence of certain attributes to execute successfully.

Key Field

The key field of the entity must be named `evidenceID` . The rationale for this is that it results in a lot less generated entity key structs on the server side.

2.1.4 Required Operations

The evidence generator relies on the existence of certain operations to execute successfully.

insert

The `insert` operation should use the stereotype `insert` .

Auto ID

The Auto ID setting should be turned on for the `evidenceID` so as to generate the unique identifier for the insertion of records onto the database. The evidence generator assumes that this is turned on.

Pre Data Access Operation

This option should be set to `Yes`

modify

The `modify` operation should use the stereotype `modify` .

Pre Data Access Operation

This option should be set to `Yes`

Optimistic Locking

This option should be set to `Yes`

read

The `read` operation should use the stereotype `read` .

remove

The `remove` operation should use the stereotype `remove` .

2.1.5 Customizing an Out-Of-The-Box Evidence Entity

Customizing an out-of-the-box evidence entity involves creating an entity extension in the custom model as described in the `Cúram Server Modeling Guide` .

Code Package

The code package for the extension must be specified in the model, for example:

```
CODE_PACKAGE=custom.seg.evidence.entity
```

Note

The code package must correspond with the product.ejb.package property (see **Cúram Evidence Generator Cookbook, section 8.2**) that is configured in the out-of-the-box product, prepended with the text **custom.** , as above.

2.2 Required Structs

Rather than creating similar / identical structs at each layer, the evidence generator uses the structs created at the entity layer to pass data through to the façade layer. For this reason it is important for the generator that certain structs are created and named with the correct naming convention. It should also be noted that additional aggregations will be required under certain conditions. These conditions are explained in detail in the next section.

2.2.1 <EntityName>EvidenceDetails

A struct named **<EntityName>EvidenceDetails** must be created. This struct must have no attributes of its own, and must include the following aggregations:

Object	Aggregation Name	Multiplicity
The entity being modeled	dtls	1:1
core.sl.EvidenceDescriptorDetails	descriptor	1:1
core.sl.CaseIDKey	caseIDKey	1:1

Table 2.1 Required Aggregations

2.2.2 Read<EntityName>EvidenceDetails

A struct named **Read<EntityName>EvidenceDetails** must be created. This struct must have no attributes of its own, and must include the following aggregations:

Object	Aggregation Name	Multiplicity
The entity being modeled	dtls	1:1

Object	Aggregation Name	Multiplicity
core.sl.EvidenceDescriptorDetails	descriptor	1:1

Table 2.2 Required Aggregations

2.3 Build Process

The reason no additional modeling is required beyond the entity layer is that the evidence generator infers the classes required at service and façade layer. The reason that this is possible is that:

- the temporal evidence solution provides the necessary tools for maintaining evidence records
- the evidence generator uses a combination of the structs that have been created at the entity layer by the developer and a number of structs which are provided by the temporal evidence solution

2.3.1 Service Layer

During the build the generator creates a process class for each evidence entity at the service layer level. The process class created has at least the following operations

- create<Entity Name>
- read<Entity Name>
- modify<Entity Name>

There may be more functions created to deal with some of more specialized scenarios but these are generated without the developer's knowledge and the implemented code required to execute these functions is also generated by the evidence generator.

2.3.2 Façade Layer

At the façade layer the evidence generator generates a single process class per product which contains all of the functions required for evidence maintenance. For each single entity at least the following functions are added to this process class:

- create<Entity Name>Evidence
- read<Entity Name>Evidence
- modify<Entity Name>Evidence

Similar to the service layer, there may be additional functions created to deal with some of the more specialized scenarios but again these will not require any input from the developer.

Note

No list<Entity Name>Evidence function is listed above as the generic listEvidence function on the Evidence facade is used instead. This also accounts for no list<Entity Name> function on the Service Layer.

Chapter 3

Parent / Child Relationships

3.1 Introduction

If an evidence entity is taking the role of a child in a Parent / Child relationship the following additional aggregations must be specified

3.2 Additional Aggregations

3.2.1 <EntityName>EvidenceDetails

This struct must now also aggregate the following structs:

Object	Aggregation Name	Multiplicity
core.sl.EvidenceKey	parEvKey	1:1
core.sl.ParentSelectDetails	selectedParent	1:1

Table 3.1 Additonal Aggregations

Chapter 4

Multiple Mandatory Parent Relationships

4.1 Introduction

If an evidence entity is taking the role of a child with multiple mandatory parents, the following additional aggregations must be specified

4.2 Additional Aggregations

4.2.1 <EntityName>EvidenceDetails

This struct must now also aggregate the following struct for each of the mandatory parent types:

Object	Aggregation Name	Multiplicity
core.sl.EvidenceKey	<Parent Entity Name>ParEvKey	1:1

Table 4.1 Additonal Aggregations

Note

It should be noted that the first letter of the above aggregation name should be lowercase, in keeping with standard Java® naming practices

4.2.2 Read<EntityName>EvidenceDetails

This struct must now also aggregate the following struct for each of the mandatory parent types:

Object	Aggregation Name	Multiplicity
core.sl.ParentEvidenceLink	<Parent Entity Name>ParentEvidenceLink	1:1

Table 4.2 Additonal Aggregations

Note

It should be noted that the first letter of the above aggregation name should be lowercase, in keeping with standard Java naming practices

Chapter 5

Pre Association Relationships

5.1 Introduction

If the developer requires the use of the PreAssociation pattern then the following additional modeling is required.

5.2 Additional Aggregations

5.2.1 <EntityName>EvidenceDetails

This struct must now also aggregate the following struct:

Object	Aggregation Name	Multiplicity
core.sl.EvidenceKey	preAssocKey	1:1

Table 5.1 Additonal Aggregations

Chapter 6

Case Participant Attributes

6.1 Introduction

If adding a case participant attribute to the entity, some further aggregations are required to allow the details be added correctly

6.2 Additional Aggregations

6.2.1 <EntityName>EvidenceDetails

This struct must now also aggregate the following struct:

Object	Aggregation Name	Multiplicity
core.sl.CaseParticipantDetails	caseParticipant-Details	1:1

Table 6.1 Additonal Aggregations

6.2.2 Read<EntityName>EvidenceDetails

This struct must now also aggregate the following struct:

Object	Aggregation Name	Multiplicity
core.sl.ReadCaseParticipantDetailss	caseParticipant-Details	1:1

Table 6.2 Additonal Aggregations

6.3 Additional Case Participant Attributes

In certain circumstances it might be a business requirement to have a case participant, other than the primary case participant, stored as a piece of evidence data. A simple example of this would be for a piece of evidence named 'Medical Report'. Not only is it necessary to store the ID of the person for whom the medical report was commissioned, it's also necessary to store the ID of the medical practitioner who compiled the report. Other examples of case participants might be Education Faculties, Unions or Employers.

As the above scenario has in the past caused difficulties for developers, a new feature now exists which allows a developer to flag an attribute, via metadata, as being a special 'case participant' attribute. This means that this attribute will store the role ID of the case participant. The developer must provide the `name` attribute in the `CaseParticipant` element of the EUIM metadata, and use this name when aggregating the structs.

6.3.1 Additional Aggregations

In order to facilitate the generator in its handling of this special flag, the two required structs must aggregate some additional structs.

`<EntityName>EvidenceDetails`

This struct must now also aggregate the following structs:

Object	Aggregation Name	Multiplicity
core.sl.CaseParticipantDetails	<name>CaseParticipantDetails	1:1

Table 6.3 Additonal Aggregations

`Read<EntityName>EvidenceDetails`

This struct must now also aggregate the following structs:

Object	Aggregation Name	Multiplicity
core.sl.ReadCaseParticipantDetails	<name>CaseParticipantDetails	1:1

Table 6.4 Additonal Aggregations

Chapter 7

Related Entity Attributes

7.1 Introduction

In certain circumstances it might be a business requirement to have a field value from a related entity available either for display or for use when maintaining an entity. This can act as a helpful hint for users, when entering information.

Typically the information used would be from a parent evidence record.

For example, displaying the remaining unallocated amount of an income record, when attempting to allocate this income against expenses.

7.2 Additional Structs

In a scenario similar to this, an additional struct must be created at the entity layer to hold the related information.

7.2.1 <EntityName>RelatedEntityAttributesDetails

This struct must have as attributes any attribute to be shared between the entities and this attribute must be of the appropriate type.

7.3 Additional Aggregations

7.3.1 Read<EntityName>EvidenceDetails

This struct must now also aggregate the following structs:

Object	Aggregation Name	Multiplicity
<EntityName>RelatedEntityAttributesDetails	relatedEntityAttributes	1:1

Table 7.1 Additonal Aggregations

Chapter 8

Non Evidence Attributes

8.1 Introduction

If an entity is using the Non Evidence Details pattern, an extra struct must be modeled and aggregated into the standard evidence structs

8.2 Additional Structs Required

8.2.1 <EntityName>NonEvidenceDetails

This struct must be modeled and hold all the extra attributes required for this entity.

8.3 Additional Aggregations

8.3.1 <EntityName>EvidenceDetails

This struct must now also aggregate the following struct:

Object	Aggregation Name	Multiplicity
<EntityName>NonEvidenceDetails	nonEvidenceDetails	1:1

Table 8.1 Additonal Aggregations

8.3.2 Read<EntityName>EvidenceDetails

This struct must now also aggregate the following struct:

Object	Aggregation Name	Multiplicity
<EntityName>NonEvidenceDetails	nonEvidenceDetails	1:1

Table 8.2 Additonal Aggregations

Chapter 9

Non Modifiable Entities

9.1 Introduction

If the developer requires that the entity should not be modifiable then additional modeling is required.

9.2 Additional Aggregations

9.2.1 Read<EntityName>EvidenceDetails

This struct must now also aggregate the following struct:

Object	Aggregation Name	Multiplicity
curam.core.sl.infrastructure.struct.ECWarningsDtlsList	warnings	1:1

Table 9.1 Additonal Aggregations

Notices

This information was developed for products and services offered in the U.S.A. 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

Armonk, NY 10504-1785

U.S.A.

For license inquiries regarding double-byte (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.

1623-14, Shimotsuruma, Yamato-shi

Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: 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 states 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 typograph-

ical 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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply 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:

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources.

IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products

should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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

COPYRIGHT LICENSE:

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

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

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