IBM Cúram Social Program Management

Utilisation du moteur de mappage de données

Version 6.0.5

IBM Cúram Social Program Management

Utilisation du moteur de mappage de données

Version 6.0.5

Important
Avant d'utiliser ces informations et le produit associé, lisez les informations dans «Remarques», à la page 37

Dernière révision : mai 2013

Cette édition s'applique à IBM Cúram Social Program Management v6.0 5 et à toutes les versions ultérieures, sauf indication contraire dans de nouvelles éditions.

Table des matières

Figures

Tableaux

Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	IBM Canada
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.

OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

France	Canada	Etats-Unis
(Pos1)		Home
Fin	Fin	End
(PgAr)		PgUp
(PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
(Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

Brevets

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

Assistance téléphonique

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

Chapitre 1. Introduction

1.1 Finalité

Le but du présent guide est de décrire l'utilisation du moteur de mappage de données Cúram (CDME) pour mapper les données d'un citoyen capturées au cours du processus d'examen préalable et d'admission à l'un des éléments suivants :

- Entités de preuve et de non preuve contenues dans des dossiers Cúram, ou
- Formulaires de demande remplis au format PDF

Les deux options rationalisent la demande de droits à différentes prestations du citoyen. Le mappage de données à une preuve de dossier permet à ces données d'être utilisées pour déterminer l'éligibilité d'une demande dans le cadre du traitement de dossier Cúram. Le mappage de données à un formulaire de demande PDF accélère le processus de remplissage du formulaire pour une soumission manuelle.

Le présent guide doit être utilisé avec le Guide de l'éditeur de mappage de données IBM Cúram. L'éditeur de mappage de données est l'outil le plus simple qui permet de créer rapidement des mappages à des preuves. L'éditeur de mappage de données sauvegarde ces mappages en langage de mappage XML. Ce guide décrit les détails du langage de mappage XML, qui s'avère utile pour comprendre comment les mappages existants sont exécutés, pour analyser davantage de mappages avancés, pour créer des mappages à des formulaires PDF et pour conserver des mappages plus anciens qui peuvent ne pas être compatibles avec l'éditeur de mappage de données.

1.2 Public concerné

Le mappage de données est un projet de collaboration entre les analystes métier et les développeurs. Le rôle d'un analyste métier consiste à mapper de manière logique les données d'un citoyen à des zones d'un formulaire de demande PDF ou à des entités de preuve. Le rôle d'un développeur consiste à traduire les efforts de l'analyste métier en rédigeant des spécifications de mappage et des configurations de mappage en langage XML.

1.3 Prérequis

Le tableau ci-dessous répertorie les prérequis pour le moteur de mappage de données et fournit une liste de lectures recommandées pour en savoir plus sur ces prérequis :

Tableau 1. Prérequis pour le moteur de mappage de données Cúram

Prérequis	Lecture recommandée	Public visé
XML	http://www.w3.org/XML/	Développeurs
Dossiers Cúram	Description des entités de dossier dans le modèle de référence principal ; `Guide de gestion des dossiers intégrés Cúram`	Analystes métier
	`Inside Eligibility and Entitlement Using Cúram Express Rules`	Développeurs
Participants Cúram	Description des entités de participant dans le modèle de référence principal ; `Guide des participants Cúram`	Analystes métier
Magasin de données Cúram (CDS)	`Creating Datastore Schemas`	Développeurs
IBM Cúram Universal Access™	`Universal Access Customization Guide`	Développeurs

Tableau 1. Prérequis pour le moteur de mappage de données Cúram (suite)

Prérequis	Lecture recommandée	Public visé
Preuves Cúram	Descriptions des entités de preuve dans le modèle de référence principal ; `Guide Cúram - Preuves`	Analystes métier
	`Designing Cúram Evidence Solutions`	Développeurs
Formulaires PDF	http://www.adobe.com/products/acrobat/?promoid=BPDDU	Développeurs

1.4 Chapitres contenus dans ce guide

Les chapitres suivants sont contenus dans le présent guide :

Chapitre 2 - Compréhension du mappage de données
Ce chapitre décrit comment le moteur de mappage de données Cúram convertit les données d'un citoyen (capturées dans le portail d'Universal Access et stockées dans le magasin de données Cúram) en zones dans des formulaires de demande PDF ou en entités de preuve dans des dossiers.

Chapitre 3 - Rédaction de spécifications et de configurations de mappage pour des preuves statiques et des PDF
Ce chapitre décrit comment rédiger des spécifications de mappage et des configurations de mappage en langage XML pour des preuves statiques et des PDF. Les spécifications de mappage mappent des données provenant d'un formulaire à un autre, par exemple d'entités de la structure du magasin de données Cúram à des entités de la base de données. Les configurations de mappage décrivent comment remplir des demandes au format PDF ou comment convertir des données en entités de preuve.

Chapitre 4 - Rédaction de spécifications et de configurations de mappage pour des preuves dynamiques
Ce chapitre décrit comment rédiger des spécifications de mappage et des configurations de mappage en langage XML pour des preuves dynamiques.

Chapitre 5 - Mappage à des tiers
Ce chapitre décrit comment procéder au mappage de tiers en tant que participants d'un dossier dans des dossiers.

Annexe A - Schéma des spécifications de mappage
Cette annexe définit les règles grammaticales à suivre lors de la rédaction de spécifications de mappage en langage XML.

Annexe B - Schéma des configurations de mappage
Cette annexe définit les règles grammaticales à suivre lors de la rédaction de configurations de mappage en langage XML.

Annexe C - Conformité
Cette annexe décrit comment développer du code de manière conforme.

Chapitre 2. Compréhension du mappage de données

2.1 Introduction

Le moteur de mappage de données Cúram (CDME) utilise le portail de Cúram Universal Access pour aider les citoyens à examiner leurs droits à différentes prestations. Au cours du processus d'examen préalable et d'admission, un citoyen soumet ses renseignements. Le moteur CDME est chargé de convertir les renseignements de ce citoyen soit en formulaire demande au format PDF (rempli avec les détails du citoyen), soit en dossier Cúram contenant de nouvelles entités de preuve.

Le moteur CDME fonctionne comme suit. Lorsque le citoyen soumet ses renseignements, ces derniers sont stockés dans le magasin de données Cúram (CDS). Le moteur CDME lit les données et utilise des règles provenant d'une spécification de mappage pour transformer les données en texte lisible à l'aide d'un générateur de demandes. Un générateur de demandes au format PDF utilise une configuration de mappage pour déterminer la manière dont les données du citoyen vont apparaître sur un formulaire de demande PDF. Un générateur de demandes de preuves utilise une configuration de mappage pour appeler l'API de preuve afin de créer les nouvelles entités de preuve pour le nouveau dossier.

2.2 Compréhension du mode de stockage des données dans le magasin de données Cúram

Au cours du mappage des données du citoyen à une preuve de dossier ou un formulaire de demande PDF, vous devez commencer par examiner la sortie souhaitée pour les données du citoyen, à savoir une preuve de dossier ou un formulaire de demande complété. L'étape suivante consiste à examiner la méthode de stockage des données d'un citoyen dans le magasin de données Cúram dans le cadre du processus d'examen préalable et d'admission. Une fois que vous avez identifié les informations requises dans la preuve de dossier ou le formulaire de demande PDF et les informations stockées dans le magasin de données Cúram, vous pouvez créer des mappes logiques entre les formulaires de données.

Le but du processus d'examen préalable et d'admission est d'aider les citoyens à déposer une demande de droits. Un formulaire de demande ou une preuve de dossier nécessite au minimum les noms des membres d'un foyer et leurs relations à la personne qui dépose la demande de droits. Les scripts IEG capturent ces informations et les stocke dans le magasin de données Cúram conformément à un schéma prédéfini. La rubrique 2.2, «Compréhension du mode de stockage des données dans le magasin de données Cúram» affiche un exemple de structure de données dans le magasin de données Cúram pour une demande de droits unique et inclut les membres d'un foyer et leurs relations entre eux.

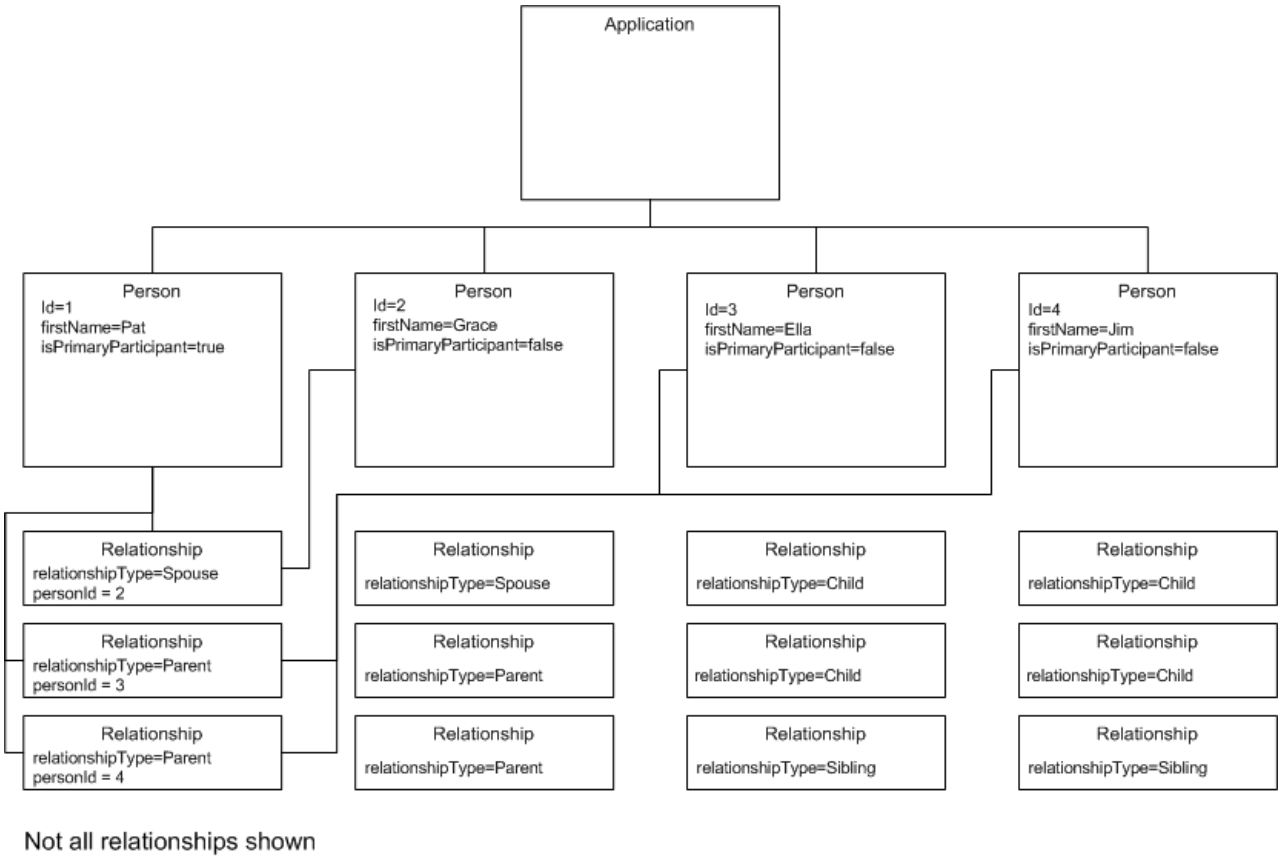

Figure 1. Exemple de structure de données dans le magasin de données Cúram

2.3 Création de mappes logiques

Une mappe logique décompose les renseignements d'une personne stockés dans le magasin de données Cúram en entités de preuve de dossier : membre du foyer, condition de logement et handicap.

L'une des étapes de la création d'une mappe logique consiste à reconnaître les règles métier qui peuvent affecter la façon dont le moteur CDME mappe les données. Par exemple si le citoyen indique qu'il ou elle est aveugle et handicapé(e), les règles métier dictent que deux enregistrements de preuve d'handicap doivent être créés pour le citoyen (le premier pour la cécité, le second pour le handicap).

2.4 Analyse des problèmes de validation dans le mappage de données

Dans le cadre du processus d'examen préalable et d'admission, il est nécessaire de trouver le juste milieu entre la validation d'une preuve de sorte qu'elle puisse être correctement insérée et l'assurance que des questions inutiles ne sont pas posées au citoyen. L'une de ces approches consiste à vérifier qu'au cours de l'admission, la preuve est créée avec des valeurs de validation temporaires, définies par défaut ou minimales, qui peuvent être mises à jour ultérieurement.

Chapitre 3. Rédaction de spécifications et de configurations de mappage pour des preuves statiques et des PDF

3.1 Introduction

Les développeurs peuvent utiliser la mappe de données logiques comme spécification des exigences pour rédiger les spécifications de mappage et les configurations de mappage. Une spécification de mappage décrit comment mapper des données stockées dans une structure spécifique à d'autres données. Chaque spécification de mappage fait référence à une source, d'où les données proviennent, et à une cible où les données sont acheminées.

Bien que la spécification de mappage contienne les règles nécessaires à la transformation des données d'un formulaire à un autre, davantage d'informations sont requises pour convertir les données transformées en preuve de dossier ou en formulaires de demandes au format PDF complétés. Ces renseignements supplémentaires sont fournis dans la configuration de mappage. Le présent chapitre fournit des exemples de rédaction de spécifications de mappage et de configurations de mappage.

3.2 Rédaction des spécifications de mappage

Les spécifications de mappage sont utilisées par le moteur CDME pour transformer des données du magasin de données Cúram dans un autre formulaire. Toutes les données contenues dans la demande d'admission et d'examen préalable du citoyen peuvent être récupérées en lisant l'entité Application, les données contenues dans son entité enfant, dans les enfants de ses enfants, etc. Le moteur de mappage de données passe à travers les données dans le magasin de données Cúram et applique les règles exprimées en code XML pour procéder à la transformation des données.

3.3 Spécification de mappage simple

Cette spécification de mappage simple mappe une entité de personne dans le magasin de données Cúram à une entité de preuve de membre du ménage :

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <map xmlns="http://www.curamsoftware.com/schemas/GUMBO/Map"
3     name="TestMapping">
4    <map-entity source="Person">
5        <target-entity name="HouseholdMember"
         id="HouseholdMemberTarget">
6            <map-attribute from="isNativeAmerican"
             to="natAlaskOrAmerInd"/>
7            <map-attribute from="comments" to="comments"/>
8        </target-entity>
9    </map-entity>
10 </map>
```

La ligne 4 indique la source du mappage tandis que la ligne 5 indique la cible. Cette règle peut être paraphrasée comme suit : "Pour chaque entité Person rencontrée dans le magasin de données Cúram, créer une entité HouseholdMember correspondante". L'élément <target-entity> contient deux éléments <map-attribute> aux lignes 6 et 7.

L'élément <map-attribute> à la ligne 6 indique que l'attribut isNativeAmerican de l'entité Person est mappé à l'attribut natAlaskOrAmerInd de l'entité HouseholdMember. Les attributs ne sont pas mappés à moins qu'il n'existe un élément <map-attribute> spécifique. Voilà pourquoi la ligne 6 indique que l'attribut comments de l'entité Person est mappé à l'attribut comments de HouseholdMember.

Dans certains cas, il est nécessaire de spécifier qu'un mappage se produit uniquement selon des circonstances particulières. Par exemple, une entité HeadOfHousehold devrait uniquement être créée dans le système cible lorsque le mappage rencontre une entité Person dans le magasin de données Cúram dans lequel un indicateur isPrimaryParticipant est défini sur true. L'exemple ci-dessus peut être étendu de sorte à inclure cette règle, comme suit :

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <map xmlns="http://www.curamsoftware.com/schemas/GUMBO/Map"
3     name="TestMapping">
4     <map-entity source="Person">
5         <target-entity name="HouseholdMember"
          id="HouseholdMemberTarget">
6             <map-attribute from="isNativeAmerican"
              to="natAlaskOrAmerInd"/>
7             <map-attribute from="comments" to="comments"/>
8         </target-entity>
9     </map-entity>
10    <condition expression="Person.isPrimaryParticipant==true">
11        <target-entity name="HeadOfHousehold/>
12    </condition>
13 </map>
```

3.4 Mappage des expressions de condition

Dans cet exemple, les valeurs Yes, No et Unanswered sont représentées sous la forme de valeurs de la table de codes et sont utilisées pour enregistrer le statut d'étranger de la personne : Citoyen américain ou non. La valeur ITYN4001 correspond à la réponse "Yes" du client à cette question. Notez l'utilisation de " : les symboles de guillemets "" ne peuvent pas être utilisés directement dans le code XML. La syntaxe du mappage conditionnel d'attributs est strictement identique.

```
1    <condition expression="Person.isBlind==&amp;quot;ITYN4001&amp;quot;">
2      <target-entity
3        name="Disability"
4        id="BlindDisabilityTarget"
5      >
6        <set-attribute
7          name="disabilityType"
8          value="DT1"
9        />
10      </target-entity>
11   </condition>
```

3.5 Mappage des valeurs des tables de codes

Dans certains mappages, les valeurs de table de codes enregistrées dans le magasin de données Cúram peuvent être directement converties en valeurs de table de codes dans le modèle cible. Une section peut alors être utilisée au début du script de mappage pour spécifier les mappages de la table de codes. Par exemple :

```
1  <map-code-table source-codetable="CITIZENSTATUS"
   target-codetable="AlienStatus">
2    <map-value source="US1" target="AS4"/>
3    <map-value source="US2" target="AS1"/>
4  </map-code-table>
```

Dans l'exemple 5, les valeurs provenant de la table de codes CITIZENSTATUS sont mappées aux valeurs de la table de codes AlienStatus.

3.6 Mappage à plusieurs entités cibles

Dans certains cas, il est nécessaire de créer un groupe d'entités cibles. Cette opération est généralement effectuée lors de la création d'un groupe d'entités de preuve, dont l'une d'entre elles est une entité parent, et les autres sont des entités enfants de cette entité de preuve parent. Consultez l'exemple ci-dessous pour obtenir un exemple de création de groupes d'entités cibles liées.

```
       1
       <target-entities>
2      <target-entity
3        name="BusinessAsset" id="BusinessAssetTarget"
4        type="parent"
5      >
6        <map-attribute
7          from="resourceAmount"
8          to="amount"
9        />
10       <map-attribute
11         from="amountOwed"
12         to="amountOwed"
13       />
14     </target-entity>
15     <target-entity
16       name="Ownership" id="OwnershipTarget"
17       type="child"
18     >
19       <set-attribute
20         name="percentageOwned"
21         value="100.0"
22       />
23     </target-entity>
24  </target-entities>
```

Dans cet exemple, deux entités sont créées. L'entité BusinessAsset est une entité de preuve parent, tandis que l'entité Ownership est un enfant. Cette modélisation du mappage garantit que les modèles d'entité de preuve parent/enfant corrects sont respectés lorsque la preuve est créée par le générateur de demandes de preuves.

3.7 Mise en correspondance d'une entité parent et de plusieurs entités enfants

A la rubrique 3.6, «Mappage à plusieurs entités cibles», un groupe d'entités cibles a été créé dans lequel les entités enfant et parent étaient liées les unes aux autres. Dans certains cas, il est nécessaire de créer une seule entité parent pour l'intégralité du dossier. Toutes les entités enfants suivantes sont liées à la même entité parent. L'exemple ci-dessous illustre cette procédure.

```
1 <target-entities>
2   <target-entity name="HholdMealsGroup" type="parent"
      attachment="case" id="MealGroup">
3     <set-attribute name="groupName" value="sample"/>
4   </target-entity>
5
6   <target-entity name="MealGroupMember" type="child"
      id="MealGroupMember">
7   </target-entity>
8 </target-entities>
```

Dans cet exemple, un HholdMealsGroup et un MealGroupMember sont créés lors de la première exécution de la règle. Chaque fois que la règle est réexécutée, un seul MealGroupMember est créé et associé à la même entité HholdMealsGroup.

3.8 Mise en correspondance des modèles et suivi des associations dans le magasin de données Cúram

Dans le scénario suivant, les clients ont demandé que le moteur de mappage soit utilisé pour remplir un formulaire PDF qui ressemble au formulaire suivant :

Tableau 2. Modèle de formulaire PDF

Nom	**Nom de l'employeur**	**Date de début**	**Salaire annuel avant impôts**
Pat	The Gingerman Bakery	02/01/2004	30000
Grace	Jarmin Pharmaceutical	03/01/2002	50000

Chaque zone de ce formulaire PDF possède une identité unique. Par exemple, la zone contenant le nom, Pat, est identifiée comme Job0.Name. La zone contenant 30000 est identifiée comme Job0.Salary.

Réfléchissez à la méthode de stockage des informations issues de l'admission dans le magasin de données Cúram :

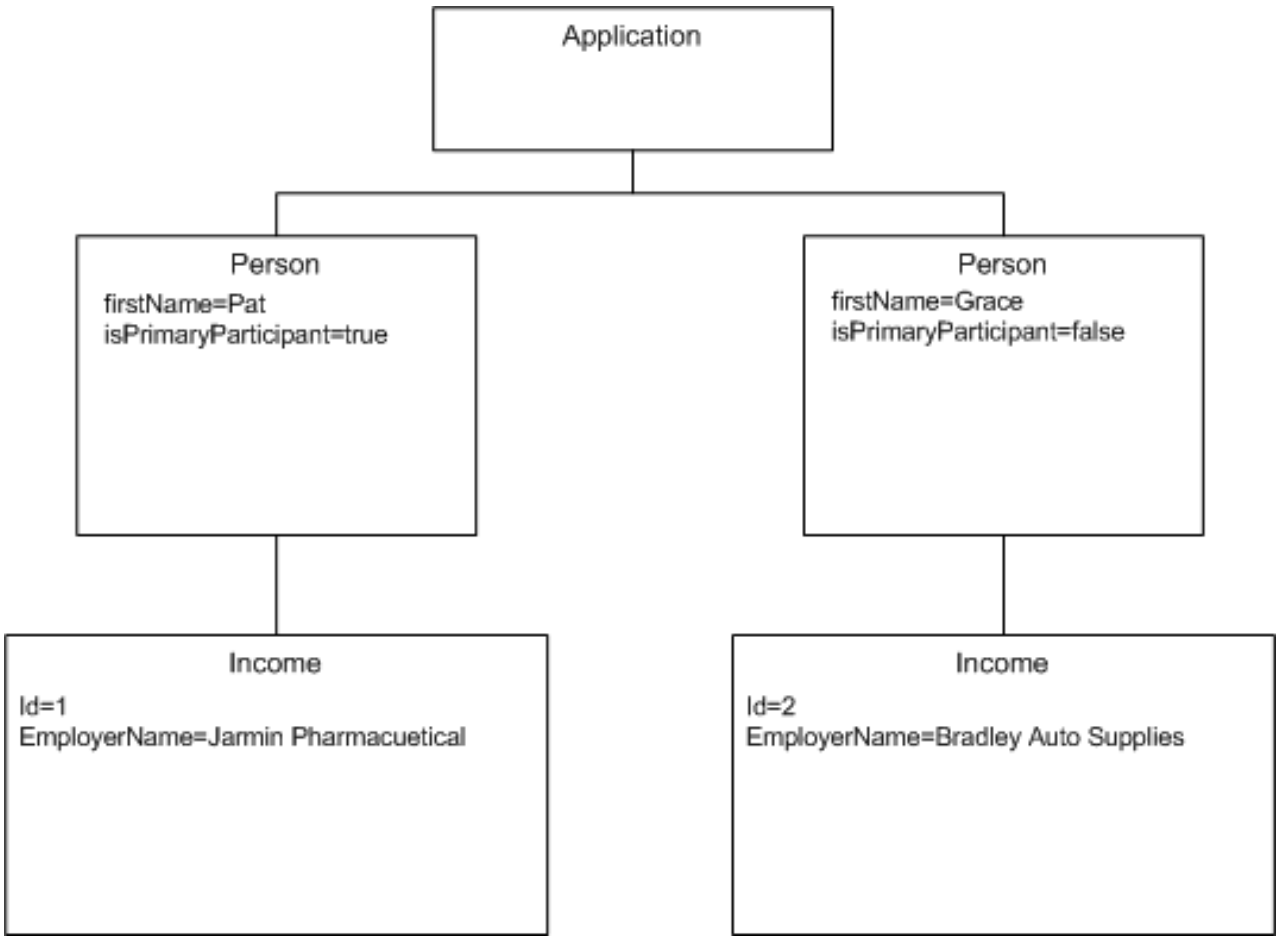

Figure 2. Revenu dans le magasin de données Cúram

Pour pouvoir remplir la zone Name du formulaire PDF ci-dessus, la spécification de mappage doit contenir une règle qui indique que pour chaque revenu (Income) appartenant à une personne (Person), le prénom (firstName) de la personne est spécifié dans la zone Name. En langage de mappage, cette règle peut être exprimée comme suit :

```
1 <map-entity source="Person">
2  <map-entity source="Income">
3    <target-entity name="Job" id="JobTarget">
4      <map-attribute from="firstName" to="Name" entity="Person"/>
5      <map-attribute from="employerName" to="Employer"/>
6      ...
7    </target-entity>
8  </map-entity>
9 </map-entity>
```

Cette règle de mappage peut être paraphrasée comme suit : "Pour chaque entité Income contenue dans une entité Person, créer une entité Target de type Job. L'attribut Name de l'entité Job est mappée à partir de l'attribut firstName de l'entité Person qui contient l'entité Income en cours de mappage."

Remarquez l'utilisation de la syntaxe entity="Person" à la ligne 4 pour indiquer que les attributs firstName proviennent de l'entité Person, et non de l'entité Income. Un exemple plus complexe de ce type de spécification de mappage implique de suivre des associations ou des liens d'une entité à une autre.

3.9 Mappage de membres à un foyer

Le tableau ci-dessous décrit la manière dont les relations sont généralement exprimées dans un formulaire de demande. Vous devez obligatoirement mapper les entités du magasin de données Cúram à un formulaire de demande pré-rempli similaire au formulaire présenté ci-dessous. La difficulté de ce dossier consiste à remplir la zone intitulée "Quelle relation cette personne a-t-elle avec vous ?" Cette zone est abrégée par "RelType" dans cet exemple.

Tableau 3. Membres du foyer

Nom	**Quelle relation cette personne a-t-elle avec vous ?**	**Date de naissance**	**Numéro de sécurité sociale**
Grace	Epouse	02/01/1981	209-57-9943
Ella	Enfant	03/01/2002	987-23-1190

Dans cet exemple, le mappage requis est rédigé comme suit :

```
1  <condition expression="Person.isPrimaryParticipant == true">
2   <map-entity source="Person">
3     <map-entity source="Relationship">
4       <follow-association source="personID">
5         <target-entity name="Householder" id="Householder">
6           <map-attribute from="firstName" to="Name"/>
7           <map-attribute from="relationshipType" to="RelType"
            entity="Relationship"/>
8         </target-entity>
9       </follow-association>
10    </map-entity>
11  </map-entity>
12 </condition>
```

Il peut être paraphrasé comme suit : "Pour chaque relation contenue dans le participant principal, suivre l'association à la personne mentionnée par cette relation. Mapper l'attribut firstName de cette entité Person à la zone Name. Mapper l'attribut relationshipType de l'entité Relationship à la zone RelType." La clé permettant de comprendre l'exemple réside à la ligne 7, où la zone RelType est mappée à partir d'un attribut dans l'entité Relationship.

3.10 Rédaction de configurations de mappage

La présente section fournit des exemples de configuration de mappage pour le générateur de demandes de preuves et le générateur de demandes au format PDF.

3.11 Pour le générateur de demandes de preuves

Lorsqu'il est configuré pour utiliser le générateur de demandes de preuves, le travail du moteur de mappage et divisé en deux phases. Dans la première phase, le moteur de mappage crée un dossier intégré contenant des membres de dossier, puis des relations de rôle de prévenance entre les membres du dossier. Les membres du dossier sont remplis en recherchant des entités dans le magasin de données appelé "Personne". Le moteur de mappage de données est conçu pour traiter toutes les entités du magasin de données commun appelé Personne comme des références à une personne ou une personne candidate dans un dossier Cúram.

En prenant la rubrique 3.11, «Pour le générateur de demandes de preuves» comme exemple, la phase 1 du mappage du générateur de demandes de preuves crée un dossier intégré contenant quatre membres de dossier, où :

- Pat est le participant principal.
- Deux enregistrements de relation de rôle de prévenance sont créés pour indiquer que Grace et Pat sont mariés.
- Le système crée également toutes les autres relations de rôle de prévenance (pour les relations avec les parents et les frères et soeurs).
- Les enregistrements d'adresse et de numéro de téléphone sont également créés.

La phase 2 du processus de mappage concerne la création d'entités de preuve, dont plusieurs exemples sont décrits dans la suite de la présente section.

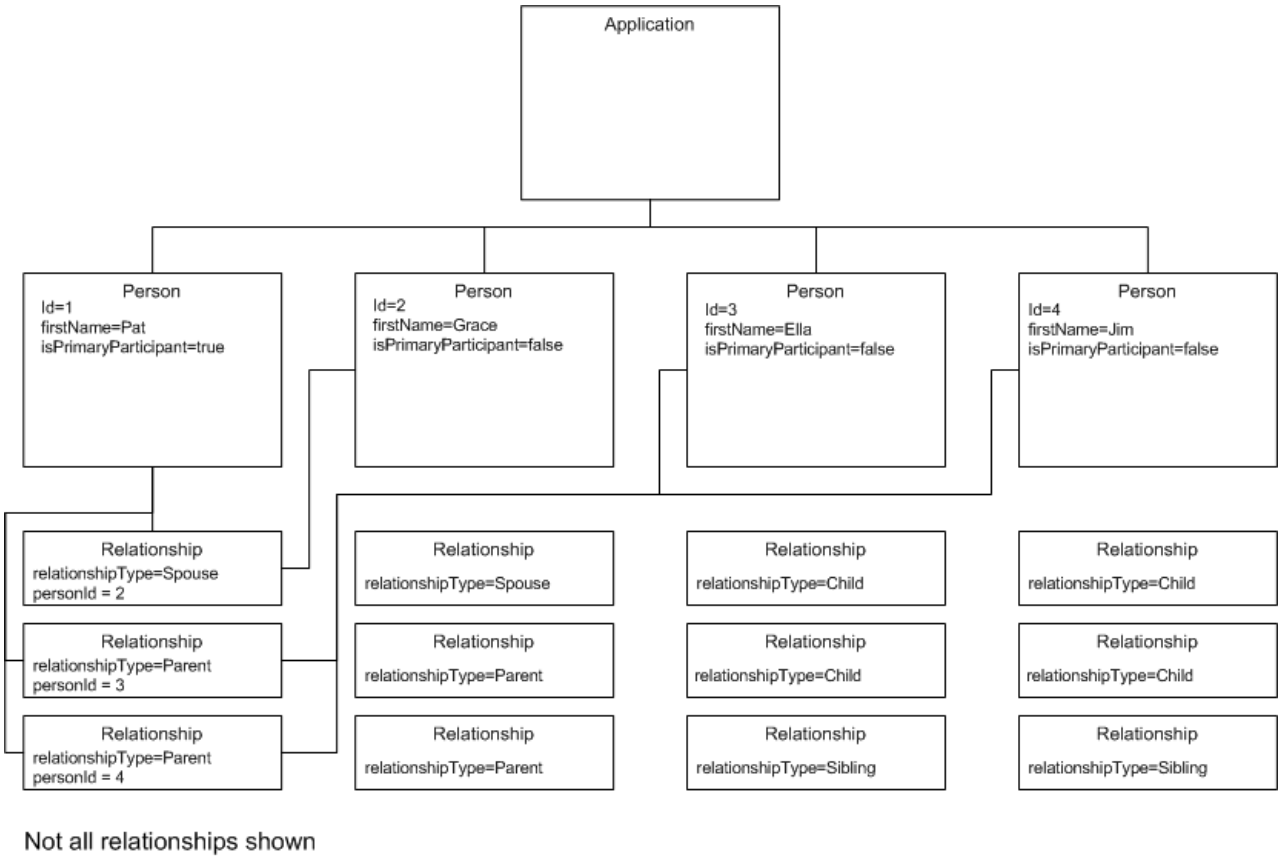

Figure 3. Utilisation de données dans un magasin de données Cúram

3.12 Configuration de mappage simple

L'exemple suivant illustre une configuration pour le générateur de demandes de preuves :

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <application-builder-config
  xmlns="http://www.curamsoftware.com/schemas/GUMBO/ApplicationBuilderConfig">
3   <evidence-config package="curam.evidence">
4     <entity name="HouseholdMember"/>
5     <entity name="HeadOfHousehold"/>
6   <evidence-config package="curam.evidence">
7 </application-builder-config>
```

Dans cet exemple, le générateur de demandes de preuves a été configuré de sorte à créer les preuves HouseholdMember et HeadOfHousehold. A la ligne 3, le nom du module Java de base est spécifié sous la forme curam.evidence. Le générateur de demandes de preuves utilise ces informations pour en déduire les détails suivants sur HouseholdMember :

1. Le nom de la classe de la couche de service de preuve est curam.evidence.service.HouseholdMember.
2. Le nom de l'opération utilisée sur cette classe pour créer la preuve est createHouseholdMemberEvidence().
3. Le nom de la classe transmise à cet appel en tant qu'argument est curam.evidence.entity.struct.HouseholdMemberEvidenceDetails.

Le générateur de demandes de preuves utilise ces informations pour générer la preuve HouseholdMember pour la personne actuelle en cours de traitement.

Les informations ci-dessus sont basées sur l'hypothèse que la preuve est codée conformément à certains modèles. Cette hypothèse est vérifiée si le générateur de preuves est utilisé pour générer la preuve. Il est possible d'utiliser le générateur de demandes de preuves avec une preuve codée manuellement tant qu'elle respecte les modèles utilisés par le générateur de preuves.

3.13 Gestion des zones caseParticipantDetails

Pour pouvoir exécuter l'opération createHouseholdMemberEvidence() sur le membre HouseholdMember, le moteur CDME doit remplir la zone caseParticipantDetails du struct HouseholdMemberEvidenceDetails, dont un extrait est illustré ci-dessous :

```
public final class HouseholdMemberEvidenceDetails
implements java.io.Serializable, curam.util.type.DeepCloneable {

  /** Attribut du struct. */
  public  curam.core.sl.struct.CaseIDKey caseIDKey;

  /** Attribut du struct. */
  public  curam.core.sl.struct.CaseParticipantDetails
    caseParticipantDetails;

  /** Attribut du struct. */
  public  curam.core.sl.struct.EvidenceDescriptorDetails descriptor;

  /** Attribut du struct. */
  public  curam.evidence.entity.struct.HouseholdMemberDtls dtls;
 ...
}
```

Les membres du struct dtls sont, de manière générale, remplis via les éléments <set-attribute> et <map-attribute> dans la spécification de mappage. Par exemple, la ligne suivante de la spécification de mappage mène à la zone natHawOrPaIsInd, remplie avec une valeur du struct dtls :

```
<map-attribute
        from="nativeAlaskanOrAmericanIndian"
        to="natHawOrPaIsInd"
      />
```

La zone caseParticipantDetails se trouve souvent dans un struct EvidenceDetails. Dans cet exemple, un participant du dossier est créé pour Grace et caseParticipantDetails correspond à ce participant. Le moteur de mappage de données effectue cette procédure automatiquement dès qu'il trouve une zone appelée caseParticipantDetails sur le struct EvidenceDetails. Dans certains cas toutefois, la gestion des participants du dossier nécessite des variantes, par exemple lorsque le struct EvidenceDetails contient des participants de dossier supplémentaires qui se réfèrent aux tiers. Tenez compte des points suivants :

```
public final class AnnuityEvidenceDetails
implements java.io.Serializable, curam.util.type.DeepCloneable {
  /** Attribut du struct. */
  public  curam.core.sl.struct.CaseIDKey caseIDKey;

  /** Attribut du struct. */
  public  curam.core.sl.struct.CaseParticipantDetails
    instCaseParticipantDetails;

  /** Attribut du struct. */
  public  curam.core.sl.struct.EvidenceDescriptorDetails descriptor;

  /** Attribut du struct. */
  public  curam.evidence.entity.struct.AnnuityDtls dtls;

  /** Attribut du struct. */
  public  curam.evidence.entity.struct.AnnuityCaseParticipantDetails
    annuityCaseParticipantDetails;
}
```

Dans cet exemple, le participant du dossier qui possède l'annuité est mentionné dans le struct AnnuityCaseParticipantDetails, agrégé sous le nom de zone annuityCaseParticipantDetails. L'institution qui détient l'annuité est décrite dans le struct CaseParticipantDetails et est agrégée sous le nom de zone instCaseParticipantDetails. Cette variante peut être utilisée pour la configuration du générateur de demandes de preuves suivante :

```
1    <entity
2      case-participant-class-name="curam.core.sl.struct.CaseParticipantDetails"
3      case-participant-relationship-name="annuityCaseParticipantDetails"
      name="Annuity"
4    >
5      <ev-field
6        aggregation="instCaseParticipantDetails"
7        referenced-as="participantName"
8        target-name="participantName"
9      />
10     <ev-field
11       aggregation="instCaseParticipantDetails"
12       referenced-as="address"
13       target-name="address"
14     />
15  </entity>
```

Les lignes 2 et 3 indiquent au générateur de demandes de preuves que les caseParticipantDetails de cette entité de preuve dont mentionnés par le nom de zone annuityCaseParticipantDetails à l'aide du struct CaseParticipantDetails. Les lignes 5 à 9 indiquent au générateur de demandes de preuves que la zone participantName du struct agrégé instCaseParticipantDetails peut être référencé dans la spécification de mappage en tant que "participantName" (ligne 7). Même chose pour l'adresse institutionnelle aux lignes 10 à 14. L'exemple suivant permet de mapper le nom et l'adresse de l'institution détenant l'annuité :

```
1  <target-entity name="Annuity" id="AnnuityTarget">
2   <map-attribute
3     from="institutionName"
```

```
4     to="participantName"
5   />
6   <map-attribute
7     from="institutionAddress"
8     to="address"
9   />
10 </target-entity>
```

Dans certains cas, il peut paraître trop lourd de demander au client de remplir ces différents types d'information tierce dans le cadre d'une admission en ligne. La spécification de mappage peut ainsi être utilisée pour définir une valeur par défaut pour ces valeurs, qui peuvent être correctement remplies au moment de l'entretien. L'exemple suivant décrit comment définir une valeur par défaut pour les valeurs d'un participant tiers, comme une institution financière :

```
1 <target-entity name="Annuity" id="AnnuityTarget">
2   <map-attribute
3          from="resourceAmount"
4          to="annuityValue"
5        />
6        <set-attribute
7          name="participantName"
8          value="Unknown"
9        />
10       <set-attribute
11         name="address"
12         value="curam.blankaddress"
13       />
14     </target-entity>
```

La valeur curam.blankaddress à la ligne 12 entraîne l'insertion d'une adresse vide pour le participant.

3.14 Définition des identificateurs d'entité cible

A la ligne 1 de l'exemple 13, comme pour un certain nombre des exemples précédents, l'élément <target-entity> inclut un attribut id "AnnuityTarget". Bien que cet attribut soit facultatif, il est recommandé d'inclure un attribut id dans tous les éléments <target-entity>. Il permet au moteur de mappage de données de faire la différence entre les différents mappages de la même entité au même type d'entité cible. Prenons l'exemple suivant : l'entité Person du magasin de données commun possède deux indicateurs booléens : isBlind et hasDisability. Ces deux indicateurs mappent au même type d'entité cible, Disability, comme suit :

```
1  <map-entity source="Person">
1    <condition expression="Person.isBlind==true">
2      <target-entity
3        id="DisabilityBlind"
4        name="Disability"
5      >
6        <set-attribute
7          name="disabilityType"
8          value="DT1"
9        />
10     </target-entity>
11   </condition>
12   <!-- Créez un enregistrement d'handicap vide. -->
13   <condition expression="Person.hasDisability==true">
14     <target-entity
15       id="DisabilityUnspecified"
16       name="Disability"
17     />
18   </condition>
19 </map-entity>
```

La première cible aux lignes 1 à 11 garantit que si un demandeur indique qu'il est aveugle, un enregistrement d'handicap de type cécité est créé. La deuxième cible, aux lignes 13 à 18, vérifie

l'indicateur hasDisability. S'il est défini sur true, un enregistrement Disability de type non spécifié est créé. En affectant un attribut id différent aux deux mappages, le moteur de mappage peut différencier ces deux mappages. Sans attribut id, le deuxième mappage ne serait pas traité.

3.15 Pour le générateur de demandes au format PDF

Un formulaire PDF contient plusieurs zones de différents types. Chaque zone possède un nom unique. Les services d'espace de travail Cúram utilisent ce nom unique pour référencer la zone de sorte à pouvoir écrire des données dans cette zone. Pour que ce processus fonctionne, l'auteur du formulaire PDF doit nommer ces zones et définir leurs propriétés conformément à certaines conventions. Si ces conventions sont respectées, le générateur de demandes au format PDF pourra mapper des données à ces zones.

ATTENTION : Cette section décrit le générateur de demandes au format PDF personnalisé, qui autorise les mappages à un formulaire PDF adapté. Les clients peuvent également utiliser le générateur de demandes au format PDF générique. Voir la section "Comment personnaliser les PDF génériques pour les applications traitées" du chapitre "Personnalisation de la gestion des applications soumises" dans le Guide de personnalisation des accès universels Cúram.

3.16 Sections et zones

Dans la convention la plus basique, les zones sont regroupées en "Sections". Ces sections ne correspondent pas nécessairement aux sections du formulaire, mais y correspondent dans certains cas. Par exemple :

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <application-builder-config
  xmlns="http://www.curamsoftware.com/schemas/GUMBO/ApplicationBuilderConfig">
3     <pdf-config>
4         <section name="Applicant">
5             <field name="Name" type="append" append-separator=" "/>
6             <field name="SSN"/>
7             <field name="DateofBirth"/>
8             <field name="Gender" type="button-radio"/>
9             <field name="USCitizen" type="button-radio"/>
10            <field name="blackOrAfricanAmerican"
                type="button-checkbox"/>
11            <field name="nativeAlaskanOrAmericanIndian"
                type="button-checkbox"/>
12            <field name="asian" type="button-checkbox"/>
13            <field name="nativeHawaiianOrPacificIslander"
                type="button-checkbox"/>
14            <field name="whiteOrCaucasian"
                type="button-checkbox"/>
15            <field name="EthnicOrigin" type="button-radio"/>
16        </section>
17    </pdf-config>
18</application-builder-config>
```

L'exemple 14 illustre un extrait d'une configuration du générateur de demandes au format PDF. Il fait référence à une section appelée "Applicant". A la ligne 4 de l'exemple 14, le générateur de demandes au format PDF s'attend à ce que le formulaire PDF cible contienne une zone de texte appelée "Applicant.Name". La ligne 8 fait référence à une zone du formulaire PDF appelée "Applicant.Gender". Cette zone est un bouton radio, tandis que les lignes 10 à 14 font toutes référence aux zones qui correspondent à des cases à cocher.

3.17 Remplissage des zones de texte

La ligne 6 de l'exemple 14 correspond à une zone de texte standard brut. Le mappage correspondant pourrait ressembler à ce qui suit :

```
<target-entity name="Applicant">
  <map-attribute from="ssn" to="SSN"/>
</target-entity>
```

La ligne 5 est différente. Le type est marqué comme "append". Cela signifie que la même zone de texte peut être écrite plusieurs fois, et chaque fois que le moteur de mappage écrit dans la zone de texte, le résultat est ajouté à la valeur actuelle de la zone de texte plutôt que de la remplacer. Chaque fois qu'un résultat est ajouté, les nouvelles données sont différenciées des anciennes données par l'élément append-separator, qui correspond dans ce cas à un caractère espace simple. Un fichier de mappage comme celui de l'exemple 16 associé à la configuration de mappage illustrée dans l'exemple 14 remplit la zone Applicant.Name avec le prénom, le deuxième prénom et le nom de famille (par ex. "Pat A Kayek") des demandeurs.

```
<target-entity name="Applicant">
  <map-attribute from="firstName" to="Name"/>
  <map-attribute from="middleInitial" to="Name"/>
  <map-attribute from="lastName" to="Name"/>
</target-entity>
```

L'ajout de zones de texte s'avère également utile pour créer une liste d'éléments séparés par des virgules. Prenons comme exemple une zone qui demande au client de fournir la liste des personnes enceintes dans son foyer. Voici un exemple d'extrait du code XML de mappage :

```
<condition expression="Person.isPregnant == true">
   <target-entity name="Pregnancy">
     <map-attribute from="firstName" to="Pregnancy"/>
     <set-attribute name="HasPregnancies" value="Yes"/>
   </target-entity>
 </condition>
```

La configuration de mappage correspondante est illustrée dans la figure 18. Chaque fois que le moteur de mappage traite une personne du foyer pour laquelle l'indicateur isPregnant est défini sur true, le prénom de cette personne est ajouté à la zone Pregnancy.Pregnancies.

```
<section name="Pregnancy">
  <field name="Pregnancies" type="button-checkbox"/>
  <field name="Pregnancy" type="append" append-separator=", "/>
</section>
```

3.18 Répétition de sections et descriptions de la table de codes

Certains formulaires contiennent des sections répétées, par exemple "Répertoriez les détails de toutes les personnes composant votre foyer" ou "Répertoriez toutes vos sources de revenu issu du travail". Le générateur de demandes au format PDF est conçu pour traiter ces sections, à condition que l'auteur du PDF nomme les zones conformément aux conventions correctes. Par exemple, les zones utilisées pour collecter des données sur les membres du foyer peuvent être nommées comme suit :

Tableau 4. Zones d'un formulaire PDF permettant d'enregistrer les membres d'un foyer

Nom	**Quelle relation cette personne a-t-elle avec vous ?**	**Date de naissance**	**Numéro de sécurité sociale**
OtherPerson0.Name	OtherPerson0 .RelType	OtherPerson0 .DateOfBirth	OtherPerson0 .SSN
OtherPerson1.Name	OtherPerson1 .RelType	ÒtherPerson1 .DateOfBirth	OtherPerson1 .SSN
OtherPerson2.Name	OtherPerson2 .RelType	OtherPerson2 .DateOfBirth	OtherPerson2 .SSN

La configuration de mappage correspondante serait la suivante :

```
1  <section name="Person" type="multiple">
2   <field name="Name" type="append" append-separator=" "/>
3   <field name="RelType" codetable-class="RelationshipTypeCode"/>
4   <field name="DateofBirth"/>
5  </section>
```

Notez que l'attribut type="multiple" à la ligne 1 provoque la répétition de la section. Notez également l'attribut codetable-class à la ligne 3 de cet exemple. Cet attribut très utile provoque la conversion des valeurs de la table de codes en descriptions localisées. Si vous l'utilisez dans le contexte ci-dessus, l'auteur du script vérifie que la deuxième colonne est remplie avec des valeurs localisées telles que "Parent" et "Sibling" plutôt que des codes dénués de sens tels que "RT1" ou "RT3".

3.19 Cases à cocher

Une case à cocher est une zone unique qui peut être soit cochée, soit décochée. Le générateur de demandes au format PDF part du principe que la définition d'une zone de case à cocher sur la valeur "Yes" permet de cocher la case, alors que sa définition sur "No" permet de la décocher. L'auteur du formulaire PDF doit vérifier que cette convention est respectée. Dès que le moteur de mappage mappe une valeur booléenne à une zone de case à cocher, celle-ci est automatiquement mappée comme suit : True mappe à "Yes" et false mappe à "No".

3.20 Boutons radio

Un ensemble de boutons radio est traité comme une zone unique dans un formulaire PDF. Un seul élément peut être sélectionné à la fois par les boutons radio. Les éléments individuels du bouton radio sont sélectionnés en écrivant une valeur spécifique dans le bouton radio. L'auteur du formulaire PDF peut déterminer quel élément est sélectionné en spécifiant une "valeur d'exportation" pour chaque élément. L'une des utilisations typiques du bouton radio avec Cúram consiste à utiliser les valeurs d'exportation pour désigner un nombre d'éléments de la table de codes.

Prenons l'exemple d'un bouton radio utilisé pour désigner la mention Homme ou Femme. Les valeurs de la table de codes pour Homme et Femme sont "SX1" et "SX2" respectivement. L'auteur du PDF crée une zone de bouton radio unique appelée "Applicant.Gender". L'élément "Homme" est désigné par la valeur d'exportation "SX1" tandis que l'élément "Femme" est désigné par la valeur d'exportation "SX2". Le mappage ressemble à ce qui suit :

```
<target-entity name="Applicant">
  <map-attribute from="gender" to="Gender"/>
</target-entity>
```

La configuration de mappage correspondante ressemble à ce qui suit :

```
<section name="Applicant">
   <field name="Gender" type="button-radio"/>
 </section>
```

3.21 Combinaisons de choix

Une combinaison de choix est une liste déroulante d'éléments dans laquelle l'utilisateur peut sélectionner un élément. Le nom de l'élément fournit suffisamment d'informations à l'utilisateur du formulaire pour qu'il décide quel élément sélectionner. Imaginez par exemple que la personne qui conçoit le formulaire souhaite créer une liste déroulante représentant le statut d'étranger d'une personne. Cúram possède une table de codes AlienStatus qui contient des codes correspondant aux descriptions suivantes :

- Etranger
- Citoyen américain
- Etranger sans papiers
- Réfugié
- Citoyen non ressortissant

Le concepteur de formulaires PDF crée une combinaison de choix et définit le texte de chaque élément dans la liste déroulante sur les valeurs ci-dessus. Pour garantir que la description de la table de codes est envoyée au formulaire, plutôt que le code de la table de codes, la configuration suivante est requise :

```
<section name="AlienPerson" type="multiple">
    <field name="CitizenshipStatus" type="choice-combo"
codetable-class="AlienStatus"/>
  </section>
```

3.22 Comment configurer le formulaire de demande PDF

Commencez par charger votre formulaire PDF dans le portail d'Universal Access. Il est important d'utiliser un *formulaire* PDF , et non un document PDF uniquement. Le PDF doit contenir un formulaire, qui lui-même doit inclure des zones. Si vous souhaitez utiliser le générateur de demandes au format PDF, chaque zone du formulaire doit porter un nom unique, par exemple PersonalDetails.surname et Child1Details.surname. Il est conseillé d'utiliser un programme tel que Adobe® Acrobat Writer Professional ou GlobalSCAPE CutePDF Pro pour éditer votre formulaire.

Remarque : si vous utilisez Adobe Acrobat Writer Professional, vérifiez que vous sauvegardez le formulaire sous AcroForm et non XFA. Vous pouvez télécharger votre PDF en vous connectant à Cúram en tant qu'administrateur, en choisissant la section Administration d'Universal Access, puis en sélectionnant Formulaires PDF. Vous pouvez télécharger ici le formulaire et lui donner un nom pratique. Chaque demande d'admission possède un type de demande d'admission. Vous devez ainsi vérifier que votre type de demande d'admission est associé au formulaire PDF correct. Pour ce faire, accédez à Demandes d'admission, sélectionnez le type de demande d'admission approprié et sélectionnez Editer. Cliquez sur la liste déroulante Formulaire PDF et le formulaire PDF que vous venez de charger devrait apparaître dans la liste. Sélectionnez-le. Indiquez ensuite Mappages PDF pour les programmes individuels qui vous intéressent.

Rédigez un bloc de code XML de mappage PDF ainsi qu'un bloc de code XML de configuration de la demande en suivant les instructions fournies dans la section précédente du présent guide. Accédez à l'élément de menu Programmes dans la section Administration d'Universal Access. Une liste de programmes s'affiche. Affichez les programmes qui vous intéressent. Sélectionnez l'onglet Mappage dans la partie supérieure droite. Créez un mappage. Assurez-vous de sélectionner "Création d'un formulaire PDF" plutôt que Création d'une preuve. Téléchargez le fichier de configuration de mappage et le fichier de spécification de mappage. Testez votre mappage en procédant à une admission pour le programme de votre choix. A la fin de l'admission, sélectionnez le lien permettant d'afficher le fichier PDF.

Chapitre 4. Rédaction de spécifications et de configurations pour une preuve dynamique

4.1 Introduction

L'objectif du présent document est de mapper les données de citoyens capturées au cours du processus d'admission dans des entités de preuve dynamique.

Conditions requises :

Le lecteur est supposé connaître les concepts de base des preuves dynamiques. Il est notamment censé comprendre la nature d'une preuve, la définition d'un type de preuve dynamique, la définition d'une version de preuve et les métadonnées XML d'une preuve dynamique.

4.2 Rédaction de spécifications et de configurations pour une preuve dynamique

Une preuve dynamique n'est, par sa nature, pas modélisée. Il s'agit d'une entité unique (ou d'un ensemble d'entités) qui contient des données pour tous les types de preuve dynamique. Un type de preuve dynamique possède une ou plusieurs versions de type de preuve tout au long de sa durée de vie. A n'importe quel moment, une seule version de type de preuve sera effective. Tous les éléments de métadonnées (attributs, relations, etc.) seront définis au niveau de la version de type de preuve. Ainsi, le développeur est chargé de fournir la spécification de mappage correcte ainsi que les configurations de la version de type de preuve actuellement en fonction.

4.3 Métadonnées pour une preuve dynamique simple

Ces métadonnées simples représentent la structure d'un type de preuve dynamique Membre du ménage. Les différents types d'attribut tels que Boolean, Codetable, Date et String sont définis ci-dessous :

```
<?xml version="1.0" encoding="UTF-8"?>
<EvidenceTypeVersion xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="file://DynamicEvidenceMetadata.xsd">
    <Model>
        <Attributes>
            <Attribute>
                <DataAttribute name="blkOrAfrAmerInd">
                    <DomainType dataType="Boolean" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <DataAttribute name="ssnStatus">
                    <DomainType dataType="Codetable">
                        <CodetableOptions codetableName=
                                "SSNApplicationStatus" />
                    </DomainType>
                </DataAttribute>
            </Attribute>
            <Attribute>
                <DataAttribute name="startDate">
                    <DomainType dataType="Date" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <DataAttribute name="endDate">
                    <DomainType dataType="Date" />
                </DataAttribute>
```

```
            </Attribute>
            <Attribute>
                <DataAttribute name="comments">
                    <DomainType dataType="String" />
                </DataAttribute>
            </Attribute>
        </Attributes>
    </Model>
    <Validations>
        <PatternValidations>
        </PatternValidations>
    </Validations>
    <UserInterface />
</EvidenceTypeVersion>
```

4.4 Spécification de mappage simple

Cette spécification de mappage simple mappe les données de l'entité HouseHoldMember dans un magasin de données à une entité de preuve dynamique HouseHoldMember définie dans la section précédente.

Notez que le nom d'attribut mentionné dans la zone 'to' doit correspondre à la zone 'name' de l'élément 'DataAttribute' des métadonnées de preuve dynamique. En d'autres termes, un développeur devrait vérifier que l'attribut auquel les données seront mappées est présent dans les métadonnées de ce type de preuve dynamique.

```
<?xml version="1.0" encoding="UTF-8"?>
<map xmlns="http://www.curamsoftware.com/schemas/GUMBO/Map"
                                        name="EvidenceMapping">
    <map-entity source="HouseHoldMember">
        <target-entity name="HouseHoldMember">
            <map-attribute from="blkOrAfrAmerInd"
                                        to="blkOrAfrAmerInd" />
            <map-attribute from="ssnStatus" to="ssnStatus" />
            <map-attribute from="startDate" to="startDate" />
            <map-attribute from="endDate" to="endDate" />
            <map-attribute from="comments" to="comments" />
        </target-entity>
    </map-entity>
</map>
```

4.5 Configuration de mappage simple

La configuration de mappage simple suivante est définie pour le type de preuve dynamique HouseHoldMember :

```
<?xml version="1.0" encoding="UTF-8"?>
<application-builder-config >
    <evidence-config package="curam.gumbo.evidence">
        <entity name="HouseHoldMember" ev-type-code="DE_HMEMBER"/>
    </evidence-config>
</application-builder-config>
```

Un développeur doit indiquer le code de type de preuve dynamique dans l'attribut 'ev-type-code'. Ceci permet au système de déterminer si la preuve est statique ou dynamique. Si cet attribut est vide ou contient une entrée non valide, le système part du principe que le type de preuve est statique et procède au mappage des données.

4.6 Mappage d'une preuve dynamique parent-enfant

La présente section décrit la procédure de mappage de preuves qui possèdent une relation parent-enfant.

4.7 Métadonnées pour une preuve dynamique parent-enfant simple

Les métadonnées suivantes contiennent des attributs pour le type de preuve dynamique d'adoption. Ces métadonnées d'adoption possèdent deux attributs qui indiquent que cette entité inclut deux zones CaseParticipant associées. La zone "caseParticipantRoleID" est un élément CaseParticipantRole principal et la zone "parCaseParticipantRoleID" est un élément CaseParticipantRole associé.

```
<?xml version="1.0" encoding="UTF-8"?>
<EvidenceTypeVersion xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance" xsi:noNamespaceSchemaLocation="../../../../
DynamicEvidence/source/curam/dynamicevidence/definition/impl/
                        xmlresources/DynamicEvidenceMetadata.xsd">
    <Model>
        <Attributes>
            <Attribute>
                <DataAttribute name="adoptionFinalizedDate">
                    <DomainType dataType="Date" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <RelatedCPAttribute name="caseParticipantRoleID"
                        participantType="Person" volatile="true" />
            </Attribute>
            <Attribute>
                <RelatedCPAttribute name="parCaseParticipantRoleID"
                                    participantType="Person" />
            </Attribute>
        </Attributes>
    </Model>
    <UserInterface>
        <Cluster>
            <RelCPCluster fullCreateParticipant="true">
                <StandardField source="caseParticipantRoleID" />
            </RelCPCluster>
        </Cluster>
        <Cluster>
            <RelCPCluster fullCreateParticipant="true">
                <StandardField source="parCaseParticipantRoleID" />
            </RelCPCluster>
        </Cluster>
    </UserInterface>
</EvidenceTypeVersion>
```

Les métadonnées AdoptionPayment possèdent une relation avec la preuve dynamique parent.

```
<?xml version="1.0" encoding="UTF-8"?>
<EvidenceTypeVersion xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="file://DynamicEvidenceMetadata.xsd"
 javaHookClassNameForCalculatedAttributes="curam.dynamicevidencetest.
 hook.impl.AdoptionPaymentCalcualtedAttributeHook"
 useHookForCalculatedAttributes="true">
    <Model>
        <Attributes>
            <Attribute>
                <DataAttribute name="amount">
                    <DomainType dataType="Money" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <CalculatedAttribute name="parentName">
                    <DomainType dataType="String" />
                </CalculatedAttribute>
            </Attribute>
        </Attributes>
        <Relationships>
            <Relationship>
                <MandatoryParent name="adoptions"
```

```
                        evidenceTypeCode="DET004" />
            </Relationship>
        </Relationships>
    </Model>
</EvidenceTypeVersion>
```

4.8 Spécification de mappage parent-enfant simple

La spécification de mappage possède une relation parent-enfant. Quelques attributs sont également définis pour l'entité Adoption, tous utilisés pour créer un nouveau participant. Les valeurs de ces attributs sont contenues dans le magasin de données Cúram.

```
<?xml version="1.0" encoding="UTF-8"?>
<map xmlns="http://www.curamsoftware.com/schemas/GUMBO/Map"
                                name="ParentChildMapping">
    <map-entity source="Adoption">
        <target-entities>
            <target-entity name="Adoption" type="parent" id="parent">
                <map-attribute from="adoptionFinalizedDate"
                                  to="adoptionFinalizedDate" />
                <map-attribute from="adParentName"
                                       to="adParentName" />
                <map-attribute from="adParentStreet1"
                                     to="adParentStreet1" />
                <map-attribute from="adParentStreet2"
                                     to="adParentStreet2" />
                <map-attribute from="adParentCity"
                                       to="adParentCity" />
                <map-attribute from="adParentState"
                                       to="adParentState" />
                <map-attribute from="adParentZipCode"
                                     to="adParentZipCode" />
            </target-entity>
            <target-entity name="AdoptionPayment"
                                    type="child" id="child">
                <set-attribute name="amount" value="2200" />
            </target-entity>
        </target-entities>
    </map-entity>
</map>
```

4.9 Configuration de mappage simple pour la relation parent-enfant

Les éléments <def-create-participant> et <create-participant> sont présentés ici. Notez qu'un nouvel attribut 'dyn-evidence-primary-cpr-field-name' a été ajouté à l'élément <entity>. A l'aide de cet attribut, le développeur doit spécifier le nom d'attribut de l'élément CaseParticipantRole principal défini dans les métadonnées. Dans cet exemple, 'caseParticipantRoleID' est l'élément CaseParticipantRole qui a été défini dans l'entité Adoption. De même, le nom d'attribut CaseParticipantRole associé ('parCaseParticipantRoleID' dans cet exemple) est défini dans la zone 'name' de l'élément <create-participant>. Remarque : Dans le cas d'une preuve statique, cette même zone 'name' de l'élément <create-participant> est utilisée pour mentionner le nom d'agrégation correspondant.

```
<?xml version="1.0" encoding="UTF-8"?>
<application-builder-config xmlns="http://www.curamsoftware.com/
                          schemas/GUMBO/ApplicationBuilderConfig">
    <evidence-config package="curam.evidence">
     <def-create-participant id="AdoptedParentDetails" type="RL13">
            <participant-name-field name="firstName"
                                 from="adParentName" order="1" />
            <participant-address type="AT3">
                <address-field name="addressLine1"
                                      from="adParentStreet1" />
                <address-field name="addressLine2"
                                      from="adParentStreet2" />
                <address-field name="city" from="adParentCity" />
```

```
                <address-field name="state" from="adParentState" />
                <address-field name="zip" from="adParentZipCode" />
            </participant-address>
        </def-create-participant>
         <entity name="Adoption" ev-type-code="DET004"
         dyn-evidence-primary-cpr-field-name="caseParticipantRoleID">
              <create-participant refid="AdoptedParentDetails"
                          name="parCaseParticipantRoleID" role="" />
          </entity>
          <entity name="AdoptionPayment" ev-type-code="DET005"/>
    </evidence-config>
</application-builder-config>
```

Chapitre 5. Mappage à des tiers

5.1 Introduction

La fonctionnalité d'admission d'Universal Access permet aux utilisateurs d'entrer des détails sur leur situation à l'aide de scripts IEG2. Le script IEG2 insère les détails du client dans un magasin de données. Suite à la soumission, le contenu du magasin de données est mappé dans des données de preuve pour un dossier d'admission. Plusieurs types de preuve font référence à des tiers – ces tiers doivent être insérés dans le dossier en tant que participants du dossier avec leur propre rôle de participant unique. Par exemple, un enregistrement de grossesse peut inclure un élément "Père". Si le père est absent, il peut être enregistré en tant que participant du dossier "personne candidate". Dans un autre exemple, la preuve Etudiant doit être associée à une Ecole. Une école est enregistrée en tant que participant du dossier "Représentant". Ces nouveaux participant du dossier doivent être rapidement créés au cours du mappage et doivent contenir autant d'informations que possible afin de faciliter le processus d'admission pour les agents affectés qui doivent traiter le dossier.

Le mappage des adresses est obligatoire car un participant nouveau ou existant doit être associé à une nouvelle preuve. Le participant est généralement soit un représentant, soit une personne candidate. L'un des défis majeurs pour la création de participants réside dans le mappage des adresses. Les zones d'adresse enregistrées dans le magasin de données, telles que "ADD1", doivent être correctement agrégées dans une structure d'adresse Cúram correctement formatée afin de garantir que le participant peut être correctement créé.

Vous allez vous apercevoir dans le présent chapitre que la logique de création de participants et de mappage d'une adresse à ce participant est dissociée du générateur de demandes de preuves.

5.2 Comment mapper des tiers

Le schéma de configuration du générateur de demandes inclut des éléments et des attributs qui peuvent être utilisés pour créer un participant et lui mapper une adresse.

5.3 Définition du créateur des participants

L'élément <def-create-participant> fait partie de l'élément <evidence-config>. Cet élément est utilisé pour définir le comportement de création d'un participant. Le même comportement peut être réutilisé par plusieurs définitions <entity> via l'attribut id. Il convient de noter que le type de données de tous les attributs mentionnés ici doit être défini sur 'String'.

```
<def-create-participant id="SchoolRepresentative" type="RL13">
  <participant-name-field name="firstName" from="participantName"
                                                  order="1"/>
       <participant-address type="AT3">
            <address-field name="addressLine1" from="street1"/>
            <address-field name="addressLine2" from="street2"/>
            <address-field name="city" from="city"/>
            <address-field name="state" from="state"/>
            <address-field name="zip" from="zipCode"/>
        </participant-address>
</def-create-participant>
```

5.4 Création d'un participant

L'élément <create-participant> a été ajouté dans l'élément <entity>. Cet élément indique au générateur de demandes de créer un participant, comme spécifié dans la définition du créateur de participants.

```
<entity case-participant-class-name="curam.core.sl.struct.
 CaseParticipantDetails"case-participant-relationship-name=
                              "curam.none" name="Student">
          <create-participant refid="SchoolRepresentative"
               name="schCaseParticipantDetails" role="SCH"/>
</entity>
```

5.5 Exemple de schéma de mappage

Le schéma suivant est à suivre lors de la rédaction d'une spécification de mappage. Notez que dans l'entité Education, les attributs sont directement mappés à l'entité de preuve 'Student' (Etudiant). Les attributs tels que schoolName, schoolStreet1, schoolStreet2, etc. seront utilisés pour créer un participant et une adresse.

```
<?xml version="1.0" encoding="UTF-8"?>
<map xmlns="http://www.curamsoftware.com/schemas/GUMBO/Map"
  from-schema="GumboDS" name="TestMapping"to-schema="CGISS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="...\EJBServer\components\
  WorkspaceServices\lib\Mapping.xsd"><map-entity source="Person">
     <target-entity id="householdMember" name="HouseholdMember">
      <map-attribute from="ssnStatus" to="ssnStatus"/>
      <map-attribute from="blackOrAfricanAmerican"
                              to="blkOrAfrAmerInd"/>
      <map-attribute from="nativeAlaskanOrAmericanIndian"
                              to="natHawOrPaIsInd"/>
      <map-attribute from="asian" to="asianInd"/>
      <map-attribute from="nativeHawaiianOrPacificIslander"
                              to="natHawOrPaIsInd"/>
      <map-attribute from="whiteOrCaucasian"
                               to="whiteOrCaucInd"/>
      <map-attribute from="isMigrantOrSeasonalFarmWorker"
                            to="migrantFWorkerInd"/>
    </target-entity>
    <target-entity id="livingArrange" name="LivingArrange">
      <map-attribute from="accommodationType"
                                 to="livingArrangeType"/>
    </target-entity>
  </map-entity>
  <map-entity source="Education">
        <condition expression=
                         "Education.highestGrade!=&amp;quot;&amp;quot;">
            <target-entity id="highestGrade" name="Student">
                <map-attribute from="highestGrade"
                                       to="highGradeCompleted"/>
                <map-attribute from="attendanceFrequency"
                                          to="studentStatus"/>
                <map-attribute from="schoolName"
                                        to="participantName"/>
                <map-attribute from="schoolStreet1" to="street1"/>
                <map-attribute from="schoolStreet2" to="street2"/>
                <map-attribute from="schoolCity" to="city"/>
                <map-attribute from="schoolState" to="state"/>
                <map-attribute from="schoolZipCode" to="zipCode"/>
            </target-entity>
        </condition>
    </map-entity>
  <map-entity source="HealthInsuranceExpense">
    <target-entity id="healthInsuranceExpense"
                                       name="MedicalInsurance">
      <map-attribute from="policyNumber" to="policyNumber"/>
      <map-attribute from="groupNumber" to="groupPolicyNumber"/>
        <map-attribute from="policyHolderParticipantName"
             to="policyHolderParticipantName"/>
        <map-attribute from="policyHolderStreet1"
                                     to="policyHolderStreet1"/>
```

```
        <map-attribute from="policyHolderStreet2"
                                      to="policyHolderStreet2"/>
        <map-attribute from="policyHolderCity"
                                        to="policyHolderCity"/>
        <map-attribute from="policyHolderState"
                                       to="policyHolderState"/>
        <map-attribute from="policyHolderZipCode"
                                      to="policyHolderZipCode"/>
        <map-attribute from="groupParticipantName"
                                     to="groupParticipantName"/>
        <map-attribute from="groupStreet1" to="groupStreet1"/>
        <map-attribute from="groupStreet2" to="groupStreet2"/>
        <map-attribute from="groupCity" to="groupCity"/>
        <map-attribute from="groupState" to="groupState"/>
        <map-attribute from="groupZipCode" to="groupZipCode"/>
        <map-attribute from="insuranceProvider"
                                       to="insuranceProvider"/>
        <map-attribute from="InsProviderStreet1"
                                       to="InsProviderStreet1"/>
        <map-attribute from="InsProviderStreet2"
                                       to="InsProviderStreet2"/>
        <map-attribute from="InsProviderCity"
                                         to="InsProviderCity"/>
        <map-attribute from="InsProviderState"
                                        to="InsProviderState"/>
        <map-attribute from="InsProviderZipCode"
                                      to="InsProviderZipCode"/>
        <map-entity source="HealthInsuranceExpenseRelationship">
        <target-entity id="healthInsuranceExpenseRelationship"
                                                name="Coverage">
          <map-attribute from="personID"
                                    to="caseParticipantRoleID"/>
        </target-entity>
      </map-entity>
    </target-entity>
  </map-entity>
</map>
```

5.6 Exemple de configuration de mappage

Le bloc de code XML de configuration suivant est à suivre lors de la rédaction de la spécification de mappage.

```
<?xml version="1.0" encoding="UTF-8"?><application-builder-config xmlns=
"http://www.curamsoftware.com/schemas/GUMBO/ApplicationBuilderConfig">
 <evidence-config package="curam.evidence">
      <def-create-participant id="SchoolRepresentative" type="RL13">
          <participant-name-field name="firstName" from=
                                    "participantName" order="1"/>
          <participant-address type="AT3">
              <address-field name="addressLine1" from="street1"/>
              <address-field name="addressLine2" from="street2"/>
              <address-field name="city" from="city"/>
              <address-field name="state" from="state"/>
              <address-field name="zip" from="zipCode"/>
          </participant-address>
      </def-create-participant>
      <def-create-participant id="MedicalInsurancePolicyHolder"
                                                        type="RL7">
          <participant-name-field name="firstName"
                    from="policyHolderParticipantName" order="1"/>
          <participant-address type="AT3">
              <address-field name="addressLine1"
                                    from="policyHolderStreet1"/>
              <address-field name="addressLine2"
                                    from="policyHolderStreet2"/>
              <address-field name="city" from="policyHolderCity"/>
```

```
            <address-field name="state" from="policyHolderState"/>
            <address-field name="zip" from="policyHolderZipCode"/>
        </participant-address>
    </def-create-participant>
    <def-create-participant id="MedicalInsuranceGroup"
                                                    type="RL13">
        <participant-name-field name="firstName"
                        from="groupParticipantName" order="1"/>
        <participant-address type="AT3">
            <address-field name="addressLine1"
                                        from="groupStreet1"/>
            <address-field name="addressLine2"
                                        from="groupStreet2"/>
            <address-field name="city" from="groupCity"/>
            <address-field name="state" from="groupState"/>
            <address-field name="zip" from="groupZipCode"/>
        </participant-address>
    </def-create-participant>
    <def-create-participant id="MedicalInsuranceProvider"
                                                    type="RL13">
        <participant-name-field name="firstName"
                          from="insuranceProvider" order="1"/>
        <participant-address type="AT3">
            <address-field name="addressLine1"
                                    from="InsProviderStreet1"/>
            <address-field name="addressLine2"
                                    from="InsProviderStreet2"/>
            <address-field name="city" from="InsProviderCity"/>
            <address-field name="state" from="InsProviderState"/>
            <address-field name="zip" from="InsProviderZipCode"/>
        </participant-address>
    </def-create-participant>

  <entity name="HouseholdMember"/>
  <entity name="HeadOfHousehold"/>
  <entity case-participant-class-name="curam.core.sl.struct.
   CaseParticipantDetails"case-participant-relationship-name=
                               "curam.none" name="Student">
      <create-participant refid="SchoolRepresentative"
              name="schCaseParticipantDetails" role="SCH"/>
  </entity>
  <entity name="MedicalInsurance">
      <create-participant refid="MedicalInsurancePolicyHolder"
             name="plchdrCaseParticipantDetails" role="MIPH"/>
      <create-participant refid="MedicalInsuranceGroup"
              name="groupCaseParticipantDetails" role="GPP"/>
      <create-participant refid="MedicalInsuranceProvider"
                name="insCaseParticipantDetails" role="MIP"/>
  </entity>
</application-builder-config>
```

Annexe A. Schéma des spécifications de mappage

A.1 Schéma

Le schéma suivant est à suivre lors de la rédaction de spécifications de mappage :

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.curamsoftware.com/schemas/GUMBO/Map"
    xmlns:mp="http://www.curamsoftware.com/schemas/GUMBO/Map"
    elementFormDefault="qualified">

    <xs:simpleType name="TargetEntityRoleType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="parent"/>
            <xs:enumeration value="child"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="AttachmentType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="case"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="MapAttributeType">
        <xs:attribute name="from" type="xs:NCName" use="required"/>
        <xs:attribute name="to" type="xs:NCName" use="required"/>
        <xs:attribute name="mapping-function" type="xs:string"
          use="optional"/>
        <xs:attribute name="mapping-rule" type="xs:string"
          use="optional"/>
        <xs:attribute name="entity" type="xs:NCName" use="optional"/>
    </xs:complexType>

    <xs:complexType name="SetAttributeType">
        <xs:attribute name="name" type="xs:NCName"/>
        <xs:attribute name="value" type="xs:string"/>
    </xs:complexType>

    <xs:element name="set-attribute" type="mp:SetAttributeType"/>

    <xs:complexType name="TargetEntityType">
        <xs:sequence>
            <xs:element name="map-attribute" type="mp:MapAttributeType"
Valeur minimale = "0"
              maxOccurs="unbounded"/>
            <xs:element ref="mp:set-attribute" minOccurs="0"
              maxOccurs="unbounded"/>
            <xs:element ref="mp:condition" minOccurs="0"
              maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="name" type="xs:NCName"/>
        <xs:attribute name="type" type="mp:TargetEntityRoleType"/>
        <xs:attribute name="attachment" type="mp:AttachmentType"/>
        <xs:attribute name="id" type="xs:ID" use="optional"/>
    </xs:complexType>

    <xs:element name="target-entity" type="mp:TargetEntityType"/>

    <xs:complexType name="ConditionType">
        <xs:choice>
            <xs:element ref="mp:target-entity" minOccurs="0"
```

```
            maxOccurs="unbounded"/>
          <xs:element ref="mp:target-entities" minOccurs="0"
            maxOccurs="unbounded"/>
          <xs:element ref="mp:set-attribute" minOccurs="0"
            maxOccurs="unbounded"/>
          <xs:element ref="mp:map-entity" minOccurs="0"
            maxOccurs="unbounded"/>
        </xs:choice>
        <xs:attribute name="expression" type="xs:string"/>
    </xs:complexType>

    <xs:element name="condition" type="mp:ConditionType"/>

    <xs:complexType name="MapEntityType">
        <xs:sequence>
          <xs:element ref="mp:target-entity" minOccurs="0"
            maxOccurs="unbounded"/>
          <xs:element ref="mp:target-entities" minOccurs="0"
            maxOccurs="unbounded"/>
          <xs:element  ref="mp:condition" minOccurs="0"
            maxOccurs="unbounded"/>
          <xs:element ref="mp:map-entity" minOccurs="0"
            maxOccurs="unbounded"/>
          <xs:element ref="mp:follow-association" minOccurs="0"
            maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="source" type="xs:NCName"/>
    </xs:complexType>

    <xs:element name="map-entity" type="mp:MapEntityType"/>

    <xs:element name="follow-association" type="mp:MapEntityType"/>

    <xs:complexType name="MapEntitiesType">
        <xs:sequence>
          <xs:element ref="mp:target-entity" minOccurs="0"
            maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:element name="target-entities" type="mp:MapEntitiesType"/>

    <xs:complexType name="MapCodeTableValueType">
        <xs:attribute name="source" type="xs:string"/>
        <xs:attribute name="target" type="xs:string"/>
    </xs:complexType>

    <xs:complexType name="MapCodeTableType">
        <xs:sequence>
          <xs:element name="map-value"
            type="mp:MapCodeTableValueType" minOccurs="1"
              maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="context" type="xs:NCName" use="optional"/>
        <xs:attribute name="source-codetable" type="xs:NCName"/>
        <xs:attribute name="target-codetable" type="xs:NCName"/>
        <xs:attribute name="source-package" type="xs:NCName"
use="optional"/>
        <xs:attribute name="target-package" type="xs:NCName"
use="optional"/>
    </xs:complexType>

    <xs:complexType name="MapType">
        <xs:sequence>
          <xs:element name="map-code-table" type="mp:MapCodeTableType"
            minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="mp:map-entity" minOccurs="0"
```

```
            maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="name" type="xs:NCName"/>
        <xs:attribute name="from-schema" type="xs:NCName"/>
        <xs:attribute name="to-schema" type="xs:NCName"/>
    </xs:complexType>

    <xs:element name="map" type="mp:MapType"/>

</xs:schema>
```

Annexe B. Schéma des configurations de mappage

B.1 Schéma

Le schéma suivant est à suivre lors de la rédaction de configurations de mappage :

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Licensed Materials - Property of IBM

  Copyright IBM Corporation 2012. All Rights Reserved.

  US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corp.
-->

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="

 http://www.curamsoftware.com/schemas/GUMBO/ApplicationBuilderConfig"
    xmlns:abc="

http://www.curamsoftware.com/schemas/GUMBO/ApplicationBuilderConfig"
    elementFormDefault="qualified">

    <xs:complexType name="EvFieldType">
        <xs:attribute name="referenced-as" type="xs:NCName" use="optional"/>
        <xs:attribute name="target-name" type="xs:NCName" use="optional"/>
        <xs:attribute name="aggregation" type="xs:NCName" use="optional"/>
        <xs:attribute name="is-reference-attribute" type="xs:boolean"

 use="optional"/>
        <xs:attribute name="map-as-concernrole-id" type="xs:boolean"

 use="optional"/>
    </xs:complexType>

    <xs:complexType name="ParticipantCreatorType">
        <xs:attribute name="refid" type="xs:string" use="required"/>
        <xs:attribute name="name" type="xs:NCName" use="required"/>
        <xs:attribute name="role" type="xs:string" use="required"/>
    </xs:complexType>

    <xs:complexType name="ParticipantNameFieldType">
        <xs:attribute name="name" type="xs:string" use="required"/>
        <xs:attribute name="from" type="xs:NCName" use="required"/>
        <xs:attribute name="order" type="xs:positiveInteger" use="optional"/>
    </xs:complexType>

    <xs:complexType name="AddressFieldType">
        <xs:attribute name="name" type="xs:NCName" use="required"/>
        <xs:attribute name="from" type="xs:NCName" use="required"/>
    </xs:complexType>

    <xs:complexType name="ParticipantAddressType">
        <xs:sequence>
            <xs:element ref="abc:address-field" minOccurs="0"

maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="type" type="xs:string" use="required"/>
    </xs:complexType>
```

```
   <xs:element name="participant-name-field"
 type="abc:ParticipantNameFieldType"/>
   <xs:element name="participant-address"
 type="abc:ParticipantAddressType"/>
   <xs:element name="ev-field" type="abc:EvFieldType"/>
   <xs:element name="create-participant" type="abc:ParticipantCreatorType"/>
   <xs:element name="address-field" type="abc:AddressFieldType"/>

   <xs:complexType name="EntityType">
       <xs:sequence>
           <xs:element ref="abc:ev-field" minOccurs="0" maxOccurs="unbounded"/>
           <xs:element ref="abc:create-participant" minOccurs="0" maxOccurs=

 "unbounded"/>
       </xs:sequence>
       <xs:attribute name="name" type="xs:NCName"/>
       <xs:attribute name="package" type="xs:string"/>
       <xs:attribute name="case-participant-relationship-name" type="xs:NCName"/>
       <xs:attribute name="case-participant-class-name" type="xs:NCName"/>
       <xs:attribute name="ev-type-code" type="xs:NCName" use="optional"/>
       <xs:attribute name="dyn-evidence-primary-cpr-field-name" type="xs:NCName"

 use="optional"/>
       <xs:attribute name="target-entity-type" type="xs:NCName" use="optional"/>
   </xs:complexType>

   <xs:complexType name="ParticipantCreatorDefinitionType">
       <xs:sequence>
           <xs:element ref="abc:participant-name-field"
  minOccurs="0" maxOccurs="unbounded"/>
           <xs:element ref="abc:participant-address" minOccurs="0" maxOccurs="1"/>
       </xs:sequence>
       <xs:attribute name="id" type="xs:string" use="required"/>
       <xs:attribute name="type" type="xs:string" use="required"/>
   </xs:complexType>

   <xs:complexType name="EvidenceConfigType">
               <xs:sequence>
                   <xs:element name="entity" type="abc:EntityType"

 maxOccurs="unbounded"/>
                   <xs:element

name="def-create-participant" type=
     "abc:ParticipantCreatorDefinitionType"

 minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:attribute name="package" type="xs:string" use="optional"/>
   </xs:complexType>

   <xs:simpleType name="MutliplicityType">
       <xs:restriction base="xs:string">
           <xs:enumeration value="multiple"/>
           <xs:enumeration value="singleton"/>
       </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="FieldType">
       <xs:restriction base="xs:string">
           <xs:enumeration value="append"/>
           <xs:enumeration value="button-radio"/>
           <xs:enumeration value="button-checkbox"/>
           <xs:enumeration value="choice-combo"/>
           <xs:enumeration value="choice-multi-list"/>
       </xs:restriction>
   </xs:simpleType>
```

```
    <xs:complexType name="FieldConfig">
        <xs:attribute name="name" type="xs:NCName"/>
        <xs:attribute name="type" type="abc:FieldType" use="optional"/>
        <xs:attribute name="append-separator" type="xs:string" use="optional"/>
        <xs:attribute name="codetable-class" type="xs:string" use="optional"/>
    </xs:complexType>

    <xs:complexType name="SectionConfig">
        <xs:sequence>
            <xs:element name="field" type="abc:FieldConfig" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="name" type="xs:NCName" use="required"/>
        <xs:attribute name="type" type="abc:MutliplicityType" use="optional"/>
    </xs:complexType>

    <xs:complexType name="PdfConfigType">
                <xs:sequence>
                    <xs:element name="section" type="abc:SectionConfig"

maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:attribute name="pdf-form-name" type="xs:string"/>
    </xs:complexType>

    <xs:complexType name="DatastoreConfigType">
        <xs:attribute name="targetSchema" type="xs:string"/>
    </xs:complexType>

    <xs:element name="evidence-config" type="abc:EvidenceConfigType"/>

    <xs:element name="pdf-config" type="abc:PdfConfigType"/>

    <xs:element name="datastore-config" type="abc:DatastoreConfigType"/>

    <xs:complexType name="ApplicationBuilderConfigType">
        <xs:choice>
          <xs:element ref="abc:evidence-config"/>
          <xs:element ref="abc:pdf-config"/>
          <xs:element ref="abc:datastore-config"/>
        </xs:choice>
    </xs:complexType>
    <xs:element name="application-builder-config"

 type="abc:ApplicationBuilderConfigType"/>
</xs:schema>
```

Remarques

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM. IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous donne aucun droit de licence sur ces brevets. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A. Pour le Canada, veuillez adresser votre courrier à : IBM Director of Commercial Relations IBM Canada Ltd 3600 Steeles Avenue East Markham, Ontario L3R 9Z7 Canada

Les informations sur les licences concernant les produits utilisant un jeu de caractères double octet peuvent être obtenues par écrit à l'adresse suivante :

Intellectual Property Licensing

Legal and Intellectual Property Law.

IBM Japan Ltd.

19-21, Nihonbashi-Hakozakicho, Chuo-ku

Tokyo 103-8510, Japan

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales : LE PRÉSENT DOCUMENT EST LIVRÉ EN L'ÉTAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DÉCLINE NOTAMMENT TOUTE RESPONSABILITÉ RELATIVE À CES INFORMATIONS EN CAS DE CONTREFAÇON AINSI QU'EN CAS DE DÉFAUT D'APTITUDE À L'EXÉCUTION D'UN TRAVAIL DONNÉ. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies. Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

U.S.A. Pour le Canada, veuillez adresser votre courrier à : IBM Director of Commercial Relations IBM Canada Ltd 3600 Steeles Avenue East Markham, Ontario L3R 9Z7 Canada

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le programme sous licence décrit dans ce document et tous les éléments sous licence associés sont fournis par IBM selon les termes de l'IBM Customer Agreement, de l'IBM International Program License Agreement ou de tout contrat équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles.

IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis et doit être considérée uniquement comme un objectif.

Tous les tarifs indiqués sont les prix de vente actuels suggérés par IBM et sont susceptibles d'être modifiés sans préavis. Les tarifs appliqués peuvent varier selon les revendeurs.

Ces informations sont fournies uniquement à titre de planification. Elles sont susceptibles d'être modifiées avant la mise à disposition des produits décrits.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

LICENCE DE COPYRIGHT :

Les présentes informations contiennent des exemples de programmes d'application en langage source illustrant les techniques de programmation sur diverses plateformes d'exploitation. Vous avez le droit de

copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes. Les exemples de programme sont fournis "EN L'ETAT", sans garantie d'aucune sorte. IBM décline toute responsabilité relative aux dommages éventuels résultant de l'utilisation de ces exemples de programmes.

Toute copie totale ou partielle de ces exemples de programmes et des travaux qui en sont dérivés doit comprendre une mention de droits d'auteur, libellée comme suit :

© (nom de votre société) (année). Des segments de code sont dérivés des Programmes exemples d'IBM Corp.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

Documentation sur l'interface de programmation

La présente publication décrit plusieurs interfaces de programmation qui permettent au client de rédiger des programmes afin d'obtenir les services d'IBM Cúram Social Program Management.

Marques

IBM, le logo IBM et ibm.com sont des marques d'International Business Machines Corp. aux Etats-Unis et/ou dans certains autres pays. Les autres noms de produit et de service peuvent être des marques d'IBM ou d'autres sociétés. La liste actualisée de toutes les marques d'IBM est disponible sur la page Web "Copyright and trademark information" à http://www.ibm.com/legal/us/en/copytrade.shtml.

Adobe et Portable Document Format (PDF) sont des marques d'Adobe Systems Incorporated aux Etats-Unis et/ou dans certains autres pays.

Java ainsi que tous les logos et toutes les marques incluant Java sont des marques d'Oracle et/ou de ses sociétés affiliées.

D'autres sociétés sont propriétaires des autres marques qui pourraient apparaître dans ce document. Les autres noms de sociétés, de produits et de services peuvent appartenir à des tiers.

Imprimé en France