IBM Cúram Social Program Management

Working with Intelligent Evidence Gathering (IEG)

Version 6.0.3

IBM Cúram Social Program Management

Working with Intelligent Evidence Gathering (IEG)

Version 6.0.3

Important

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations contenues dans la section «Remarques», à la page 67

Dernière révision : Mai 2013

Cette édition s'applique à IBM Cúram Social Program Management v6.0 5 et à toutes les révisions suivantes, sauf indication contraire dans les 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 Public visé

Ce guide est destiné aux auteurs de script débutant avec Intelligent Evidence GatheringIntelligent Evidence Gathering (IEG) et souhaitant utiliser ses fonctions pour capturer intelligemment des données dans une application interne ou externe. Techniquement, il peut s'agir des données que vous souhaitez utiliser pour toute opération, mais généralement les données en question sont des données client requises dans une application de programme ou pour déterminer l'éligibilité potentielle. Toutes ces informations portent l'appellation "preuve" dans Cúram. En raison de son style instructif, ce guide est directement destiné aux concepteurs de script.

1.2 Objectif

L'objectif de ce guide est de fournir aux auteurs de script des informations essentielles relatives à la définition et la gestion des scripts IEG et les schémas de magasin de données (DS) à utiliser dans des applications internes ou externes.

IEG est une technologie intégrée à la suite d'applications Cúram qui permet aux clients de créer des scripts dynamiques pour collecter des données de plusieurs façons. Il existe toutefois certaines remarques lors de la création d'un script IEG et d'un schéma DS. Ce guide va exposer certaines de ces considérations, ainsi que des informations relatives à la maintenance des scripts.

1.3 Lectures complémentaires

Il est recommandé de consulter d'autres documents avant de créer ou de publier un script IEG. Premièrement, le manuel `Cúram Development Compliancy Guide` décrit les restrictions qui s'appliquent lors du développement d'applications à l'aide d'IEG et doivent être comprises avant toute implémentation. Le document `Authoring Scripts using Intelligent Evidence Gathering (IEG)` est un autre document à consulter. Ce document peut être utilisé comme guide de référence et contient des informations détaillées sur toutes les fonctions disponibles dans IEG et des instructions relatives à l'utilisation de ces fonctions. Le guide `Creating Datastore Schemas` explique comment les schémas de DS sont créés et gérés avec IEG.

Chapitre 2. Mise en route

2.1 Introduction

Ce chapitre va vous expliquer les principes de base d'IEG et de sa dépendance sur le magasin de données (DS) et le magasin de ressources (RS). Le présent chapitre va vous guider tout au long de la création d'un script IEG simple permettant de collecter des informations sur un client.

2.2 A propos d'IEG

IEG constitue une alternative efficace aux processus de collecte d'informations traditionnels. Grâce à IEG, les informations sont collectées de façon interactive en affichant un script de questions auxquelles un utilisateur peut répondre. Les questions ne s'affichent que si elles sont cohérentes avec les réponses précédentes de l'utilisateur. L'utilisateur ne doit ainsi répondre qu'aux questions correspondant à ses besoins ou à sa situation. Il en résulte un environnement convivial qui peut être efficacement implémenté pour divers processus incluant la saisie d'informations, le tri de l'évaluation des prestations, l'évaluation de l'éligibilité en ligne, etc.

Contrairement aux processus de collecte d'informations traditionnelles, IEG permet de réduire les tâches administratives de l'organisation en créant plusieurs cheminements potentiels dans un même script de question. Ceci élimine la nécessité de développer de nombreux scripts pour collecter les informations des différents types d'utilisateur.

Un autre avantage que procure IEG est la flexibilité de sa mise en oeuvre et la plage de ses utilisateurs potentiels. L'environnement d'exécution IEG peut être défini afin d'être accessible à partir de n'importe quelle page UIM. Cela signifie qu'IEG peut être accessible directement à partir d'une application d'organisation ou à distance par un utilisateur en ligne.

Les deux principaux composants d'IEG sont le moteur et le lecteur. Les scripts IEG sont définis au format XML et le moteur interprète les définitions de script au moment de l'exécution et évalue les réponses fournies par l'utilisateur pour déterminer le flux d'exécution. Le moteur détermine les pages disponibles pour l'utilisateur et le nombre de fois qu'elles doivent être affichées. Le lecteur présente les pages, les questions et d'autres contrôles à l'utilisateur. IEG s'appuie également sur d'autres éléments de la suite d'applications Cúram telles que le magasin de données (DS) et le magasin de ressources (RS).

2.2.1 Magasin de données (DS)

Les données fournies par un utilisateur lors de l'exécution du script ne sont pas directement conservées par IEG lui-même. Cette tâche est déléguée au magasin de données (DS). Le DS est une base de données configurable. A l'instar des questions et pages de questions mises à disposition de l'utilisateur qui sont déterminées par un script IEG, les données qui peuvent être stockées dans le DS sont déterminées de façon dynamique par un schéma XML. Le schéma décrit la structure des informations à stocker et les relations entre les données. Les données sont stockées dans le DS au format XML et sont conformes au langage de définition de schéma XML W3C. Pour plus d'informations sur le DS et son fonctionnement, voir le manuel `Creating Datastore Schemas` .

Le script IEG et le schéma DS sont étroitement liés. Un script IEG est défini avec les références aux éléments contenus dans un schéma. Voilà pourquoi un schéma doit être fourni lors de l'édition d'un script. Le même schéma est également requis lors de l'exécution d'un script. Les schémas peuvent être réutilisés pour éditer et exécuter plusieurs scripts afin que les mêmes structures de données puissent être utilisées dans différentes circonstances.

2.2.2 Magasin de ressources (RS)

Un script IEG peut contenir des références à des images mises à la disposition de l'utilisateur lorsqu'un script est exécuté (par exemple, icônes représentant des sections et des pages de questions). Les images sont stockées dans le magasin de ressources (RS). Un script IEG contient également plusieurs éléments textuels, par exemple, des en-têtes de page, des libellés de questions et du texte d'aide. IEG vous permet de saisir tout le texte de votre script pour l'environnement local par défaut, et ce directement dans la définition du script.

Lorsqu'un script IEG est téléchargé dans le système via les écrans d'administration IEG, tout le texte qui s'y trouve est automatiquement extrait dans un fichier de propriétés correctement nommé pour le script. Ces fichiers de propriétés sont également stockés dans le RS. Les fichiers de propriétés sont stockés sans environnement local associé (afin qu'ils agissent en tant que propriétés de secours si aucune propriété n'existe pour l'environnement local que vous exécutez). Le RS permet de télécharger les fichiers de propriétés de plusieurs environnements locaux qui simplifient la localisation des scripts. Lors de l'exécution, les fichiers de propriétés de l'environnement local approprié sont extraits et présentés à l'utilisateur dans le lecteur IEG.

2.2.3 Structure d'un script

Dans sa forme la plus simple, un script IEG est constitué de pages qui contiennent des questions posées par les utilisateurs d'IEG. La structure du script IEG est un regroupement logique de ces pages afin que les réponses aux questions puissent être capturées de manière efficace. Les séquences de pages peuvent être regroupées en sections logiques. L'objectif de ces sections consiste à donner aux utilisateurs une vue de niveau supérieur du type d'informations enregistré par le script IEG.

Outre le fait d'inclure un nombre variable de pages, chaque section doit contenir une page récapitulative. Cette page est utilisée pour transmettre à l'utilisateur des commentaires sur les informations saisies dans une section des pages. Les pages récapitulatives contiennent généralement des clusters et des listes affichant des versions en lecture seule des réponses aux questions posées. La page récapitulative sera toujours la dernière page affichée dans une section et sera également affichée lorsqu'un utilisateur clique sur le lien de cette section dans la barre latérale du lecteur IEG.

Pour résumer, les scripts IEG se composent d'une hiérarchie d'éléments structurés de la manière suivante :

- Script
  - Section
    - Page
      - Cluster
        - Question
    - Page récapitulative

2.3 Evaluation de l'utilisation d'IEG

Il existe certaines questions clés à poser lors de l'évaluation de l'utilisation d'IEG dans toute application :

- Quelles informations sont capturées ?
- Quelle est la source de ces informations ?
- Comment ces informations doivent-elles être utilisées ?
- Combien de temps ces informations résident-elles dans l'application ?

La plupart des utilisations en cours d'IEG découlent du besoin de prendre en charge une application pour les produits et les services offerts par des agences, en interne ou en externe. Les informations capturées sont généralement des informations liées au client, telles que les détails personnels du client, les détails sur leur famille ou leur foyer et les détails de leurs besoins.

Souvent, les agences possèdent déjà des données sur un client ; par conséquent, elles peuvent déterminer la source des informations d'un autre système à l'aide d'éléments d'informations clés tels qu'un numéro de sécurité sociale. Cela leur permet de vérifier les informations client en cours de saisie ou d'extraction pour faciliter la saisie de l'application.

Certaines applications sont complexes et nécessitent des informations de plusieurs sources. Les clients peuvent devoir entrer des informations qu'ils ne possèdent pas à proximité. Par exemple, les informations requises peuvent être conservées par leur employeur. Ils peuvent avoir besoin de stocker les informations qu'ils ont saisies et de retourner ultérieurement à l'application une fois qu'ils disposent de toutes les données requises.

Les clients peuvent être exposés à des applications de balayage simples qui les informent de leurs droits en vertu de la législation en cours et de la nouvelle législation. Ces informations sont souvent peu fiables et les données temporaires doivent être supprimées du système après déconnexion du client ou une période de temps définie.

Ces exigences régissent l'utilisation d'IEG et fournissent des informations importantes sur l'utilisation des données au cours de sa durée de vie.

Par conséquent, démarrons par les bases : nous voulons capturer et stocker les informations sur un client.

2.4 Bases

2.4.1 Création d'un schéma

La première étape de la capture d'un client consiste à créer un schéma DS. Cette section présente comment créer un schéma de base qui définit la capture de données client générales.

Le DS stocke les données collectées des utilisateurs lors du balayage en ligne et de la saisie des applications. Le contenu du DS est définissable de façon dynamique par le biais d'une définition de schéma. Les conditions requises pour la capture et le stockage des données sur un client peuvent être complexes. Toutefois, avec un schéma approprié, ces données peuvent être gérées de façon efficace sur tout son cycle de vie.

L'objectif de l'exemple suivant consiste à capturer ce qui suit :

Tableau 1. Données client à capturer

attributs	**Type**
Prénom	Chaîne
Deuxième nom	Chaîne
Nom de famille	Chaîne
Sexe	Masculin/Féminin
Date de naissance	Date

Un schéma doit comporter un ensemble minimum de définitions. Pour utiliser un schéma dans IEG, la configuration suivante est requise :

- Inclusion des domaines de base
- Inclusion des domaines IEG
- Une entité racine

Pour plus d'informations sur l'ensemble minimum de définitions requises, voir le manuel `Creating Datastore Schemas` .

Le schéma doit ressembler à ceci avant d'ajouter un nouveau contenu, tel que l'entité Person décrite ci-dessus :

```
<xsd:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:d="http://www.curamsoftware.com/BaseDomains">
  <xsd:import namespace="http://www.curamsoftware.com/BaseDomains"/>
  <xsd:include schemaLocation="IEGDomains"/>
  <xsd:element name="Application">
    <xsd:complexType>
      <xsd:sequence minOccurs="0">
    </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Figure 1. Schéma de démarrage

Le contenu du schéma indique qu'il s'agit d'un schéma XML qui importe un schéma BaseDomains et contient le schéma IEGDomains. Le premier élément appelé `Application` est l'entité racine du schéma. IEG nécessite de toujours appeler l'entité racine `Application`.

Le schéma IEGDomains contient les domaines requis pour définir les attributs des entités à utiliser avec IEG. Les types des attributs doivent être dérivés des domaines IEG plutôt que des domaines de base. Une entité Person peut être définie pour représenter un client comme suit :

```
<xsd:element name="Person">
  <xsd:complexType>
    <xsd:attribute name="firstName" type="IEG_STRING"/>
    <xsd:attribute name="middleName" type="IEG_STRING"/>
    <xsd:attribute name="lastName" type="IEG_STRING"/>
    <xsd:attribute name="gender" type="IEG_GENDER"/>
    <xsd:attribute name="dateOfBirth" type="IEG_DATE"/>
  </xsd:complexType>
</xsd:element>
```

Figure 2. Entité Person

Plusieurs éléments sont à noter sur la définition d'une entité comme personne :

- Comme les tables de base de données relationnelle, un champ ID est requis et une clé est définie pour cette table à l'aide de cet ID unique.
- L'entité Person est ajoutée en tant qu'entité enfant de l'entité racine.

Le schéma permettant de capturer les informations de base relatives à une personne peut être défini comme suit :

```
<xsd:element name="Application">
  <xsd:complexType>
    <xsd:sequence minOccurs="0">
       <xsd:element ref="Person" minOccurs="0"
                                 maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="Person">
  <xsd:complexType>
    <xsd:attribute name="personID" type="d:SVR_KEY"/>
    <xsd:attribute name="firstName" type="IEG_STRING"/>
    <xsd:attribute name="middleName" type="IEG_STRING"/>
    <xsd:attribute name="lastName" type="IEG_STRING"/>
    <xsd:attribute name="gender" type="IEG_GENDER"/>
    <xsd:attribute name="dateOfBirth" type="IEG_DATE"/>
  </xsd:complexType>
  <xsd:key name="Person_Key">
    <xsd:selector xpath="./Person"/>
    <xsd:field xpath="@personID"/>
  </xsd:key>
</xsd:element>
```

Figure 3. Schéma de base

Une fois le schéma défini, vous pouvez ensuite créer un script pour utiliser le schéma.

2.4.2 Création d'un script

IEG vous permet de créer des scripts dynamiques de collecte des données. Les scripts IEG peuvent contenir des sections, des pages de questions et une logique conditionnelle qui vous permettent de choisir les informations à capturer, les pages à afficher et le nombre de fois qu'elles s'affichent.

Veuillez consulter le manuel `Authoring Scripts using Intelligent Evidence Gathering (IEG)` pour plus d'informations sur la définition de chaque élément d'un script IEG.

Pour les exigences ci-dessus, où il est nécessaire de capturer les information sur une personne, vous devez définir le script et définir la disposition des pages pour capturer les informations.

Un nouveau script peut être créé dans l'application d'administration et l'éditeur peut être utilisé pour ajouter des éléments dans ce script. Le contenu d'un nouveau script créé est identique à ce qui suit :

```
<?xml version="1.0" encoding="UTF-8"?>
<ieg-script xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="ieg-schema.xsd">
  <identifier id="WorkingWithIEG" scriptversionnumber="V1"
              type="Intake" />
</ieg-script>
```

Figure 4. Nouveau script

L'`ID`, le `type` et la `version` fournis lors de la création du script sont combinés pour créer un identificateur de script visant à identifier de façon unique la définition de script.

Une fois un nouveau script créé, des éléments tels que des sections, des pages de questions et des pages récapitulatives peuvent être ajoutés au script. Les exemples des deux sections suivantes vont vous présenter comment ajouter une section et une page de questions dans un script et ajouter une page récapitulative qui affiche les informations pour l'utilisateur. Les pages récapitulatives permettent à l'utilisateur de confirmer que les données saisies sont correctes avant de poursuivre et de modifier les données.

2.4.2.1 Ajout d'une section et d'une page de questions à un script IEG

Une section et une page de questions doivent être ajoutées. Une section peut être utilisée pour regrouper les pages associées et permettre à l'utilisateur de parcourir les écrans de façon logique. Les sections peuvent également vous aider à informer l'utilisateur de sa progression dans un script. La section et la page de questions peuvent posséder un titre et la page de questions peut éventuellement contenir une description.

L'exemple de code suivant montre une section contenant une page de questions ajoutée à un script :

```
<?xml version="1.0" encoding="UTF-8"?>
<ieg-script xmlns:xsi="http://www.w3.org/2001/XMLSchema-in stance"
 xsi:noNamespaceSchemaLocation="ieg-schema.xsd">
  <identifier id="WorkingWithIEG" scriptversionnumber="V1"
              type="Intake" />
  <section>
    <title id="AboutYouSection.Title">
      <![CDATA[A propos de vous]]>
    </title>
    <question-page id="AboutYouPage" entity="Person">
      <title id="PrimaryPersonPage.Title">
        <![CDATA[A propos de vous]]>
      </title>
      <description id="PrimaryPersonPage.Description">
        <![CDATA[Veuillez entrer vos informations personnelles]]>
      </description>
    </question-page>
  </section>
</ieg-script>
```

Figure 5. Nouvelle section

La page de questions nécessite les questions appropriées pour capturer les données. Les données à stocker dans le DS doivent être associées à l'attribut d'une entité dans le schéma DS à utiliser avec ce script. Si toutes les questions d'une page se rapportent à la même entité, la page peut être mappée sur ce type d'entité. Dans l'exemple suivant, la page est mappée sur l'entité Person.

Pour ajouter des questions à une page, un cluster est requis. Les clusters permettent de contrôler la présentation des questions sur la page. Une page peut contenir plusieurs clusters pour vous permettre de regrouper logiquement les questions sur la page. Les clusters peuvent également contenir un titre et une description.

Notre exemple ci-dessous contient deux clusters : un pour afficher uniquement du texte informatif au niveau de l'utilisateur et un autre pour contenir les questions des détails personnels. Vous pouvez ajouter des questions et du texte d'affichage à chaque cluster. Les questions doivent posséder un ID correspondant à l'un des attributs du type d'entité sur lequel la page est mappée. Si une réponse doit être fournie à une question, l'indicateur mandatory de la question peut être défini sur `true`. Le fragment de script ci-dessous contient les questions visant à capturer les données requises présentées dans notre exemple.

```
<question-page ...
    <cluster>
    <display-text id="RequiredFields.Text">
      <![CDATA[<span style="color: orange;">
                * indique un champ obligatoire</span>]]>
    </display-text>
  </cluster>
  <cluster>
    <title id="DetailsCluster.Title">
      <![CDATA[Détails personnels]]>
    </title>
    <description id="DetailsCluster.Description">
      <![CDATA[Entrez vos détails ici]]>
    </description>
    <question id="firstName" mandatory="true">
      <label id="FirstName.Label">
        <![CDATA[Prénom :]]>
      </label>
    </question>
    <question id="middleName">
      <label id="MiddleName.Label">
        <![CDATA[Deuxième prénom :]]>
      </label>
    </question>
    <question id="lastName">
      <label id="lastName.Label">
        <![CDATA[Nom :]]>
      </label>
    </question>
    <question id="gender" mandatory="true">
      <label id="Gender.Label">
        <![CDATA[Sexe :]]>
      </label>
    </question>
    <question id="dateOfBirth" mandatory="true">
      <label id="DateOfBirth.Label">
        <![CDATA[Date de naissance :]]>
      </label>
    </question>
  </cluster>
</question-page>
```

Figure 6. Clusters, questions et texte d'affichage

Notez qu'il existe d'autres propriétés de scripts, de sections, de pages de questions, de clusters, de questions et de textes d'affichage que celles couvertes ici. Ces propriétés sont traitées dans le manuel `Authoring Scripts using Intelligent Evidence Gathering (IEG)`, dont certaines seront étudiées ultérieurement dans ce manuel.

2.4.3 Ajout d'une page récapitulative à un script IEG

L'étape finale de cet exemple de base consiste à afficher un récapitulatif des informations capturées. En règle générale, chaque section disposera d'une page récapitulative. Une page récapitulative est utilisée pour afficher les données les plus importantes à l'utilisateur pour qu'ils puissent vérifier que les données ont été correctement capturées et calculées. Une page récapitulative peut afficher les données capturées sur plusieurs pages de question. Une page récapitulative ne doit pas contenir toutes les informations capturées dans la section car elle pourrait devenir très volumineuse et, par conséquent, moins pratique.

Evidemment, si les données affichées dans une page récapitulative sont incorrectes, l'utilisateur souhaitera très probablement les modifier. Les utilisateurs peuvent revenir en arrière dans l'exécution du script en appuyant sur le bouton Précédent du lecteur IEG jusqu'à la page où les données ont été saisies, mettre à jour les données, puis reprendre le script dans le sens approprié. Vous pouvez sinon ajouter des liens d'édition aux clusters dans la page récapitulative. Lorsque l'utilisateur clique sur un lien d'édition dans la page récapitulative, la page de question indiquée dans le lien d'édition s'affiche pou l'utilisateur

dans le lecteur IEG. L'utilisateur peut ensuite modifier les données et, selon que les données sont référencées ailleurs dans le script, la page récapitulative s'affiche à nouveau lorsque l'utilisateur appuie sur le bouton Suivant du lecteur IEG.

Dans ce cas, la page récapitulative sera très simple et identique à la page de question précédemment ajoutée. A l'instar d'une page de question, si tous les attributs mentionnés dans la page se rapportent à la même entité, la page récapitulative peut être mappée sur ce type d'entité, comme suit :

```
<section>
...
  <summary-page id="AboutYouSummary" entity="Person">
    <title id="AboutYouSummary.Title">
      <![CDATA[Informations personnelles]]>
    </title>
    <description id="AboutYouSummary.Description">
      <![CDATA
        [Voici vos informations personnelles]]>
    </description>
    <cluster>
      <title id="DetailsCluster.Title">
        <![CDATA[Détails de la personne]]>
      </title>
      <description id="DetailsCluster.Description">
        <![CDATA[Entrez les détails de la personne ici]]>
      </description>
      <edit-link start-page="AboutYouPage" />
      <question id="firstName">
        <label id="FirstName.Label">
          <![CDATA[Prénom :]]>
        </label>
      </question>
      <question id="middleName">
        <label id="MiddleName.Label">
          <![CDATA[Deuxième prénom :]]>
        </label>
      </question>
      <question id="lastName">
        <label id="lastName.Label">
          <![CDATA[Nom :]]>
        </label>
      </question>
      <question id="gender">
        <label id="Gender.Label">
          <![CDATA[Sexe :]]>
        </label>
      </question>
      <question id="dateOfBirth">
        <label id="DateOfBirth.Label">
          <![CDATA[Date de naissance :]]>
        </label>
      </question>
    </cluster>
  </summary-page>
</section>
```

Figure 7. Page récapitulative

Ce script et ce schéma de base permettant de capturer les informations sur une personne et d'afficher une page récapitulative sont désormais terminés et peuvent être exécutés.

2.4.4 Exécution d'un script

Pour exécuter un script IEG, les définitions du script et du schéma associé doivent être téléchargées dans le système. Les diverses manières permettant de procéder ainsi seront traitées ultérieurement dans ce

manuel. La méthode la plus simple de télécharger les définitions consiste à utiliser les écrans d'administration dans la section Système de regroupement de preuves intelligent de l'espace de travail Administration.

Pour accéder aux écrans d'administration IEG, vous devez vous connecter en tant qu'utilisateur admin. Une fois connecté, une section IEG nommée Système de regroupement de preuves intelligent est disponible dans votre panneau de raccourcis. Cliquez dessus pour afficher un menu IEG qui contient un lien nommé Scripts. Si vous cliquez dessus, une page contenant une liste des scripts IEG actuellement dans le système et divers liens vous permettant d'effectuer des activités sur ces scripts s'affiche.

Dans la partie supérieure de la page 'Scripts', un lien 'Import' (Importer) vous permet de télécharger ou d'importer une nouvelle définition de script IEG.

De même, si vous cliquez sur le lien 'Datastore Schemas' (Schémas de magasin de données) du menu 'IEG', une page contenant une liste des schémas DS actuellement dans le système s'affiche. Dans la partie supérieure de la page 'Datastore Schemas' (Schémas de magasin de données), un lien Importer (Import) vous permet de télécharger ou d'importer une nouvelle définition de schéma.

Pour des raisons pratiques, IEG fournit un type de routine de test qui permet de tester les scripts IEG sans qu'il soit nécessaire de les intégrer à l'application Cúram. La routine de test comporte certaines limitations mais elle permet de tester la plupart des scripts dès qu'ils sont téléchargés dans le système. Les scripts IEG peuvent être exécutés soit dans un onglet ou dans une fenêtre modale via les écrans d'administration.

Un script peut être exécuté à l'aide des options 'Run' (Exécuter) ou 'Run in Modal' (Exécuter les options modales) du script dans la page 'Scripts'. Comme il n'existe aucune association entre le script IEG et un schéma DS, il vous sera demandé de sélectionner un schéma avec lequel exécuter le script dans une liste déroulante lorsque vous sélectionnez l'option pour exécuter un script. Si vous cliquez sur le bouton 'Exécuter un script', le lecteur IEG démarre et vous êtes redirigé vers la première page du script.

2.4.4.1 Validation d'un script

Lorsqu'un script est exécuté de cette manière via les écrans d'administration, le script est validé avant son exécution. Vous pouvez également choisir l'option 'Validate' (Valider) pour le script dans la page 'Scripts'. Tous les scripts doivent être validés avant leur exécution. Si la validation du script échoue, une liste des erreurs de validation s'affiche. Les erreurs de validation doivent être traitées avant même que le script puisse être exécuté dans la page 'Scripts'.

Indiquez des exemples de données sur la première page du script, puis sélectionnez le bouton Suivant. Ces mêmes exemples de données doivent maintenant figurer dans la page récapitulative. Les réponses ne sont pas modifiables, mais un lien d'édition est fourni pour accéder à la page où les données ont été entrées.

Si vous appuyez sur le bouton Suivant du lecteur IEG dans la page récapitulative du script qui a été implémenté dans cet exemple, une erreur s'affiche. Cela est dû au fait que toutes les propriétés du script ont été définies. Les propriétés requises seront traitées ultérieurement dans ce manuel.

Chapitre 3. Capture des informations client

3.1 Introduction

Le chapitre précédent fournissait un exemple de base d'utilisation d'IEG pour capturer les données d'un client. Certains formulaires de demande de prestations et de services peuvent être complexes et les informations requises sur les candidats peuvent être très détaillées.

Nous allons commencer par nous appuyer sur l'exemple initial couvert dans le chapitre précédent en prenant un foyer où nous avons capturé certaines données initiales sur un membre principal et souhaitons ajouter les détails des autres membres du ménage.

3.2 Familles et foyers

Nous disposons actuellement d'un script simple relatif à une seule personne. Souvent, les applications nécessitent davantage d'informations sur la situation du client, en commençant par leur situation.

En général, les informations sont requises sur la personne principale, puis suivies par une question simple qui permettra au client de passer à une autre zone de l'application. Par exemple, une fois les informations personnelles saisies, il est demandé au client s'il vit seul. Si la réponse est oui, la personne peut être traitée comme une personne seule qui ne vit pas dans un foyer ou avec d'autres personnes. La plupart des clients souhaitent parcourir le processus d'application le plus vite possible. Par conséquent, de telles questions permettent d'accéder à des parties plus pertinentes de l'application.

Si le client vit avec d'autres personnes, il est nécessaire de poser des questions sur chaque personne. Les boucles permettent de capturer les informations de chaque personne. Selon la façon dont l'auteur du script souhaite présenter ces questions, plusieurs types de boucle sont disponibles : for, while et for-each.

IEG dispose également d'un onglet Personne qui permet au client de voir les personnes concernées par ces questions lors de la saisie des données. Cet onglet apparaît automatiquement pour chaque entité Person dans le magasin de données. Chaque personne est représentée par une icône (basée sur le sexe et l'âge) et un nom. La personne actuellement sélectionnée est mise en surbrillance.

Prenons un scénario pour la gestion des données de la famille/du foyer comme une extension des exigences dans le modèle de base. Dans ce cas, il est demandé au client de connaître le nombre de personnes du foyer, y compris le client. De nouvelles pages de question doivent être ajoutées pour capturer ces informations.

La première page de question concerne la situation. Dans cet exemple, l'unique question à poser est la suivante : Hormis vous-même, de combien de personnes se compose la famille ?

```
<question-page id="HouseholdPage" progress="10">
    <title id="LoopControlPage.Title">
        <![CDATA[Détails du foyer]]>
    </title>
    <description id="LoopControlPage.Description">
        <![CDATA[Veuillez nous fournir les informations sur votre
        foyer]]>
    </description>
    <icon image="sample_title_household" />
    <cluster>
        <title id="DetailsCluster.Title">
            <![CDATA[Détails]]>
        </title>
        <question id="numPeople" control-question="true"
            control-question-type="IEG_INT32"
            mandatory="true">
            <label id="NumPeople.Label">
                <![CDATA[De combien de personnes se compose votre
                foyer ?]]>
            </label>
        </question>
    </cluster>
</question-page>
```

Figure 8. Obtention de la taille du foyer

Il s'agit d'une question de contrôle, c'est-à-dire une question qui permet de contrôler la taille d'une boucle, et non d'une question à des fins de collecte de données. Les questions de contrôle ne sont pas stockées dans le schéma de magasin de données. Elle seront utilisées dans l'expression de boucle 'for' de la page de questions suivante.

La page de questions aux membres du foyer se trouve à l'intérieur d'une boucle 'for' qui va itérer sur le nombre de membres de la famille.

```
<loop loop-type="for" loop-expression="numPeople"
    entity="Person" criteria="isPrimary==false">
    <question-page id="PersonDetailsPage"
        show-person-tabs="true"
        progress="20">
        <title id="PersonDetailsPage.Title">
            <![CDATA[Détails du membre du foyer]]>
        </title>
        <description id="PersonDetailsPage.Description">
            <![CDATA[Veuillez nous fournir les informations de la
                personne suivante de votre foyer]]>
        </description>
        <icon image="sample_title_household" />
        <cluster>
            <title id="DetailsCluster.Title">
                <![CDATA[Détails de la personne]]>
            </title>
            <description id="DetailsCluster.Description">
                <![CDATA[Entrez les détails de la personne
                ci-dessous]]>
            </description>
            <question id="firstName" mandatory="true">
                <label id="FirstName.Label">
                    <![CDATA[Prénom :]]>
                </label>
            </question>
            <question id="lastName">
                <label id="lastName.Label">
                    <![CDATA[Nom :]]>
                </label>
            </question>
            <question id="gender" mandatory="true">
                <label id="Gender.Label">
                    <![CDATA[Sexe :]]>
                </label>
            </question>
        </cluster>
    </question-page>
</loop>
```

Figure 9. Utilisation de la boucle 'for' pour collecter les membres du foyer

L'exemple ci-dessus présente comment le client entre le nombre de membres qui composent le foyer. La question aurait pu être posée différemment : 'Vivez-vous avec votre famille ?' Dans ce cas, un élément de condition dans le script peut être utilisé pour vérifier la valeur de cette question. La page du membre de la famille s'affiche s'il vit avec sa famille. Sur cette page de questions, une question de contrôle est posée pour déterminer s'il souhaite capturer les détails d'un autre membre de la famille.

Cette question de contrôle est utilisée dans une boucle 'while' autour de la page de questions du membre de la famille, comme suit :

```
<question-page id="HouseholdPage" progress="10">
    <title id="LoopControlPage.Title">
        <![CDATA[Détails du foyer]]>
    </title>
    <description id="LoopControlPage.Description">
        <![CDATA[Veuillez nous fournir les informations sur votre
        foyer]]>
    </description>
    <icon image="sample_title_household" />
    <cluster>
        <title id="DetailsCluster.Title">
            <![CDATA[Détails]]>
        </title>
        <question id="livesWithFamily" control-question="true"
            control-question-type="IEG_BOOLEAN"
            mandatory="true">
            <label id="NumPeople.Label">
                <![CDATA[Vivez-vous avec votre famille ?]]>
            </label>
        </question>
    </cluster>
</question-page>
```

Figure 10. Utilisation de la boucle 'while' pour collecter les informations sur les membres du foyer

Grâce à cette méthode, la question de contrôle est de type booléen, car elle est utilisée dans une expression de condition qui indique si la boucle while doit être saisie ou non. Une fois saisie, la boucle est itérée jusqu'à la collecte de tous les membres du foyer, comme suit :

```
<condition expression="livesWithFamily==true">
  <loop loop-type="while" loop-expression="
    anotherMember==true"
    entity="Person">
      <question-page id="PersonDetailsPage"
       show-person-tabs="true"
         progress="20">
           <title id="PersonDetailsPage.Title">
               <![CDATA[Détails du membre du foyer]]>
           </title>
           <description id="PersonDetailsPage.Description">
               <![CDATA[Veuillez fournir les informations de la
                 personne suivante de votre foyer]]>
           </description>
           <icon image="sample_title_household" />
           <cluster>
               <title id="DetailsCluster.Title">
                   <![CDATA[Détails de la personne]]>
               </title>
               <description id="DetailsCluster.Description">
                   <![CDATA[Entrez les détails de la personne
                   ci-dessous]]>
               </description>
           <question id="firstName" mandatory="true">
               <label id="FirstName.Label">
                   <![CDATA[Prénom :]]>
               </label>
           </question>
           <question id="lastName">
               <label id="lastName.Label">
                   <![CDATA[Nom :]]>
               </label>
           </question>
           <question id="gender" mandatory="true">
               <label id="Gender.Label">
                   <![CDATA[Sexe :]]>
               </label>
           </question>
           </cluster>
           <cluster>
               <question id="anotherMember"
               control-question="true"
                 control-question-type="IEG_BOOLEAN">
                   <label id="AnotherMember.Label">
                       <![CDATA[Y a-t-il un autre
                       membre dans votre foyer ?]]>
                   </label>
               </question>
           </cluster>
      </question-page>
  </loop>
</condition>
```

Figure 11. Utilisation de la boucle while pour collecter les membres du foyer

3.3 Relations du foyer

Lors de la collecte des informations sur les membres d'un foyer, il peut être nécessaire de déterminer comment ces gens sont liés les uns aux autres. IEG fournit un mécanisme pour capturer les relations via l'utilisation de pages de relation et une structure de schéma de magasin spécifique.

Une entité Relationship doit être définie dans le schéma de magasin de données, sous la forme suivante :

```
<xsd:element name="Person">
  <xsd:complexType>
    <xsd:sequence minOccurs="0">
      <xsd:element ref="Relationship" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:sequence>
    ...
</xsd:element>
<xsd:element name="Relationship">
  <xsd:complexType>
    <xsd:attribute name="relationshipType"
    type="IEG_STRING"/>
    <xsd:attribute name="isNonParentPrimaryCaretaker"
        type="IEG_BOOLEAN" default="false"/>
    <xsd:attribute name="personID" type="D:SVR_KEY"/>
  </xsd:complexType>
</xsd:element>
```

Figure 12. Entité Relationship dans un schéma de magasin de données

Une page de relation pour le foyer peut être définie comme suit, sous réserve que l'entité Relationship est un enfant de l'entité Person :

```
<relationship-page id="RelationshipPage" show-person-tabs="true"
    progress="40">
    <title id="RelationshipPage.Title">
        <![CDATA[Relations du foyer]]>
    </title>
    <description id="RelationshipPage.Description">
    <![CDATA[Veuillez entrer les informations de %1s ci-dessous]]>
      <argument id="Person.firstName" />
    </description>
    <icon image="sample_title_household" />
    <question id="caretakerInd">
        <label id="CaretakerInd.Label">
            <![CDATA[S'agit-t-il d'un tuteur
            non parent ?]]>
        </label>
    </question>
</relationship-page>
```

Figure 13. Page de relation

Il est uniquement nécessaire de définir la page de relations une fois. IEG affiche alors la page autant de fois que nécessaire pour regrouper les relations d'une personne à la fois. Ceci équivaut à une fois moins que le nombre de membres du foyer, car les relations de la dernière personne auront été regroupées dans leur intégralité via le processus.

Par défaut le champ Type de relation se présente sous la forme d'une liste déroulante, remplie à partir d'une table de codes (configurable via la propriété `relationship.type.domain.name`) :

La page de relation contient un onglet Personne dans la partie supérieure. Il renferme la liste des membres du foyer et la personne en cours est mise en évidence. Ensuite, chaque relation entre la personne en cours et les autres membres s'affiche.

L'indicateur intérimaire est la seule question qui peut être ajoutée directement à la page de relations. Les questions concernant les autres attributs d'une entité Relationship doivent être ajoutées à des clusters qui ont été ajoutés à la page de relations.

3.4 Récapitulation des informations client

Les listes sont utilisées dans les pages récapitulatives pour afficher les informations collectées dans les boucles. La structure de la liste doit refléter la structure de la boucle ou la hiérarchie des boucles qui a collecté les données. Cela signifie que l'entité et les critères de la liste doivent correspondre à l'entité et les critères de la boucle. Par exemple, pour enregistrer les membres de la famille décrite dans 3.2, «Familles et foyers», à la page 13, une boucle for a été utilisée :

```
<loop loop-type="for" loop-expression="numPeople"
    entity="Person" criteria="isPrimary==false">
  ...
</loop>
```

Figure 14. Boucle For pour collecter les informations de membre du foyer

Dans la page récapitulative de la section, les informations regroupées dans cette boucle sont affichées dans une liste. La liste, comme la boucle, possède une entité 'Person' et des critères 'isPrimary==false' :

```
<list entity="Person" criteria="isPrimary==false">
  ...
</list>
```

Figure 15. Liste des personnes

Les informations de relation regroupées à l'aide d'une page de relation peuvent être affichées sur des pages récapitulatives dans les listes récapitulatives de relation :

```
<relationship-summary-list>
    <title id="RelationshipSummaryList.Title">
        <![CDATA[Récapitulatif des relations de la personne]]>
    </title>
    <description id="PersonRelationshipSummaryList.Description">
        <![CDATA[Détails du récapitulatif des relations de la personne]]>
    </description>
    <column id="caretakerInd">
        <title id="CaretakerInd.Title">
            <![CDATA[Tuteur non parent]]>
        </title>
    </column>
    <edit-link start-page="RelationshipPage" />
</relationship-summary-list>
```

Figure 16. Liste récapitulative de relation

Chapitre 4. Capture de données associées

4.1 Introduction

Maintenant que nous avons capturé des informations sur les membres du foyer telles que leurs informations personnelles et leurs relations, nous voudrons peut-être capturer les données associées. Cette opération est possible par composition (utilisation d'entités DS imbriquées) ou association (utilisation d'entités DS associées et non imbriquées).

4.2 Capture de données composite

Nous avons vu précédemment qu'il est possible de capturer des relations dans IEG. La combinaison de l'entité Relationship et de l'entité RelationshipPage permet de capturer les relations entre les membres d'un foyer. La relation entre les membres d'un foyer est uniquement une forme de relation. IEG prend en charge d'autres types de relation. IEG et le magasin de données permettent d'imbriquer des entités en créant une relation parent-enfant. Vous pouvez observer cette opération dans l'exemple nécessitant de capturer les revenus des membres d'un foyer. L'entité Income est définie de la même manière que toute autre entité. Elle est imbriquée dans l'entité Person en la référençant dans une séquence, comme l'illustre l'exemple de fragment de code suivant :

```
<xsd:element name="Person">
  <xsd:complexType>
    <xsd:sequence minOccurs="0">
      <xsd:element ref="Income" minOccurs="0"
        maxOccurs="unbounded"/>
    </xsd:sequence>
    ...
    <xsd:attribute name="hasIncome" type="IEG_BOOLEAN"
      default="false"/>
  </xsd:complexType>
  ...
</xsd:element>
<xsd:element name="Income">
  <xsd:complexType>
    <xsd:attribute name="type" type="IEG_STRING" />
    <xsd:attribute name="amount" type="IEG_MONEY" />
  </xsd:complexType>
</xsd:element>
```

Figure 17. Schéma parent/enfant

Les informations relatives aux revenus peuvent ensuite être regroupées dans un foyer en mettant en boucle chaque membre possédant un revenu. Le critères de boucle vont utiliser une question booléenne "hasIncome" qui sera posée lors du regroupement des informations de chaque personne. Une page dans la boucle peut être mappée sur l'entité Income. La relation imbriquée est ainsi créée, comme illustré ci-dessous :

```
<loop loop-type="for-each" entity="Person"
        criteria="hasIncome==true">
  <loop loop-type="while" loop-expression="hasMoreIncome"
          entity="Income">
    <question-page id="IncomePage" entity="Income"
    ...
```

Figure 18. Création d'entités imbriquées

4.3 Affichage de données composite dans un récapitulatif

Les informations collectées pour les entités imbriquées peuvent être affichées dans une page récapitulative à l'aide d'une liste imbriquée. Comme pour les listes standard, les listes imbriquées doivent correspondre aux entités et aux critères utilisés dans les boucles imbriquées qui ont capturé les données.

```
<list entity="Person" show-icons="true" criteria="hasIncome==true">
  <title id="IncomeList.Title">
      <![CDATA[Revenus]]>
  </title>
  <description id="IncomeList.Description">
    <![CDATA[Voici les détails des revenus entrés pour tous les
                  membres de votre foyer]]>
  </description>
  <column id="firstName">
    <title id="FirstName.Title">
      <![CDATA[Prénom]]>
    </title>
  </column>
  <list entity="Income">
    <column id="type">
      <title id="IncomeType.Title">
        <![CDATA[Type de revenu]]>
      </title>
    </column>
    <column id="amount">
      <title id="IncomeAmount.Title">
        <![CDATA[Montant du revenu]]]>
      </title>
    </column>
  </list>
</list>
```

Figure 19. Affichage de entités imbriquées dans la page récapitulative

L'exemple de fragment de code d'une liste récapitulative de revenus ci-dessus s'affiche dans le lecteur IEG sous la forme d'une liste standard avec des revenus regroupés par personne. Il va également contenir les Editer et Supprimer pour chaque revenu et un lien Ajouter avec une liste de toutes les personnes.

4.4 Capture des données associées

IEG permet de créer des relations d'association entre les entités. Cette fonction est utile car une restriction s'applique aux entités et aux listes imbriquées qui ne peuvent être imbriquées qu'à deux niveaux. L'utilisation de relations associées propose une alternative efficace à l'imbrication d'entités à trois niveaux.

Par exemple, supposons qu'il soit nécessaire d'enregistrer les informations relatives aux emplois des membres d'un foyer. Les informations relatives à l'emploi peuvent être regroupées indépendamment des informations relatives aux revenus car il existe plusieurs revenus pour un emploi donné.

Une fois les informations relatives aux revenus et à l'emploi regroupées et les entités créées, l'association entre les entités peut être effectuée. Pour ce faire, il est nécessaire de créer une entité de relation. L'entité de relation est normalement la propriété de l'une des entités participant à la relation et est représentée sous la forme d'une séquence comme avec d'autres types de relation.

La définition d'une entité de relation nécessite de pouvoir identifier l'entité associée. Une clé doit donc être définie dans l'entité associée. Pour appliquer cela à l'exemple des revenus/d'emploi, le type d'entité Employment disposera d'une clé, un type d'entité Employment Relationship sera défini et l'entité Income disposera d'une séquence d'entitéEmploymentRelationship, comme suit :

```
<xsd:element name="Employment">
  <xsd:complexType>
    <xsd:attribute name="employmentID" type="d:SVR_KEY" />
    <xsd:attribute name="employer" type="IEG_STRING" />
    <xsd:attribute name="employmentType" type="IEG_STRING" />
  </xsd:complexType>
  <xsd:key name="Employment_Key">
    <xsd:selector xpath="./Employment" />
    <xsd:field xpath="@employmentID" />
  </xsd:key>
</xsd:element>
<xsd:element name="Income">
  <xsd:complexType>
    <xsd:sequence minOccurs="0">
      <xsd:element ref="EmploymentRelationship" minOccurs="0"
                maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="type" type="IEG_STRING" />
    <xsd:attribute name="amount" type="IEG_MONEY" />
  </xsd:complexType>
</xsd:element>
<xsd:element name="EmploymentRelationship">
  <xsd:complexType>
    <xsd:attribute name="employmentID" type="d:SVR_KEY" />
  </xsd:complexType>
</xsd:element>
```

Figure 20. Schéma des entités associées

L'association peut ensuite être capturée dans le script en définissant un attribut list-question et en indiquant un attribut link-entity faisant référence à la clé de l'entité associée. Pour poursuivre notre exemple, une liste de question peut être définie sur une page mappée sur l'entité Income en indiquant la clé à partir de l'entité Employment Relationship utilisée pour identifier l'entité Employment.

Les listes de questions sont des structures qui permettent à l'utilisateur de faire son choix dans une liste d'entités. Pour plus d'informations, voir 5.2, «Questions de liste», à la page 27.

```
<question-page id="IncomePage" entity="Income"  ...
  <cluster>
    <layout>
      <label-width>0</label-width>
    </layout>
    <list-question link-entity="EmploymentRelationship.employmentID"
        entity="Employment" single-select="true">
      <label id="SelectEmployer.Label">
        <![CDATA[Sélectionner l'employeur]]>
      </label>
      <item-label>
        <label-element attribute-id="employer" />
      </item-label>
    </list-question>
  </cluster>
</question-page>
```

Figure 21. Création de relations d'association

4.5 Affichage des données associées sur un récapitulatif

L'association entre les entités peut s'afficher dans une page récapitulative en ajoutant une colonne à la liste des entités d'un type, pour afficher les informations de l'entité associée. Un attribut lien-entité doit être indiqué dans cette colonne pour identifier l'entité associée.

L'exemple suivant illustre comment afficher le nom de l'employeur associé lorsque les revenus d'une personne sont répertoriés dans une page récapitulative :

```
<summary-page id="IncomeSummary"
  ...
  <list entity="Person" criteria="hasIncome==true"
                          show-icons="true">
    <title id="IncomeList.Title">Income</title>
    <description id="IncomeList.Description">Voici les détails des revenus
        entrés pour tous les membres de votre
        foyer</description>
    <column id="firstName">
      <title id="FirstName.Title">First Name</title>
    </column>
    <list entity="Income" show-icons="false">
      <column id="type">
        <title id="IncomeType.Title">Income Type</title>
      </column>
      <column id="amount">
        <title id="IncomeAmount.Title">Income Amount</title>
      </column>
      <column id="employer"
              link-entity="EmploymentRelationship.employmentID"
              entity="Employment">
        <title id="Employer.Title">Employer</title>
      </column>
    </list>
  </list>
</summary-page>
```

Figure 22. Page récapitulative d'association des entités

4.6 Suppression de données associées

Lorsque des entités forment des relations parent-enfant et que l'entité parent est supprimée, toutes ses entités enfant le sont également. Lorsqu'une entité qui participe à une relation est supprimée, les relations de cette entité sont supprimées par défaut, mais les entités associés sont conservées.

Par exemple, supposons que les informations de tous les membres d'un foyer ont été regroupées et les entités Person ont été créées et que les relations entre les membres du foyer ont également été capturées et les entités Relationship créées. Si l'utilisateur choisit de supprimer une personne, les relations auxquelles participe cette personne sont également supprimées, mais aucune autre personne du foyer n'est supprimée.

Ce comportement par défaut s'applique également à l'exemple d'emploi/de revenus. Si l'utilisateur choisit de supprimer un revenu, les entitésEmployment Relationship du revenu seront supprimées, mais toutes les entités Employment seront conservées.

Vous pouvez modifier le comportement par défaut lors de la suppression des entités associées de sorte que les entités associées à l'entité en cours seront également supprimées.

Pour modifier le comportement par défaut, une annotation contenant un élément de documentation peut être ajoutée à la définition d'une entité de relation dans le schéma DS. Un élément de documentation contenant le texte `"@curam.ieg.cascading.delete=true"` indique que les entités associées doivent être supprimées lorsque la relation est supprimée.

```
<xsd:element name="EmploymentRelationship">
  <xsd:annotation>
    <xsd:documentation>@curam.ieg.cascading.delete=true
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:attribute name="employmentID" type="d:SVR_KEY" />
  </xsd:complexType>
</xsd:element>
```

Figure 23. Schéma de suppressions en cascade

Dans l'exemple de revenu/d'emploi, si `curam.ieg.cascading.delete` est défini sur true pour l'entité Employment Relationship lorsqu'une entité Income est supprimée, les entités Employment associées le sont également. La suppression des entités Employment de cette manière ne provoque pas la suppression d'autres entités Income.

Chapitre 5. Capturer efficacement des données

5.1 Introduction

Ce chapitre présente certaines fonctions d'IEG permettant de regrouper les informations plus efficacement et intuitivement.

5.2 Questions de liste

Dans un exemple précédent, nous avons abordé un cas dans lequel il était nécessaire de regrouper les informations sur les revenus des personnes d'un foyer. Pour ne collecter que les informations sur les revenus des personnes qui en ont réellement, une question a été ajoutée à la page 'Détails des membres du foyer' pour indiquer si la personne a des revenus ou non.

IEG fournit une alternative permettant d'éviter de poser la même question booléenne à plusieurs entités. Une question de liste peut être utilisée pour regrouper simultanément toutes les réponses.

Poursuivons avec l'exemple précédent où des informations sur les personnes ont été collectées. L'attribut `hasIncome` a été ajouté à l'entité Person pour indiquer si les informations doivent être collectées pour la personne, comme suit :

```
<xs:element name="Person">
  <xs:complexType>
    ...
    <xs:attribute name="hasIncome" type="IEG_BOOLEAN"/>
```

Figure 24. Schéma Has Income Person

Comme les questions, les questions de liste doivent être ajoutées à un cluster. Les questions de liste nécessitent que vous indiquiez le type des entités qui seront affichées dans la liste. L'ID de la question de liste correspond au nom de l'attribut booléen qui doit être défini si l'utilisateur sélectionne un élément dans la liste. Comme avec les questions, une question de liste doit posséder un libellé pour indiquer le but de la question. Les questions de liste doivent également posséder un libellé d'élément. Le libellé d'élément indique l'attribut des entités à utiliser pour identifier les entités dans la liste. Dans l'exemple suivant, le premier nom des membres du foyer s'affiche pour les identifier.

```
<question-page id="AnyoneHaveIncome">
...
  <cluster>
    <list-question id="hasIncome" entity="Person">
      <label id="HasIncome.Label">
             <![CDATA[Quelles personnes ont des revenus ?]]>
      </label>
      <item-label>
        <label-element attribute-id="firstName" />
      </item-label>
    </list-question>
  </cluster>
</question-page>
```

Figure 25. Question de liste

Plutôt que d'ajouter une question à la boucle où les détails des membres du foyer sont collectés, une liste contenant les membres du foyer peut être affichée, une fois les détails capturés. L'utilisateur peut ensuite sélectionner les membres qui ont des revenus.

Les questions de liste sont particulièrement utiles lorsqu'elles sont utilisés avec une boucle for-each, faisant référence à la question qui a été définie dans l'expression de critère de la boucle de la question de liste. Les questions de liste peuvent également être utilisées avec des types d'entité autres que Person.

5.2.1 A sélection unique

Les questions de liste peuvent également être utilisées lorsque la sélection doit être mutuellement exclusive. Lorsque l'attribut `single-select` d'une question de liste est défini sur `true`, seul l'un des éléments de la liste peut être sélectionné.

Si, par exemple, il est nécessaire d'indiquer le membre du foyer qui est le tuteur principal, un attribut peut être ajouté à l'entité Person et une question de liste à sélection unique peut être ajoutée au script :

```
<xsd:element name="Person">
  <xsd:complexType>
    ...
    <xsd:attribute name="primaryCareGiver" type="IEG_BOOLEAN"/>
```

Figure 26. Schéma de tuteur principal

```
<question-page id="PrimaryCareGiver" ...>
...
  <cluster>
    <list-question id="primaryCareGiver" entity="Person"
          single-select="true" criteria="age > 14">
      <label id="PrimaryCareGiver.Label">
        <![CDATA[Qui est le tuteur principal ?]]>
      </label>
      <item-label>
        <label-element attribute-id="firstName" />
      </item-label>
    </list-question>
  </cluster>
```

Figure 27. Q-uestion de liste à sélection unique

La question de liste ci-dessus affiche la liste des membres du foyer de plus de 14 ans avec un bouton d'option en regard de chaque personne. Vous ne pouvez sélectionner qu'un membre à la fois.

5.3 Question de table de codes

Si un attribut est défini dans un schéma DS en tant que table de codes et que la question s'affiche, le comportement par défaut consiste à afficher la question en tant que liste déroulante. Vous ne pouvez sélectionner qu'une réponse dans la liste déroulante.

Par exemple, s'il s'agit d'une condition requise pour capturer l'état d'un membre du foyer, une nouvelle définition de domaine peut être ajoutée pour représenter la table de codes `AddressState` et un attribut peut être ajouté à l'entité Person pour stocker l'état comme suit :

```
...
<xsd:simpleType name="IEG_STATE_ADDRESS">
  <xsd:annotation>
    <xsd:appinfo>
      <D:options>
        <D:option name="code-table-name">AddressState</D:option>
      </D:options>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:restriction base="IEG_CODETABLE_CODE" />
</xsd:simpleType>
...

<xsd:element name="Person">
...
  <xsd:attribute name="homeState" type="IEG_STATE_ADDRESS" />
```

Figure 28. Table de codes et attribut d'état

Une question relative à la capture des informations d'état peut être ajoutée au script comme suit :

```
<question-page id="AboutYouPage" entity="Person">
...

  <cluster>
    <question id="homeState">
      <label id="State.Label">
        <![CDATA[Veuillez sélectionner votre état de résidence :]]>
      </label>
    </question>
  </cluster>
```

Figure 29. Question Table de codes d'état

Lorsque le script est exécuté, la question est présentée à l'utilisateur sous forme de liste déroulante.

IEG prend également en charge la définition des questions de table de codes de sorte que l'utilisateur puisse effectuer plusieurs sélections.

Lorsqu'une question de table de codes ne dispose que d'une seule sélection, la question peut être stockée dans l'attribut unique d'une entité. Une question de table de codes à sélection multiple pouvant posséder plusieurs réponses, une séquence doit être ajoutée pour stocker toutes les réponses et un nouveau type d'entité doit être défini pour représenter les réponses dans la séquence.

```
<xsd:element name="Person">

  <xsd:complexType>
    <xsd:sequence minOccurs="0">
      <xsd:element ref="State" minOccurs="0"
          maxOccurs="unbounded"/>
      </xsd:sequence>
...
  </xsd:complexType>
</xsd:element>

<xsd:element name="State">
  <xsd:complexType>
    <xsd:attribute name="stateCode" type="IEG_STATE_ADDRESS" />
  </xsd:complexType>
</xsd:element>
```

Figure 30. Entité State

La création d'une question de table de codes à sélection multiple s'effectue en définissant l'attribut `multi-select` de la question sur `true`. Lors de l'ajout d'une question de table de codes à sélection multiple, le cluster auquel la question est ajouté doit être mappé sur le nouveau type d'entité représentant les réponses à la question. Dans notre exemple, le cluster doit être mappé sur l'entité `State` . La page qui contient la question à sélection multiple doit être mappée sur l'entité qui contient la séquence. Dans cet exemple, la page doit être mappée sur l'entité Person. Enfin, pour qu'un certain nombre d'options dans une question de table de codes à sélection multiple soit visible, une présentation doit être ajoutée à la question. La présentation doit indiquer le nombre de lignes visibles pour la question. Si le nombre d'options disponibles de la question dépasse le nombre de lignes indiquées dans la présentation, une barre de défilement est ajoutée à la question.

```
<question-page id="AboutYouPage" entity="Person">
...

  <cluster entity="State">
    <question id="stateCode" multi-select="true">
      <label id="State.Label">
        <![CDATA[Veuillez sélectionnez vos états de résidence :]]>
      </label>
      <layout>
        <num-rows>4</num-rows>
      </layout>
    </question>
  </cluster>
```

Figure 31. Question Table de codes à sélection multiple

Lorsque le script est exécuté, la question est proposée à l'utilisateur sous la forme d'une liste de descriptions de table de codes avec une case à cocher pour chaque élément.

5.4 Eléments conditionnels

Les scripts IEG peuvent posséder plusieurs éléments conditionnels : sections, pages ou clusters. Les éléments conditionnels peuvent être affichés ou masqués en fonction des réponses des pages précédentes ou des données pre-renseignées dans le DS.

5.4.1 Sections conditionnelles

Vous pouvez supprimer des sections à partir d'un script en évaluant une expression au début de l'exécution. Si la section n'est pas visible, elle n'est pas répertoriée dans le panneau des sections et l'expression n'est pas réévaluée au cours de l'exécution du script.

A l'aide d'un DS pré-renseigné tel que décrit dans 8.3, «Pré-renseignement des scripts à l'aide des données capturées», à la page 55, nous pouvons définir un indicateur sur une entité en fonction de circonstances externes au script. Supposons que nous avons une entité appelée IntakeInformation qui a un attribut booléen "collectIncomeInformation". Nous pouvons définir une section de revenu dans notre script :

```
...
<section visible="IntakeInformation.collectIncomeInformation==true">
   ...
</section>
...
```

Figure 32. Attribut visible d'une section

La section Revenu est masquée si l'attribut "collectIncomeInformation" est défini sur false, comme si la section n'était pas présente dans la définition de script.

Si une section doit être activée ou désactivée selon les réponses des sections précédentes, il est possible d'encapsuler toutes les pages d'une section dans une seule condition. Contrairement à l'attribut visible, cette condition sera évaluée à chaque détection de la section. Cela signifie qu'il est possible de revenir en arrière et de modifier une réponse qui affecte la navigabilité d'une section. La section apparaît toujours dans le panneau Sections mais est grisé afin que l'utilisateur ne puisse pas cliquer dessus.

L'exemple précédent peut être modifié de sorte que la question "collectIncomeInformation" soit posée en début de script. La section Revenu peut ensuite être modifiée comme suit :

```
...
<section>
   <condition
     expression="IntakeInformation.collectIncomeInformation">
     ...
   </condition>
</section>
...
```

Figure 33. Section conditionnelle

5.4.2 Pages conditionnelles

Les pages peuvent s'afficher ou non, selon la valeur d'une expression de condition. Les boucles peuvent également être encapsulées dans ces conditions.

La section conditionnelle précédemment mentionnée où une condition encapsule tout le contenu de la section constitue un exemple de page conditionnelle.

5.4.3 Clusters conditionnels

Les clusters peuvent également être enveloppés dans un élément de condition. Si l'expression de l'élément de condition ne fait référence à aucune des questions sur la même page, le cluster est un cluster conditionnel statique. Cela est dû au fait qu'il peut être déterminé, avant l'affichage des pages, d'afficher ou non le cluster.

Si des informations sur les membres du foyer ont été regroupées, vous pouvez souhaiter ajouter une nouvelle page contenant d'autres informations personnelles (par exemple, la personne est-elle enceinte ?). Un nouvel attribut `isPregnant` doit être ajouté à l'entité Person pour stocker ces informations :

```
<xsd:element name="Person">
  <xsd:complexType>
    ...
    <xsd:attribute name="isPregnant" type="IEG_BOOLEAN"/>
```

Figure 34. Attribut Personne supplémentaire

Cette question ne s'applique évidemment qu'aux femmes. Par conséquent, le cluster peut être encapsulé dans une condition et ne s'affiche que si l'expression de condition est évaluée "true". La nouvelle page Détails de la personne supplémentaire peut être définie comme suit :

```
<question-page id="AboutTheClientContinued" entity="Person" ...>
  <condition expression="Person.gender==&quot;SX2&quot;">
    <cluster>
      <question id="isPregnant" mandatory="true">
          <label id="IsPregnant.Label">
              Etes-vous enceinte ?
          </label>
          <help-text id="IsPregnant.HelpText">
              Etes-vous enceinte ?
          </help-text>
      </question>
    </cluster>
  </condition>
</question-page>
```

Figure 35. Cluster statique conditionnel

Si l'une des questions référencées dans l'expression de condition se trouve sur la même page, le cluster devient un cluster conditionnel dynamique. Cela signifie que le l'affichage du cluster est activé ou désactivé lorsque l'utilisateur modifie les réponses aux autres questions de la page. Cette fonction dynamique d'IEG nécessite d'activer JavaScript dans le navigateur. Les expressions de cluster conditionnel dynamique peuvent ne pas faire référence aux fonctions personnalisées, car les expressions sont évaluées sans appel serveur.

Sans modification du schéma du DS, il s'agit d'un cluster conditionnel dynamique si l'exemple ci-dessus est modifié afin que le cluster conditionnel soit défini sur la même page que la question Sexe.

```
<question-page id="AboutTheClient" entity="Person" ...>
...
  <cluster>
    <title id="DetailsCluster.Title">
      <![CDATA[Détails personnels]]>
    </title>
...
    <question id="gender" mandatory="true">
      <label id="Gender.Label">
        <![CDATA[Sexe :]]>
      </label>
    </question>
...
  <condition expression="Person.gender==&quot;SX2&quot;">
    <cluster>
      <question id="isPregnant" mandatory="true">
        <label id="IsPregnant.Label">
        <![CDATA[Etes-vous enceinte ?]]>
        </label>
      </question>
    </cluster>
  </condition>
</question-page>
```

Figure 36. Cluster conditionnel dynamique

La question relative à la grossesse apparaît ou disparaît de façon dynamique lorsque vous modifiez la valeur associée au sexe. Le comportement dynamique sur une page peut être déclenché par des champs de texte, des champs de date, des cases à cocher, des boutons d'option et des éléments de sélection. Le comportement dynamique ne peut pas être déclenché par la réponse à une question à sélection multiple ou une matrice de questions en raison des restrictions de la syntaxe de l'expression.

Il convient de noter qu'un seul niveau de condition n'est autorisé dans un cluster. Par exemple, les clusters conditionnels ne peuvent pas être imbriqués dans d'autres conditions. L'expression de condition

pour une condition de cluster dynamique peut faire référence à des questions sur la même page qui sont elles-mêmes définies dans un cluster conditionnel dynamique. Les clusters deviennent alors dépendants en cascade.

5.5 Matrices de questions

Les questions de liste présentées dans 5.2, «Questions de liste», à la page 27 posent la même question booléenne sur un groupe d'entités. Il est possible de poser la question de table de codes pour un groupe d'entités à l'aide de matrices de questions.

Une matrice de questions affiche une liste de questions en fonction d'une table de codes. Pour chaque valeur de table de codes et chaque entité, une case à cocher s'affiche pour permettre à l'utilisateur de sélectionner toutes les valeurs qui s'appliquent à une entité particulière.

Par exemple, supposons qu'il s'agit d'une condition requise pour capturer les niveaux possibles d'abus de substance pour chaque membre du foyer. Une nouvelle définition de domaine peut être ajoutée pour représenter la table de codes `SubstanceAbuse` et un attribut permettant de stocker le niveau d'abus de substance peut être ajouté à l'entité Personne comme suit :

```
<xsd:simpleType name="IEG_SUBSTANCEABUSE">
  <xsd:annotation>
    <xsd:appinfo>
      <D:options>
        <D:option name="code-table-name">SubstanceAbuse</D:option>
      </D:options>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:restriction base="IEG_CODETABLE_CODE" />
</xsd:simpleType>

<xsd:element name="Person">
  <xsd:complexType>
    ...
    <xsd:attribute name="substanceAbuse"
                   type="IEG_SUBSTANCEABUSE" />
```

Figure 37. Attribut Substance Abuse

La matrice de questions est alors définie comme une question de liste standard. Son affichage diffère car elle se base sur une table de codes au lieu d'un booléen.

```
...
<list-question entity="Person" id="substanceAbuse"
      criteria="age > 14">
  <label id="SubstanceAbuse.Label">
    <![CDATA[Abus de substance :]]>
  </label>
  <item-label>
    <label-element attribute-id="firstName" />
  </item-label>
</list-question>
```

Figure 38. Exemple de code de matrice de questions

L'exemple ci-dessous présentant une matrice de questions collectant les informations d'abus de substance de plusieurs membres d'un foyer s'affiche dans le lecteur IEG sous la forme d'une matrice, avec chaque ligne et chaque colonne correspondant respectivement à une description de table de codes et une personne.

5.6 Navigation Fast Path

Par défaut, lorsqu'un utilisateur effectue une réitération dans un script, toutes les pages sont à nouveau affichées. Cela peut être difficile, particulièrement dans les foyers importants. La navigation Fast Path permet aux utilisateurs finaux de parcourir les scripts IEG plus rapidement en passant automatiquement les boucles ou les pages conditionnelles qui ont déjà été traitées.

Cette fonctionnalité est facultative et désactivée par défaut. Elle peut être activée dans des boucles et des conditions (pour activer la navigation Fast Path, voir le manuel `Authoring Scripts using Intelligent Evidence Gathering (IEG)`).

La première fois qu'un élément Fast Path est détecté, il se comporte normalement. Lorsque l'utilisateur parcourt le script ultérieurement, seules les nouvelles pages contenues dans ces éléments Fast Path seront affichées. Les pages précédemment affichées seront désormais ignorées. Cette fonctionnalité n'empêche pas d'éditer les données via les liens d'édition dans une page récapitulative, si nécessaire.

L'élément Fast Path peut être utilisé dans les scénarios suivants :

- Question de liste générant une boucle
- Critères d'éligibilité
- Conditions Fast Path
- Condition dans une boucle Fast Path

5.6.1 Question de liste générant une boucle

À l'aide de la même question de liste décrite dans 5.2, «Questions de liste», à la page 27, nous voulons regrouper des informations relatives aux revenus d'une personne d'un foyer. Nous allons utiliser une boucle Fast Path imbriquée comme décrit dans l'exemple suivant :

```
...
<loop loop-type="for-each" entity="Person"
  criteria="hasIncome==true" fast-path="true">
  <loop loop-type="while" loop-expression="hasMoreIncome"
    entity="Income">
    <question-page id="IncomePage" entity="Income"
      show-person-tabs="true">
      <title id="IncomePage.Title">
        <![CDATA[Détails des revenus]]>
      </title>
      <cluster>
        <title id="IncomeDetails.Title">
          <![CDATA[Détails des revenus]]>
        </title>
        <question id="type">
          <label id="Type.Label">
            <![CDATA[Type :]]>
          </label>
        </question>
        <question id="amount">
          <label id="Amount.Label">
            <![CDATA[Montant :]]>
          </label>
        </question>
        <question id="hasMoreIncome"
          control-question="true"
          control-question-type="IEG_BOOLEAN">
          <label id="ContinueQuestion.Label">
            <![CDATA[%1s a-t-il d'autres revenus ?]]>
            <argument id="Person.firstName" />
          </label>
        </question>
      </cluster>
    </question-page>
  </loop>
</loop>
```

Figure 39. Exemple de code de question de liste Fast Path générant une boucle

La première fois que la question de liste est détectée, les pages suivant la boucle vont regrouper les revenus des personnes qui ont été sélectionnées. Lors d'une nouvelle visite de la page contenant la question de liste, les situations suivantes peuvent alors se produire:

- Si les cases ne sont pas modifiées, un clic sur Suivant permet de passer la boucle des revenus et affiche la page qui suit la boucle.
- Si certaines cases sont décochées, un clic sur Suivant supprime les revenus correspondant aux personnes désélectionnées, passe la boucle des revenus et affiche la page qui suit la boucle.
- Si de nouvelles cases sont cochées, un clic sur Suivant passe les pages de revenus existantes, affiche le pages de revenus des nouvelles personnes sélectionnées et affiche la page qui suit la boucle.
- Si de nouvelles cases sont cochées et que d'autres ne le sont pas, un clic sur Suivant supprime les revenus correspondant aux personnes désélectionnées, passe les pages de revenus existantes, affiche les pages de revenus des nouvelles personnes sélectionnées et affiche la page qui suit la boucle.

5.6.2 Critères d'éligibilité

En nous appuyant sur le scénario précédent, nous pouvons filtrer les personnes qui seront affichées dans la question de liste (il n'est pas nécessaire de modifier la boucle). Seules les personnes de plus de 18 ans seront autorisées à saisir des revenus. Par conséquent, un critère est ajouté à la question de liste. Lorsque vous réitérez le script, les personnes du script peuvent ne plus correspondre aux critères et, par conséquent, ne plus apparaître dans la liste.

```
...
<list-question id="hasIncome" entity="Person" criteria="age > 18">
  <label id="HasIncome.Label">
    <![CDATA[Quelles personnes ont des revenus ?]]>
  </label>
  <item-label>
    <label-element attribute-id="firstName" />
  </item-label>
</list-question>
```

Figure 40. Exemple de code de question de liste avec critères d'éligibilité générant une boucle

Le comportement sera identique à celui mentionné lors du scénario précédent, mais si la date de naissance d'une personne est modifiée, la situation suivante va se produire :

- Si la personne n'est pas éligible (moins de 18 ans) et que des revenus ont été saisis, les revenus correspondants seront automatiquement supprimés dès que la nouvelle date de naissance aura été soumise.
- Si la personne devient éligible (plus de 18 ans), elle apparaît dans la question de liste (mais n'est pas sélectionnée) lors du prochain affichage de la page de question de liste.

5.6.3 Conditions Fast Path

Nous pouvons demander des informations relatives à la grossesse aux membres féminins du foyer via une page conditionnelle. Si la condition est définie comme Fast Path, les informations relatives à la grossesse seront masquées lors de la réitération des membres du foyer car les pages de la condition ne seront affichées que lors de la réitération dans le script si la condition a été précédemment évaluée false et qu'un élément a été modifié de sorte que la condition soit désormais évaluée true.

```
...
<question-page id="AboutYouPage" entity="Person">
  <title id="PrimaryPersonPage.Title">
    <![CDATA[A propos de vous]]>
  </title>
  <cluster>
    <title id="DetailsCluster.Title">
      <![CDATA[Détails personnels]]>
    </title>
    <question id="firstName" mandatory="true">
      <label id="FirstName.Label">
        <![CDATA[Prénom :]]>
      </label>
    </question>
    <question id="middleName">
      <label id="MiddleName.Label">
        <![CDATA[Deuxième prénom :]]>
      </label>
    </question>
    <question id="lastName">
      <label id="lastName.Label">
        <![CDATA[Nom :]]>
      </label>
    </question>
    <question id="gender" mandatory="true">
      <label id="Gender.Label">
        <![CDATA[Sexe :]]>
      </label>
    </question>
    <question id="dateOfBirth" mandatory="true">
      <label id="DateOfBirth.Label">
        <![CDATA[Date de naissance :]]>
      </label>
    </question>
  </cluster>
</question-page>
<condition expression="Person.gender==&quot;SX2&quot;"
  fast-path="true">
  <question-page id="PregnancyPage" entity="Person">
    <title id="PregnancyPage.Title">
      <![CDATA[A propos de vous : grossesse]]>
    </title>
    <cluster>
      <title id="DetailsCluster.Title">
        <![CDATA[Détails personnels sur votre grossesse]]>
      </title>
      <question id="isPregnant" >
        <label id="IsPregnant.Label">
          <![CDATA[Etes-vous enceinte ?]]>
        </label>
      </question>
    </cluster>
  </question-page>
</condition>
```

Figure 41. Exemple de code de conditions Fast Path

Lors de l'édition de détails personnels, les situations suivantes peuvent se produire :

- Si aucun changement n'a été apporté au sexe, la condition est ignorée si vous cliquez sur Suivant, que la condition ait été affichée ou non la première fois.
- Si le sexe passe de Masculin à Féminin, un clic sur Suivant affiche la page conditionnelle permettant de saisir les détails de la grossesse.
- Si le sexe passe de Féminin à Masculin, un clic sur Suivant supprime les détails de la grossesse et affiche la page après la condition.

5.6.4 Condition dans une boucle Fast Path

Lorsqu'une condition est définie dans une boucle Fast Path, le comportement est identique à celui observé lorsqu'un critère est utilisé sur la boucle au lieu d'imbriquer une condition, avec l'exception suivante : si la condition devient true, la page contenue dans la condition ne peut pas être affichée car la boucle ne possède pas de nouvelle itération à afficher et sera par conséquent ignorée. Si la condition devient false, la page et les données associées ne seront pas supprimées car la condition n'est pas réévaluée. Il est par conséquent recommandé d'utiliser un critère sur la boucle au lieu d'une condition.

```
...
<loop loop-type="for-each" entity="Person"
  fast-path="true">
  <condition expression="Person.hasIncome==true">
    <loop loop-type="while" loop-expression="hasMoreIncome"
      entity="Income">
      <question-page id="IncomePage" entity="Income"
        show-person-tabs="true">
        <title id="IncomePage.Title">
          <![CDATA[Détails des revenus]]>
        </title>
        <cluster>
          <title id="IncomeDetails.Title">
            <![CDATA[Détails des revenus]]>
          </title>
          <question id="type">
            <label id="Type.Label">
              <![CDATA[Type :]]>
            </label>
          </question>
          <question id="amount">
            <label id="Amount.Label">
              <![CDATA[Montant :]]>
            </label>
          </question>
          <question id="hasMoreIncome"
            control-question="true"
            control-question-type="IEG_BOOLEAN">
            <label id="ContinueQuestion.Label">
              <![CDATA[%1s a-t-il d'autres revenus ?]]>
              <argument id="Person.firstName" />
            </label>
          </question>
        </cluster>
      </question-page>
    </loop>
  </condition>
</loop>
```

Figure 42. Exemple de code de condition dans une boucle Fast Path

5.7 Suppression implicite

Chaque fois que possible, le moteur IEG tente de supprimer les données dès qu'il constate que celui-ci n'est plus utile.

Si une réponse est explicitement modifiée par l'utilisateur (par le biais d'une question standard, d'un attribut list-question ou set-, et non via un appel de fonction personnalisé), le moteur détecte si cette réponse est utilisée dans une expression de condition, un critère de question de liste ou un critère de boucle. Si c'est le cas, l'expression ou le critère est réévalué. S'il devient false, les pages correspondantes sont supprimées et les données associées sont supprimées sans avoir besoin de parcourir le script pour rencontrer les expression ou le critère.

Chapitre 6. Autres éléments à prendre en compte lors du développement d'un script

6.1 Introduction

Les différentes constructions qui ont été présentées jusqu'ici répondent à différents besoins de regroupement de preuves, mais certaines situations peuvent nécessiter des fonctionnalités supplémentaires telles que l'affichage des données en lecture seule ou le recours à des fonctionnalités externes. Ce chapitre détaille ces éléments.

Ce chapitre décrit également des éléments à prendre en compte lors de la gestion des scripts IEG en plaçant des scripts sous contrôle de la source et en chargeant les scripts dans la base de données.

6.2 Affichage des données en lecture seule

Parfois, les réponses à certaines questions doivent être affichées pour l'utilisateur de sorte qu'elles ne puissent pas être modifiées. C'est déjà le cas dans les pages récapitulatives où les utilisateurs peuvent examiner les réponses et utiliser le bouton Précédent ou éditer les liens afin de modifier ces réponses.

Sur une page de questions, un attribut booléen en lecture seule peut être défini sur true pour indiquer que toutes les questions qui s'y trouvent ne peuvent pas être éditées.

Il existe un mécanisme plus sophistiqué : les attributs d'expression en lecture seule peuvent être utilisés sur différents éléments de script (sections, tous types de page, clusters, questions et questions de liste). Si l'expression est définie sur true, cela s'applique à toutes les questions contenues dans l'élément. Sous sa forme la plus simple, l'expression est définie sur "true" si l'élément doit être absolument en lecture seule. Sur une page récapitulative, il en résulte que les liens d'ajout, d'édition et de suppression ne sont pas affichés.

En cas d'expression en lecture seule définie pour le cluster, les éléments de question et de script de question de liste, l'élément de script est ensuite activé ou désactivé, et non plus disponible en lecture seule, si l'une des questions référencées dans l'expression se trouve sur la même page que l'élément de script. Cela signifie que les questions seront activées et désactivées si l'utilisateur modifie les réponses aux autres questions de la page. Si l'expression en lecture seule d'un cluster référence une question sur la même page, toutes les questions contenues dans le cluster sont activées et désactivées. Cette fonction dynamique d'IEG nécessite d'activer JavaScript dans le navigateur. Les expressions permettant d'activer et de désactiver dynamiquement les questions peuvent ne pas faire référence aux fonctions personnalisées, car les expressions sont évaluées sans appel serveur.

Les expressions en lecture seule dynamiques peuvent également faire référence à des questions sur la même page qui se sont activées et désactivées dynamiquement. Cette opération crée une dépendance en cascade entre les questions. Une attention particulière doit être prise lors de la définition des expressions avec des dépendances en cascade car IEG ne tient pas compte du fait que les questions référencées dans l'expression en lecture seule sont activées ou non. IEG ne tient compte que de la valeur de la question. Cette situation peut être déroutante pour l'utilisateur car l'élément qui contrôle l'activation et la désactivation d'une questions peut ne pas être clairement déterminé.

Si l'attribut de magasin de données possède une valeur lorsqu'une question est affichée, il s'affiche, même si la question est initialement désactivée. La question peut alors être activée par l'utilisateur et l'utilisateur peut modifier la réponse. Si la question est désactivée, la valeur qu'elle possédait initialement est rétablie. Lorsqu'une page est soumis, l'attribut de magasin de données n'est pas mis à jour, à moins que la question ne soit activée. Par conséquent, si la page est réaffichée, la valeur d'origine de l'attribut de données de magasin de données est réaffichée.

Il n'est pas possible de marquer une question obligatoire si elle contient également une expression en lecture seule dynamique sur la question elle-même ou l'un de ses éléments parents.

L'activation et la désactivation dynamiques d'éléments de script ne sont pas prises en charge sur les pages de relation.

Les informations regroupées en boucles peuvent s'afficher dans les pages récapitulatives à l'aide de listes, mais il est également possible d'utiliser cette structure de liste sur des pages standard sans qu'il soit nécessaire d'indiquer une expression en lecture seule dans l'un des éléments qui enveloppe la liste. Les liaisons ne sont pas autorisées et constituent l'unique différence avec les listes récapitulatives.

Il est également possible de créer un script entier en lecture seule. Par exemple, cela peut être utile si un assistant social doit analyser un script sans pouvoir modifier aucune réponse. Le script est défini en lecture seule via IEGRuntimeAPI en définissant un indicateur en lecture seule sur l'exécution du script, comme indiqué ci-dessous :

```
...
//Définir l'indicateur en lecture seule.
IEGRuntime runtimeAPI = new IEGRuntime();
IEGScriptExecutionID runtimeExecID = new IEGScriptExecutionID();
runtimeExecID.executionID = execution.getExecutionID();
IEGReadOnlyFlag readOnlyFlag = new IEGReadOnlyFlag();
readOnlyFlag.readOnlyFlag = true;
runtimeAPI.setReadOnlyFlag(runtimeExecID, readOnlyFlag);
...
```

Figure 43. Définition de l'indicateur en lecture seule de l'exécution d'un script

6.3 Recours aux fonctionnalités externes à l'aide d'expressions

Les expressions sont disponibles à divers emplacements d'un script pour définir le comportement des boucles, des conditions, etc. Voir l'annexe Expression Syntaxe du manuelAuthoring Scripts using Intelligent Evidence Gathering (IEG) pour plus d'informations.

Ces expressions peuvent faire référence à des réponses et être combinées à l'aide de différents opérateurs. Elles peuvent même appeler les fonctions (sauf lorsqu'elles sont utilisées sur des clusters conditionnels dynamiques car ces expressions sont évaluées dans le navigateur).

Les fonctions décrites ci-dessus sont appelées Fonctions personnalisées et définies à l'aide du code Java™ . Selon leur utilisation, elles peuvent être de deux types :

- Les fonctions personnalisées qui peuvent prendre des paramètres (en appelant éventuellement une fonctionnalité externe) et renvoient une valeur. Elles ne modifient pas le contenu du DS. Elles sont utilisées dans la plupart des expressions.
- Lorsque l'objectif consiste à mettre à jour le contenu du DS, la fonction personnalisée peut être utilisée dans un élément autonome : `callout`. La valeur renvoyée est inutile (mais il doit s'agir d'une valeur booléenne). La fonction personnalisée ne doit pas mettre à jour les valeurs qui ont été données avant l'appel. Cela est dû au fait que le moteur IEG n'est pas informé des mises à jour effectuées en dehors du script. Par conséquent, il n'est pas possible d'effectuer les actions requises par les mises à jour.

Les exemples qui peuvent nécessiter l'appel de fonctionnalités externes sont la validation d'un code postal américain qu'un utilisateur a fourni et le remplissage d'une zone d'état en fonction du code postal indiqué. Nous allons maintenant démontrer ces deux usages.

Le schéma du DS devra être développé pour ajouter les 2 attributs suivants à l'entité Personne, comme suit :

```
<xsd:attribute name="state" type="IEG_STRING"/>
<xsd:attribute name="zipCode" type="IEG_STRING"/>
```

Figure 44. Attributs de personne supplémentaires dans le schéma du DS

Tout d'abord, commençons par valider un code postal américain : celui-ci se compose de cinq chiffres et les trois premiers indiquent l'état.

La page de détails de la personne mentionnée précédemment et la page récapitulative correspondante peuvent être modifiées via deux questions obligatoires supplémentaires : Etat et Code postal américain :

```
<question id="state" mandatory="true">
    <label id="State.Label">
        State:
    </label>
    <help-text id="State.HelpText">
        Votre état de résidenced
    </help-text>
</question>
<question id="zipCode" mandatory="true">
    <label id="ZipCode.Label">
        ZIP Code:
    </label>
    <help-text id="ZipCode.HelpText">
        Votre code postal américain
    </help-text>
</question>
```

Figure 45. Questions Etat et Code postal américain dans la définition du script

La fonction personnalisée qui procèdera à la validation doit alors être créée en tant que classe Java dans le module `curam.rules.functions`:

```
...
public class CustomFunctionValidateZipCode extends CustomFunctor {

  public Adaptor getAdaptorValue(final RulesParameters rp)
  throws AppException, InformationalException {

    final List<Adaptor> parameters = getParameters();
    final String zipCode =
      ((StringAdaptor) parameters.get(0)).getStringValue(rp);
    final String state =
      ((StringAdaptor) parameters.get(1)).getStringValue(rp);
    boolean valid = false;

    if (zipCode.length() == 5) {
      final String prefix = zipCode.substring(0, 3);
      //rechercher les préfixes d'état
      if (prefix.equals("100")
        && state.equalsIgnoreCase("New York")) {
        valid = true;
      }
      if (prefix.equals("900")
        && state.equalsIgnoreCase("Californie")) {
        valid = true;
      }
    }

    return AdaptorFactory.getBooleanAdaptor(Boolean.valueOf(valid));
  }

}
```

Figure 46. Fonction personnalisée pour valider le code postal américain

Les métadonnées suivantes de la fonction personnalisée doivent être insérées dans<yourcomponent>/rulesets/functions/CustomFunctionMetaData.xml:

```
<CustomFunctor name="CustomFunctionValidateZipCode">
  <parameters>
    <parameter>
      curam.util.rules.functor.Adaptor$StringAdaptor
    </parameter>
    <parameter>
      curam.util.rules.functor.Adaptor$StringAdaptor
    </parameter>
  </parameters>
  <returns>curam.util.rules.functor.Adaptor$BooleanAdaptor</returns>
</CustomFunctor>
```

Figure 47. Métadonnées de fonction personnalisée

Pour plus d'informations sur la définition des fonctions personnalisées, voir le manuel Cúram Rules Codification Guide.

Dans notre exemple, la fonction personnalisée ValidateZipCode n'accède pas à une base de données externe de recherche pour recherche l'état correspondant. Idéalement, elle devrait effectuer cette recherche, puis vérifier l'état renvoyé par rapport à l'état saisi. A des fins de simplification, seuls deux préfixes de code postal américain sont codés en dur ci-dessus.

La validation sera alors insérée dans la page des détails personnels :

```
<validation
  expression="ValidateZipCode(Person.zipCode, Person.state)">
    <message id="InvalidZipCode">
        Le code postal américain est non valide.
    </message>
</validation>
```

Figure 48. Validation du code postal américain dans la définition du script

Lorsque l'utilisateur clique sur Suivant, les réponses aux questions Code postal américain et Etat sont transmises à la fonction personnalisée, qui renvoie la valeur true si les réponses sont valides. La page suivante est ensuite affichée.

Si la fonction personnalisée renvoie la valeur false, le message indiqué dans la validation s'affiche en haut de la page Détails de la personne, bloquant l'accès à la page Suivant jusqu'à la soumission des réponses valides.

La fonction personnalisée n'a pas d'effet secondaire car elle ne modifie aucune donnée. Elle n'effectue qu'une opération en fonction des paramètres et renvoie un résultat.

Il serait également possible de supprimer l'indicateur obligatoire sur les deux nouvelles questions et de valider les réponses uniquement si elles ont toutes deux été fournies. L'expression de validation doit alors être modifiée comme suit, à l'aide de la fonction personnalisée prête à l'emploi isNotNull qui vérifie si le paramètre fourni possède la valeur NULL :

```
"not(isNotNull(Person.zipCode) and isNotNull(Person.state))
    or ValidateZipCode(Person.zipCode, Person.state)"
```

Figure 49. Expression de validation alternative

Vous pouvez sinon renseigner la question Etat en fonction du code postal américain. Pour ce faire, la page Détails de la personne ne demande que le code postal américain (avec l'indicateur obligatoire) et la page récapitulative affiche l'état et le code postal américain.

La fonction personnalisée suivante doit être définie :

```
...
public class CustomFunctionpopulateState extends CustomFunctor {

  public Adaptor getAdaptorValue(final RulesParameters rp)
  throws AppException, InformationalException {

    final IEG2Context ieg2Context = (IEG2Context) rp;
    final long rootEntityID = ieg2Context.getRootEntityID();
    String schemaName;
    //schemaName doit être codé en dur ou extrait en dehors d'IEG
    Datastore ds = null;
    try {
      ds =
        DatastoreFactory.newInstance().openDatastore(
            schemaName);
    } catch (NoSuchSchemaException e) {
      throw new AppException(IEG.ID_SCHEMA_NOT_FOUND);
    }

    Entity applicationEntity = ds.readEntity(rootEntityID);

    Entity personEntity =
      applicationEntity.getChildEntities(
        ds.getEntityType("Person"))[0];
    String zipCode = personEntity.getAttribute("zipCode");
    String state = "Unknown";
    final String prefix = zipCode.substring(0, 3);
    //rechercher les préfixes d'état
    if (prefix.equals("100")) {
      state = "New York";
    }
    if (prefix.equals("900")) {
      state = "Californie";
    }
    personEntity.setAttribute("state", state);
    personEntity.update();
    return AdaptorFactory.getBooleanAdaptor(new Boolean(true));
  }

}
```

Figure 50. Fonction personnalisée pour renseigner l'état

Et ses métadonnées :

```
<CustomFunctor name="CustomFunctionpopulateState">
  <returns>curam.util.rules.functor.Adaptor$BooleanAdaptor</returns>
</CustomFunctor>
```

Figure 51. Métadonnées de fonction personnalisée

Un élément d'appel doit être inséré entre la page Détails de la personne et la page récapitulative pour appel cette fonction personnalisée :

```
<callout id="populateAddress" expression="populateState()"/>
```

Figure 52. Appel pour renseigner l'état dans la définition de script

Cette fois, la fonction personnalisée modifie le DS en renseignant l'état dans l'entité Person. Le contexte contient l'ID d'entité racine et l'ID d'exécution, afin de faciliter la mise à jour du DS. Si l'appel se trouve dans une boucle, le contexte contient également l'ID d'entité en cours.

6.4 Réutilisation des scripts

Il est possible de décomposer une définition de script en plusieurs fichiers pour définir un mécanisme de réutilisation.

Pour ce faire, une définition de script doit faire référence à des sous-scripts. Chaque sous-script est un script autonome qui peut être exécuté indépendamment.

Voici un exemple de script utilisable comme sous-script :

```
<?xml version="1.0" encoding="UTF-8"?>
<ieg-script ...>
  <identifier id="Subscript" scriptversionnumber="V1" type="Test" />
  <question-page ...>
    ...
  </question-page>
  ...
</ieg-script>
```

Figure 53. Sous-script contenant des pages

Le script du fragment de code ci-dessus peut être inclus dans un autre script en tant que sous-script, comme suit :

```
<?xml version="1.0" encoding="UTF-8"?>
<ieg-script ...>
  <identifier id="Script" scriptversionnumber="V1" type="Test" />
  <section>
    <ieg-sub-script>
         <identifier id="Subscript"
         scriptversionnumber="V1" type="Test" />
    </ieg-sub-script>
  </section>
  <section>
    ...
  </section>
  ...
</ieg-script>
```

Figure 54. Inclusion d'un sous-script dans un script

Le point possible d'insertion d'un sous-script dans un script peut être comme suit :

- Si le script contient des sections et que le sous-script en contient également, le sous-script doit être inséré au niveau supérieur, sous l'élément ieg-script parent.
- Si le script contient des sections et que le sous-script n'en contient pas, le sous-script doit être inséré dans une section du script parent.
- Si le script ne contient pas de sections, le sous-script ne contient pas de sections. Il sera inséré au niveau supérieur, sous l'élément ieg-script.

Une autre limite à garder à l'esprit est qu'un sous-script ne peut apparaître qu'une fois dans un script car les ID de page doivent être uniques dans le script obtenu.

Notez qu'un script peut être utilisé comme sous-script à un autre emplacement. Lors de la modification des scripts, assurez-vous que les scripts auxquels il est fait référence sont retestés pour vérifier que les modifications n'ont pas un impact non souhaité.

6.5 Contrôle de la source et gestion des versions

Les définitions de script IEG son stockées dans la base de données. Lors de l'édition d'un script IEG à l'aide de l'éditeur IEG, le script est édité en place et mis à jour directement dans la base de données. Les définitions de script IEG sont les artefacts de développement. Du point de vue de la gestion de la configuration, il est important de placer ces artefacts sous contrôle de la source, comme vous le feriez avec d'autres artefacts.

Il est possible de télécharger une définition de script à partir des écrans d'administration du script IEG. Lorsque l'option permettant de télécharger un script est sélectionnée, le script est d'abord extrait de la base de données, les fichiers de propriétés associés à la définition de script sont extraits du magasin de ressources et les propriétés textuelles sont "injectées" dans la définition de script avant qu'il ne soit disponible. Toutefois, le téléchargement d'un script de cette manière ne fournit pas toutes les ressources qui peuvent être associées à une définition de script. Par exemple, il ne fournit pas de fichiers de propriétés dans plusieurs environnements locaux, ni d'images et d'icônes. Veuillez consulter l'annexe `Compliancy` du Guide du développeur `Authoring Scripts using Intelligent Evidence Gathering (IEG)` pour plus d'informations sur la représentation de la base de données d'un script IEG.

Lors du renseignement de la base de données avec les définitions de script, il est important d'être conscient des différences de fonctionnalités entre l'importation d'un script via les écrans d'administration de script IEG et le chargement des définitions de script via des fichiers DMX.

Chapitre 7. Intégration d'IEG dans une application Cúram

7.1 Introduction

Ce chapitre décrit comment IEG peut être intégré dans une application. IEG peut être intégré de deux manières : en ouvrant le lecteur dans un onglet ou dans une boîte de dialogue modale. Les tâches d'intégration qui sont traitées ici incluent la création de l'exécution de script, la configuration des pages de fin et de sortie, l'exécution dans un onglet, l'exécution dans une boîte de dialogue modale, le nettoyage des données d'application et la reprise des scripts.

7.2 Création de l'exécution d'un script

Avant d'ouvrir le lecteur IEG d'une application, il est recommandé de créer l'exécution du script à l'aide de l'API publique. L'ID d'exécution peut ensuite être transmis au lecteur.

L'exemple de fragment de code suivant illustre la création d'une exécution de script à l'aide de l'API publique :

```
...

// création de l'exécution de script
final IEGRuntime runtimeAPI = new IEGRuntime();
final IEGScriptExecutionIdentifier executionIdentifier =
  runtimeAPI.createScriptExecution(iegScriptID, schemaName);
```

Figure 55. Création d'une exécution de script

7.3 Définition d'une URL de redirection

Les attributs `finish-page` et `quit-page` d'un script IEG indiquent l'URL de redirection lorsque vous quittez le lecteur IEG. De cette manière, ils permettent de connecter le lecteur IEG et une application. Ces attributs sont détaillés dans le chapitre `IEG Script Element Reference` du guide du développeur `Authoring Scripts using Intelligent Evidence Gathering (IEG)`.

Modifiez l'exemple de script pour inclure ces attributs comme illustré ci-dessous :

```
<ieg-script
  finish-page="IEG2_listAllIEG2Scripts"
  quit-page="IEG2_listAllIEG2Scripts"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="ieg-schema.xsd">
    ...
<ieg-script>
```

Figure 56. Script avec attributs finish-page et quit-page définis

Dans l'exemple ci-dessous, la fin ou la sortie du script vous redirige vers la liste de tous les scripts IEG fournis dans les écrans d'administration.

7.4 Exécution du lecteur IEG dans un onglet

L'exécution d'un lecteur IEG dans un onglet nécessite moins d'efforts que l'exécution dans une boîte de dialogue modale. Le lien 'opening' (ouverture) pointe vers `ieg/Screening.do` et transmet le paramètre `executionID`. `Screening.do` appelle le lecteur IEG. Les paramètres sont les suivants :

Voici un exemple d'UIM de résolution qui ouvre le lecteur IEG dans un onglet :

```
<?xml version="1.0" encoding="UTF-8"?>
  <PAGE PAGE_ID="System_IEGResolver">
    <JSP_SCRIPTLET>
    <![CDATA[

      String scriptID = request.getParameter("scriptID");
      String scriptType = request.getParameter("scriptType");
      String scriptVersion = request.getParameter(
          "scriptVersion");
      String schemaName = request.getParameter("schemaName");
      String name = request.getParameter("name");

      String executionIDParam =
          request.getParameter("executionIDParam");
      String url = null;

      curam.omega3.request.RequestHandler
          rh = curam.omega3.request.
      RequestHandlerFactory.getRequestHandler(request);

      String context = request.getContextPath() + "/";

      if (executionIDParam == null) {
        // Besoin de vérifier la présence d'erreurs de validation de script
        // avant l'exécution du script.

        String contextWithUserPreferences = context +
          curam.omega3.user.UserPreferencesFactory
          .getUserPreferences(
              pageContext.getSession()).getLocale() + "/";

        curam.interfaces.IEGScriptAdminPkg.
        IEGScriptAdmin_checkForScriptErrors_TH
            iegScriptAdminCheckForErrors
                = new curam.interfaces.IEGScriptAdminPkg.
        IEGScriptAdmin_checkForScriptErrors_TH();

        iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$scriptID_idx, scriptID);
        iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$scriptType_idx,
            scriptType);
        iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$scriptVersion_idx,
            scriptVersion);
        iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$schemaName_idx,
            schemaName);
        //Appel de la méthode.
        iegScriptAdminCheckForErrors.callServer();

        String errorsPresentInScript =
            iegScriptAdminCheckForErrors.getFieldValue(
                iegScriptAdminCheckForErrors
                    .result$errorsExist_idx);
        boolean errorsPresent =
            Boolean.valueOf(errorsPresentInScript).booleanValue();

        if (errorsPresent) {

          // En cas d'erreurs, redirection vers la page d'erreur
          // de validation.
          String redirectTo = contextWithUserPreferences
          + "System_listValidationErrorsForRunPage.do"
          + "?name=" + name
          + "&scriptID=" + scriptID
          + "&scriptType=" + scriptType
          + "&scriptVersion=" + scriptVersion
          + "&schemaName=" + schemaName;
          url = redirectTo + "&amp;" + rh.getSystemParameters();

        } else {
```

7.5 Exécution du lecteur IEG dans une boîte de dialogue modale

Le lecteur IEG peut être ouvert dans une boîte de dialogue modale et certaines considérations spécifiques doivent être prises en compte à ce sujet par un développeur de script.

7.5.1 Ouverture du lecteur IEG dans une boîte de dialogue modale

Pour ouvrir le lecteur IEG dans une boîte de dialogue modale, ouvrez `Screening.do` dans le modal en transmettant les paramètres `executionID` et système à l'aide d'un UIM de résolution. `System_IEGResolverModal.uim` est fourni prêt à l'emploi pour effectuer cette opération :

```
<PAGE PAGE_ID="System_IEGResolverModal">
        <JSP_SCRIPTLET>
    <![CDATA[

      String scriptID = request.getParameter("scriptID");
      String scriptType = request.getParameter("scriptType");
      String scriptVersion =
        request.getParameter("scriptVersion");
      String schemaName = request.getParameter("schemaName");
      String name = request.getParameter("name");

      // Besoin de vérifier la présence d'erreurs de validation
      // de script avant l'exécution du script.
      curam.omega3.request.RequestHandler
        rh = curam.omega3.request.
          RequestHandlerFactory.getRequestHandler(request);

      String context = request.getContextPath() + "/";
      String contextWithUserPreferences = context +
       curam.omega3.user.UserPreferencesFactory
          .getUserPreferences(
              pageContext.getSession()).getLocale() + "/";

      String url = null;

      curam.interfaces.IEGScriptAdminPkg.
        IEGScriptAdmin_checkForScriptErrors_TH
            iegScriptAdminCheckForErrors
          = new curam.interfaces.IEGScriptAdminPkg.
            IEGScriptAdmin_checkForScriptErrors_TH();

      iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$scriptID_idx,
            scriptID);
      iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$scriptType_idx,
            scriptType);
      iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$scriptVersion_idx,
            scriptVersion);
      iegScriptAdminCheckForErrors.setFieldValue(
        iegScriptAdminCheckForErrors.key$schemaName_idx,
            schemaName);
      //Appel de la méthode.
      iegScriptAdminCheckForErrors.callServer();

      String errorsPresentInScript =
         iegScriptAdminCheckForErrors.getFieldValue(
           iegScriptAdminCheckForErrors.result$errorsExist_idx);
      boolean errorsPresent =
          Boolean.valueOf(errorsPresentInScript).
              booleanValue();
```

```
      if (errorsPresent) {

        // En cas d'erreurs, redirection vers la
        // page d'erreur de validation.
        String redirectTo = contextWithUserPreferences
          + "System_listValidationErrorsForModalPage.do"
          + "?name=" + name + "&scriptID=" + scriptID
                 + "&scriptType=" + scriptType
          + "&scriptVersion=" + scriptVersion
          + "&schemaName=" + schemaName;
        url = redirectTo + "&&" + rh.getSystemParameters();

      } else {

        // Appel de la méthode du script d'exécution et redirection
        // vers le lecteur IEG.
        curam.interfaces.IEGScriptAdminPkg.
          IEGScriptAdmin_runScript_TH iegScriptAdminRunScript
            = new curam.interfaces.IEGScriptAdminPkg.
              IEGScriptAdmin_runScript_TH();

        iegScriptAdminRunScript.setFieldValue(
          iegScriptAdminRunScript.key$dtls$scriptID_idx,
              scriptID);
              iegScriptAdminRunScript.setFieldValue(
          iegScriptAdminRunScript.key$dtls$scriptType_idx,
              scriptType);
        iegScriptAdminRunScript.setFieldValue(
          iegScriptAdminRunScript.key$dtls$scriptVersion_idx,
              scriptVersion);
        iegScriptAdminRunScript.setFieldValue(
          iegScriptAdminRunScript.key$schemaName_idx,
              schemaName);
        //Appel de la méthode.
        iegScriptAdminRunScript.callServer();

        String executionID =
            iegScriptAdminRunScript.getFieldValue(
          iegScriptAdminRunScript.result$executionID_idx);
        executionID = executionID.replaceAll(",", "");

        url = context + "ieg/Screening.do?"
              + "executionID=" + executionID
              + "&" + rh.getSystemParameters();
      }

      // Redirection vers la page correcte.
      response.sendRedirect(
          response.encodeRedirectURL(url));
    ]]>
  </JSP_SCRIPTLET>
</PAGE>
```

7.5.2 Sortie d'une exécution de script dans une boîte de dialogue modale

Un développeur de script dispose de deux méthodes pour terminer ou quitter une exécution de script IEG dans une boîte de dialogue modale :

- Fermeture directe de la boîte de dialogue modale et actualisation ou redirection dans l'onglet parent.
- Progression vers les autres écrans UIM de la boîte de dialogue modale.

7.5.2.1 Fermeture directe de la boîte de dialogue modale à la fin du script

Pour fermer une boîte de dialogue modale directement à la fin ou à la sortie (actions Quitter, Enregistrer & quitter) d'une exécution de script IEG, le développeur du script doit indiquer un UIM de résolution en

tant que page de fin et/ou de sortie. L'UIM de résolution doit à son tour appeler une page JSP personnalisée qui appelle la fonction JavaScript appropriée pour fermer la boîte de dialogue.

Par exemple, pour effectuer une redirection vers l'écran d'administration `IEG2_listAllIEG2Scripts`, incluez le scriptlet JSP suivant dans votre fichier UIM :

```
<PAGE
   PAGE_ID="IEG2_resolveFinishScript"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="file://Curam/UIMSchema.xsd"
 >
 <JSP_SCRIPTLET>
   <![CDATA[

     curam.omega3.request.RequestHandler
       rh = curam.omega3.request.RequestHandlerFactory
           .getRequestHandler(request);

     String context = request.getContextPath() + "/";
     context += curam.omega3.user.UserPreferencesFactory
         .getUserPreferences(
           pageContext.getSession()).getLocale() + "/";

     String url = "";
     url = context + "IEG2_listAllIEG2ScriptsPage.do";

     String forwardParams =
         request.getParameter("forwardParams");

     if (screenContext != null && screenContext
         .hasContextBits(
         curam.omega3.taglib.ScreenContext.MODAL)) {
       url += "?" +  rh.getSystemParameters();
       String encodeRedirectURL = response.encodeURL(url);
       response.sendRedirect(response.encodeRedirectURL(
           request.getContextPath() +
               "/ieg/CloseAndRedirect.jspx?redirect="
               + encodeRedirectURL));
     } else {
       url += "?" +  rh.getSystemParameters();
       response.sendRedirect(
           response.encodeRedirectURL(url));
     }

     ]]>
 </JSP_SCRIPTLET>
</PAGE>
```

`CloseAndRedirect.jspx` est fourni prêt à l'emploi pour fermer la boîte de dialogue modale et effectuer une redirection vers un UIM indiqué (si fourni) dans le parent.

7.5.2.2 Progression vers les autres écrans UIM de la boîte de dialogue modale

Pour maintenir la boîte de dialogue ouverte afin d'afficher les autres écrans UIM après l'exécution du script, indiquez la page UIM requise, comme la page de fin et/ou la page de sortie dans la définition de script IEG. Une fois l'UIM chargé, vous sortez d'IEG et le traitement UIM standard via une boîte de dialogue modale s'applique.

7.6 Nettoyage des données d'application

Le nettoyage des données d'application nécessite de supprimer les données de la table de base de données IEGEXECUTIONSTATE et du magasin de données (DS), le cas échéant. Cette section décrit en détail les tâches de nettoyage des données manuelles et automatiques que les auteurs de script doivent connaître et fournit des recommandations visant à garantir un nettoyage simple des données d'application.

Pour prendre en charge l'exécution d'un script IEG, des informations sur les exécutions de chaque script doivent être conservées par le moteur IEG. Par exemple, le moteur IEG doit conserver la page en cours de l'exécution du script. Le moteur IEG doit également conserver une liste des pages qui ont été présentées à l'utilisateur pour prendre en charge la navigation. Les réponses aux questions de contrôle ne sont pas conservées dans le DS et le moteur IEG doit également conserver une trace de celles-ci. Toutes les informations permettant de prendre en charge l'exécution d'un script IEG sont conservées dans la table IEGEXECUTIONSTATE. Lorsqu'un nouvel objet IEGScriptExecution est créé via l'API IEGScriptExecutionFactory, une entrée correspondante est créée dans la table IEGEXECUTIONSTATE. La table IEGEXECUTIONSTATE est une table "interne" destinée à être utilisée par le moteur IEG. Elle ne doit pas être modifiée ou étendue. Les auteurs de script ne doivent pas dépendre du/émettre des hypothèses en se basant sur le contenu de cette table car celui-ci peut être modifié sans préavis.

IEG n'a aucun moyen de savoir quand une entrée de la table IEGEXECUTIONSTATE n'est plus nécessaire. Par conséquent, les entrées sont conservées jusqu'à ce qu'elles soient explicitement supprimées. Pour éviter tout encombrement inutile de la table IEGEXECUTIONSTATE, si une exécution de script s'est terminée ou n'est pas reprise/réexécutée, son entrée dans la table doit être supprimée via la méthode removeScriptExecutionObject de l'API IEGScriptExecutionFactory.

IEG ne peut effectuer aucune supposition quant aux données utilisées pour identifier de façon unique et logique une exécution de script particulière, car celles-ci peuvent varier d'une définition de script à une autre. IEG ne peut identifier une exécution de script que via l'ID généré qui est affecté à l'exécution de script lors de sa création. Il est fortement recommandé que les auteurs de script implémentent un mécanisme permettant d'identifier les exécutions de script en associant des données uniques aux ID d'exécution de script. Une nouvelle table peut être créée pour gérer la relation entre les données qui identifient l'exécution et l'ID d'exécution pour faciliter la reprise de l'exécution de scripts. Lorsqu'elle n'est plus nécessaire, elle peut être supprimée. La suppression d'un script d'exécution ne supprime pas les données collectées qui sont conservées dans le DS.

A l'instar d'IEGEXECUTIONSTATE, le moteur IEG et le DS n'ont aucun moyen de savoir quand les données qui sont collectées lors de l'exécution du script et conservées dans le DS ne sont plus requises. Encore une fois, le DS peut être inutilement encombré par des entités qui ne sont plus requises. Il est prévu que les entités ne seront pas conservées indéfiniment, mais que les données collectées seront déplacées vers les tables d'application, puis supprimées du DS. Lorsqu'une entité est supprimée du DS, ses entités sont également supprimées. En conséquence, lorsque les données collectées à l'aide de l'exécution d'un script ont été déplacées vers les tables d'application et ne sont plus nécessaire, il suffit de supprimer l'entité racine pour l'exécution.

L'exemple de fragment de code suivant illustre la suppression de l'entité racine :

```
final Long applicationID = execution.getRootEntityID();
      final Entity rootEntity = datastore.readEntity(applicationID);
      rootEntity.delete();
```

Figure 58. Suppression de l'entité racine

7.7 Reprise des scripts exécutés

Il est possible d'arrêter l'exécution d'un script et de le reprendre ultérieurement.Pour ce faire, l'application doit prendre soin de stocker l'ID d'exécution dans une table personnalisée et de l'associer à un ID utilisateur. Voir 7.6, «Nettoyage des données d'application», à la page 53 pour plus d'informations.

Si la table IEGEXECUTIONSTATE n'a pas été nettoyée et que la définition de script n'a pas été modifiée, une exception de script est reprise en transmettant le paramètre executionID au lecteur IEG de la même manière que lors du démarrage d'une nouvelle exécution de script.

Chapitre 8. Gestion des données capturées

8.1 Introduction

Comme nous l'avons mentionné précédemment, les données capturées lors de l'exécution du script sont conservées dans le magasin de données (DS). Ce chapitre vous expliquera comment extraire les données capturées du magasin de données. Ce chapitre va également vous expliquer comment les données peuvent être insérées dans le magasin de données afin de le mettre à disposition d'IEG lors de l'exécution des scripts.

8.2 Extraction des données capturées

Le magasin de données (DS) possède une API publique que vous pouvez utiliser dans votre code d'application. Cette API est le plus souvent utilisée pour extraire des informations à partir d'un schéma renseigné, mais elle peut également être utilisée pour pré-renseigner un schéma. Par exemple, une fois qu'un client a terminé une application, il peut soumettre ses informations. A ce stade, l'API peut être utilisée pour extraire les données du schéma et renseigner les tables dans la base de données relationnelle.

Un exemple de pré-renseignement consiste à disposer d'informations sur le client avant de démarrer l'application. Si certaines de ces informations sont nécessaires pour naviguer dans l'application, le magasin DS peut être pré-renseigné avec les informations.

Pour lire des données à partir d'un schéma, l'exécution appropriée du script doit être connue. Cela signifie que vous extrayez les informations d'application correctes pour un client. Par conséquent, l'ID d'exécution et le nom du schéma sont essentiels pour accéder aux données.

L'exemple de fragment de code suivant illustre l'obtention de l'entité racine :

```
final IEGRuntime runtimeAPI = new IEGRuntime();
final IEGRootEntityID rootEntityID =
  runtimeAPI.getScriptExecutionRootEntityID(executionID);

Datastore ds = DatastoreFactory.newInstance()
  .openDatastore(kSchemaName);

final Entity rootEntity =
  ds.readEntity(rootEntityID.entityID));
```

Figure 59. Obtention de l'entité racine

L'entité racine peut ensuite être utilisée pour extraire d'autres entités sous cette entité racine.

8.3 Pré-renseignement des scripts à l'aide des données capturées

Il est possible de pré-renseigner les valeurs que pourra voir l'utilisateur afin celui-ci ait juste besoin de confirmer ou de modifier les réponses.

Par exemple, nous pouvez pré-renseigner le nom et la date de naissance d'un utilisateur dans une page Détails personnels sous réserve que l'utilisateur s'est déjà connecté et qu'une autre base de données contient les détails personnels.

Le magasin DS peut être pré-renseigné avant le démarrage d'une exécution de script comme suit :

```
...
Datastore ds = null;

try {
  // ouverture du magasin de données et création de l'entité racine
  ds = DatastoreFactory.newInstance().openDatastore(schemaName);
} catch (NoSuchSchemaException e) {
  throw new AppException(IEG.ID_SCHEMA_NOT_FOUND);
}

final EntityType appType = ds.getEntityType("Application");
final Entity rootElement = ds.newEntity(appType);

ds.addRootEntity(rootElement);

final EntityType personType = ds.getEntityType("Person");
final Entity person = ds.newEntity(personType);

person.setAttribute("firstName", "TestFirstName");
person.setAttribute("lastName", "TestLastName");
person.setAttribute("dateOfBirth", "19700101");
//...

rootElement.addChildEntity(person);
```

Figure 60. Fragment de code pour pré-renseigner le magasin DS

L'entité racine peut ensuite être utilisée pour créer une nouvelle exécution de script comme suit :

```
...

// création de l'exécution de script
final IEGRootEntityID rootEntityID = new IEGRootEntityID();
rootEntityID = rootElement.getUniqueID();
final IEGRuntime runtimeAPI = new IEGRuntime();
final IEGScriptExecutionIdentifier executionIdentifier =
  runtimeAPI.createScriptExecutionExistingRootEntity(
    iegScriptID, schemaName, rootEntityID);
```

Figure 61. Création d'une exécution de script

Le lecteur IEG peut ensuite être exécuté à l'aide de la nouvelle exécution de script comme suit :

```
<?xml version="1.0" encoding="UTF-8"?>
<PAGE PAGE_ID="IEGScriptLauncher">
  <JSP_SCRIPTLET>
    <![CDATA[
curam.omega3.request.RequestHandler rh =
  curam.omega3.request.RequestHandlerFactory.getRequestHandler(
    request);

String context = request.getContextPath() + "/";

String url =
  context + "ieg/Screening.do?" + "executionID=" + executionID
    + "&" + rh.getSystemParameters();

// Redirection vers la page correcte.
response.sendRedirect(response.encodeRedirectURL(url));
    ]]>
  </JSP_SCRIPTLET>
</PAGE>
```

Figure 62. Demarrage du lecteur IEG

Notez qu'il est uniquement possible de pré-renseigner le magasin de données DS, et non les questions de contrôle ou d'autres informations de script car elles sont stockées dans l'exécution de script et non dans le magasin de données DS. Cela signifie qu'il n'est pas possible de pré-renseigner les données affichées dans la première section du script et de démarrer à la deuxième section. La première section s'affiche et l'utilisateur peut confirmer les données pré-renseignées.

Chapitre 9. Utilisation du magasin de ressources

9.1 Introduction

Le magasin de ressources est une zone de la base de données d'infrastructure, utilisée pour stocker les ressources employées dans une application active. Les ressources peuvent être de tout type, mais les plus couramment utilisées par IEG sont les images et les ressources de fichier de propriétés.

9.2 Liste de toutes les ressources

Pour accéder aux écrans d'administration de ressources, vous devez vous connecter en tant qu'utilisateur admin. Une fois connecté, une section IEG est disponible dans votre panneau de navigation. Lorsque vous cliquez sur la section, un menu contenant un lien 'Ressources de demande' s'affiche. Si vous cliquez sur ce lien, une liste des ressources s'affiche avec une zone de recherche basée sur la catégorie.

Les ressources sont organisées en catégories. Pour afficher les ressources, sélectionnez une catégorie dans les critères de filtrage et sélectionnez 'rechercher'. Les catégories de ressources utilisées par IEG sont les suivantes :

- CSS

  Modèles de feuille de style qui peuvent être modifiés pour personnaliser l'apparence du lecteur IEG.
- Image

  Images utilisées dans le lecteur et les scripts IEG.
- Propriété

  Fichiers de propriétés contenant un texte spécifique à l'environnement local pour les scripts et les pages de questions.

9.3 Téléchargement d'une nouvelle ressource

Un lien permettant d'ajouter une nouvelle ressource se trouve en haut de l'écran qui répertorie toutes les ressources. Lorsque vous cliquez sur ce lien, un écran vous invitant à saisir les détails de la ressource s'affiche.

Vous devez saisir les informations suivantes :

- Name

  Il s'agit du nom unique de la ressource qui peut être utilisée dans un script IEG pour y faire référence. Selon le type de ressource, une convention de dénomination peut être appliquée pour une utilisation dans un script IEG. Les sections 9.7, «Ajout d'images», à la page 60 et 9.8, «Modification du texte statique», à la page 60 contiennent plus de détails.
- Content Type (Type de contenu)

  Lors de l'exécution d'une ressource dans un navigateur Web, un type de contenu est requis pour indiquer au navigateur comment gérer la ressource. Les types de contenu les plus courants utilisés dans un script IEG sont `image/png` pour une image PNG et `text/plain` pour un fichier de propriétés.
- Content (Contenu)

  Le sélecteur de fichier permet à l'utilisateur de sélectionner la ressource à télécharger.

Les informations suivantes sont facultatives :

- Category (Catégorie)

  Catégorie dans laquelle la nouvelle ressource doit être ajoutée.
- Content Disposition (Disposition du contenu)

S'il s'agit de ressources utilisées dans les scripts IEG, cette zone d'informations peut être laissée vide.

- Locale (Environnement local)

  Si vous souhaitez disposer d'une version spécifique de l'environnement local d'une ressource, entrez le code de l'environnement local ici. Lorsque le système recherche une ressource, il utilise un mécanisme de rétromigration similaire à Java. Par exemple, si l'environnement local en cours est `en_US`, le système tente de localiser la ressource pour l'environnement local `en_US`, puis `en` et enfin la ressource «default». La ressource «par défaut» est indiquée en laissant le champ d'environnement local vide lors du téléchargement de la ressource.

- Interne

  Indique si la ressource est destinée à un usage interne et ne doit jamais être exécutée dans le navigateur Web. Dans cette première édition d'IEG, ce paramètre peut être ignoré.

- Description

  Description de la ressource.

9.4 Suppression d'une ressource existante

Pour supprimer une ressource existante, sélectionnez le lien 'view' (vue) sur la ressource et, à partir de la 'View Resource Page' (Affiche la page de ressources), sélectionnez 'delete' (supprimer) pour supprimer cette ressource du système. Lorsque vous cliquez sur ce lien, une boîte de dialogue de confirmation vous demandant de confirmer la suppression de cette ressource du système s'affiche.

9.5 Mise à jour d'une ressource existante

Pour mettre à jour une ressource existante, sélectionnez le lien 'edit' (édition) de la page 'Application Resources' (Ressources de demande) ou de la page 'View Resource' (Afficher les ressources). Vous pouvez ensuite accéder à la ressource mise à jour sur votre système de fichiers dans le champ 'Nouveau contenu'.

9.6 Téléchargement d'une ressource existante

Pour télécharger une entrée de la liste des ressources, cliquez sur le lien 'télécharger' de la page 'Ressources de la demande'. Ce lien ouvre la boîte de dialogue de téléchargement du fichier du navigateur pour permettre à l'utilisateur de sauvegarder la ressource ou de l'ouvrir directement.

9.7 Ajout d'images

Les scripts IEG vous permettent d'indiquer des images à utiliser dans vos sections (panneau de navigation à gauche d'une page, par défaut) et vos pages (dans la zone de titre de la page) et proposent également des images intégrées au système (comme les différentes images de personnes utilisées dans les onglets, etc.). Toutes ces images doivent être stockées dans le magasin de ressources afin que les nouvelles images puissent être ajoutées et que les images existantes puissent être mises à jour sans avoir à reconstruire et redéployer votre application. Lors du téléchargement d'une ressource image, définissez «Content Type» (Type de contenu) de manière appropriée pour l'image (par exemple, image/png, image/gif etc.) et laissez la zone «Content Disposition (Disposition du contenu)» vide.

9.8 Modification du texte statique

Le moteur IEG vous permet d'entrer la totalité du texte de votre script pour l'environnement local par défaut, directement dans la définition du script. Cependant, il ne s'agit pas de l'endroit à partir duquel le texte affiché sur les écrans est lu. En fait, tout le texte référencé dans un script est stocké dans des fichiers de propriétés spécifiques à l'environnement local dans le magasin de ressources. Pour chaque script, il y aura un minimum d'un fichier de propriétés pour le script lui-même et un fichier de propriétés pour chaque page du script. Pour garantir l'unicité de ces fichiers, la convention de dénomination suivante est employée (la dernière partie ne s'appliquant évidemment qu'aux fichiers de propriétés spécifiques à la page) :

```
IDScript_VersionScript_TypeScript_IDPage
```

Lorsque vous utilisez les écrans d'administration IEG pour télécharger un nouveau script dans le système, tout le texte statique qui s'y trouve (par exemple, libellés, titres, descriptions, etc.) est automatiquement extrait dans les fichiers de propriétés correctement nommés de votre script et stockés dans le magasin de ressources sans environnement local associé (afin qu'il y agit comme propriété de secours si aucune propriété n'existe pour l'environnement local exécuté). Tout ce texte peut alors être modifié en téléchargeant simplement le fichier de propriétés en cours et en gardant à l'esprit la convention de dénomination décrite ci-dessus pour localiser la ressource dans la liste des ressources. Apportez ensuite les modifications nécessaires et mettez à jour les ressources, comme décrit dans la section 9.5, «Mise à jour d'une ressource existante», à la page 60. Il n'est pas obligatoire d'apporter de modifications au script lui-même.

De même, les versions de ces fichiers pour d'autres environnements locaux peuvent être facilement ajoutées et seront sélectionnées au lieu des propriétés de l'environnement local par défaut lors de la prochaine exécution du script dans cet environnement local.Lors du téléchargement d'une ressource de fichier de propriétés, définissez «Content Type» (Type de contenu) sur `text/plain` et laissez la zone «Content Disposition» (Disposition du contenu) vide.

9.9 Modification du codage du fichier par défaut

Lors du téléchargement d'une ressource en texte clair, le fichier doit posséder un codage UTF-8. Si vous souhaitez utiliser un codage différent lors du téléchargement du fichier, la zone "Content Type" (Type de contenu) peut être utilisée pour l'indiquer via le paramètre facultatif `charset`. Par exemple :

```
text/plain; charset=ISO-8859-1
```

Chapitre 10. Utilisation d'IBM Rational AppScan dans le cadre de l'analyse d'IEG

10.1 Introduction

Ce chapitre décrit les étapes requises pour effectuer des analyses de sécurité des applications de style IEG à l'aide de l'outil IBM®Rational AppScan.

10.2 Préparation

IEG coordonne la communication entre le lecteur et le moteur à l'aide d'un jeton de synchronisation. Le jeton de synchronisation est utilisé pour s'assurer que la page soumise par le navigateur est cohérente avec la page via laquelle le moteur IEG est censé être soumis. Cela facilite la détection lorsque l'utilisateur a recours aux boutons du navigateur au lieu des boutons du lecteur. Le jeton de synchronisation change pour chaque page de questions affichée dans le lecteur IEG. Il est par conséquent très difficile d'analyser les scripts de questions IEG exécutés dans le lecteur IEG.

Pour cette raison, avant l'analyse, il est recommandé de définir sur true la propriété de configuration de script appscan.mode.enabled. Lorsque cette propriété est définie sur true, le moteur ne vérifie pas la valeur du jeton de synchronisation transmise par le lecteur. La désactivation de la vérification du jeton de synchronisation est acceptable lors de l'exécution d'une analyse, mais la vérification du jeton de synchronisation doit toujours être activée dans un environnement de production.

En outre, pour réduire le volume d'informations superflues signalées dans une analyse, la trace de pile doit être désactivée. Pour désactiver la trace de pile :

- Accédez au dossier webclient\JavaSource\curam\omega3\
- Renommez Initial_ApplicationConfiguration.properties en ApplicationConfiguration.properties
- Ouvrez ApplicationConfiguration.properties
- Ajoutez l'entrée : `errorpage.stacktrace.output=false`

10.3 Pages Relation

Les pages Relation sont une fonction spéciale d'IEG qui facilitent la collecte des informations relatives aux relations entre les personnes d'un foyer. Contrairement aux autres pages d'un script IEG, les pages Relation ont une apparence plus dynamique et contiennent un nombre variable de champs. Actuellement, les noms des champs qui peuvent être générés pour les pages Relation varient selon les exécutions. Cela signifie qu'il est actuellement impossible d'exécuter une analyse sur un script de question IEG qui contient une page Relation.

10.4 Configuration des examens

Une fois AppScan lancé, vous pouvez créer une nouvelle analyse en sélectionnant l'option 'Créer une nouvelle analyse...' dans l'écran d'accueil.

Sélectionnez ensuite 'Regular Scan' (Analyse standard) dans la zone Predefined Templates (Modèles prédéfinis) de l'écran suivant.

Sélectionnez 'Web Application Scan' (Analyse d'application Web) sur la première page de l'assistant de configuration, puis cliquez sur 'Suivant'.

Dans la page 'URL et serveurs' de l'assistant, saisissez l'URL de départ de l'application. Une fois l'URL saisie, vous pouvez la vérifier en cliquant sur l'icône en regard du champ d'entrée. Le navigateur AppScan s'affiche et tente d'ouvrir l'URL. Confirmez que l'URL est correcte et accessible (cliquez sur Oui si un avertissement de sécurité s'affiche). Fermez le navigateur et cliquez sur 'Suivant' dans l'assistant de configuration.

Entrez les détails de gestion de la connexion nécessaires. Les applications qui s'exécutent sous Eclipse/Tomcat ne nécessitent pas à l'utilisateur de se connecter. L'option 'Aucun' peut donc être sélectionnée. Cliquez sur 'Suivant'.

Sur l'écran 'Test Policy' (Stratégie de test), cliquez sur le lien 'Full Scan Configuration' (Configuration d'analyse complète) dans le panneau 'General Tasks' (Tâches générales). La boîte de dialogue 'Configuration d'analyse s'affiche'.

10.5 Stratégie de test

Vérifiez que l'option 'Test Policy' (Stratégie de test) est sélectionnée dans le panneau d'affichage de la sélection à gauche de la boîte de dialogue de configuration. L'approche la plus simple lors de la configuration d'une analyse consiste à activer tous les tests, puis à désactiver les tests de faible valeur qui ralentissent l'exécution de l'analyse.

Sélectionnez 'Enabled/Disabled' (Activé/Désactivé) dans la liste déroulante 'sort tests by' (trier les tests par). Commencez par cocher la case 'Partially Enabled' (Partiellement activé), puis la case 'Disabled' (Désactivé). La seule entrée affichée doit être 'Enabled' (Activé). Sélectionnez 'Gravité' dans la liste déroulante. Décochez les cases 'Faible' et 'Information'. Pour les besoins de l'analyse IEG, il n'est pas nécessaire d'exécuter les tests invasifs car ces tests sont surtout destinés à tester la plateforme. Sélectionnez 'Invasif' dans la liste déroulante. Décochez la case 'Invasif'.

10.6 Options d'exploration

Sélectionnez 'Options d'exploration' dans le panneau d'affichage de la sélection. Définissez 'Limite de chemin redondante' sur 1. Sélectionnez 'Largeur en premier' comme 'Méthode d'exploration'.

10.7 Communications et proxy

Sélectionnez 'Communications et proxy' dans le panneau d'affichage de la sélection. Définissez 'Nombre d'unités d'exécution' sur 1.

10.8 Options de test

Sélectionnez 'Test Options' (Options de test) dans le panneau d'affichage de la sélection. Décochez la case 'Use Adaptive Testing based on application behavior' (Utiliser le test adaptatif en fonction du comportement de l'application).

10.9 Opérations en plusieurs étapes

IEG nécessite d'utiliser des données correctement formatées dans certains paramètres. En tant que tel, AppScan doit être habitué à utiliser l'application en cours de test. Sélectionnez 'Multi-Step Operations' (Opérations en plusieurs étapes) dans le panneau d'affichage de la sélection. Cliquez sur le bouton Enregistrer. Le navigateur AppScan s'affiche alors et tente d'ouvrir l'URL indiquée dans la page 'URL et serveurs' de l'assistant de configuration. Vous devez ensuite parcourir l'application comme requis, en entrant les données pertinentes. AppScan enregistre les valeurs saisies et utilise ces valeurs dans chaque test exécuté ultérieurement. Une fois terminé, fermez simplement le navigateur. La boîte de dialogue Configuration d'analyse est mise à jour avec la séquence qui vient d'être enregistrée. Cochez la case 'Enable playback of this sequence' (Activer la diffusion de cette séquence) et décochez la case 'Autorisation l'optimisation de la lecture'.

Prenez note de toutes les étapes de séquence qui contiennent `Screening.do`. Vous devez définir ces étapes de séquence en expressions régulières et les ajouter en tant que chemins d'exception aux options de chemin d'exclusion AppScan. AppScan peut être facilement placé hors synchronisation lorsqu'il s'agit d'opérations enregistrées. Vous devez donc vous assurer qu'AppScan va ignorer le chemin incorrect et conserver le script enregistré lors de l'exécution de ses tests. Pour ce faire, demandez à AppScan d'ignorer toutes les étapes de séquence contenant `screening.do`, à l'exception de celles que vous spécifiez dans des expressions régulières. Prenez note de chaque valeur `__u=x` trouvée dans la liste des étapes de séquence.

10.10 Exclusion de chemins et de fichiers

Sélectionnez 'Exclure les chemins et les fichiers' dans le panneau d'affichage de la sélection. Cliquez sur le bouton pour ajouter un chemin d'exclusion. Sélectionnez 'Exclure' dans le champ 'Type' et sélectionnez 'Expression régulière' dans la liste déroulante 'Match' (Correspondance). Saisissez `.*/Curam/ieg/Screening.do.*` dans le champ 'Chemin', puis cliquez sur 'OK'.

Ajoutez un autre chemin d'exclusion. Sélectionnez 'Exception' dans le champ 'Type' et sélectionnez 'Expression régulière' dans la liste déroulante Correspondance. Saisissez `.*/Curam/ieg/Screening.do?executionID=.\d*` dans le champ 'Chemin', puis cliquez sur 'OK'.

Une exception doit également être ajoutée pour chaque valeur `__u=x` dans la liste des étapes de la séquence. Encore une fois, sélectionnez 'Expression régulière' dans le menu déroulant 'Match' (Correspondance), puis entrez une expression au format suivant dans le champ 'Path':
`.*/Curam/ieg/Screening.do?executionID=.*&__u=[valeur affichée dans l'écran récapitulatif]`

Cliquez sur 'OK'.

Cliquez sur 'OK' pour retourner à l'assistant de configuration.

Cliquez sur 'Suivant' dans l'assistant de configuration.

10.11 Fin

A ce stade, la configuration de l'analyse est terminée. Sélectionnez l'option 'I will start my scan later' (Démarrer mon analyse ultérieurement) pour enregistrer l'analyse configurée au lieu d'autoriser AppScan à analyse de façon aléatoire l'application entière. Cliquez sur 'Finish' (Terminer).

10.12 Exécution de l'analyse

Pour démarrer l'analyse, sélectionnez l'élément de menu 'Scan' (Analyser) dans la fenêtre principale AppScan, puis sélectionnez 'Test Multi-Step Operations Only' (Tester uniquement les opérations en plusieurs étapes). Selon l'application à tester, l'exécution d'une analyse peut prendre plusieurs jours. Une fois l'analyse terminée, AppScan affiche les résultats de l'analyse dans un écran récapitulatif. Ces résultats doivent ensuite être examinés pour déterminer les problèmes signalés qui sont des vulnérabilités réelles et ceux qui sont de faux positifs.

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 :

Licence sur la propriété intellectuelle

Mentions légales et droit de propriété intellectuelle.

IBM Japon Ltd.

19-21, Nihonbashi-Hakozakicho, Chuo-ku

Tokyo 103-8510, Japon

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales. INTERNATIONAL BUSINESS MACHINES CORPORATION FOURNIT CETTE PUBLICATION "EN L'ETAT" SANS GARANTIE D'AUCUNE SORTE, EXPLICITE OU IMPLICITE, Y COMPRIS NOTAMMENT, LES GARANTIES IMPLICITES DE NON-CONTREFACON, DE QUALITE MARCHANDE OU D'ADEQUATION A UN USAGE PARTICULIER. 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 logiciel sous licence décrit dans ce document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM, conformément aux dispositions du Livret contractuel, des Conditions Internationales d'Utilisation de Logiciels IBM ou de tout autre accord 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 :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes 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 programmes 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 intégrale ou partielle de ces exemples de programmes et des oeuvres qui en sont dérivées doit inclure une mention de droits d'auteur libellée comme suit :

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

© Copyright IBM Corp. _entrez l'année ou les années_. All rights reserved.

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

Marques

IBM, le logo IBM et ibm.com sont des marques ou des marques déposées d'International Business Machines Corp. dans de nombreux pays. Les autres noms de produit et de service peuvent être des marques d'IBM ou d'autres sociétés. Une liste des marques commerciales actuelles d'IBM est disponible sur Internet sous "Droits d'auteur et marques" à l'adresse http://www.ibm.com/legal/us/en/copytrade.shtml.

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

D'autres noms peuvent être des marques de leurs propriétaires respectifs. Les autres noms de sociétés, de produits et de services peuvent appartenir à des tiers.

Imprimé en France