IBM Cúram Social Program Management
Version 6.0.5

Engine für Datenzuordnung verwenden

Hinweis

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die Informationen in „Bemerkungen" auf Seite 35 gelesen werden.

Überarbeitung: März 2014

Diese Ausgabe bezieht sich auf IBM Cúram Social Program Management v6.0.5 und alle nachfolgenden Releases, sofern nicht anderweitig in neuen Ausgaben angegeben.

Licensed Materials - Property of IBM.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellen

Mit der Data Mapping Engine entwickeln

Unter Datenzuordnung wird hier der Prozess verstanden, bei dem Daten aus dem Cúram-Datenspeicher (Cúram Datastore, CDS) den Cúram-Angabenentitäten zugeordnet werden. Datenzuordnungen werden von der Cúram-Engine für die Datenzuordnung anhand von Zuordnungen vorgenommen. Zuordnungen werden für ein Programm oder einen Anliegenantrag erstellt.

Einführung

Zweck

In diesem Handbuch wird beschrieben, wie mit der Cúram-Engine für Datenzuordnung (Cúram Data Mapping Engine, CDME) Daten von Bürgern, die während der Anliegen- und Screeningverarbeitung erfasst wurden, zugeordnet werden:

- Zu Angabenentitäten und Nicht-Angabenentitäten in Cúram-Fällen
- Zu ausgefüllten Antragsformularen im PDF-Format

Beide Optionen optimieren den Antrag eines Bürgers auf ein Programm für eine bestimmte Leistung. Die Zuordnung der Daten zu den Fallangaben ermöglicht die Verwendung dieser Daten für die Feststellung der Anspruchsberechtigung im Rahmen der Cúram-Fallverarbeitung. Die Zuordnung der Daten zu PDF-Antragsformularen beschleunigt den Prozess zum Ausfüllen dieser Formulare bei der manuellen Übermittlung.

Dieses Handbuch sollte in Verbindung mit dem Handbuch zu IBM Cúram Data Mapping Editor verwendet werden. Der Datenzuordnungseditor (Data Mapping Editor) ist das einfachste Tool, um schnell Zuordnungen zu Angaben erstellen. Vom Datenzuordnungseditor werden diese Zuordnungen in einer XML-Zuordnungssprache gespeichert. In diesem Handbuch werden die Details der XML-Zuordnungssprache beschrieben, die wichtig sind, um die Ausführung der vorhandenen Zuordnungen zu verstehen, um anspruchsvollere Zuordnungen zu verstehen, um Zuordnungen zu PDF-Formularen erstellen zu können und zum ältere Zuordnungen verwalten zu können, die unter Umständen nicht mit den Datenzuordnungseditor kompatibel sind.

Zielgruppe

Datenzuordnung ist ein Kooperationsprojekt zwischen Geschäftsanalysten und Entwicklern. Die Aufgabe der Geschäftsanalysten besteht darin, die Daten der Bürger entweder den Feldern in einem PDF-Antragsformular oder den Angabenentitäten logisch zuzuordnen. Die Entwickler haben die Aufgabe, die Arbeit der Geschäftsanalysten zu übersetzen und anhand dieser Informationen Zuordnungsspezifikationen und Zuordnungskonfigurationen in XML zu schreiben.

Voraussetzungen

In der folgenden Tabelle werden die Voraussetzungen für die Engine für Datenzuordnung aufgelistet und die Quellen angegeben, die zur Erlangung der Kenntnisse über diese Voraussetzungen empfohlen werden:

Tabelle 1. Voraussetzungen für die Cúram-Engine zur Datenzuordnung

Voraussetzung	Empfohlene Quelle	Zielgruppe
XML	http://www.w3.org/XML/	Entwickler
Cúram-Fälle	Beschreibungen von Fallentitäten im Kernreferenzmodell; `Cúram-Handbuch zur Verwaltung integrierter Fälle`	Geschäftsanalysten
	`Cúram-Handbuch zu den Grundlagen der Anspruchsberechtigung und Leistungshöhe mit Cúram Express Rules (CER)`	Entwickler
Cúram-Beteiligte	Beschreibungen von Beteiligtenentitäten im Kernreferenzmodell; `Cúram-Handbuch zu Beteiligten`	Geschäftsanalysten
Common Datastore (CDS)	`Datenspeicherschemas erstellen`	Entwickler
IBM Cúram Universal Access	`Universal Access Customization Guide`	Entwickler
Cúram Evidence	Beschreibungen von Angabenentitäten im Kernreferenzmodell; `Handbuch zu Cúram Evidence`	Geschäftsanalysten
	`Lösungen für Angaben in Cúram entwickeln`	Entwickler
PDF-Formulare	http://www.adobe.com/products/acrobat/?promoid=BPDDU	Entwickler

Kapitel in diesem Handbuch

Dieses Handbuch umfasst die folgenden Kapitel:

Kapitel 2 - Grundlagen der Datenzuordnung

In diesem Kapitel wird beschrieben, wie die Daten der Bürger (die in Universal Access Portal erfasst und im Cúram-Datenspeicher gespeichert sind) von der Cúram-Engine für Datenzuordnung in Felder in PDF-Antragsformulare oder Angabenentitäten in Fällen umgesetzt werden.

Kapitel 3 - Zuordnungsspezifikationen und -konfigurationen für statische Angaben und PDF-Dateien schreiben

In diesem Kapitel wird beschrieben, wie Zuordnungsspezifikationen und Zuordnungskonfigurationen in XML für statische Angaben und PDF-Dateien geschrieben werden. Von Zuordnungsspezifikationen werden Daten aus einem Formular einem anderen zugeordnet, zum Beispiel aus Entitäten in der CDS-Struktur zu Entitäten in der Datenbank. Von Zuordnungskonfigurationen wird beschrieben, wie PDF-Anträge mit Daten gefüllt werden oder wie Daten in Angabenentitäten konvertiert werden.

Kapitel 4 - Zuordnungsspezifikationen und -konfigurationen für dynamische Angaben schreiben
In diesem Kapitel wird beschrieben, wie Zuordnungsspezifikationen und Zuordnungskonfigurationen in XML für dynamische Angaben geschrieben werden.

Kapitel 5 - Zuordnungen zu Dritten
In diesem Kapitel wird beschrieben, wie die Zuordnung von Dritten als Fallbeteiligte in Fällen durchgeführt wird.

Anhang A - Schema für Zuordnungsspezifikationen
In diesem Anhang wird die Grammatik beschrieben, die beim Schreiben von Zuordnungsspezifikationen in XML eingehalten werden muss.

Anhang B - Schema für Zuordnungskonfigurationen
In diesem Anhang wird die Grammatik beschrieben, die beim Schreiben von Zuordnungskonfigurationen in XML eingehalten werden muss.

Anhang C - Konformität
In diesem Anhang wird beschrieben, wie die Konformität während der Entwicklung berücksichtigt wird.

Grundlagen der Datenzuordnung

Einführung

Die Cúram-Engine für Datenzuordnung (Cúram Data Mapping Engine, CDME) und Cúram Universal Access Portal sollen Bürger dabei unterstützen, zu überprüfen, ob sie Anspruch auf Leistungen haben. Während dieser Überprüfung (des sog. Screenings) und der Anliegenerfassung übergibt der Bürger seine Informationen. Mithilfe von CDME werden die Informationen des Bürgers entweder in ein PDF-Antragsformular (mit den Detailangaben des Bürgers) oder in einen Cúram-Fall mit neuen Angabenentitäten konvertiert.

Die Verarbeitung durch CDME wird nachfolgend beschrieben. Wenn der Bürger seine Informationen übergibt, werden diese Informationen im Cúram-Datenspeicher (CDS) gespeichert. Die Daten werden von CDME gelesen und unter Verwendung der Regeln einer Zuordnungsspezifikation in ein Format umgewandelt, das vom Antragsbuilder gelesen werden kann. Vom PDF-Antragsbuilder wird mithilfe einer Zuordnungskonfiguration ermittelt, wie die Daten des Bürgers in einem PDF-Antragsformular angezeigt werden. Von einem Angabenantragsbuilder wird mithilfe einer Zuordnungskonfiguration eine Angaben-API aufgerufen, um die neuen Angabenentitäten für den neuen Fall zu erstellen.

Vorgehensweise zum Speichern von Daten im CDS

Wenn die Daten eines Bürgers einer Fallentität oder PDF-Antragsformularen zugeordnet werden, sollte in einem ersten Schritt die gewünschte Ausgabe für die Daten des Bürgers überprüft werden, also die Fallangaben oder das ausgefüllte Antragsformular. Im nächsten Schritt sollte geprüft werden, wie die Daten des Bürgers im Verlauf der Anliegenaufnahme und des Screeningprozesses im CDS gespeichert werden. Sobald Sie die Informationen, die in den Fallangaben oder PDF-Antragsformularen erforderlich sind und die Informationen, die im CDS gespeichert sind, kennen, können Sie logische Zuordnungen zwischen den Formularen und den Daten erstellen.

Das Screening und die Anliegenaufnahme sollen die Bürger dabei unterstützen, Leistungen zu beantragen. Als Mindestvoraussetzung für ein Antragsformular oder

die Fallangaben sind die Namen der Mitglieder des Haushalts und ihre Beiziehungen zu der Person erforderlich, die die Leistungen beantragt. Diese Informationen zum Bürger werden mithilfe von IEG-Scripts erfasst und im CDS gemäß einem vordefinierten Schema gespeichert. In „Vorgehensweise zum Speichern von Daten im CDS" auf Seite 3 wird ein Beispiel für eine Datenstruktur im CDS für einen einzelnen Leistungsantrag dargestellt, in dem die Mitglieder eines Haushalts und ihre Beziehungen angegeben werden.

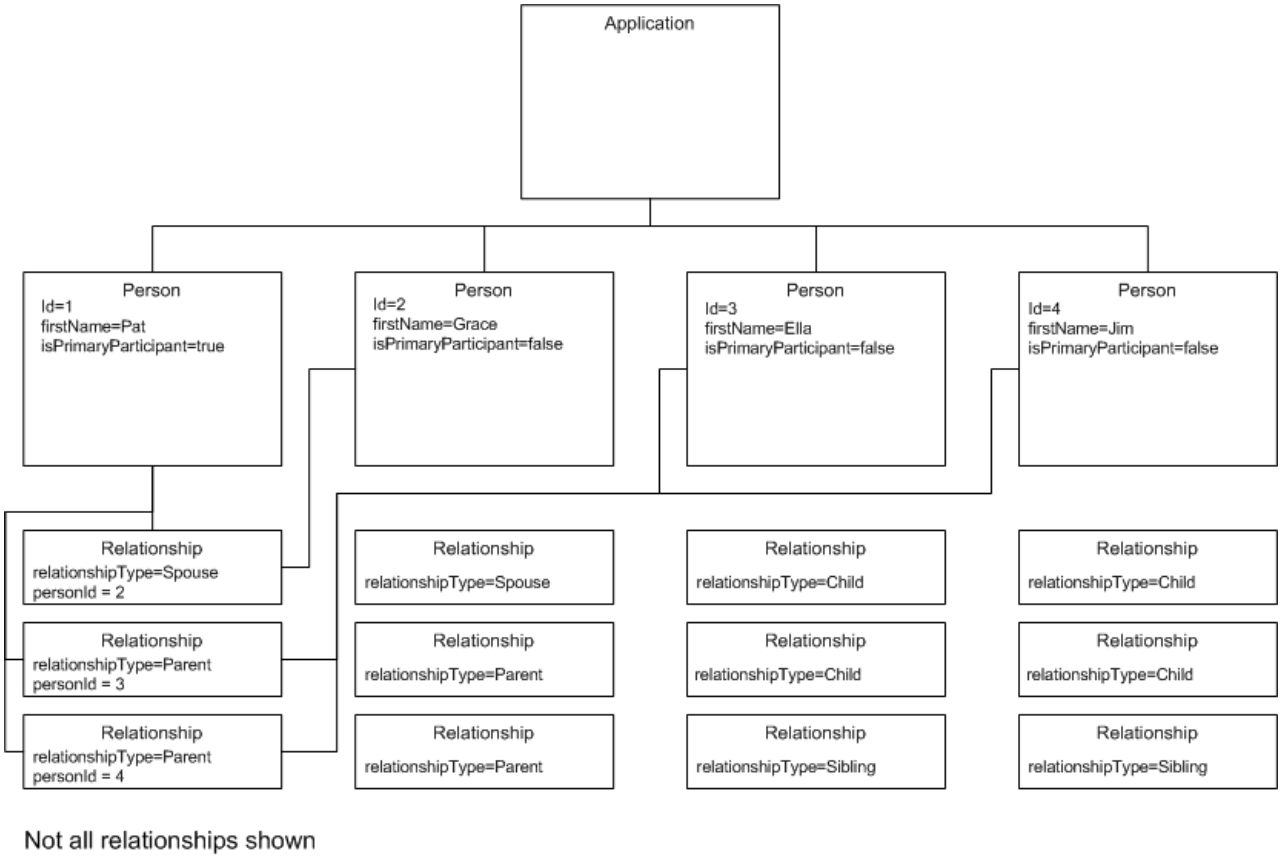

Abbildung 1. Beispiel für die Datenstruktur im Cúram-Datenspeicher

Logische Zuordnungen erstellen

Von einer logischen Zuordnung werden Informationen zu einer im CDS gespeicherten Person auf die Angabenentitäten eines Falls verteilt: Haushaltsmitglied, Wohnsituation und Behinderung.

Im Rahmen der Erstellung einer logischen Zuordnung müssen Geschäftsregeln beachtet werden, die sich auf die Vorgehensweise der Datenzuordnung durch CDME auswirken. Wenn ein Bürger zum Beispiel angibt, dass er blind und behindert ist, legen die Geschäftsregeln fest, dass zwei Angabendatensätze für eine Behinderung für den Bürger erstellt werden müssen (einer für die Blindheit und einer für die Behinderung).

Validierungsprobleme bei Datenzuordnungen berücksichtigen

Im Rahmen eines Anliegen- und Screeningprozesses ist es erforderlich, auf das Gleichgewicht beim Validieren der Angaben zu achten, sodass sie vernünftig eingegeben werden können und gleichzeitig sichergestellt wird, dass dem Bürger keine unnötigen Fragen gestellt werden. Ein Ansatz zur Berücksichtigung der Validie-

rungsprobleme bei Datenzuordnungen ist die Vorgehensweise, während der Anliegenaufnahme minimale Validierungen durchzuführen und mit Standardwerten oder temporären Werten zu arbeiten, die später aktualisiert werden können.

Zuordnungsspezifikationen und -konfigurationen für statische Angaben und PDF-Dateien schreiben

Einführung

Entwickler können die logische Datenzuordnung als Spezifikation der Anforderungen zum Schreiben von Zuordnungsspezifikationen und Zuordnungskonfigurationen verwenden. Von einer Zuordnungsspezifikation wird beschrieben, wie Daten, die in einer bestimmten Struktur gespeichert sind, einer anderen zugeordnet werden. Von jeder Zuordnungsspezifikation wird auf die Quelle, aus der die Daten stammen, und auf das Ziel, dem die Daten zugeordnet werden, verwiesen.

Die Zuordnungsspezifikation enthält zwar die Regeln, die zum Transformieren der Daten aus einem Format in ein anderes erforderlich sind, es sind jedoch noch weitere Informationen erforderlich, um die transformierten Daten in Fallangaben oder vollständige PDF-Antragsformulare umzuwandeln. Diese zusätzlichen Informationen werden in der Zuordnungskonfiguration bereitgestellt. In diesem Kapitel wird anhand von Beispielen veranschaulicht, wie Zuordnungsspezifikationen und Zuordnungskonfigurationen geschrieben werden.

Zuordnungsspezifikationen schreiben

Zuordnungsspezifikationen werden von CDME zum Transformieren von Daten im Cúram Datastore (CDS) in ein anderes Format verwendet. Alle Daten, die in einem Anliegen- oder Screeningantrag eines Bürgers enthalten sind, können durch Lesen der Antragsentität, der in ihren untergeordneten Elementen enthaltenen Daten und der Daten aller weiteren untergeordneten Elemente abgerufen werden. Von der Engine für Datenzuordnung werden die Daten im CDS traversiert und die Regeln angewendet, die in der XML ausgedrückt werden, um die Transformation der Daten auszuführen.

Einfache Zuordnungsspezifikation

Durch die folgende einfache Zuordnungsspezifikation wird eine Personenentität im CDS einer Angabenentität eines Haushaltsmitglieds zugeordnet:

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

In Zeile 4 wird die Quelle der Zuordnung angegeben, in Zeile 5 das Ziel. Diese Regel lässt sich wie folgt umschreiben: "Erstellen Sie für jede im CDS befindliche Entität 'Person' die entsprechende Entität 'HouseholdMember'". Das Element <target-entity> enthält zwei Elemente des Typs <map-attribute> in den Zeilen 6 und 7.

Vom Element <map-attribute> in Zeile 6 wird festgelegt, dass das Attribut 'isNativeAmerican' in der Entität 'Person' dem Attribut 'natAlaskOrAmerInd' in der Entität 'HouseholdMember' zugeordnet wird. Attribute werden nicht zugeordnet, wenn kein spezifisches Element <map-attribute> vorhanden ist. Aus diesem Grund wird in Zeile 6 festgelegt, dass das Kommentarattribut in 'Person' dem Kommentarattribut in 'HouseholdMember' zugeordnet wird.

In manchen Fällen ist es erforderlich, anzugeben, dass eine Zuordnung nur unter bestimmten Umständen durchgeführt wird. So kann zum Beispiel die Entität 'HeadOfHousehold' nur im Zielsystem erstellt werden, wenn von der Zuordnung eine Entität 'Person' im CDS ermittelt werden kann, für die der Indikator 'isPrimaryParticipant' auf 'true' gesetzt ist. Das obige Beispiel kann wie nachfolgend dargestellt um diese Regel erweitert werden:

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

Bedingungsausdrücke zuordnen

Im folgenden Beispiel werden die Werte 'Yes', 'No' und 'Unanswered' als Codetabellenwerte dargestellt und dazu verwendet, aufzuzeichnen, ob die Person ein US-Staatsbürger ist. Der Wert ITYN4001 besagt, dass der Kunde auf diese Frage mit 'Yes' geantwortet hat. Beachten Sie hierbei, dass " verwendet wird, weil doppelte Anführungszeichen ("") nicht direkt in der XML verwendet werden können. Die Syntax für bedingte Zuordnungsattribute ist mit dieser identisch.

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

Codetabellenwerte zuordnen

In manchen Zuordnungen können die im CDS aufgezeichneten Codetabellenwerte direkt in die Codetabellenwerte umgesetzt werden, die im Zielmodell verwendet werden. In solchen Fällen kann ein Abschnitt am Anfang des Zuordnungsscripts zum Angeben der Codetabellenzuordnungen verwendet werden. Beispiel:

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

In Beispiel 5 werden die Werte aus der Codetabelle CITIZENSTATUS den Werten in der Codetabelle 'AlienStatus' zugeordnet.

Zuordnung zu mehreren Zielentitäten

Manchmal ist es erforderlich, eine Gruppe aus Zielentitäten zusammen zu erstellen. In der Regel wird so vorgegangen, wenn eine Gruppe aus Angabenentitäten erstellt wird, in der eine dieser Angabenentitäten übergeordnet ist und die anderen untergeordnete Elemente dieser übergeordneten Angabenentität sind. Am folgenden Beispiel wird erläutert, wie Gruppen aus zusammengehörigen Zielentitäten erstellt werden.

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

In diesem Beispiel werden zwei Entitäten erstellt. Die Entität 'BusinessAsset' ist die übergeordnete Angabe, die Entität 'Ownership' ist das untergeordnete Element. Durch eine solche Modellierung wird sichergestellt, dass die korrekten Muster für unter- und übergeordnete Angabenentitäten beachtet werden, wenn die Angaben vom Angabenantragsbuilder erstellt werden.

Eine übergeordnete Entität mehreren untergeordneten Entitäten zuordnen

In „Zuordnung zu mehreren Zielentitäten" wurde eine Gruppe aus Zielentitäten erstellt, in der unter- und übergeordnete Entitäten miteinander verknüpft wurden. In manchen Fällen ist es erforderlich, nur eine übergeordnete Entität für den gesamten Fall zu erstellen. Alle nachfolgenden untergeordneten Entitäten sind mit derselben übergeordneten Entität verknüpft. Ein Beispiel hierfür wird nachfolgend dargestellt.

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

In diesem Beispiel werden bei der ersten Ausführung der Regel die Entitäten 'HholdMealsGroup' und 'MealGroupMember' erstellt. Bei jeder nachfolgenden Ausführung der Regel wird nur eine Entität mit der Bezeichnung 'MealGroupMember' erstellt und derselben Entität wird 'HholdMealsGroup' zugeordnet.

Muster abgleichen und Zuordnungen im CDS verfolgen

Im folgenden Szenario fordern die Kunden, dass die Zuordnungsengine zum Füllen eines PDF-Formulars verwendet wird, das dem folgenden ähnelt:

Tabelle 2. PDF-Beispielformular

Name	**Arbeitgebername**	**Startdatum**	**Jahreseinkommen vor Steuern**
Pat	The Gingerman Bakery	2.1.2004	30.000
Grace	Jarmin Pharmaceutical	3.1.2002	50.000

Jedes Feld in diesem PDF-Formular verfügt über eine eindeutige Identität. Das Feld, in dem der Name 'Pat' enthalten ist, wird zum Beispiel als 'Job0.Name' bezeichnet. Das Feld, in dem die Zahl '30.000' enthalten ist, trägt die Bezeichnung 'Job0.Salary'.

Hierbei sollte bedacht werden, wie die Informationen aus dem Anliegen im CDS gespeichert werden können:

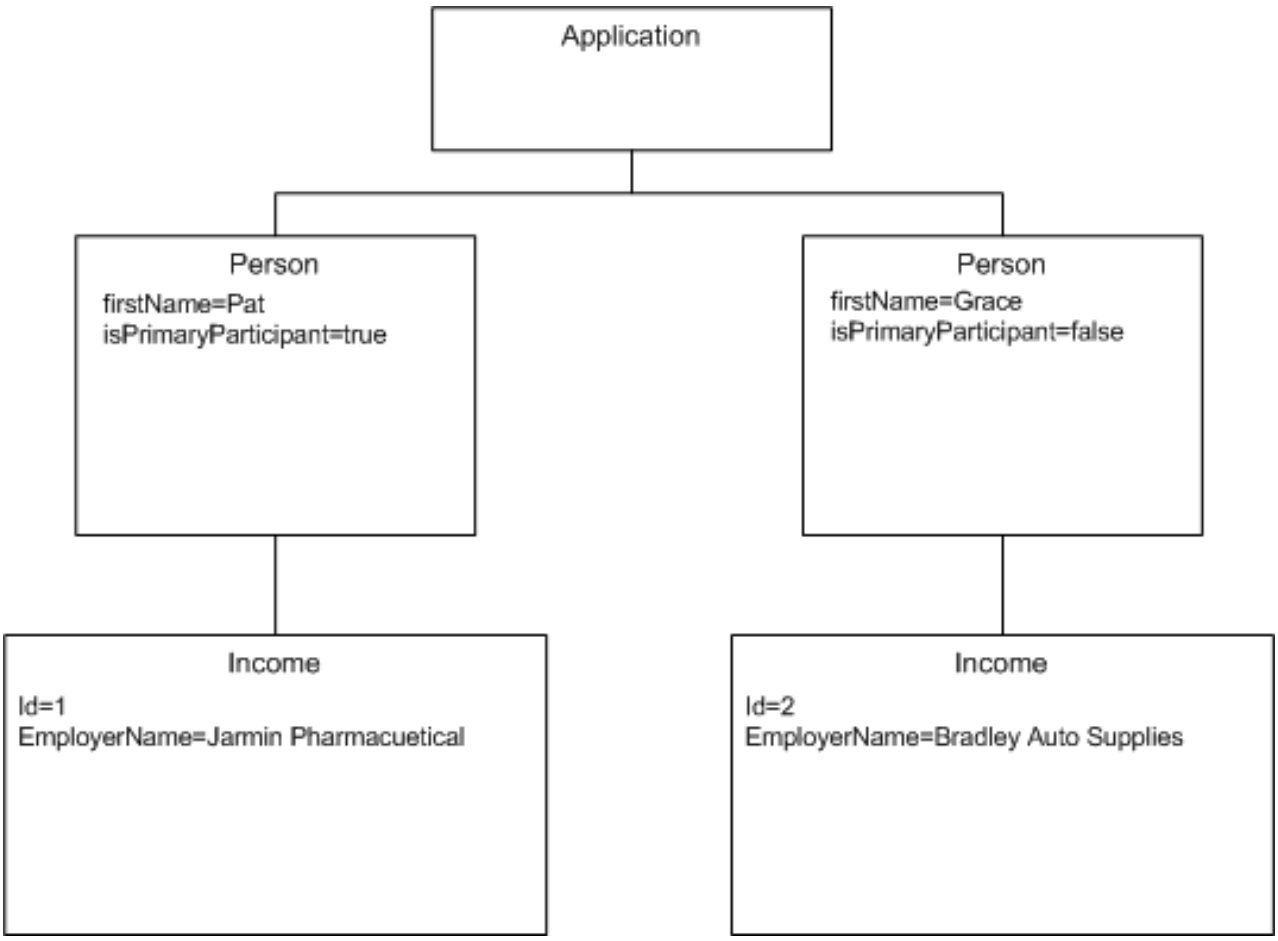

Abbildung 2. Arbeitseinkommen im CDS

Damit das Feld 'Name' im obigen PDF-Formular ausgefüllt werden kann, muss in der Zuordnungsspezifikation eine Regel enthalten sein, von der bestimmt wird, dass für jedes Einkommen, das zu einer Person gehört, der Vorname der Person in das Feld 'Name' ausgegeben werden muss. In der Zuordnungssprache kann dies wie folgt ausgedrückt werden:

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

Diese Zuordnungsregel lässt sich wie folgt umschreiben: "Erstellen Sie für jede Entität mit der Bezeichnung 'Income' innerhalb der Entität 'Person' die Entität 'Target' des Typs 'Job'. Das Attribut 'Name' der Entität 'Job' wird vom Attribut 'firstName' der Entität 'Person' zugeordnet, in dem die Entität 'Income' enthalten ist, die zugeordnet wird".

Beachten Sie die Verwendung der Syntax 'entity="Person"' in Zeile 4, die angibt, dass das Attribut 'firstName' von der Entität 'Person' und nicht von der Entität 'In-

come' stammt. In einem komplexeren Beispiel dieser Art der Zuordnungsspezifikation müssen auch noch Zuordnungen oder Verknüpfungen von einer Entität zu einer befolgt werden.

Mitglieder eines Haushalts zuordnen

In der folgenden Tabelle wird angegeben, wie Beziehungen in der Regel in einem Antragsformular ausgedrückt werden. Voraussetzung ist die Zuordnung der CDS-Entitäten zu einem vorab gefüllten Antragsformular, das dem nachfolgend dargestellten ähnelt. Schwierig ist in diesem Zusammenhang das Ausfüllen des Felds 'In welcher Beziehung stehen Sie zu dieser Person?'. Dieses Feld trägt im folgenden Beispiel die Bezeichnung 'RelType'.

Tabelle 3. Mitglieder des Haushalts

Name	In welcher Beziehung stehen Sie zu dieser Person?	Geburtsdatum	Sozialversicherungsnummer
Grace	Ehepartner	2.1.1981	209-57-9943
Ella	Kind	3.1.2002	987-23-1190

In diesem Beispiel wird die erforderliche Zuordnung wie folgt geschrieben:

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

Dies lässt sich wie folgt umschreiben: "Folgen Sie für jede im primären Beteiligten enthaltene Beziehung der Zuordnung zu der Person, auf die von dieser Beziehung verwiesen wird. Ordnen Sie das Attribute 'firstName' der Entität 'Person' dem Feld 'Name' zu. Ordnen Sie das Attribut 'relationshipType' der Entität 'Relationship' dem Feld 'RelType' zu." Der Schlüssel zum Verständnis dieses Beispiels befindet sich in Zeile 7, in der das Feld 'RelType' von einem Attribut in der Entität 'Relationship' zugeordnet wird.

Zuordnungskonfigurationen schreiben

In diesem Abschnitt werden Beispiele für Zuordnungskonfigurationen sowohl für den Angabenantragsbuilder als auch für den PDF-Antragsbuilder bereitgestellt.

Für den Angabenantragsbuilder

Wenn die Verwendung des Angabenantragsbuilders konfiguriert ist, unterteilt sich die Arbeit der Zuordnungsengine in zwei Phasen. In der ersten Phase werden von der Zuordnungsengine ein integrierter Fall mit Fallmitgliedern und die Beziehungen zwischen den Betroffenenrollen und den Fallmitgliedern erstellt. Die Fallmitglieder werden durch das Suchen nach Entitäten im Datenspeicher 'Person' gefüllt. Die Engine für Datenzuordnung ist so konzipiert, dass alle allgemeinen Datenspeicherentitäten mit der Bezeichnung 'Person' in einem Cúram-Fall als Person oder Antragsteller interpretiert werden.

Wenn „Für den Angabenantragsbuilder" auf Seite 10 als Beispiel verwendet werden würde, würde in Phase 1 der Zuordnung vom Angabenantragsbuilder ein integrierter Fall mit den folgenden vier Fallmitgliedern erstellt:

- Pat wird als der primäre Beteiligte angegeben.
- Zwei Beziehungen zwischen Betroffenenrollen werden erstellt, mit denen angegeben wird, dass Grace und Pat verheiratet sind.
- Vom System werden auch alle weiteren Beziehung zwischen Betroffenenrollen erstellt (für elterliche und geschwisterliche Beziehungen).
- Außerdem werden Datensätze für Adresse und Telefonnummer erstellt.

In Phase 2 des Zuordnungsprozesses werden die Angabenentitäten erstellt; Beispiele hierfür werden im restlichen Abschnitt aufgeführt.

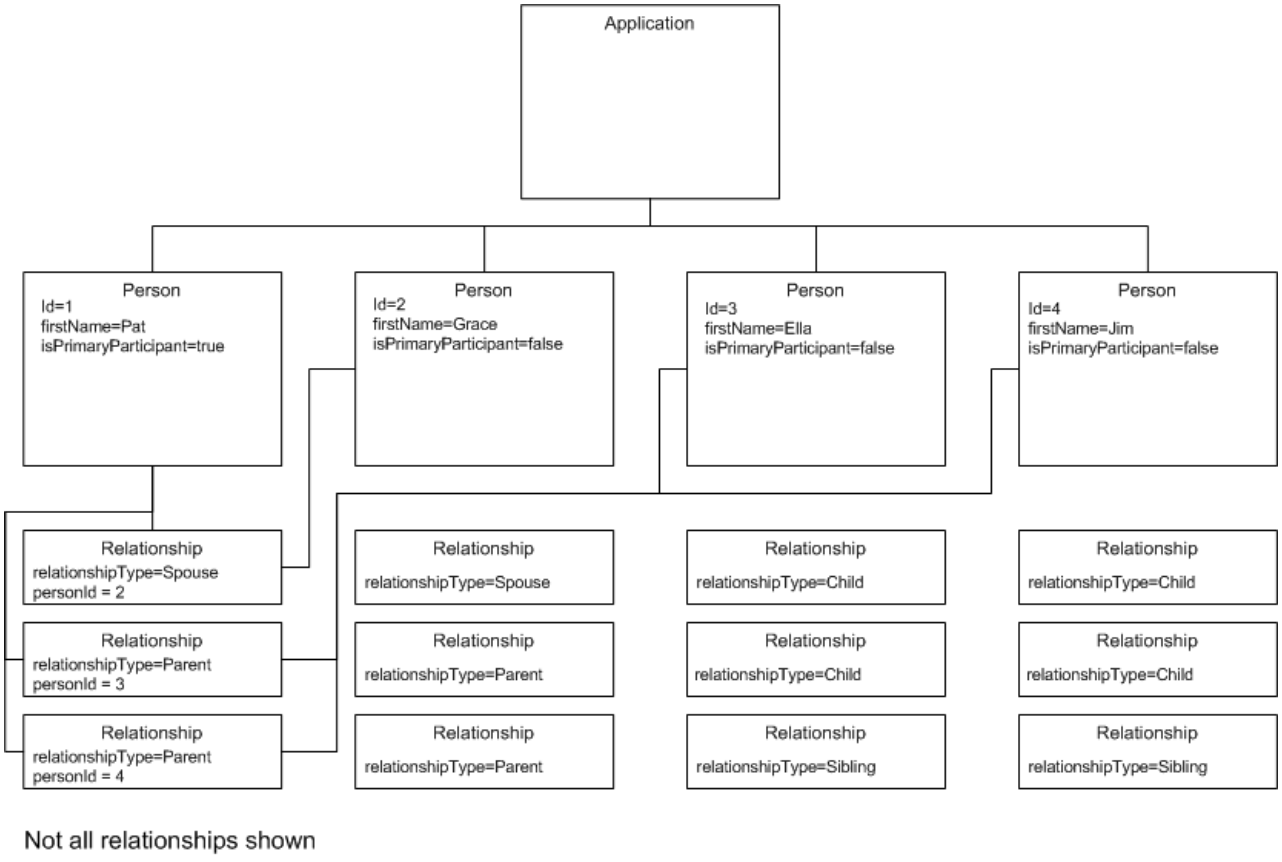

Abbildung 3. Daten im CDS verwenden

Einfache Zuordnungskonfiguration

Am folgenden Beispiel wird eine Konfiguration für den Angabenantragsbuilder veranschaulicht:

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

In diesem Beispiel wurde der Angabenantragsbuilder so konfiguriert, dass von ihm die Angaben 'HouseholdMember' und 'HeadOfHousehold' erstellt werden. In Zeile 3 ist als Java-Basispaketname 'curam.evidence' angegeben. Vom Angabenantragsbuilder werden diese Informationen zum Ableiten der folgenden Angaben zum Mitglied des Haushalts verwendet:

1. Der Name der Klasse für die Angabenserviceschicht (Evidence Service Layer) lautet 'curam.evidence.service.HouseholdMember'.
2. Der Name der Operation in dieser Klasse, die zum Erstellen der Angaben verwendet wird, lautet 'createHouseholdMemberEvidence()'.
3. Der Name der Klasse, der an diesen Aufruf als Argument übergeben wird, lautet 'curam.evidence.entity.struct.HouseholdMemberEvidenceDetails'.

Vom Angabenantragsbuilder werden diese Informationen zum Erstellen der 'HouseholdMember'-Angaben für die Person erstellt, die derzeit verarbeitet wird.

Alle oben aufgeführten Punkte basieren auf der Annahme, dass die Angaben gemäß bestimmter Muster codiert sind. Dies ist sichergestellt, wenn der Angabengenerator (Evidence Generator) zum Generieren der Angaben verwendet wird. Vom Angabenantragsbuilder können auch handcodierte Angaben verarbeitet werden, wenn diese den Mustern entsprechen, die vom Angabengenerator verwendet werden.

Felder des Typs 'caseParticipantDetails' verarbeiten

Zur Verarbeitung der Operation 'createHouseholdMemberEvidence()' für 'HouseholdMember' müssen die Felder des Typs 'caseParticipantDetails' der Struktur 'HouseholdMemberEvidenceDetails' von CDME gefüllt werden. Ein Auszug dieser Struktur wird nachfolgend dargestellt:

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

  /** Attribute of the struct. */
  public  curam.core.sl.struct.CaseIDKey caseIDKey;

  /** Attribute of the struct. */
  public  curam.core.sl.struct.CaseParticipantDetails
    caseParticipantDetails;

  /** Attribute of the struct. */
  public  curam.core.sl.struct.EvidenceDescriptorDetails descriptor;

  /** Attribute of the struct. */
  public  curam.evidence.entity.struct.HouseholdMemberDtls dtls;
 ...
}
```

Die Elemente der Struktur 'dtls' werden weitgehend mithilfe der Elemente <set-attribute> und <map-attribute> in der Zuordnungsspezifikation gefüllt. Beispiel: Die folgende Zeile in der Zuordnungsspezifikation hat zur Folge, dass das Feld 'natHawOrPaIsInd' mit einem Wert in der Struktur 'dtls' gefüllt wird:

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

Das Feld 'caseParticipantDetails' ist oft in der Struktur 'EvidenceDetails' enthalten. In vorliegenden Beispiel wird ein Fallbeteiligter für Grace erstellt und von 'caseParticipantDetails' wird auf diesen Fallbeteiligten verwiesen. Von der Engine für Datenzuordnung wird dies automatisch ausgeführt, wenn ein Feld mit dem Namen

'caseParticipantDetails' in der Struktur 'EvidenceDetails' gefunden wird. Manchmal sind für die Verarbeitung von Fallbeteiligten jedoch Variationen erforderlich, zum Beispiel, wenn in der Struktur 'EvidenceDetails' weitere Fallbeteiligte enthalten sind, von denen auf Dritte verwiesen wird. Beachten Sie Folgendes:

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

  /** Attribute of the struct. */
  public  curam.core.sl.struct.CaseParticipantDetails
    instCaseParticipantDetails;

  /** Attribute of the struct. */
  public  curam.core.sl.struct.EvidenceDescriptorDetails descriptor;

  /** Attribute of the struct. */
  public  curam.evidence.entity.struct.AnnuityDtls dtls;

  /** Attribute of the struct. */
  public  curam.evidence.entity.struct.AnnuityCaseParticipantDetails
    annuityCaseParticipantDetails;
}
```

In diesem Beispiel wird auf den Fallbeteiligten, der Empfänger der Rente ist, in der Struktur 'AnnuityCaseParticipantDetails' verwiesen, die im Feldnamen 'annuityCaseParticipantDetails' aggregiert wird. Die Einrichtung, die die Rente empfängt, wird in der Struktur 'CaseParticipantDetails' beschrieben und im Feldnamen 'instCaseParticipantDetails' aggregiert. Diese Variation kann mithilfe der folgenden Konfiguration des Angabenantragsbuilders verwendet werden:

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

In den Zeilen 2 und 3 wird dem Angabenantragsbuilder mitgeteilt, dass auf die Details des Fallbeteiligten (caseParticipantDetails) für diese Angabenentität vom Feldnamen 'annuityCaseParticipantDetails' unter Verwendung der Struktur 'CaseParticipantDetails' verwiesen wird. Aus den Daten in den Zeilen 5 bis 9 entnimmt der Angabenantragsbuilder die Information, dass auf das Feld 'participantName' der zusammengefassten Struktur 'instCaseParticipantDetails' in der Zuordnungsspezifikation als 'participantName' verwiesen werden kann (Zeile 7). Ähnliches gilt für die Adresse der Einrichtung in den Zeilen 10 bis 14. Aus dem folgenden Beispiel geht hervor, dass es so möglich ist, den Namen und die Adresse der Einrichtung zuzuordnen, die der Empfänger der Rente ist:

```
1  <target-entity name="Annuity" id="AnnuityTarget">
2   <map-attribute
3     from="institutionName"
4     to="participantName"
5   />
```

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

In manchen Fällen kann es zu mühsam sein, den Kunden dazu aufzufordern, all diese Arten an Informationen von Dritten im Rahmen eines Onlineanliegens anzugeben. Stattdessen können für die Zuordnungsspezifikation Standardwerte verwendet werden, die später im Interviewstadium durch die korrekten Werte ersetzt werden. Am folgenden Beispiel wird erläutert, wie für einen beteiligten Dritten wie einem Geldinstitut, zunächst Standardwerte verwendet werden.

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

Der Wert 'curam.blankaddress' in Zeile 12 hat eine leere Adresse zur Folge, in die der Beteiligte eingegeben werden kann.

Zielentitätskennungen festlegen

In Zeile 1 von Beispiel 13 ist wie in einigen vorherigen Beispielen im Element <target-entity> das ID-Attribut 'AnnuityTarget' enthalten. Dieses Attribut ist zwar optional, es ist doch ratsam, in alle Elemente des Typs <target-entity> eine ID einzuschließen. Dies ermöglicht es der Engine für Datenzuordnung, zwischen unterschiedlichen Zuordnungen von derselben Entität zu demselben Zielentitätstyp zu unterscheiden. Beispiel: Die Entität 'Person' im allgemeinen Datenspeicher weist die beiden booleschen Kennungen 'isBlind' und 'hasDisability' auf. Beide werden wie nachfolgend dargestellt demselben Zielentitätstyp 'Disability' zugeordnet:

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

Vom ersten Ziel in den Zeilen 1 bis 11 wird sichergestellt, dass ein Datensatz für Behinderungen des Typs Blindheit erstellt wird, wenn ein Antragsteller angibt, dass er blind ist. Vom zweiten Ziel in den Zeilen 13 bis 18 wird der Indikator 'has-Disability' überprüft; wenn für ihn der Wert 'true' eingestellt ist, wird ein Datensatz für eine Behinderung (Disability) mit nicht weiter spezifiziertem Typ erstellt. Wenn die beiden Zuordnungen über unterschiedliche IDs verfügen, können sie von der Zuordnungsengine unterschieden werden. Ohne die ID wird die zweite Zuordnung nicht verarbeitet.

Für den PDF-Antragsbuilder

Ein PDF-Formular enthält eine Reihe von Feldern unterschiedlichen Typs. Jedes Feld verfügt über einen eindeutigen Namen. Von den Cúram-Arbeitsbereichsservices wird dieser eindeutige Name zum Verweisen auf das Feld verwendet, damit in dieses Feld Daten geschrieben werden können. Damit dieser Prozess ordnungsgemäß ausgeführt werden kann, muss der Autor des PDF-Formulars diese Felder benennen und die Eigenschaften dieser Felder in Übereinstimmung mit bestimmten Konventionen festlegen. Wenn diese Konvention eingehalten werden, können die Daten diesen Feldern vom PDF-Antragsbuilder zugeordnet werden.

Anmerkung: In diesem Abschnitt wird der benutzerdefinierte PDF-Antragsbuilder beschrieben, der das Zuordnen von Daten zu einem angepassten PDF-Formular ermöglicht. Kunden können auch den generischen PDF-Builder verwenden. Informationen hierzu finden Sie im Abschnitt zur Anpassung der generischen PDF für verarbeitete Anträge im Kapitel zur Anpassung der Verarbeitung übermittelter Anträge im Handbuch zur Anpassung von Cúram Universal Access.

Abschnitte und Felder

Die grundlegendste Konvention besagt, dass Felder in "Abschnitten" zusammengefasst werden. Diese Abschnitte müssen nicht unbedingt den Abschnitten im Formular entsprechen, tun dies aber in vielen Fällen. Beispiel:

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

In Beispiel 14 wird ein Auszug aus einer Konfiguration des PDF-Antragsbuilders dargestellt. Er betrifft den Abschnitt mit der Bezeichnung 'Antragsteller' (Applicant). Basierend auf Beispiel 14, Zeile 4, erwartet der PDF-Antragsbuilder, dass im PDF-Zielformular ein Textfeld mit dem Namen 'Applicant.Name' enthalten ist. In

Zeile 8 wird auf ein Feld mit dem Namen 'Applicant.Gender' im PDF-Formular verwiesen. Dieses Feld ist ein Optionsfeld, in den Zeilen 10 bis 14 wird dagegen auf Felder verwiesen, die Kontrollkästchen sind.

Textfelder füllen

In Beispiel 14 wird in Zeile 6 auf ein Klartextfeld verwiesen. Die entsprechende Zuordnung kann der nachfolgend aufgeführten ähneln:

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

Zeile 5 weicht etwas ab. Als Typ ist 'append' angegeben. Dies bedeutet, dass in dasselbe Textfeld mehrfach geschrieben werden kann und bei jedem Schreibvorgang der Zuordnungsengine in das Textfeld das Ergebnis an den aktuellen Wert des Textfelds angehängt und dieser nicht überschrieben wird. Bei jedem Anhängevorgang werden die neuen Daten von den alten Daten durch ein Anhangtrennzeichen getrennt, im vorliegenden Fall durch ein einzelnes Leerzeichen. Wenn beispielsweise die Zuordnungsdatei aus Beispiel 16 mit der Zuordnungskonfiguration aus Beispiel 14 kombiniert wird, wird das Feld 'Applicant.Name' mit dem Vornamen des Antragstellers, der Initiale des zweiten Vornamens und dem Nachnamen gefüllt, zum Beispiel "Pat A Kayek".

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

Das Anhängen in Textfeldern ist auch zum Erstellen von durch Kommas getrennten Listeneinträgen sinnvoll. Beispiel: Der Kunde wird aufgefordert, in einem Feld eine Liste der Personen im Haushalt anzugeben, die schwanger sind. Ein Auszug der Zuordnungs-XML kann im Normalfall wie nachfolgend dargestellt lauten:

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

Die entsprechende Zuordnungskonfiguration wird in Abbildung 18 dargestellt. Bei jeder Verarbeitung einer Person im Haushalt durch die Zuordnungsengine, für die für den Indikator 'isPregnant' der Wert 'true' festgelegt ist, wird der Vorname dieser Person an das Feld 'Pregnancy.Pregnancies' angehängt.

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

Wiederkehrende Abschnitte und Codetabellenbeschreibungen

Manche Formulare enthalten wiederkehrende Abschnitte, zum Beispiel 'Listen Sie die Details aller Personen im Haushalt auf' oder 'Listen Sie alle Einkommen aus Arbeit auf'. Der PDF-Antragsbuilder ist hierfür konzipiert, sofern der Autor der PDF-Datei die Felder gemäß den korrekten Konventionen angibt. Die Felder zum Erfassen der Daten zu den Mitgliedern des Haushalts könnten zum Beispiel wie folgt lauten:

Tabelle 4. Felder in einem PDF-Formular für die Erfassung von Mitgliedern eines Haushalts

Name	In welcher Beziehung stehen Sie zu dieser Person?	Geburtsdatum	Sozialversicherungsnummer
OtherPerson0.Name	OtherPerson0 .RelType	OtherPerson0 .DateOfBirth	OtherPerson0 .SSN
OtherPerson1.Name	OtherPerson1 .RelType	OtherPerson1 .DateOfBirth	OtherPerson1 .SSN
OtherPerson2.Name	OtherPerson2 .RelType	OtherPerson2 .DateOfBirth	OtherPerson2 .SSN

Die entsprechende Zuordnungskonfiguration würde wie folgt geschrieben werden:

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

Beachten Sie, dass das Attribut 'type="multiple"' in Zeile 1 zur Folge hat, dass der Abschnitt wiederholt wird. Beachten Sie auch das Attribut 'codetable-class' in Zeile 3 dieses Beispiels. Hierbei handelt es sich um ein sehr nützliches Attribut, das zur Folge hat, dass die Codetabellenwerte in lokalisierte Beschreibungen umgesetzt werden. Wenn der Autor des Scripts es im obigen Kontext verwendet, stellt er damit sicher, dass die zweite Spalte mit lokalisierten Werten wie 'Elternteil' und 'Geschwister' anstatt bedeutungsloser Codes wie 'RT1' oder 'RT3' gefüllt wird.

Kontrollkästchen

Ein Kontrollkästchen ist ein einzelnes Feld, das markiert oder nicht markiert sein kann. Vom PDF-Antragsbuilder wird davon ausgegangen, dass die Einstellung 'Ja' (Yes) für ein Kontrollkästchenfeld bedeutet, dass es markiert ist, während die Einstellung 'Nein' (No) bedeutet, dass es nicht markiert ist. Vom Autor des PDF-Formulars muss sichergestellt werden, dass diese Konvention eingehalten wird. Wenn von der Zuordnungsengine einem Kontrollkästchenfeld ein boolescher Wert zugeordnet wird, wird die Zuordnung automatisch wie folgt durchgeführt: 'true' wird als 'Ja' (Yes) und 'false' als 'Nein' (No) zugeordnet.

Optionsfelder

Eine Auflistung von Optionsfeldern wird in einem PDF-Formular als einzelnes Feld behandelt. Nur eines der Optionsfelder kann gleichzeitig ausgewählt werden. Die einzelnen Elemente im Optionsfeld werden durch das Schreiben eines bestimmten Werts in das Optionsfeld ausgewählt. Der Autor des PDF-Formulars kann durch Angeben eines 'Exportwerts' für jedes Element festlegen, welches Element ausgewählt wird. Eine typische Verwendung eines Optionsfelds in Cúram ist die Verwendung von Exportwerten zum Angeben einer Anzahl von Codetabellenelementen.

Mit einem Optionsfeld könnten zum Beispiel die beiden Optionen 'Männlich' oder 'Weiblich' angegeben werden. Der Codetabellenwert für 'Männlich' lautet 'SX1', der für 'Weiblich' ist 'SX2'. Der Autor der PDF-Datei erstellt ein einzelnes Optionsfeld mit der Bezeichnung 'Applicant.Gender'. Für das Element 'Männlich' wird der Exportwert 'SX1' angegeben, für das Element 'Weiblich' der Exportwert 'SX2'. Die Zuordnung ähnelt der nachfolgend aufgeführten:

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

Die entsprechende Zuordnungskonfiguration ähnelt der folgenden:

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

Auswahllisten

Eine Auswahlliste ist eine Dropdown-Liste mit Einträgen, aus der der Benutzer einen Eintrag auswählen kann. Der Name des Elements bietet dem Benutzer des Formulars ausreichend Informationen, um zu entscheiden, welches Element die richtige Auswahl ist. Beispiel: Eine Person, die ein Formular entwirft, möchte eine Dropdown-Liste zur Darstellung des Ausländerstatus einer Person bereitstellen. Im Lieferumfang von Cúram ist die Codetabelle 'AlienStatus' enthalten, die Codes für die folgenden Beschreibungen enthält:

- Ausländer
- US-Staatsbürger
- Unregistrierter Ausländer
- Flüchtling
- Ausländischer Staatsangehöriger

Der Entwickler des PDF-Formats erstellt eine Auswahlliste und legt den Elementtext für jedes Element in der Dropdown-Liste die oben genannten Werte fest. Um sicherzustellen, dass die Codetabellenbeschreibung und nicht der Codetabellencode an das Formular gesendet wird, ist die folgende Konfiguration erforderlich:

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

Verwendungshinweise zum Konfigurieren des PDF-Antragsformulars

Laden Sie zunächst das PDF-Formular in Universal Access Portal. Hierbei ist wichtig, ein PDF-Formular und nicht nur ein PDF-Dokument zu verwenden. In der PDF-Datei muss ein Formular enthalten sein und das Formular muss Felder enthalten. Wenn Sie den PDF-Antragsbuilder verwenden möchten, muss jedes Feld im Formular über einen eindeutigen Namen verfügen, zum Beispiel 'PersonalDetails.surname' im Gegensatz zu 'Child1Details.surname'. Sie müssen zum Bearbeiten des Formulars ein Programm wie Adobe Acrobat Writer Professional oder GlobalSCAPE CutePDF Pro verwenden.

Stellen Sie bei Verwendung von Adobe Acrobat Writer Professional sicher, dass Sie das Formular als 'AcroForm' und nicht als 'XFA' speichern. Wenn Sie die PDF-Datei hochladen möchten, melden Sie sich als Administrator an Cúram an, wählen Sie den Abschnitt für die Universal Access-Verwaltung aus und wählen Sie 'PDF-Formulare' aus. An dieser Stelle können Sie das Formular hochladen und einen sinnvollen Namen festlegen. Da jeder Anliegenantrag über einen Anliegenantragstyp verfügt, müssen Sie als nächstes sicherstellen, dass der Anliegenantragstyp dem richtigen PDF-Formular zugeordnet ist. Wechseln Sie hierzu zu 'Anliegenanträge', wählen Sie den entsprechenden Anliegenantragstyp aus und wählen Sie 'Bearbeiten' aus. Klicken Sie auf die Dropdown-Liste für PDF-Formulare; das neu geladene

PDF-Formular muss in der Liste angezeigt werden. Wählen Sie es aus. Im nächsten Schritt müssen Sie PDF-Zuordnungen für einzelne Programme angeben, die für Sie von Interesse sind.

Schreiben Sie mithilfe der Anweisungen im vorherigen Abschnitt dieses Handbuchs eine XML für die PDF-Zuordnung und eine XML für die PDF-Anwendungskonfiguration. Wechseln Sie zum Menüpunkt 'Programme' im Abschnitt für die Verwaltung von Universal Access. Eine Programmliste wird angezeigt. Zeigen Sie das Programm an, das für Sie von Interesse ist. Wählen Sie die Registerkarte 'Zuordnung' auf der rechten Seite aus. Erstellen Sie eine neue Zuordnung. Stellen Sie sicher, dass 'PDF-Formularerstellung' anstatt 'Angabenerstellung' ausgewählt ist. Laden Sie die Zuordnungskonfigurationsdatei und die Zuordnungsspezifikationsdatei hoch. Testen Sie die Zuordnung durch Ausführung eines Anliegens für das jeweilige Programm. Wählen Sie am Ende des Anliegens die Verknüpfung zum Anzeigen der PDF-Datei aus.

Zuordnungsspezifikationen und -konfigurationen für dynamische Angaben schreiben

Einführung

In diesem Dokument wird beschrieben, wie Bürgerdaten während eines Anliegenprozesses dynamischen Angabenentitäten zugeordnet werden.

Voraussetzungen:

Es wird davon ausgegangen, dass Sie mit den grundlegenden Konzepten für dynamische Angaben vertraut sind. Insbesondere wird vorausgesetzt, dass Sie über ausführliche Kenntnisse der Angabenarten, der Definitionen dynamischer Angabentypen, der Definitionen von Angabenversionen und der XML-Metadaten der dynamischen Angaben verfügen.

Zuordnungsspezifikationen und -konfigurationen für dynamische Angaben schreiben

Dynamische Angaben werden ihrem Wesen nach nicht modelliert. In einer einzelnen Entität (oder mehreren Entitäten) sind die Daten für alle dynamische Angabentypen enthalten. Ein dynamischer Angabentyp weist in seiner Lebensdauer mindestens eine Angabentypversion auf. Gültig ist jedoch immer nur eine Angabentypversion. Alle Metadatenelemente (Attribute, Beziehungen, etc) sind auf der Ebene der Angabentypversion definiert. Somit ist der Entwickler für die Bereitstellung der korrekten Zuordnungsspezifikation und -konfigurationen für die derzeit gültige Angabentypversion verantwortlich.

Einfache Metadaten dynamischer Angaben

Die folgenden einfachen Metadaten stellen die Struktur für den dynamischen Angabentyp 'Haushaltsmitglied' dar. Die unterschiedlichen Attributtypen, wie zum Beispiel Boolesch (Boolean), Codetabelle (Codetable), Datum (Date) und Zeichenfolge (String), werden nachfolgend definiert:

```
<?xml version="1.0" encoding="UTF-8"?>
<EvidenceTypeVersion xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="file://DynamicEvidenceMetadata.xsd">
    <Model>
        <Attributes>
            <Attribute>
                <DataAttribute name="blkOrAfrAmerInd">
```

```
                    <DomainType dataType="Boolean" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <DataAttribute name="ssnStatus">
                    <DomainType dataType="Codetable">
                        <CodetableOptions codetableName=
                                "SSNApplicationStatus" />
                    </DomainType>
                </DataAttribute>
            </Attribute>
            <Attribute>
                <DataAttribute name="startDate">
                    <DomainType dataType="Date" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <DataAttribute name="endDate">
                    <DomainType dataType="Date" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <DataAttribute name="comments">
                    <DomainType dataType="String" />
                </DataAttribute>
            </Attribute>
        </Attributes>
    </Model>
    <Validations>
        <PatternValidations>
        </PatternValidations>
    </Validations>
    <UserInterface />
</EvidenceTypeVersion>
```

Einfache Zuordnungsspezifikation

Mit den folgenden einfachen Zuordnungsspezifikationen werden die Daten aus der Entität 'HouseHoldMember' in einem Datenspeicher der dynamischen Angabenentität 'HouseHoldMember' zugeordnet, die im vorherigen Abschnitt definiert ist.

Beachten Sie hierbei, dass der im Feld 'to' angegebene Attributname mit dem Attributnamen übereinstimmen muss, der im Feld 'name' des Elements 'DataAttribute' der Metadaten der dynamischen Angaben angegeben ist. Dies bedeutet somit, dass ein Entwickler sicherstellen muss, dass das Attribut, dem die Daten zugeordnet werden sollen, in den Metadaten dieses dynamischen Angabentyps vorhanden ist.

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

Einfache Zuordnungskonfiguration

Die folgende einfache Zuordnungskonfiguration ist für den dynamischen Angabentyp für Haushaltsmitglieder (HouseHoldMember) definiert:

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

Ein Entwickler muss einen dynamischen Angabentypcode im Attribut 'ev-type-code' angeben. So soll es dem System ermöglicht werden, festzustellen, ob die Angaben statisch oder dynamisch sind. Wenn für dieses Attribut kein Wert oder ein ungültiger Eintrag angegeben wird, wird vom System davon ausgegangen, dass die Angaben statisch sind, und das Zuordnen der Daten wird fortgesetzt.

Über- und untergeordnete dynamische Angaben zuordnen

In diesem Abschnitt wird beschrieben, wie eine Zuordnung für Angaben durchgeführt wird, die über eine Beziehung zwischen einem übergeordnetem und einem untergeordnetem Element verfügen.

Einfache Metadaten über- und untergeordneter dynamischer Angaben

In den folgenden Metadaten sind Attribute für den dynamischen Angabentyp 'Adoption' enthalten. Da die Metadaten für 'Adoption' über zwei Attribute verfügen, weist diese Entität zwei zugehörige Felder des Typs 'CaseParticipant' auf. Das Feld 'caseParticipantRoleID' ist eine primäre Fallbeteiligtenrolle (CaseParticipantRole) und 'parCaseParticipantRoleID' ist eine zugeordnete Fallbeteiligtenrolle (CaseParticipantRole).

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

Die Metadaten 'AdoptionPayment' verfügen über eine Beziehung zu den übergeordneten dynamischen Angaben.

```
<?xml version="1.0" encoding="UTF-8"?>
<EvidenceTypeVersion xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="file://DynamicEvidenceMetadata.xsd"
 javaHookClassNameForCalculatedAttributes="curam.dynamicevidencetest.
 hook.impl.AdoptionPaymentCalcualtedAttributeHook"
 useHookForCalculatedAttributes="true">
    <Model>
        <Attributes>
            <Attribute>
                <DataAttribute name="amount">
                    <DomainType dataType="Money" />
                </DataAttribute>
            </Attribute>
            <Attribute>
                <CalculatedAttribute name="parentName">
                    <DomainType dataType="String" />
                </CalculatedAttribute>
            </Attribute>
        </Attributes>
        <Relationships>
            <Relationship>
                <MandatoryParent name="adoptions"
                           evidenceTypeCode="DET004" />
            </Relationship>
        </Relationships>
    </Model>
</EvidenceTypeVersion>
```

Einfache Zuordnungsspezifikation mit Beziehung zwischen über- und untergeordnetem Element

Die Zuordnungsspezifikation verfügt über eine Beziehung zwischen übergeordnetem und untergeordnetem Element, und für die Adoptionsentität sind wenige Attribute definiert, die alle zum Erstellen eines neuen Beteiligten verwendet werden. Die Werte für diese Attribute werden aus dem IBM Cúram-Datenspeicher abgerufen.

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

```
            </target-entity>
        </target-entities>
    </map-entity>
</map>
```

Einfache Zuordnungskonfiguration für Beziehung zwischen übergeordnetem und untergeordnetem Element

In diesem Abschnitt werden die Elemente <def-create-participant> und <create-participant> vorgestellt. Wichtig ist in diesem Zusammenhang, dass das neue Attribut 'dyn-evidence-primary-cpr-field-name' zum Element <entity> hinzugefügt wird. Der Entwickler muss mithilfe dieses Attributs den Attributnamen der primären Fallbeteiligtenrolle (CaseParticipantRole) angeben, der in den Metadaten definiert ist. Im folgenden Beispiel steht 'caseParticipantRoleID' für die primäre 'CaseParticipantRole', die in der Entität 'Adoption' definiert ist. Auf ähnliche Weise ist der zugehörige Attributname 'CaseParticipantRole' (im folgenden Beispiel 'parCaseParticipantRoleID') im Feld 'name' des Elements <create-participant> definiert. Anmerkung: Bei statischen Angaben wird dasselbe Feld 'name' des Elements <create-participant> zum Angeben des Zusammenfassungsnamens verwendet.

```
<?xml version="1.0" encoding="UTF-8"?>
<application-builder-config xmlns="http://www.curamsoftware.com/
                            schemas/GUMBO/ApplicationBuilderConfig">
    <evidence-config package="curam.evidence">
     <def-create-participant id="AdoptedParentDetails" type="RL13">
           <participant-name-field name="firstName"
                                   from="adParentName" order="1" />
           <participant-address type="AT3">
               <address-field name="addressLine1"
                                        from="adParentStreet1" />
               <address-field name="addressLine2"
                                        from="adParentStreet2" />
               <address-field name="city" from="adParentCity" />
               <address-field name="state" from="adParentState" />
               <address-field name="zip" from="adParentZipCode" />
           </participant-address>
      </def-create-participant>
       <entity name="Adoption" ev-type-code="DET004"
       dyn-evidence-primary-cpr-field-name="caseParticipantRoleID">
           <create-participant refid="AdoptedParentDetails"
                       name="parCaseParticipantRoleID" role="" />
        </entity>
        <entity name="AdoptionPayment" ev-type-code="DET005"/>
    </evidence-config>
</application-builder-config>
```

Zuordnungen zu Dritten

Einführung

Die Funktion für die Anliegenaufnahme in Universal Access ermöglicht dem Benutzer das Eingeben der Details seiner Umstände mithilfe eines IEG2-Scripts. Über das IEG2-Script werden die Detaildaten des Benutzers in den Datenspeicher eingefügt. Im Rahmen der folgenden Übertragung wird der Inhalt des Datenspeichers den Angabendaten für einen Anliegenfall zugeordnet. Von vielen Angabentypen wird auf Dritte verwiesen; diese Dritten müssen in den Fall als neue Fallbeteiligte mit einer eindeutigen Fallbeteiligtenrolle eingefügt werden. So kann zum Beispiel einem Datensatz für eine Schwangerschaft (Pregnancy) ein 'Vater' (Father) zugeordnet sein. Wenn der Vater abwesend ist, kann dieser Vater als Fallbeteiligter eines Anwärters aufgezeichnet werden. Weiteres Beispiel: Den Angaben eines Schülers (Student) muss eine Schule zugeordnet sein. Eine Schule wird als Fallbeteiligter des

Typs 'Repräsentant' (Representative) eingegeben. Diese neuen Fallbeteiligten müssen während der Verarbeitung der Zuordnung erstellt werden und so viele Informationen wie möglich enthalten, damit dem zugeordneten Fallbearbeiter des jeweiligen Falls die Verarbeitung des Anliegens erleichtert wird.

Die Zuordnung von Adressen ist erforderlich, weil ein neuer oder vorhandener Beteiligter einer neuen Angabe zugeordnet werden muss. Der Beteiligte ist in der Regel entweder ein Repräsentant oder ein Anwärter. Eine besondere Herausforderung beim Erstellen neuer Beteiligter ist das Zuordnen der Adressen. Die im Datenspeicher gespeicherten Adressfelder, wie zum Beispiel 'ADD1', müssen in einer formatierten Cúram-Adressenstruktur korrekt zusammengefasst werden, um sicherzustellen, dass der Beteiligte ordnungsgemäß erstellt werden kann.

Aus diesem Kapitel geht hervor, dass die Logik für das Erstellen eines neuen Beteiligten und seine Zuordnung zu einer Adresse vom Angabenantragsbuilder isoliert ist.

Verwendungshinweise zum Zuordnen von Dritten

Im Konfigurationsschema des Antragsbuilders sind Elemente und Attribute enthalten, die zum Erstellen eines neuen Beteiligten und Zuordnen einer Adresse zu diesem Beteiligten verwendet werden können.

Beteiligtenerstellerdefinition

Das Element <def-create-participant> ist Bestandteil des Elements <evidence-config>. Mithilfe dieses Elements wird das Verhalten für die Erstellung eines Beteiligten definiert. Dasselbe Verhalten kann von mehreren Definitionen des Typs <entity> über die ID erneut verwendet werden. Hierbei muss beachtet werden, dass der Datentyp aller hier aufgeführten Attribute eine Zeichenfolge (String) ist.

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

Beteiligten erstellen

Das Element <create-participant> wird im Element <entity> hinzugefügt. Von diesem Element wird der Antragsbuilder angewiesen, wie in der Beteiligtenerstellerdefinition angegeben, einen Beteiligten zu erstellen.

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

Beispiel für Zuordnungsschema

Das folgende Schema dient als Orientierung beim Schreiben einer Zuordnungsspezifikation. Beachten Sie, dass die Attribute in der Entität 'Education' direkt der An-

gabenentität 'Student' zugeordnet werden. Anhand von Attributen wie 'schoolName', 'schoolStreet1', 'schoolStreet2', etc werden ein neuer Beteiligter und eine neue Adresse erstellt.

```
<?xml version="1.0" encoding="UTF-8"?>
<map xmlns="http://www.curamsoftware.com/schemas/GUMBO/Map"
  from-schema="GumboDS" name="TestMapping"to-schema="CGISS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="...\EJBServer\components\
  WorkspaceServices\lib\Mapping.xsd"><map-entity source="Person">
     <target-entity id="householdMember" name="HouseholdMember">
      <map-attribute from="ssnStatus" to="ssnStatus"/>
      <map-attribute from="blackOrAfricanAmerican"
                              to="blkOrAfrAmerInd"/>
      <map-attribute from="nativeAlaskanOrAmericanIndian"
                              to="natHawOrPaIsInd"/>
      <map-attribute from="asian" to="asianInd"/>
      <map-attribute from="nativeHawaiianOrPacificIslander"
                              to="natHawOrPaIsInd"/>
      <map-attribute from="whiteOrCaucasian"
                               to="whiteOrCaucInd"/>
      <map-attribute from="isMigrantOrSeasonalFarmWorker"
                            to="migrantFWorkerInd"/>
    </target-entity>
    <target-entity id="livingArrange" name="LivingArrange">
      <map-attribute from="accommodationType"
                                to="livingArrangeType"/>
    </target-entity>
  </map-entity>
  <map-entity source="Education">
        <condition expression=
                         "Education.highestGrade!=&amp;quot;&amp;quot;">
            <target-entity id="highestGrade" name="Student">
                <map-attribute from="highestGrade"
                                      to="highGradeCompleted"/>
                <map-attribute from="attendanceFrequency"
                                         to="studentStatus"/>
                <map-attribute from="schoolName"
                                        to="participantName"/>
                <map-attribute from="schoolStreet1" to="street1"/>
                <map-attribute from="schoolStreet2" to="street2"/>
                <map-attribute from="schoolCity" to="city"/>
                <map-attribute from="schoolState" to="state"/>
                <map-attribute from="schoolZipCode" to="zipCode"/>
            </target-entity>
        </condition>
    </map-entity>
  <map-entity source="HealthInsuranceExpense">
    <target-entity id="healthInsuranceExpense"
                                       name="MedicalInsurance">
      <map-attribute from="policyNumber" to="policyNumber"/>
      <map-attribute from="groupNumber" to="groupPolicyNumber"/>
        <map-attribute from="policyHolderParticipantName"
              to="policyHolderParticipantName"/>
        <map-attribute from="policyHolderStreet1"
                                     to="policyHolderStreet1"/>
        <map-attribute from="policyHolderStreet2"
                                     to="policyHolderStreet2"/>
        <map-attribute from="policyHolderCity"
                                       to="policyHolderCity"/>
        <map-attribute from="policyHolderState"
                                       to="policyHolderState"/>
        <map-attribute from="policyHolderZipCode"
                                     to="policyHolderZipCode"/>
        <map-attribute from="groupParticipantName"
                                    to="groupParticipantName"/>
        <map-attribute from="groupStreet1" to="groupStreet1"/>
```

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

Beispiel für Zuordnungskonfiguration

Die folgende Konfigurations-XML dient als Orientierung beim Schreiben einer Zuordnungsspezifikation.

```
<?xml version="1.0" encoding="UTF-8"?><application-builder-config xmlns=
"http://www.curamsoftware.com/schemas/GUMBO/ApplicationBuilderConfig">
 <evidence-config package="curam.evidence">
      <def-create-participant id="SchoolRepresentative" type="RL13">
          <participant-name-field name="firstName" from=
                                      "participantName" order="1"/>
          <participant-address type="AT3">
              <address-field name="addressLine1" from="street1"/>
              <address-field name="addressLine2" from="street2"/>
              <address-field name="city" from="city"/>
              <address-field name="state" from="state"/>
              <address-field name="zip" from="zipCode"/>
          </participant-address>
      </def-create-participant>
      <def-create-participant id="MedicalInsurancePolicyHolder"
                                                    type="RL7">
          <participant-name-field name="firstName"
                    from="policyHolderParticipantName" order="1"/>
          <participant-address type="AT3">
              <address-field name="addressLine1"
                                      from="policyHolderStreet1"/>
              <address-field name="addressLine2"
                                      from="policyHolderStreet2"/>
              <address-field name="city" from="policyHolderCity"/>
              <address-field name="state" from="policyHolderState"/>
              <address-field name="zip" from="policyHolderZipCode"/>
          </participant-address>
      </def-create-participant>
      <def-create-participant id="MedicalInsuranceGroup"
                                                   type="RL13">
          <participant-name-field name="firstName"
                         from="groupParticipantName" order="1"/>
          <participant-address type="AT3">
              <address-field name="addressLine1"
                                        from="groupStreet1"/>
```

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

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

Schema für Zuordnungsspezifikationen

Schema

Das folgende Schema dient als Orientierung beim Schreiben einer Zuordnungsspezifikation:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

</xs:schema>
```

Schema für Zuordnungskonfigurationen

Schema

Das folgende Schema dient als Orientierung beim Schreiben einer Zuordnungskonfiguration:

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

  Copyright IBM Corporation 2012. All Rights Reserved.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
        <xs:attribute name="dyn-evidence-primary-cpr-field-name" type="xs:NCName"

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

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

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

 maxOccurs="unbounded"/>
                    <xs:element

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Fehlerprotokolle und Diagnoseprogramme

Fehlercodes

Während der Entwicklung von Zuordnungen stoßen Anwendungsentwickler oft auf unerwartete Fehler. Diese Anwendung kann so konfiguriert werden, dass die Protokollierungsstufe erhöht wird, wenn ein Entwickler einem solchen Problem begegnet.

Die Anwendungseigenschaft

`curam.workspaceservices.application.processing.logging.on`

kann auf "true" gesetzt werden, um diese Nachrichten weiter zu unterteilen.

Es folgt eine Liste der Fehlercodes (und ihrer Codetabellenbeschreibungen), die von dieser Zuordnungs-Engine zurückgegeben werden:

```
APROCER001 Beim Erstellen einer Person ist ein Fehler aufgetreten.
APROCER002 Beim Erstellen eines Anwärters ist ein Fehler aufgetreten.
APROCER003 Beim Erstellen eines Anwärters ist ein Fehler aufgetreten.
APROCER004 Beim Erstellen eines Falls ist ein Fehler aufgetreten.
APROCER005 Beim Ausführen der Zuordnung "map-attribute" ist ein Fehler aufgetreten.
APROCER006 Beim Ausführen der Zuordnung "set-attribute" ist ein Fehler aufgetreten.
APROCER007 Beim Ausführen der Zuordnung "map-address" ist ein Fehler aufgetreten.
APROCER008 Allgemeiner Zuordnungsfehler.
APROCER009 Fehler beim Erstellen von Angaben.
APROCER010 Für den Programmtyp ist mehr als ein PDF-Formular registriert.
APROCER011 Fehler beim Festlegen des alternativen ID-Typs für einen Anwärter.
APROCER012 Ungültiger Wert für die alternative ID.
APROCER013 Fehler: Der Angabenantragsbuilder wurde nicht korrekt konfiguriert.
APROCER014 Der Angabentyp ist in der Zuordnungskonfiguration nicht aufgeführt.
APROCER015 Keine übergeordnete Angabenentität gefunden.
APROCER016 Fehler beim Versuch, das Marshalling der XML-Daten dieses Antrags rückgängig zu machen.
APROCER017 Fehler beim Versuch, einen Feldwert festzulegen.
APROCER018 Fehler beim Versuch, das PDF-Dokument zu erstellen.
APROCER019 Fehler beim Versuch, das PDF-Dokument zu erstellen.
Ein Formularcode konnte keiner Codetabellenbeschreibung zugeordnet werden.
```

```
APROCER020 Fehler beim Versuch, einen Erweiterungshandler für die Zuordnung "WorkspaceServices" au
 APROCER021 Fehlendes Quellenattribut in Datenspeicherentität.
APROCER022 Ein Attribut in einem Ausdruck ist ungültig.
APROCER023 Konfigurationsfehler im Antragsbuilder.
APROCER024 Fehler bei der Erstellung von "DataStoreMappingConfig". Kein Name angegeben.
APROCER025 Fehler bei der Erstellung von "DataStoreMappingConfig". Der Name ist nicht eindeutig.
APROCER026 Die Zuordnung zum Datenspeicher musste abgebrochen werden, da das Schema
 nicht registriert ist.
APROCER027 Problem beim Analysieren der Mapping-Spezifikation.
APROCER028 Allgemeiner Zuordnungsfehler. Die XML-Zuordnungsdatei ist enthalten.
APROCER029 Es kann nicht mehrere primär Beteiligte geben.
APROCER030 Es wurde kein Programm beantragt.
APROCER031 Fehler beim Versuch, eine Zuordnung zu Personendaten herzustellen.
APROCER032 Fehler beim Versuch, eine Zuordnung zu Beziehungsdaten herzustellen.
APROCER033 Fehler bei der Erstellung von Fällen.
APROCER034 Fehler bei der Erstellung von Angaben.
APROCER035 Es wurde kein Programm beantragt.
APROCER036 Beim Lesen von Daten aus dem Datenspeicher ist ein Fehler aufgetreten.
APROCER037 Der angegebene Falltyp ist nicht vorhanden.
APROCER038 Der angegebene Falltyp ist nicht vorhanden.
APROCER039 Es wurde eine doppelte Sozialversicherungsnummer eingegeben.
APROCER040 Es wurde eine doppelte Sozialversicherungsnummer eingegeben.
APROCER041 Es ist ein Problem mit dem Workflowprozess aufgetreten.
APROCER042 Es wurde kein primär Beteiligter als Teil des Anliegenprozesses angegeben-
.
```

Dieser Satz von Fehlercodes, der von der Anwendung zurückgegeben wird, ist der Satz, der in der Codetabellendatei "CT_ApplicationProcessingError.ctx." definiert ist. Der Codebereich, der für interne Prozesse reserviert ist, ist APROCER001 – APROCER500. Das bedeutet, dass Kunden den Bereich zwischen APROCER501 und APROCER999 nutzen können, wenn für die angepasste Verarbeitung, z. B. eine Handlerklasse für eine Erweiterungszuordnung, Fehler protokolliert werden sollen.

Die eigentlichen Fehlernachrichten, die in den Protokollen erscheinen, sind in folgenden Nachrichtendateien zu finden: EJBServer\components\WorkspaceServices\message\Mapping.xml, EJBServer\components\WorkspaceServices\message\WorkspaceServicesDataMapping.xml

Bemerkungen

Die vorliegenden Informationen wurden für Produkte und Services entwickelt, die auf dem deutschen Markt angeboten werden. Möglicherweise bietet IBM die in dieser Dokumentation beschriebenen Produkte, Services oder Funktionen in anderen Ländern nicht an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim zuständigen IBM Ansprechpartner erhältlich. Hinweise auf IBM-Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Services von IBM verwendet werden können. Anstelle der IBM Produkte, Programme oder Services können auch andere, ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder anderen Schutzrechte von IBM verletzen. Die Verantwortung für den Betrieb von Produkten, Programmen und Services anderer Anbieter liegt beim Kunden. Für die in diesem Handbuch beschriebenen Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanforderungen sind schriftlich an folgende Adresse zu richten (Anfragen an diese Adresse müssen auf Englisch formuliert werden):

IBM Director of Licensing

IBM Europe, Middle East & Africa

Tour Descartes

2, avenue Gambetta

92066 Paris La Defense

France

Trotz sorgfältiger Bearbeitung können technische Ungenauigkeiten oder Druckfehler in dieser Veröffentlichung nicht ausgeschlossen werden.

Die hier enthaltenen Informationen werden in regelmäßigen Zeitabständen aktualisiert und als Neuausgabe veröffentlicht. IBM kann ohne weitere Mitteilung jederzeit Verbesserungen und/oder Änderungen an den in dieser Veröffentlichung beschriebenen Produkten und/oder Programmen vornehmen. Verweise in diesen Informationen auf Websites anderer Anbieter werden lediglich als Service für den Kunden bereitgestellt und stellen keinerlei Billigung des Inhalts dieser Websites dar.

Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt. Die Verwendung dieser Websites geschieht auf eigene Verantwortung.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht. Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängig voneinander erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

U.S.A.

Die Bereitstellung dieser Informationen kann unter Umständen von bestimmten Bedingungen - in einigen Fällen auch von der Zahlung einer Gebühr - abhängig sein.

Die Bereitstellung des in diesem Dokument beschriebenen Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt auf der Basis der IBM Rahmenvereinbarung bzw. der Allgemeinen Geschäftsbedingungen von IBM, der IBM Internationalen Nutzungsbedingungen für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer kontrollierten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Gewährleistung, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können davon abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Alle Informationen zu Produkten anderer Anbieter stammen von den Anbietern der aufgeführten Produkte, deren veröffentlichten Ankündigungen oder anderen allgemein verfügbaren Quellen.

IBM hat diese Produkte nicht getestet und kann daher keine Aussagen zu Leistung, Kompatibilität oder anderen Merkmalen machen. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten von IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele von IBM.

Alle von IBM angegebenen Preise sind empfohlene Richtpreise und können jederzeit ohne weitere Mitteilung geändert werden. Händlerpreise können u. U. von den hier genannten Preisen abweichen.

Diese Veröffentlichung dient nur zu Planungszwecken. Die in dieser Veröffentlichung enthaltenen Informationen können geändert werden, bevor die beschriebenen Produkte verfügbar sind.

Diese Veröffentlichung enthält Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufs. Sie sollen nur die Funktionen des Lizenzprogramms illustrieren und können Namen von Personen, Firmen, Marken oder Produkten enthalten. Alle diese Namen sind frei erfunden; Ähnlichkeiten mit tatsächlichen Namen und Adressen sind rein zufällig.

COPYRIGHTLIZENZ:

Diese Veröffentlichung enthält Musteranwendungsprogramme, die in Quellensprache geschrieben sind und Programmiertechniken in verschiedenen Betriebsumgebungen veranschaulichen. Sie dürfen diese Musterprogramme kostenlos kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, zu verwenden, zu vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle für die Betriebsumgebung konform sind, für die diese Musterprogramme geschrieben werden. Diese Beispiele wurden nicht unter allen denkbaren Bedingungen getestet. IBM kann daher die Zuverlässigkeit, Wartungsfreundlichkeit oder Funktion dieser Programm nicht garantieren oder implizieren. Die Beispielprogramme werden ohne Wartung (auf "as-is"-Basis) und ohne jegliche Gewährleistung zur Verfügung gestellt. IBM übernimmt keine Haftung für Schäden, die durch Ihre Verwendung der Musterprogramme entstehen.

Kopien oder Teile der Musterprogramme bzw. daraus abgeleiteter Code müssen folgenden Copyrightvermerk beinhalten:

© (Name Ihres Unternehmens) (Jahr). Teile des vorliegenden Codes wurden aus Musterprogrammen der IBM Corp. abgeleitet.

© Copyright IBM Corp. _Jahreszahl oder Jahreszahlen eingeben_. Alle Rechte vorbehalten.

Wird dieses Buch als Softcopy (Book) angezeigt, erscheinen keine Fotografien oder Farbabbildungen.

Hinweise zur Datenschutzrichtlinie

IBM Softwareprodukte, einschließlich Software as a Service-Lösungen ("Softwareangebote"), können Cookies oder andere Technologien verwenden, um Informationen zur Produktnutzung zu erfassen, die Endbenutzererfahrung zu verbessern und Interaktionen mit dem Endbenutzer anzupassen oder zu anderen Zwecken. In vielen Fällen werden von den Softwareangeboten keine personenbezogenen Daten erfasst. Einige der IBM Softwareangebote können Sie jedoch bei der Erfassung personenbezogener Daten unterstützen. Wenn dieses Softwareangebot Cookies zur Erfassung personenbezogener Daten verwendet, sind nachfolgend nähere Informationen über die Verwendung von Cookies durch dieses Angebot zu finden.

Je nachdem, welche Konfigurationen implementiert wurden, ist es möglich, dass dieses Softwareangebot Sitzungscookies und persistente Cookies zum Erfassen der Namen, Benutzernamen, Kennwörter, Profilnamen oder anderer personenbezogener Daten einzelner Benutzer für die Sitzungsverwaltung, Authentifizierung, Single-Sign-on-Konfiguration oder für einen besseren Bedienungskomfort und/oder andere Zwecke der Nutzungsverfolgung bzw. funktionale Einsatzmöglichkeiten. Diese Cookies oder ähnliche Technologien können nicht inaktiviert werden.

Wenn die für dieses Softwareangebot genutzten Konfigurationen Sie als Kunde in die Lage versetzen, personenbezogene Daten von Endbenutzern über Cookies und andere Technologien zu erfassen, müssen Sie sich zu allen gesetzlichen Bestimmungen in Bezug auf eine solche Datenerfassung, einschließlich aller Mitteilungspflichten und Zustimmungsanforderungen, rechtlich beraten lassen.

Weitere Informationen zur Nutzung verschiedener Technologien, einschließlich Cookies, für diese Zwecke finden Sie in der "IBM Online-Datenschutzerklärung, Schwerpunkte" unter http://www.ibm.com/privacy und in der "IBM Online-Da-

tenschutzerklärung" unter http://www.ibm.com/privacy/details im Abschnitt "Cookies, Web-Beacons und sonstige Technologien" und unter "IBM Software Products and Software-as-a-Service Privacy Privacy Statement" unter http://www.ibm.com/software/info/product-privacy.

Informationen zu Programmierschnittstellen

In der vorliegenden Veröffentlichung werden vorgesehene Programmierschnittstellen dokumentiert, mit deren Hilfe Kunden Programme für den Zugriff auf IBM Cúram Social Program Management-Services schreiben können.

Marken

IBM, das IBM Logo und ibm.com sind Marken oder eingetragene Marken der International Business Machines Corporation. Weitere Produkt- und Servicenamen können Marken von IBM oder anderen Unternehmen sein. Weitere Produkt- und Servicenamen können Marken von IBM oder anderen Unternehmen sein. Eine aktuelle Liste der IBM Marken finden Sie auf der Website "Copyright and trademark information" unter http://www.ibm.com/legal/us/en/copytrade.shtml.

Adobe und das Portable Document Format (PDF) sind Marken oder eingetragene Marken der Adobe Systems Incorporated in den USA und/oder anderen Ländern.

Java und alle auf Java basierenden Marken und Logos sind eingetragene Marken der Oracle Corporation und/oder ihrer verbundenen Unternehmen.

Andere Namen können Marken der jeweiligen Rechtsinhaber sein. Weitere Firmen-, Produkt- und Servicenamen können Marken oder Servicemarken anderer Unternehmen sein.

Gedruckt in Deutschland