XFall Arbeitsgruppe

Umwandlung von FIM Stammdatenschemata in XFall-Nachrichten

Konzept

- ENTWURF -

Stand 2019-10-02

© 2019 FJD Information Technologies AG. All rights reserved.

Historie

Datum	Bearbeiter	Änderung
2017-06-26	Andreas Huber, FJD	Konzept erstellt
2017-07-06	Andreas Huber, FJD	Ergebnis der internen Abstimmung eingearbeitet
2017-11-13	Andreas Huber, FJD	Konzept überarbeitet
2018-12-10	Andreas Huber, FJD	Umbenennung des Bausteins in Datenfelder, weitere Anpassungen
2019-07-04	Andreas Huber, FJD	Fortschreibung
2019-07-12	Andreas Huber, FJD	Fortschreibung
2019-07-31	Andreas Huber, FJD	Ausarbeitung der Abbildungsvorschrift
2019-10-02	Andreas Huber, FJD	Schema-Element aus XFall 6 aufgenommen

Inhaltsverzeichnis

1 Vorgehen....4
1.1 Vorbedingungen....4
1.2 Ergebnis....4
1.2.1 Root Element....4
1.2.2 Namespace....4
1.2.3 Inhalte....5
1.2.4 Feldarten....5
1.2.5 Datentypen....5
1.2.6 Präzisierung....5
1.2.7 Beispiele....6
1.3 XFall-Container....7
2 Anhang....8
2.1 Offene Punkte....8
2.1.1 String.Latin....8
2.2 Beispiel....9

1 Vorgehen

1.1 Vorbedingungen

- Unter einem Element darf es ein anderes Element nur einmal geben. Mehrfache Vorkommen sind immer mit Multiplizitäten abzudecken.
- Ein Element darf ein anderes Element nur in einer Version enthalten, auch in beliebiger Tiefe. Dies stellt sicher, dass in jedem Stammdatenschema jedes enthaltene Element in genau einer Version enthalten ist.
  - Dies wäre umgehbar, wenn auch die Versionsnummer in den Elementnamen für XFall mit eingeht.

1.2 Ergebnis

1.2.1 Root Element

`fim.s00000009.00000009`

`SNNNNNNNN` = FIM-ID

`NNNNNNNN` = Numerische FIM-ID (ohne das führende „S“)

1.2.2 Namespace

Der Namespace setzt sich aus folgenden Teilen zusammen:

- „urn:xoev-de:“ – Prefix gemäß Kosit/XÖV
- „xfall“ – Name des Standards, hier: „XFall“
- „:standard:“ – Kennung für Standards gemäß Kosit/XÖV
- Name der Nachricht, bestehend aus:
  - „fim-“ – Prefix für aus FIM generierte Nachrichten
  - FIM-ID mit kleinem ‚S‘
  - „_“ – Trennzeichen zum Anhängen der Version gemäß Kosit/XÖV
  - FIM-Version

In Summe also wie folgt:

urn:xoev-de:xfall:standard:fim-<FIM-ID mit kleinem ‚S‘>_<FIM-Version>

Zum Beispiel:

urn:xoev-de:xfall:standard:fim-s00000009_1.0

1.2.3 Inhalte

Bei den normalen Feldern (Eingabe und Auswahl) wird der technische Wert übertragen. Dies bedeutet:

- Datumsangaben werden im ISO-Format (z.B. „2019-03-17“) übertragen.
- Wahrheitswerte werden als „true“ und „false“ übertragen.
- Dezimalzahlen verwenden den Punkt als Dezimaltrenner und keinen Tausendertrennzeichen.
- Bei Codelisten wir der Code und nicht der Name übertragen. Also z.B. beim Geschlecht (f=„weiblich“, m=„männlich“, d=„divers“) wird „f“, „m“ oder „d“ übertragen.

1.2.4 Feldarten

- **Eingabe:** Eingabefelder werden auf Elemente abgebildet. Die verwendeten Datentypen werden im folgenden Kapitel definiert.
- **Auswahl:** Die Abbildung von Codelisten ist noch offen.
- **Statisch:** Die statischen Felder werden entfernt, da deren Inhalt dem empfangenen System bekannt ist.

1.2.5 Datentypen

Die folgende Liste enthält die in FIM definierten Datentypen und wie sie im XFall-Schema abgebildet werden.

- **Text** (text): W3C Datentyp „string“; siehe hierzu die Anmerkung „String.Latin“ im Anhang unter offene Punkte
- **Datum** (date): W3C Datentyp „date“
- **Wahrheitswert** (bool): W3C Datentyp „boolean“
- **Nummer** (num): W3C Datentyp „decimal“
- **Ganzzahl** (num_int): W3C Datentyp „integer“
- **Geldbetrag** (num_currency): W3C Datentyp „decimal“
- **Anlage** (file): W3C Datentyp „base64Binary“
- **Objekt** (obj): W3C Datentyp „base64Binary“

1.2.6 Präzisierung

Die definierten Präzisierungen werden durch die lokale Definition eines „simpleType“ mit einer „restriction“ in das XFall-Schema übernommen:

- **Minimale Länge:** „minLength“
- **Maximale Länge:** „maxLength“
- **Minimaler Wert:** „minInclusive“
- **Maximaler Wert:** „maxInclusive“

- **Pattern:** Der reguläre Ausdruck wird auf eine „restriction“ abgebildet. Hierfür muss ein anonymer „simpleType“ gebildet werden.
- **Hinweis:** Der Freitext-Hinweis kann nicht abgebildet werden.

1.2.7 Beispiele

Eine Zahl zwischen 0 und 10:

```
<xs:simpleType>
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="10"/>
  </xs:restriction>
</xs:simpleType>
```

Ein Text mit 3 bis 12 Zeichen:

```
<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:minLength value="3"/>
    <xs:maxLength value="12"/>
  </xs:restriction>
</xs:simpleType>
```

Regulärer Ausdruck für die BIC:

```
<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:pattern value="[a-zA-Z]{6}[0-9a-zA-Z]{2}([0-9a-zA-Z]{3})?"/>
  </xs:restriction>
</xs:simpleType>
```

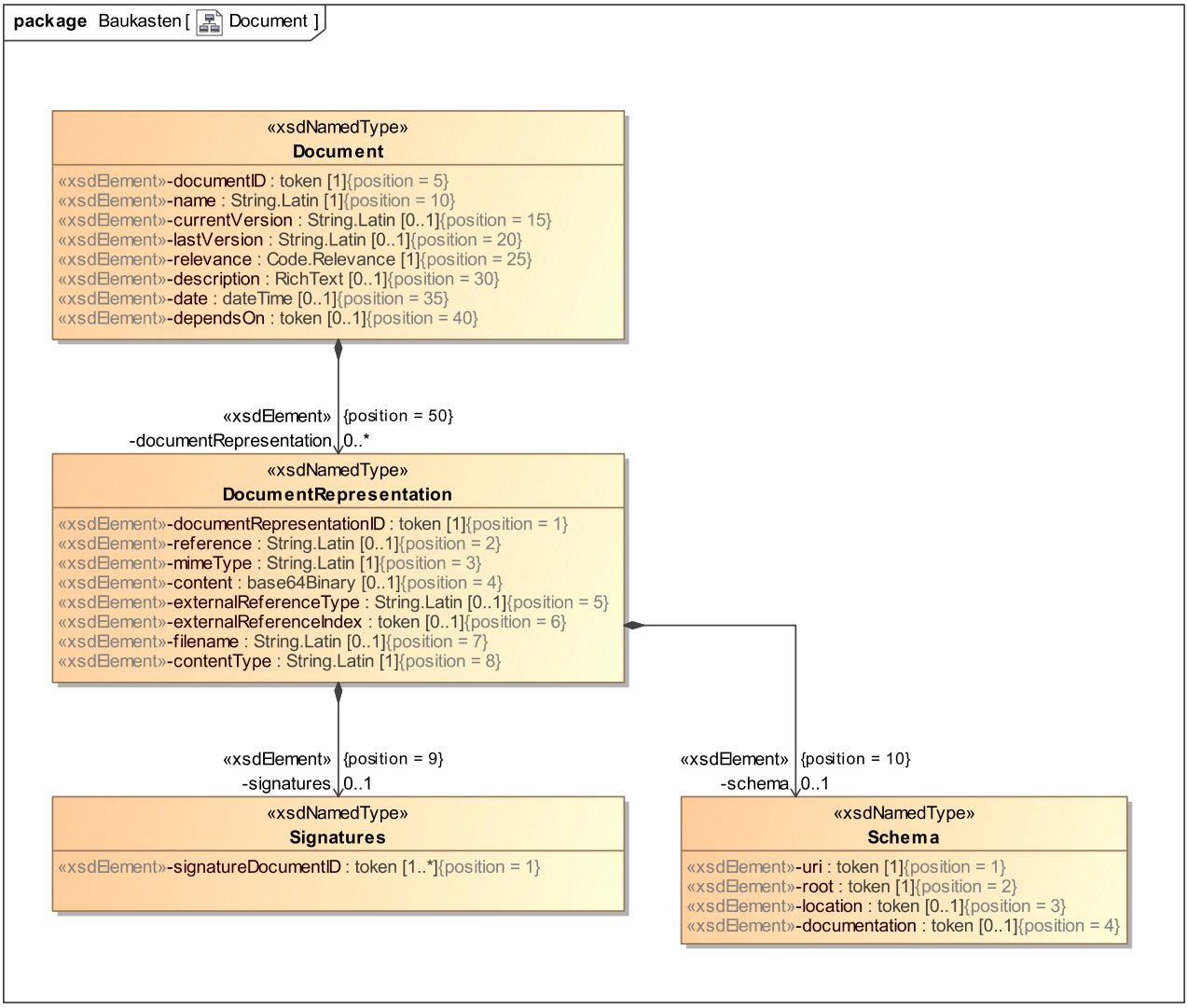
Postleitzahl:

```
<xs:element name="F60000019">
    <xs:annotation>
        <xs:documentation>
            <name>Postleitzahl</name>
            <definition>Eine Postleitzahl ist eine Angabe, um postalische
Zustellgebiete unabhängig von Gebietskörperschaften zu bezeichnen.</definition>
            <status>inVorbereitung</status>
        </xs:documentation>
    </xs:annotation>
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:minLength value="5"/>
            <xs:maxLength value="5"/>
            <xs:pattern value="[0-9]{5}"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

1.3 XFall-Container

Die DocumentRepresentation sollte um optionale Elemente für „schema“ und „schemaLocation“ erweitert werden. Diese enthalten die Schema-Kennung und eine URL, von der das Schema bezogen werden kann.

Dieser Vorschlag wurde in XFall 6 umgesetzt. Die DocumentRepresentation wurde um ein Unterelement „Schema“ mit den Angaben „uri“ (Kennung), „root“ (Wurzelelement), „location“ (XML Schemadatei) und „documentation“ (Dokumentation) ergänzt.



2 Anhang

2.1 Offene Punkte

- Welche Angaben werden im Header benötigt? Vorschläge:
  - Erstellungsdatum
  - Nachrichten-ID
- Wie bilden wir Codelisten ab?
  - XÖV Codeliste & Code
  - XML Enum

2.1.1 String.Latin

Es sollte, wo möglich, String.Latin für Textfelder verwendet werden. Es kann jedoch nicht ausgeschlossen werden, dass einzelne Felder auch nicht-lateinische Zeichen benötigen. Daher können nicht alle Textfelder auf String.Latin abgebildet werden.

Es wird empfohlen, dass das FIM Datenfelder Redaktionssystem um ein Unterscheidungskriterium erweitert wird, welches den Redakteuren ermöglich, festzulegen, ob ein Textfeld vom Typ „String“ oder „String.Latin“ ist.

2.2 Beispiel

```
<fim.S00000009.00000009>
  <header>
    <id>
      <id>S00000009</id>
      <version>1.0</version>
    </id>
  </header>
  <S00000009>
    <!-- Name der entgegennehmenden Gemeinde -->
    <F00000152>Kirchheim bei München</F00000152>
    <!-- Betriebsinhaber -->
    <G00000091>
      <!-- Rechtsform -->
      <F00000103></F00000103>
      <!-- Registereintrag -->
      <G00000100>
        <!-- Ort des Registereintrags -->
        <F00000104>Amtsgericht München<F00000104>
        <!-- Nummer des Registereintrages -->
        <F00000115>HRB 135350</F00000115>
      <!-- Person -->
      <G00000093>
        <!-- Name -->
        <F00000160>Jorga</F00000160>
        <!-- Vorname -->
        <F00000045>Frank</F00000045>
        <!-- Geschlecht -->
        <F00000065 listURI="urn:xpersonenstand:schluesseltabelle:geschlecht"
listVersionID="2.0"><code>m</code></F00000065>

[…]

  </S00000009>
</fim.S00000009.00000009>
```