Memo

Van	Henri Korver
Aan	StUF Expertgroep & Discussieforum StUF 3.01
Afdeling	KING/E-Diensten
Onderwerp	RFC voor StUF 3.01: Overbodig maken nillable=”true” in StUF-schema’s
Vergaderstuk ter	Besluitvorming
Datum	15-10-2015

Inleiding

In het onlangs verschenen rapport “20150925_Eindrapport_DenHaag_StUF_standaard.pdf” (Analyse van de StUF-BG 3.10 standaard), dat de Software Improvement Group in opdracht van de gemeente Den Haag heeft opgesteld, wordt onder andere het gebruik van de **nillable=”true”** constructie in de StUF 3.01 standaard aan de kaak gesteld. Ruwweg gesproken heeft deze constructie van XML Schema twee nadelen:

- Standaard tooling zoals die van Microsoft .NET en Java hebben moeite met deze constructie. Dit leidt bijvoorbeeld tot allerlei problemen en work-arounds bij codegeneratie.
- De combinatie van het gebruik van minOccurs=”0” en nillable=”true” maakt StUF een complexe standaard.

In deze notitie wordt een voorstel gedaan om de schema’s van de nieuwe StUF 3.02 standaard op een andere manier in te richten zodat de nillable-constructie niet meer nodig is. Het nieuwe voorgestelde schema heeft bovendien als voordeel dat het afdwingt dat verplichte elementen altijd voorzien zijn van een noValue waarde indien ze zelf geen inhoud hebben.

Huidige situatie

In de schema’s van StUF 3.01 wordt momenteel de XSD-constructie nillable="true" gebruikt voor het specificeren van lege waarden. Onderstaand voorbeeld geeft aan hoe deze constructie in zijn werk gaat:[1]

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
	<xs:element name="persoon" type="Persoon"/>
	<xs:complexType name="Persoon">
		<xs:sequence>
```

[1] Dit schema voldoet niet aan de officiële StUF-syntax maar een vereenvoudigd voorbeeld hoe StUF met de nillable-constructie omgaat.

```
                    <xs:element name="achternaam" type="Achternaam" nillable="true"
minOccurs="0"/>
                    <xs:element name="voorvoegsel" type="Voorvoegsel"
nillable="true" minOccurs="0"/>
             </xs:sequence>
       </xs:complexType>
       <xs:complexType name="Achternaam">
             <xs:simpleContent>
                    <xs:extension base="AchternaamSimpel">
                          <xs:attribute ref="noValue"/>
                    </xs:extension>
             </xs:simpleContent>
       </xs:complexType>
       <xs:simpleType name="AchternaamSimpel">
             <xs:restriction base="xs:string">
                    <xs:minLength value="1"/>
             </xs:restriction>
       </xs:simpleType>
       <xs:complexType name="Voorvoegsel">
             <xs:simpleContent>
                    <xs:extension base="VoorvoegselSimpel">
                          <xs:attribute ref="noValue"/>
                    </xs:extension>
             </xs:simpleContent>
       </xs:complexType>
       <xs:simpleType name="VoorvoegselSimpel">
             <xs:restriction base="xs:string">
                    <xs:minLength value="1"/>
             </xs:restriction>
       </xs:simpleType>
       <xs:attribute name="noValue" type="NoValue"/>
       <xs:simpleType name="NoValue">
             <xs:restriction base="xs:string">
                    <xs:enumeration value="geenWaarde"/>
                    <xs:enumeration value="waardeOnbekend"/>
             </xs:restriction>
       </xs:simpleType>
</xs:schema>
```

In het complextype “Persoon” wordt het nillable="true" attribuut gebruikt om aan te geven dat de elementen “achternaam” en “voorvoegsel” een lege inhoud mogen hebben. Onderstaand XML-bericht voldoet aan het bovenstaande schema.

```
<persoon xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
       <achternaam>Korver</achternaam>
       <voorvoegsel xsi:nil="true" noValue="geenWaarde"/>
</persoon>
```

Het element “voorvoegsel” is leeg en bevat alleen de attributen xsi:nil="true" noValue="geenWaarde". Het laatste attribuut geeft aan dat het element “voorvoegsel” in de werkelijkheid geen waarde heeft. Het was ook mogelijk geweest dat er wel een voorvoegsel had

bestaan, maar dat het niet bekend is voor de zender van het bericht. In dat geval had het bericht er zo uitgezien:

```
<persoon xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
	<achternaam>Korver</achternaam>
	<voorvoegsel xsi:nil="true" noValue="waardeOnbekend"/>
</persoon>
```

Voorstel

Hieronder een alternatief schema met dezelfde functionaliteit als het eerdere schema maar dan zonder nillable-constructie:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
	<xs:element name="persoon" type="Persoon"/>
	<xs:complexType name="Persoon">
		<xs:sequence>
			<xs:element name="achternaam" type="Achternaam" minOccurs="0"/>
			<xs:element name="voorvoegsel" type="Voorvoegsel"
minOccurs="0"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="Achternaam">
		<xs:choice>
			<xs:element name="waarde" type="AchternaamSimpel"/>
			<xs:element ref="noValue"/>
		</xs:choice>
	</xs:complexType>
	<xs:simpleType name="AchternaamSimpel">
		<xs:restriction base="xs:string">
			<xs:minLength value="1"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:complexType name="Voorvoegsel">
		<xs:choice>
			<xs:element name="waarde" type="VoorvoegselSimpel"/>
			<xs:element ref="noValue"/>
		</xs:choice>
	</xs:complexType>
	<xs:simpleType name="VoorvoegselSimpel">
		<xs:restriction base="xs:string">
			<xs:minLength value="1"/>
		</xs:restriction>
	</xs:simpleType>
	<xs:element name="noValue" type="NoValue"/>
	<xs:simpleType name="NoValue">
		<xs:restriction base="xs:string">
			<xs:enumeration value="geenWaarde"/>
			<xs:enumeration value="waardeOnbekend"/>
		</xs:restriction>
```

```
	</xs:simpleType>
</xs:schema>
```

In bovenstaand schema is het attribuut “noValue” omgezet naar een gelijknamig element en is het element “waarde” erbij gekomen. De gegevens van een Persoon zoals achternaam en voorvoegsel worden nu als een choice van de elementen “waarde” en “noValue” gemodelleerd. Dus als de waarde beschikbaar is dan wordt er gekozen voor het element “waarde” en anders voor het element “noValue”, zie onderstaand voorbeeld:

```
<persoon xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
	<achternaam>
		<waarde>Korver</waarde>
	</achternaam>
	<voorvoegsel>
		<noValue>geenWaarde</noValue>
	</voorvoegsel>
</persoon>
```

Naast het feit dat dit voorstel de in de inleiding genoemde nadelen wegneemt, biedt het ook het extra voordeel dat het schema veel scherper is geworden. In de huidige schema’s wordt niet afgedwongen dat een element met xsi:nil="true" het attribute noValue moet bevatten. Bijvoorbeeld, onderstaand bericht wordt (onterecht) gevalideerd door het schema van de vorige sectie:

```
<persoon xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
	<achternaam>Korver</achternaam>
	<voorvoegsel xsi:nil="true"/>
</persoon>
```

In het nieuwe voorgestelde schema is dit probleem opgelost. Immers als een element verplicht is, dan moet nu hetzij een element met een geldige waarde worden opgenomen, hetzij een noValue element met een geldige noValue waarde.