Zorginstituut Nederland

Ontwerprichtlijnen voor XML-schemadefinities (XSD's)

Datum	1 juli 2019
Status	Definitief
Versienummer	1.4
Volgnummer intern	2015038977
Afdeling	Informatiemanagement
Team	iStandaarden
Contact	info@istandaarden.nl

Versies:

Versie	Datum	Opmerkingen
1.4	01-07-2019	Aanpassingen m.b.t. vastgesteld versiebeheer van releases en bijbehorende documentatie.
1.3	01-09-2017	De beschrijving van de "formatting" van het XML bericht aangepast conform bedrijfsregel OP192.
1.2	18-07-2017	Toevoeging opname releasenummer in document element Toevoeging omgang lege elementen.
1.1	03-07-2017	Aanpassingen n.a.v. nieuwe releases iWlz 2.0, iWmo 2.2 en iJw 2.2
1.0	01-06-2016	Definitief 1e versie.

Inhoud

Inleiding—3

1 Namespaces—4
1.1 Target namespace—4
1.2 Default namespace—4
1.3 Prefix—5
1.3.1 elementFormDefault en attributeFormDefault—5
1.3.2 xs-prefix—6
1.3.3 basisschema-prefix—6
1.3.4 bericht-prefixes—7
1.4 Versionering—9
1.5 Voorbeeld—10

2 Design Pattern—11
2.1 Inleiding—11
2.2 Venetian Blind—11
2.3 Basisschema—12

3 Overige richtlijnen—15
3.1 Richtlijnen elementen en attributen—15
3.2 Naamgevingsrichtlijnen—15
3.3 Richtlijnen typedefinities—15
3.4 Annotations—17
3.5 Default en vaste waarden—17
3.6 Substitution Groups en Choice—17
3.7 any en anyAttribute—17
3.8 minOccurs en maxOccurs—17
3.9 Repeterende elementen—18

4 Afspraken bij het opstellen van het xml-bericht—19
4.1 Gebruik Byte-Order-Mark (BOM)—19
4.2 Formatting—19
4.3 Bericht validatie—19
4.4 Lege elementen—19

Inleiding

Vanaf 1 januari 2016 zijn ketenpartijen voor de uitvoering van de Wet Langdurige Zorg (Wlz) verplicht om gebruik te maken van de XML-standaard binnen het iWlz berichtenverkeer. Binnen de andere twee zorgdomeinen, Wet Maatschappelijke Ondersteuning (Wmo) en Jeugdwet (Jw), geldt dit per 12 juni 2017 ook voor het iWmo- en iJw berichtenverkeer. Voor het iPgb berichtenverkeer geldt ook de XML-standaard.

Binnen de iStandaarden worden XML-schemadefinities (XSD's) gebruikt voor het definiëren van berichtdefinities. De W3C-specificaties van XSD's zijn echter omvangrijk en complex waardoor het bijna ondoenlijk is om die tot in detail te kennen. Daarnaast biedt de W3C-standaard geen handreiking op het gebied van *best practices* of richtlijnen voor de implementatie van XML-schemadefinities.

Het doel van dit document is het definiëren van 'high level best practices' en richtlijnen die binnen de iStandaarden gevolgd zullen worden om consistentie binnen alle XSD's te bewerkstelligen. Deze XSD's definiëren gegevenssets die tussen ketenpartners kunnen worden uitgewisseld.

Omdat de W3C-XSD-specificatie zich blijft ontwikkelen en steeds volwassener wordt, blijft dit document zich ontwikkelen. Periodiek werken we dit document daarom bij met de recentste inzichten.

Hebt u vragen over de ontwerprichtlijnen voor XSD's? Neem dan contact op met het Team iStandaarden van Zorginstituut Nederland: info@istandaarden.nl.

1 Namespaces

De volgende paragrafen geven richtlijnen voor het definiëren van namespaces binnen XML-schemadefinities. Nadat de richtlijnen zijn beschreven, wordt een gedeelte van een XML schemadefinitie als voorbeeld gepresenteerd.

Omdat de overgang naar XML niet voor alle iStandaarden (iWlz, iWmo, iJw en iPgb) gelijk is heeft elke iStandaard een eigen namespace. Dit gaat in de toekomst mogelijk veranderen.

1.1 Target namespace

De XML-schemadefinitie moet een "target namespace" definiëren. Deze namespace moet gedefinieerd worden als een URL die dit schema en de definities daarin uniek identificeert.

Vermijd het XML schemadefinities zonder "target namespace" ("chameleon"). Hoewel "chameleon" schema's flexibiliteit bieden, heeft het een negatieve invloed op de performance. De meeste parsers zijn bij "chameleon" schema's namelijk niet in staat om de componenten van het schema te cachen op basis van de namespace. Daarnaast moet er zeer voorzichtig met "chameleon" schema's worden omgegaan om naamconflicten te voorkomen bij het importeren van schemadefinities.

Elke XML-schemadefinitie binnen een iStandaard heeft een eigen namespace. Hiermee worden naamgevingsconflicten voorkomen met schema's die geïmporteerd of "geinclude" worden in andere schema's. De namespace is een URL die wordt opgebouwd uit een hiërarchie, waarin de naam van de iStandaard, het releasenummer (major.minor) en de berichtidentificatie is opgenomen.

Het volledige releasenummer van een iStandaard kent het format major.minor.patch en een optionele toevoeging "-rc" (zie het document Versiebeheer iStandaarden[1]). In de namespace wordt echter alleen het major.minor nummer opgenomen. Dit maakt het mogelijk om gedurende de looptijd van een release kleine verbeteringen ("patches") door te voeren, zonder dat álle XSD's noodzakelijkerwijs aangepast moeten worden.

De root-URL voor de iWlz namespace is http://www.istandaarden.nl/iwlz. Deze wordt gevolgd door een iWlz releasenummer, een iWlz berichtidentificatie en gevolgd door "/schema".

Voorbeeld:

```
targetNamespace="http://www.istandaarden.nl/iwlz/2_0/IO31/schema"

Wlz-release 2.0; berichtindentificatie IO31;
```

De namespace URL kan een locatie betreffen waar de schemadefinitie en –specificaties gepubliceerd zijn, maar dit is niet gegarandeerd.

1.2 Default namespace

De XML-schemadefinitie moet een "default namespace" definiëren die gelijk is aan de "target namespace". Met andere woorden: "XML Schema" (standaard) moet niet

[1] https://www.istandaarden.nl/ibieb/versiebeheer-istandaarden

de "default namespace" zijn.

Voorbeeld:

```
xmlns="http://www.istandaarden.nl/iwlz/2_0/IO31/schema"
```

1.3 Prefix

1.3.1 elementFormDefault en attributeFormDefault

Wanneer een XML-schemadefinitie wordt opgesteld, kunnen er een tweetal attributen worden gedefinieerd (elementFormDefault en attributeFormDefault) waarvan de impact pas zichtbaar is wanneer een XML-bestand conform de XML_schemadefinitie wordt samengesteld.

De <schema> element attributen **elementFormDefault** en **attributeFormDefault** geven aan of voor de lokale elementen en attributen van het XML bestand expliciet moet worden aangegeven in welke namespace deze zijn gedefinieerd. Met de waarde "qualified" wordt aangegeven dat voor de lokale elementen en attributen van het XML bestand expliciet aangegeven moet worden in welke namespace deze zijn gedefinieerd. Dit gaat door middel van een namespace prefix. Default staan deze attributen op "unqualified".

Voor de XML-schemadefinities is gekozen voor de volgende standaard:

- **elementFormDefault** moet op "qualified" staan. Voor alle elementen moet daarmee expliciet aangegeven worden in welke namespace ze zijn gedefinieerd (prefix aangeven). Deze keuze is gemaakt omdat het de leesbaarheid en de performance van het verwerken van de XML-bestanden verbetert.
- **attributeFormDefault** moet op "unqualified" staan. Voor alle attributen mag niet expliciet aangegeven worden in welke namespace ze zijn gedefinieerd (geen prefix). Dit is de defacto standaard voor XML-schemadefinities.

Omdat de default waarde "unqualified" is, wordt het attributeFormDefault attribuut niet gedefinieerd in de XML-schemadefinities. Het elementFormDefault attribuut wordt expliciet op "qualified" gezet.

Bedenk dat aliases voor verwijzing naar een namespace (de prefix) lokaal (in het xml-bericht) worden gedefinieerd. In de voorbeelden is voor de leesbaarheid de verwijzing naar de namespace in het header-element opgenomen en heeft de namespace een prefix gekregen die overeenkomt met het bericht. In de xml voorbeelden in dit document is er voor de leesbaarheid voor gekozen om als alias naar het basisschema altijd de istandaard te kiezen en als alias naar het berichtschema altijd het bericht te kiezen. Hieronder gaat het om een voorbeeld op basis van een aw33-bericht uit de iWlz. De alias van het basisschema is daarom 'iwlz' en is de alias van het berichtschema 'aw33', waardoor de prefixes respectievelijk 'iwlz' (voor elementen uit het basisschema) en 'aw33' (voor elementen uit het berichtschema.

Voorbeeld XSD:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:iwlz="http://www.
istandaarden.nl/iwlz/2_0/basisschema/schema" xmlns:aw310="http://www.
istandaarden.nl/iwlz/2_0/aw310/schema" targetNamespace="http://www.
istandaarden.nl/iwlz/2_0/aw310/schema" elementFormDefault="qualified">
<xs:import namespace="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema"
schemaLocation="basisschema.xsd"/>
…
```

Voorbeeld XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<aw310:Bericht xmlns:aw310="http://www.istandaarden.nl/iwlz/2_0/aw310/schema”
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema”
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<aw310:Clienten>
<aw310:Client>
<aw310:Bsn>12345678910</aw310:Bsn>
<aw310:Geboortedatum>
<iwlz:Datum>1966-03-29</iwlz:Datum>
</aw310:Geboortedatum>
…
</aw310:Client>
</aw310:Clienten>
</aw310:AW310Bericht>
```

1.3.2 *xs-prefix*

Binnen de XML schemadefinitie, moet de namespace prefix voor XML-schema **xs** zijn.

Voorbeeld:

```
<schema targetNamespace=http://www.istandaarden.nl/iwlz/2_0/AW310/schema
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
xmlns:aw310="http://www.istandaarden.nl/iwlz/2_0/AW310/schema”
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2_0/Basisschema/schema”>
```

1.3.3 *basisschema-prefix*

Binnen de XML-schemadefinities van de iStandaarden is er voor gekozen om alle generieke typedefinities (simple en complex) binnen één iStandaard in één XML-schemadefinitie te definiëren. Generiek wil in dit geval zeggen dat de definitie van een type binnen de iStandaard over alle berichten heen gelijk zijn. Er zijn momenteel dus vijf basisschema’s. Een iWmo basisschema, iJw basisschema, iPgb basisschema, iEb basisschema en een iWlz basisschema. In dit laatstgenoemde XML-schemadefinitie worden alle iWlz generieke typedefinities (simple en complex) op een eenduidige manier en centrale plaats gedefinieerd en beheerd. En gelden dus voor alle iWlz berichtdefinities. De typedefinities worden in de schemadefinities van de berichten geïmporteerd en gebruikt. Al deze geïmporteerde generieke iWlz typeberichtdefinities hebben ‘iwlz’ als prefix.

Voorbeeld:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema”
xmlns:aw33="http://www.istandaarden.nl/iwlz/2 0/aw33/schema”
targetNamespace="http://www.istandaarden.nl/iwlz/2_0/aw33/schema”
elementFormDefault="qualified">
<xs:import namespace="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema”
schemaLocation="basisschema.xsd"/>
<xs:element name="Bericht" type="aw33:Root"/>
<xs:complexType name="Root">
<xs:annotation>
<xs:documentation>Bericht voor een zorgtoewijzing (zorgkantoor naar
zorgaanbieder).</xs:documentation>
```

```
</xs:annotation>
<xs:sequence>
<xs:element name="Header" type="aw33:Header"/>
<xs:element name="Clienten" type="aw33:Clienten"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Header">
<xs:annotation>
<xs:documentation>De header bevat de identificerende gegevens van het
bestand.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="BerichtCode">
<xs:annotation>
<xs:documentation>Code ter identificatie van een soort
bericht.</xs:documentation>
</xs:annotation>
<xs:simpleType>
<xs:restriction base="iwlz:LDT_BerichtCode">
<xs:pattern value="352"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:complexType name="Clienten">
<xs:sequence>
<xs:element name="Client" type="aw33:Client" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Client">
<xs:annotation>
<xs:documentation>Persoonsgegevens van de client.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Bsn" type="iwlz:LDT_BurgerServicenummer">
<xs:annotation>
<xs:documentation>Een door de overheid toegekend identificerend nummer in het
kader van het vereenvoudigen van het contact tussen overheid en burgers en het
verminderen van de administratieve lasten.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:complexType>
</xs:schema>
```

1.3.4 *bericht-prefixes*

Iedere XML schemadefinitie van een bericht heeft een eigen namespace (waarin de berichtnaam is opgenomen, bijv.: "http://www.istandaarden.nl/iwlz/2_0/aw33/schema" met een eigen prefix (de afkorting van het bericht, bijv: 'aw33'). Daarmee wordt aangegeven dat de gedefinieerde typen in de XML schemadefinitie van het bericht specifiek zijn voor het betreffende bericht. Tevens worden problemen voorkomen indien typedefinities in de toekomst toch worden geïmporteerd in andere XML-schemadefinities.

Voorbeeld:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2 0/basisschema/schema”
xmlns:aw33="http://www.istandaarden.nl/iwlz/2_0/aw33/schema”
targetNamespace="http://www.istandaarden.nl/iwlz/2_0/aw33/schema”
elementFormDefault="qualified">
<xs:import namespace="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema”
schemaLocation="basisschema.xsd"/>
<xs:element name="Bericht" type="aw33:Root"/>
<xs:complexType name="Root">
<xs:annotation>
<xs:documentation>Bericht voor een zorgtoewijzing (zorgkantoor naar
zorgaanbieder).</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Header" type="aw33:Header"/>
<xs:element name="Clienten" type="aw33:Clienten"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Header">
<xs:annotation>
<xs:documentation>De header bevat de identificerende gegevens van het
bestand.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="BerichtCode">
<xs:annotation>
<xs:documentation>Code ter identificatie van een soort
bericht.</xs:documentation>
</xs:annotation>
<xs:simpleType>
<xs:restriction base="iwlz:LDT_BerichtCode">
<xs:pattern value="352"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:complexType name="Clienten">
<xs:sequence>
<xs:element name="Client" type="aw33:Client" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Client">
<xs:annotation>
<xs:documentation>Persoonsgegevens van de client.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Bsn" type="iwlz:LDT_BurgerServicenummer">
<xs:annotation>
<xs:documentation>Een door de overheid toegekend identificerend nummer in het
kader van het vereenvoudigen van het contact tussen overheid en burgers en het
verminderen van de administratieve lasten.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:complexType>
</xs:schema>
```

1.4 Versionering

Het versiebeheer van de iStandaarden wordt beschreven in een apart document[2]. In deze paragraaf staat beschreven hoe dit zich vertaalt naar de XSD's.

Het major.minor releasenummer van de iStandaard wordt opgenomen in de "default namespace" en de "target namespace". In namespaces is de punt vervangen door een liggend streepje. Dit maakt het mogelijk om in verschillende releases van de iStandaarden, indien nodig, verschillende typedefinities te gebruiken met dezelfde naam.

In 2017 is een documentatie element <appinfo> toegevoegd aan de XSD's waarin naast het releasenummer ook de publicatieversie van het XSD bestand zelf is opgenomen. In 2020 is daar nog aan toegevoegd de minimale en maximale publicatieversie die compatibel zijn met deze publicatieversie.

Wanneer een nieuwe release van iWlz, iWmo, iJw, iPgb en/of iEb van kracht wordt verandert het releasenummer in de namespaces en in het documentatie element <appinfo><release>. Er wordt alleen gesproken van een nieuwe release wanneer het major of minor releasenummer wordt opgehoogd. Bij het ophogen van het patch nummer betreft het bijv. een correctie op een bestaande release (revisierelease).

Wanneer er een nieuwe versie van de XML-schemadefinitie wordt vastgesteld binnen een bestaande release wordt alleen het publicatienummer van het XSD in het documentatie element <appinfo> opgehoogd. Een nieuwe publicatie van een XSD kan het gevolg zijn van een revisierelease, maar ook van een geconstateerde fout in het XSD zelf.

Vanaf de releases die in 2020 in productie gaan, moet het publicatienummer van het XSD dat gebruikt is om het XML bericht op te stellen bovendien worden meegegeven in het XML bestand. De ontvanger van een XML bericht kan op die manier bij uitval van een bericht altijd toetsen of de verzender en ontvanger compatibele versies hebben geimplementeerd.

*Voorbeeld: de komende versie van de iJw is **2.4.0**. Bij de publicatie van de definitieve specificaties krijgen de XSD's publicatienummer **1.0.0.** Indien er een correctie plaatsvindt waarvoor een nieuwe publicatie van het XSD's nodig is, wordt het publicatienummer opgehoogd naar **1.1.0.** en blijft het releasenummer in de namespace dus ongewijzigd.*

Voorbeeld:

```
targetNamespace=http://www.istandaarden.nl/ijw/2_4/jw305/schema

<xs:annotation>
  <xs:appinfo>
    <ijw:standaard>ijw</ijw:standaard>
    <ijw:bericht>jw305</ijw:bericht>
    <ijw:release>2.4</ijw:release>
    <ijw:BerichtXsdVersie>1.0.0</ijw:BerichtXsdVersie>
    <ijw:BerichtXsdMinVersie>1.0.0</ijw:BerichtXsdMinVersie>
    <ijw:BerichtXsdMaxVersie>1.0.0</ijw:BerichtXsdMaxVersie>
```

[2] https://www.istandaarden.nl/ibieb/versiebeheer-istandaarden

```
    <ijw:BasisschemaXsdVersie>1.0.0</ijw:BasisschemaXsdVersie>
    <ijw:BasisschemaXsdMinVersie>1.0.0</ijw:BasisschemaXsdMinVersie>
    <ijw:BasisschemaXsdMaxVersie>1.0.0</ijw:BasisschemaXsdMaxVersie>
  </xs:appinfo>
</xs:annotation>
```

1.5 Voorbeeld

Het volgende XML voorbeeld omvat alle boven genoemde richtlijnen.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2 1/basisschema/schema”
xmlns:aw33="http://www.istandaarden.nl/iwlz/2 1/aw33/schema”
targetNamespace="http://www.istandaarden.nl/iwlz/2_1/aw33/schema”
elementFormDefault="qualified">
<xs:import namespace="http://www.istandaarden.nl/iwlz/2_1/basisschema/schema”
schemaLocation="basisschema.xsd"/>
<aw33:Header>
  <aw33:BerichtVersie>2</aw33:BerichtVersie>
  <aw33:BerichtSubversie>1</aw33:BerichtSubversie>
  <aw33:BerichtIdentificatie>
    <iwlz:Identificatie>3434334rrrt</iwlz:Identificatie>
  </aw33:BerichtIdentificatie>
  <aw33:XsdVersie>
    <iwlz:BasisschemaXsdVersie>1.0.0</iwlz:BasisschemaXsdVersie>
    <iwlz:BerichtXsdVersie>1.0.0</iwlz:BerichtXsdVersie>
  </aw33:XsdVersie>
</aw33:Header>
<aw33:Clienten>
<aw33:Client>
<aw33:Bsn/>
<aw33:Naam>
<iwlz:Geslachtsnaam>
<iwlz:Achternaam>Jansens</iwlz:Achternaam>
</iwlz:Geslachtsnaam>
<iwlz:NaamGebruik>1</iwlz:NaamGebruik>
</aw33:Naam>
</aw33:Client>
</aw33:Clienten>
</aw33:Bericht>
```

2 Design Pattern

2.1 Inleiding

Net als bij softwareontwikkeling, zijn er design patterns voor het ontwerpen van XML-schemadefinities. De meest populaire XML-schemadefinitie patterns zijn Russian Doll, Salami, Bologna, Venetian Blind, en Garden of Eden[3]. Deze patterns hebben allen eigen voor- en nadelen. Voor de iStandaarden XML-schemadefinities is gekozen voor het Venetian Blind design-pattern.

2.2 Venetian Blind

In het Venetian Blind design pattern is er in de XML-schemadefinitie één enkel globaal element gedefinieerd (waarbinnen andere lokale elementen genest zijn). Dit globale element wordt het "root" element genoemd en wordt gedefinieerd in de globale namespace. De lokale elementen worden gedefinieerd op basis van typedefinities (complex en simple).

Voordelen Venetian Blind pattern:

- Herbruikbaarheid wordt gestimuleerd, doordat typedefinities (simple en complex) eenvoudig in andere XML-schemadefinities kunnen worden geïmporteerd.
- Naamgevingsconflicten worden voorkomen, doordat de typedefinities (simple en complex) in een eigen bericht specifieke namespace worden gedefinieerd.
- Omdat er maar één root element is, is er ook maar één geldige vorm van het XML document.

Nadelen Venetian Blind pattern:

- Er zijn geen grote nadelen bij het gebruik van het Venetian Blind pattern.

Het voordeel van herbruikbaarheid in combinatie met slechts één root-element, maakt dat het Venetian Blind pattern als best-practice is aangemerkt voor de XML-schemadefinities.

Voorbeeld:

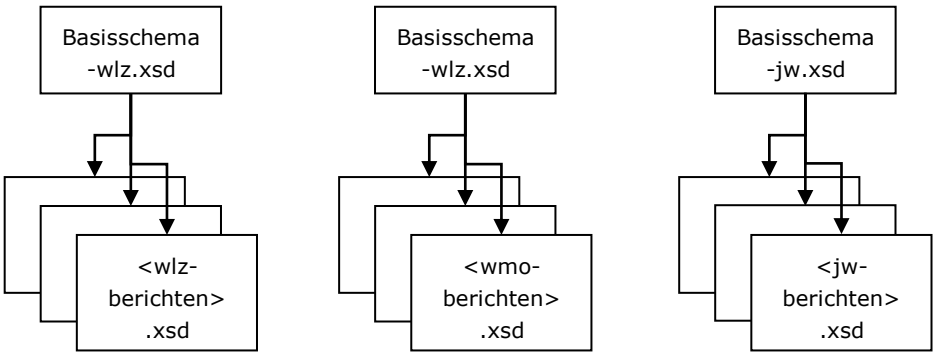
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2 0/basisschema/schema"
xmlns:aw33="http://www.istandaarden.nl/iwlz/2 0/aw33/schema"
targetNamespace="http://www.istandaarden.nl/iwlz/2_0/aw33/schema"
elementFormDefault="qualified">
<xs:import namespace="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema"
schemaLocation="basisschema.xsd"/>
<aw33:Header>
<aw33:BerichtVersie>2</aw33:BerichtVersie>
<aw33:BerichtSubversie>0</aw33:BerichtSubversie>
<aw33:BerichtIdentificatie>
<iwlz:Identificatie>3434334rrrt</iwlz:Identificatie>
</aw33:BerichtIdentificatie>
</aw33:Header>
<aw33:Clienten>
<aw33:Client>
<aw33:Bsn/>
```

[3] http://www.xfront.com/GlobalVersusLocal.html

```
<aw33:Naam>
<iwlz:Geslachtsnaam>
<iwlz:Achternaam>Jansens</iwlz:Achternaam>
</iwlz:Geslachtsnaam>
<iwlz:NaamGebruik>1</iwlz:NaamGebruik>
</aw33:Naam>
</aw33:Client>
</aw33:Clienten>
</aw33:Bericht>
```

2.3 Basisschema

Het Venetian Blind design pattern maakt het mogelijk om eenvoudig typedefinities (complex en simple) in andere XML-schemadefinities te hergebruiken. Binnen de XML-schemadefinities is er voor gekozen om alle generieke[4] typedefinities (simple en complex) in één XML-schemadefinitie te definiëren: het Basisschema. In deze XML-schemadefinitie worden alle generieke typedefinities (simple en complex) op een eenduidige manier en centrale plaats gedefinieerd en beheerd. Daarmee wordt het beheer vereenvoudigd en wordt voor alle berichten een eenduidige definitie van elementen nagestreefd. Het Basisschema wordt middels een "import" statement in de verschillende XML-schemadefinities geïmporteerd. De typedefinities uit het basisschema behouden daarbij een eigen namespace en worden in de XML-schemdefinitie van het bericht gemarkeerd door een iStandaard specifieke prefix. Bijvoorbeeld: Voor het Wlz domein is de prefix: iwlz



Voorbeeld iWlz Basissschema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema"
targetNamespace="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema"
elementFormDefault="qualified">
<xs:annotation>
  <xs:appinfo>
    <iwlz:standaard>iwlz</iwlz:standaard>
    <iwlz:bericht>basisschema</iwlz:bericht>
    <iwlz:release>2.1</iwlz:release>
    <iwlz:BasisschemaXsdVersie>0.1.0</iwlz:BasisschemaXsdVersie>
```

[4] Generiek wil zeggen dat de definitie van het type over alle berichten heen gelijk zijn. Dit geldt onder andere voor veel simple typedefinities in WLZ met codelijsten (toegestane waarden).

```
    <iwlz:BasisschemaXsdMinVersie>0.1.0</iwlz:BasisschemaXsdMinVersie>
    <iwlz:BasisschemaXsdMaxVersie>0.1.0</iwlz:BasisschemaXsdMaxVersie>
  </xs:appinfo>
</xs:annotation>
<xs:complexType name="CDT_Achternaam">
<xs:annotation>
<xs:documentation>De achternaam van een natuurlijk persoon, aangeduid als Naam
met Voorvoegsel.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Achternaam" type="iwlz:LDT_Naam">
<xs:annotation>
<xs:documentation>De achternaam van een persoon, indien nodig verkort
weergegeven.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="Voorvoegsel" type="iwlz:LDT_Voorvoegsel" minOccurs="0">
<xs:annotation>
<xs:documentation>Verzameling van een of meer voorzetsels/lidwoorden, die aan
het significante deel van de achternaam van een persoon vooraf gaat en daar een
onderdeel van is.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:schema>
```

Voorbeeld import en gebruik iWlz-Basisschema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema”
xmlns:aw33="http://www.istandaarden.nl/iwlz/2_0/aw33/schema”
targetNamespace="http://www.istandaarden.nl/iwlz/2_0/aw33/schema”
elementFormDefault="qualified">
<xs:import namespace="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema”
schemaLocation="basisschema.xsd"/>
<xs:element name="Bericht" type="aw33:Root"/>
<xs:complexType name="Root">
<xs:annotation>
<xs:documentation>Bericht voor een zorgtoewijzing (zorgkantoor naar
zorgaanbieder).</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Header" type="aw33:Header"/>
<xs:element name="Clienten" type="aw33:Clienten"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Header">
<xs:annotation>
<xs:documentation>De header bevat de identificerende gegevens van het
bestand.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="BerichtCode">
```

```
<xs:annotation>
<xs:documentation>Code ter identificatie van een soort
bericht.</xs:documentation>
</xs:annotation>
<xs:simpleType>
<xs:restriction base="iwlz:LDT_BerichtCode">
<xs:pattern value="352"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:complexType name="Clienten">
<xs:sequence>
<xs:element name="Client" type="aw33:Client" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Client">
<xs:annotation>
<xs:documentation>Persoonsgegevens van de client.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Bsn" type="iwlz:LDT_BurgerServicenummer">
<xs:annotation>
<xs:documentation>Een door de overheid toegekend identificerend nummer in het
kader van het vereenvoudigen van het contact tussen overheid en burgers en het
verminderen van de administratieve lasten.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:complexType>
</xs:schema>
```

3 Overige richtlijnen

In dit hoofdstuk worden een aantal richtlijnen gepresenteerd die niet in één van de voorgaande categorieën vallen.

3.1 Richtlijnen elementen en attributen

De richtlijn voor het gebruik van elementen en attributen is om elementen te gebruiken voor gegevens die door applicaties worden verwerkt en attributen voor metadata. Binnen de iWlz, iWmo, iJw, iPgb en iEb XML-schemadefinities worden attributen gebruikt om metadata van het betreffende bericht vast te leggen en worden elementen gebruikt om de inhoudelijke data van het bericht vast te leggen.

3.2 Naamgevingsrichtlijnen

De naamgevingsrichtlijnen zijn gebaseerd op de richtlijnen van ebXML (http://www.oasis-open.org/). Onderstaand overzicht toont de vastgestelde naamgevingsrichtlijnen voor de XML-schemadefinities binnen de iStandaarden.

Richtlijn	Omschrijving	Voorbeeld
Hoofdlettergebruik in element- en typedefinities	Element- en typedefinities worden gedefinieerd op basis van *Upper Camel Case*.	*<VoorvoegselClient>*
Hoofdlettergebruik in attribuutdefinities	Attribuutdefinities worden gedefinieerd op basis van *Lower Camel Case*.	*<aw310:AW310Bericht codeExterneIntegratiebericht= "010">*
Acroniemen	Acroniemen worden zoveel mogelijk vermeden, maar wanneer ze worden toegepast dan worden hoofdletters gebruikt.	<clientID>
Consistente naamgeving	Zorg voor een consistente naamgeving.	<WoonAdres> <ContactAdres>
Meervoudsaanduidingen	Gebruik enkel meervoudsaanduidingen voor verzamelingen van elementen.	<Adressen>
Lengte naamgeving	Er is geen beperking op de lengte van de naamgeving. De naamgeving moet zinvol zijn.	<MutatieZorgzwaartepakket>

3.3 Richtlijnen typedefinities

De volgende richtlijnen zijn van toepassing op "simple" en "complex" typedefinities. Typedefinities moeten altijd in de globale namespace worden gedefinieerd en vervolgens gebruikt worden door de lokale elementen. Deze regel ondersteunt het Venetian Blind design pattern. Dit heeft ook tot gevolg dat typedefinities van een naam voorzien moeten worden in plaats van gebruik te maken van anonieme typedefinities.

Voorbeeld:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:iwlz="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema"
xmlns:aw33="http://www.istandaarden.nl/iwlz/2_0/aw33/schema"
targetNamespace="http://www.istandaarden.nl/iwlz/2_0/aw33/schema"
elementFormDefault="qualified">
<xs:import namespace="http://www.istandaarden.nl/iwlz/2_0/basisschema/schema"
schemaLocation="basisschema.xsd"/>
<xs:element name="Bericht" type="aw33:Root"/>
<xs:complexType name="Root">
<xs:annotation>
<xs:documentation>Bericht voor een zorgtoewijzing (zorgkantoor naar
zorgaanbieder).</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Header" type="aw33:Header"/>
<xs:element name="Clienten" type="aw33:Clienten"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Header">
<xs:annotation>
<xs:documentation>De header bevat de identificerende gegevens van het
bestand.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="BerichtCode">
<xs:annotation>
<xs:documentation>Code ter identificatie van een soort bericht.</xs:documentation>
</xs:annotation>
<xs:simpleType>
<xs:restriction base="iwlz:LDT_BerichtCode">
<xs:pattern value="352"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:complexType name="Clienten">
<xs:sequence>
<xs:element name="Client" type="aw33:Client" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Client">
<xs:annotation>
<xs:documentation>Persoonsgegevens van de client.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Bsn" type="iwlz:LDT_BurgerServicenummer">
<xs:annotation>
<xs:documentation>Een door de overheid toegekend identificerend nummer in het kader
van het vereenvoudigen van het contact tussen overheid en burgers en het verminderen
van de administratieve lasten.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:complexType>
</xs:schema>
```

3.4 Annotations

Annotations zijn een special mechanisme om XML-schemadefinities te documenteren. Verschillende tools kunnen op basis van deze ingebedde documentatie overzichtelijke documentatie van de XML-schemadefinitie genereren. De XML-schemadefinities worden zoveel mogelijk met behulp van annotations gedocumenteerd.

Voorbeeld:

```
<element name="Bsn" type="iwlz:LDT_BurgerServicenummer">
<xs:annotation>
<xs:documentation>Een door de overheid toegekend identificerend nummer in
het kader van het vereenvoudigen van het contact tussen overheid en
burgers en het verminderen van de administratieve lasten. Voor meer
informatie over het BSN kunt u terecht bij het Informatiepunt BSN in de
zorg (www.infobsnzorg.nl).
</xs:documentation>
<xs:annotation>
</element>
```

3.5 Default en vaste waarden

Het is een best practice om geen gebruik te maken van zogenaamde default en fixed values omdat er niet gegarandeerd kan worden dat de ontvangende partij het XML-bestand ook daadwerkelijk valideert tegen de XML-schemadefinitie. Wanneer de ontvangende partij dit niet doet zal de default of vaste waarde ook niet verkregen worden. Andersom vult validatie mogelijk onbedoeld een waarde in wanneer het element opzettelijk is leeg gelaten met een default waarde. Er wordt daarom binnen de XML-schemadefinities geen gebruik gemaakt van default en vaste waarden.

3.6 Substitution Groups en Choice

Omdat het Venetian Blind design pattern als best-practice voor de XML-schemadefinities geldt, is er geen mogelijkheid om “substitution groups” te gebruiken. Het gebruik van “substitution groups” vereist namelijk dat alle elementen (kandidaten voor vervanging) in de globale namespace worden gedefinieerd, hetgeen binnen het Venetian Blind pattern niet is toegestaan. In plaats daarvan moet de “Choice” optie worden gebruikt om toegestane waarden te definiëren.

3.7 any en anyAttribute

Het “any” element en “anyAttribute” attribuut zijn in de standard gedefinieerd om een bepaalde mate van vrijheid te ondersteunen. In plaats van deze vrijheid op deze manier te ondersteunen, is het binnen de XML schemadefinities de richtlijn om nieuwe versies van XML-schemadefinities op te stellen en oude interfaces (waar nodig) voor een bepaalde tijd te ondersteunen.

n.b. momenteel staat dit ter discussie voor de toekomst, om backwards compatibiliteit te ondersteunen.

3.8 minOccurs en maxOccurs

De default waarde voor de minOccurs en maxOccurs attributen is 1. Voor de overzichtelijkheid is het binnen de XML-schemadefinities de richtlijn om de schemadefinities niet onnodig te vervuilen met deze attributen indien de default waarde wordt gebruikt. Concreet betekent dit dat alleen wanneer een element optioneel is minOccurs = “0” is meegegeven en wanneer het is toegestaan dat en element meerdere keren aangeleverd mag worden maxOccurs = “unbounded” is gedefinieerd.

3.9 Repeterende elementen

Wanneer het is toegestaan om meerdere voorkomens van hetzelfde element op te nemen in het xml-bericht, zullen deze elementen binnen een gelijknamige in meervoudsvorm gespecificeerde collectie worden ondergebracht.

Voorbeeld:

```
<xs:complexType name="RetourCodes">
<xs:sequence>
<xs:element name="RetourCode" type="iwlz:LDT_RetourCode" maxOccurs="unbounded">
<xs:annotation>
<xs:documentation>Gecodeerde aanduiding in een retourbericht van het resultaat van
de beoordeling van een (deel van het) ontvangen bericht.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Clienten">
<xs:sequence>
<xs:element name="Client" type="aw34:Client" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="Client">
<xs:annotation>
<xs:documentation>Persoonsgegevens van de client.</xs:documentation>
</xs:annotation>
<xs:sequence>
<xs:element name="Bsn" type="iwlz:LDT_BurgerServicenummer">
<xs:annotation>
<xs:documentation>Een door de overheid toegekend identificerend nummer in het kader
van het vereenvoudigen van het contact tussen overheid en burgers en het verminderen
van de administratieve lasten.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="RetourCodes" type="aw34:RetourCodes"/>
</xs:sequence>
</xs:complexType>
</xs:schema>
```

4 Afspraken bij het opstellen van het xml-bericht

Hoewel dit document de richtlijnen beschrijft op basis waarvan de xml-schemadefinities zijn opgesteld, volgen in dit hoofdstuk nog enkele aanbevelingen voor het opstellen van de bijbehorende xml-berichten.

4.1 Gebruik Byte-Order-Mark (BOM)

Het toevoegen van een Byte-Order-Mark aan een XML-bericht is niet toegestaan.

Een Byte-Order-Mark is bedoeld om bij het gebruik van UTF-16 encoding aan te kunnen geven wat de gehanteerde byte volgorde is om tot de juiste encoding van de character-set te komen. Deze BOM staat altijd vooraan in de tekenreeks.

Omdat de afgesproken encoding van de xml-berichten 'UTF-8' is, is het niet nodig om een BOM mee te geven. Daarnaast is er gebleken dat het gebruik van een BOM binnen een XML-bericht tot problemen kan leiden bij de verwerking van dat XML document.

Met name Microsoft producten geven standaard wel een BOM mee bij gebruik van 'UTF-8' encoding.

De afspraak is in het Informatiemodel iStandaarden vastgelegd in bedrijfsregel OP192.

4.2 Formatting

Een XML-bericht is 'well-formed' indien deze voldoet aan de syntax-regels. Deze regels beschrijven niet hoe een XML-bericht geformatteerd moet zijn. Wanneer een XML-bericht well-formed is, is het toegestaan om de inhoud van het bericht op één regel te zetten.
Voor de leesbaarheid heeft het de voorkeur dit niet te doen en het bericht over meerdere regels te spreiden en gebruik te maken van inspringen. De consequentie is echter dat het bericht groter wordt door het toevoegen van non-essential characters. Het einderegel teken is het Windows einde-regel teken (CR & LF) en ieder element wordt afgesloten met het einde-regel-teken. Zie bedrijfsregel OP192.

4.3 Bericht validatie

Berichten kunnen worden gevalideerd tegen de bijbehorende XML-schemadefinitie (xsd) en XSL-Transformaties (xslt) voordat deze worden aangeboden en wanneer deze worden ontvangen. Indien een bericht wordt afgekeurd op basis van een XSLT controle wordt in de header van het retourbericht het element XsltVersie opgenomen met het versienummer uit het resultaat van de XSLT.

4.4 Lege elementen

Zorginstituut Nederland adviseert om lege elementen niet op te nemen in het XML-bericht. Vanaf de releases iWlz 2.0, iWmo 2.2, iJw 2.2 en iPgb 2.0 wordt dit technisch afgedwongen en mogen lege elementen in het geheel niet voorkomen in een XML-bericht. Elk element dat in het bericht is opgenomen, moet een waarde hebben.