< IH Berichttransport

Datum: 7 oktober 2020
Publicatie: V8.2.0.0

Inhoudsopgave

1 Inleiding

1.1 Doel en scope

Dit document specificeert het berichttransport over SOAP/HTTP zoals dat in AORTA wordt toegepast. Het berichttransport vormt de verbinding tussen de HTTP(S)-laag, en de er bovenliggende HL7v3-berichten. Dit document beschrijft:

- hoe HL7v3-interacties worden verpakt in SOAP voor transport op basis van webservices;
- hoe SOAP op hoofdlijnen wordt toegepast binnen AORTA;
- hoe webservices worden gedefinieerd met behulp van WSDL;
- de toepassing van HTTP(S)-transport in relatie tot de webservices;
- versionering van webservices en end point uri's;
- betrouwbaarheidsaspecten van dat transport.

1.2 Doelgroep voor dit document

Dit document is bedoeld voor softwareontwikkelaars van zorgtoepassingen en de Zorg Informatie Makelaar (ZIM), die op grond van de HL7v3-communicatiestandaard en dit document berichten in AORTA willen transporteren.

De lezer wordt verondersteld kennis te hebben van [XML], [SOAP] en [WSDL]. Lezing van de AORTA-documentatie wordt ten zeerste aanbevolen.

1.3 Documenthistorie

Versie	Datum	Omschrijving
6.10.0.0	12-okt-2011	Initiële versie na herstructurering AORTA-documentatie. RFC 46142: Wijziging in verband met elektronische handtekening: gebruik SOAP Header attributen (§4.3). RFC 46691: HTTP Fouten doorgeven door ZIM verduidelijkt in Par 4.5 en 4.5.4.
6.10.0.0	2-apr-2012	RFC 51940: toegestane waarden voor faultactor opgenomen in Par 4.5.2.
6.11.0.0	5-dec-2012	Herpublicatie als onderdeel van AORTA-Infrastructuur v6·11
6.12.0.0	13-aug-2013	RfC 52089: Fout met ZIM als tussenstation verduidelijkt.
8.0.1.0	15-mei-2017	RfC 52477: Uitwisseling op basis van bouwstenen
8.0.1.0	15-mei-2017	RfC 76206: SSL verwijderen
8.0.3.0	15-nov-2018	Opgenomen in publicatie 8.0.3.0
8.1.0.0	1-aug-2019	INI-8877: Toevoegen inschrijftoken t.b.v. conditionele query
8.2.0.0	7-okt-2020	Opgenomen in publicatie 8.2.0.0

1.4 Legenda

Dit document gebruikt de volgende symbolen:

	Let op! Dit is een aandachtpunt. Een opmerking die de aandacht vestigt op een bepaald opvallend aspect.
	Dit is een 'open issue' of 'known issue'. Een kwestie die nog open ligt voor discussie, maar onderkend is.
	Dit is een frequently asked question (FAQ) met antwoord.

1.5 Structuur

Dit document is als volgt opgebouwd:

- Inleiding (niet normatief).
- Berichtuitwisselingsstandaarden (niet normatief): inleiding in de relevante gebruikte standaarden.
- Informatiestromen (niet normatief): een overzicht van berichtuitwisseling zoals gebruikt bij het landelijk EPD.
- XML, SOAP en HTTP (normatief): richtlijnen voor het gebruik. Voor GBx'en geldt dat §4.3 alleen verplicht is bij ondersteuning van elektronische handtekening[1].
- AORTA en WSDL (normatief): algemene toelichting bij de WSDL bestanden.
- Betrouwbaar transport (normatief).

Normatief betekent dat het verplicht is om te conformeren aan de specificaties. De niet normatieve hoofdstukken zijn bedoeld als achtergrondinformatie.

Noot: dit document gebruikt veel voorbeelden van XML materialen (WSDL, XSD, XML), de XML materialen die AORTA publiceert zijn mogelijk recenter, en leidend.

[1] De elektronische handtekening wordt binnen AORTA vooralsnog niet toegepast.

2 Berichtuitwisselingsstandaarden

Het transport van documenten in AORTA is gebaseerd op een aantal standaarden: SOAP, WSDL, WS-I Basic Profile, Web Service Security (WSS) en relevante delen van de HL7v3-standaard (met name Transmission Infrastructure en Query Infrastructure). Dit hoofdstuk licht deze standaarden kort toe.

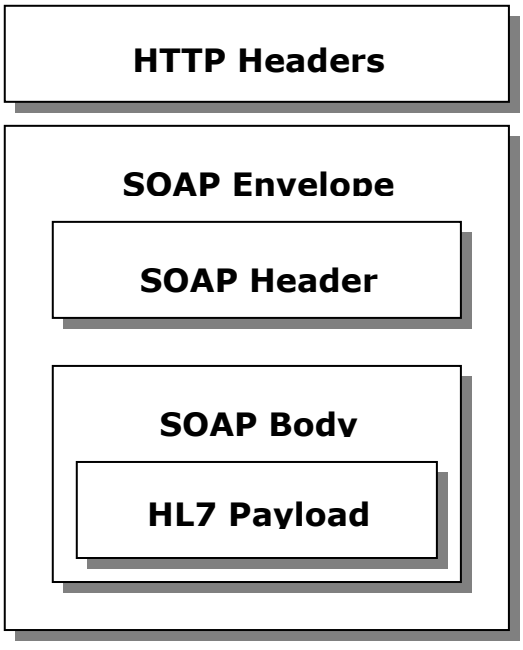
2.1 SOAP

SOAP 1.1 – Simple Object Access Protocol – is een "de facto standaard" voor uitwisseling van gegevens over – onder andere - internet. AORTA gebruikt SOAP 1.1.

Een SOAP-document is een XML-document. Het bestaat uit drie onderdelen:

1. De SOAP Envelope is het omringende deel.
2. De SOAP Header is een optionele verzameling headers die bijvoorbeeld meta-informatie over het bericht kunnen bevatten. Eventuele tokens, zoals bijvoorbeeld voor authenticatie, worden opgenomen in de SOAP Header.
3. De SOAP Body bevat het eigenlijke bericht. De Body bevat de "data", de Header de bijbehorende "metadata".

SOAP kent een mapping op HTTP, waarbij HTTP als transportmechanisme gebruikt wordt om een SOAP-document te transporteren. Zie Figuur 1 voor een overzicht van een SOAP bericht.



Figuur 1 - Overzicht SOAP bericht

De verschillende onderdelen zien er als volgt uit.

```
<?xml version="1.0" encoding="utf-8"?>
<Envelope ... namespace declaraties ... >
  <Header>
```

```
    ... headers ...
  </Header>
  <Body>
    ... payload ...
  </Body>
</Envelope>
```

De SOAP Envelope is het omsluitende XML-element. Dit element bevat de SOAP Headers en de SOAP Body. Er hoeven geen headers aanwezig te zijn; een SOAP Body is wel verplicht. SOAP Headers worden meestal gebruikt voor meta-informatie die te maken heeft met authenticatie, beveiliging, transactiemanagement of betalingen. De Body bevat de "payload", de eigenlijke gegevens die verzonden worden.

2.2 WSDL

WSDL– Web Services Description Language – is een taal om Web Services te beschrijven. Het is een hulpmiddel en het vereenvoudigt het bouwen van interoperabele Web Services. WSDL 1.1 is een W3C Note, die als "de facto" standaard geldt. AORTA gebruikt WSDL 1.1.

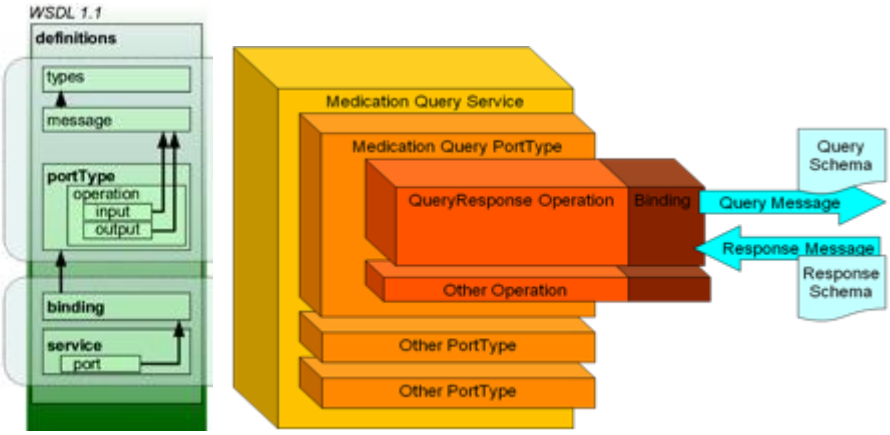
WSDL dient de volgende doelen:

- Eenduidige en machine leesbare beschrijving van de Web Service.
- Invoer voor een codegenerator die een deel van de voor de Web Service benodigde code genereert.

Een WSDL document bestaat uit de volgende onderdelen:

- Types. Hier worden abstracte XML-structuren beschreven in XML Schema formaat.
- Messages. Met de elementen en elementtypen uit de Types worden Messages samengesteld – dit zijn generieke berichten gedefinieerd in XML.
- PortTypes. Een PortType is een logische "applicatiepoort" die één of meer Operations ondersteunt. Een Operation bestaat uit één of meer Inputs en Outputs. Deze Inputs en Outputs verwijzen naar de Messages die hierboven beschreven zijn. Een Message beschrijft dus de interne structuur van één of meer Inputs of Outputs.
- Bindings. Een Binding verbindt een – logisch – PortType met een specifiek transportprotocol, zoals SOAP over HTTP. Alle Operations, Inputs en Outputs uit de Port Type worden toegewezen aan SOAP.
- Services. Een Service bestaat uit één of meer fysieke Ports. Een Port verbindt een Binding met een locatie, normaal gesproken een URI waarmee de Port over het Internet aangeroepen kan worden.

Twee manieren van een grafische weergave van de onderdelen van een WSDL zijn opgenomen in Figuur 2.



Figuur 2 – Twee grafische weergaven van wsdl onderdelen

2.3 WS-I Basic Profile

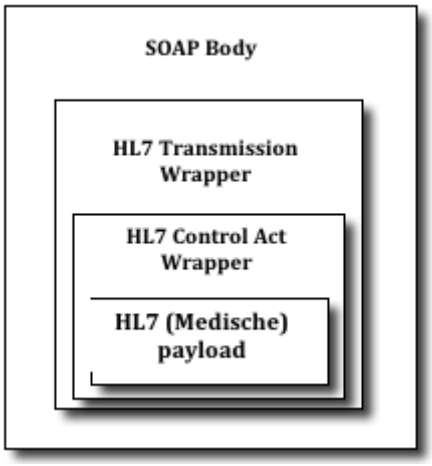
WS-I Basic Profile 1.0 is een specificatie gepubliceerd door WS-I, een ad hoc consortium van voornamelijk softwareleveranciers. Het Basic Profile beschrijft hoe SOAP 1.1 en WSDL 1.1 gebruikt kunnen worden om interoperabele Web Services te bouwen en geeft richtlijnen over hoe SOAP en WSDL het beste gebruikt kunnen worden. Op deze wijze corrigeert het Basic Profile veel van de tekortkomingen van SOAP en WSDL. AORTA volgt het Basic Profile tenzij anders aangegeven.

2.4 HL7v3

HL7v3 kent verder een aantal onderdelen die te maken hebben met het transport van gegevens. HL7v3 definieert de volgende onderdelen:

- In de Transmission Wrapper worden metagegevens opgeslagen die te maken hebben met het bericht zelf: id, datum waarop het is aangemaakt, afzender, geadresseerde en dergelijke.
- De Trigger Event Control Act Wrapper bevat gegevens over de gebeurtenis die aanleiding was het bericht te versturen. Ook zijn hierin gegevens opgenomen die te maken hebben met de verwerking van het bericht, bijvoorbeeld het aantal records dat een query terug moet sturen (er zijn veel specialisaties van deze wrapper).
- De Batch Wrapper kan gebruikt worden om meerdere berichten van dezelfde of verschillende soort in te pakken.

Tezamen met de eigenlijke (medische) gegevens, de payload, vormen deze wrappers een XML-document. De inhoud van de SOAP Body is weergegeven in Figuur 3.



Figuur 3 - Inhoud SOAP Body

3 Informatiestromen

3.1 AORTA en gegevensuitwisseling

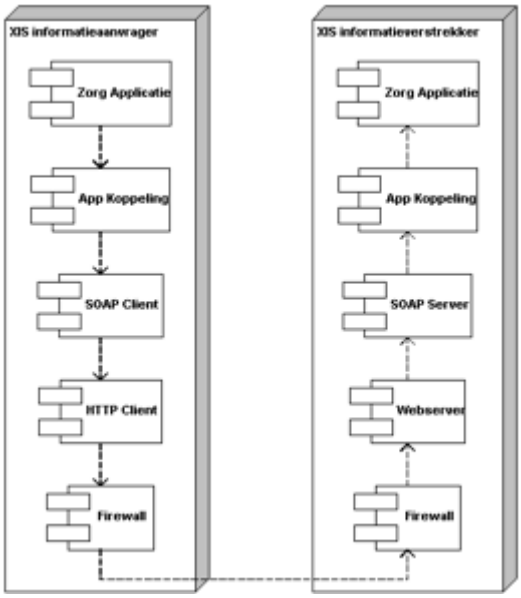
In AORTA vindt gegevensuitwisseling plaats op basis van SOAP/HTTP over met TLS beveiligde verbindingen en hierbij zijn webservices beschreven in WSDL.

WSDL is, strikt gesproken, optioneel. Veel tools genereren op basis van WSDL een deel van de code. Het is echter ook mogelijk de webservice te bouwen zonder WSDL, en in die zin is gebruik van WSDL optioneel.

3.2 Globaal overzicht informatiestroom

Een SOAP-interactie kent een client en een server. De SOAP client stuurt een request naar de server via HTTP, en ontvangt het antwoord via HTTP.

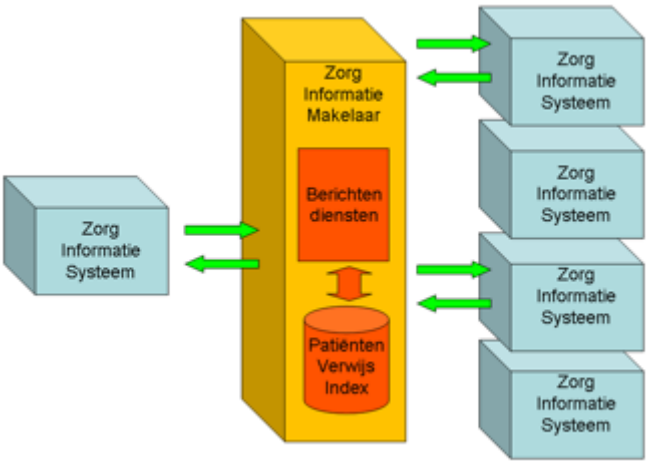
Een dergelijke interactie stelt verschillende eisen aan de client en de server. Een client hoeft alleen een verbinding te hebben en beschikbaar te zijn op het moment dat de aanroep gedaan wordt. De server zal echter continu moeten luisteren op een poort of er een aanroep binnenkomt. De pijlen in Figuur 4 geven het verloop van een SOAP request aan. Een response volgt de omgekeerde weg.



Figuur 4 - AORTA interactie

3.3 SOAP en de ZIM

In bovenstaande beschrijving is alleen gesproken over berichtenuitwisseling tussen een client en een server. In AORTA worden berichten echter grotendeels uitgewisseld tussen een client en server via de Zorg Informatie Makelaar (ZIM), zie Figuur 5.



Figuur 5 - Berichtenuitwisseling via de ZIM

Voorbeeld van AORTA berichtenuitwisseling.

Een ziekenhuis vraagt een overzicht van medicijnverstrekkingen voor een patiënt op. De opvraging van het ziekenhuis is geadresseerd aan de ZIM. De ZIM is verantwoordelijk voor het opzoeken van de applicaties die deze gegevens kunnen leveren en zendt de opvraging door aan één of meer applicaties. De ZIM zal de adressering van de berichten aanpassen: deze doorgezonden opvragingen zijn geadresseerd aan de zorginstellingen die over de gevraagde medicatiegegevens beschikken. Alle berichten zijn SOAP berichten. In SOAP client/server terminologie betekent dit het volgende. Het opvragende, initiërende zorg informatiesysteem is client, de ZIM fungeert als server voor dit systeem. Wanneer de ZIM de opvragingen doorstuurt is de ZIM client en zijn de reagerende zorg informatiesystemen die de gevraagde gegevens opleveren server. De ZIM is vervolgens weer, als server, verantwoordelijk voor het doorsturen van de antwoorden aan het initiërende ziekenhuis.

Voorbeeld van AORTA berichtenuitwisseling o.b.v. context.

Een ziekenhuis vraagt een overzicht van medicijnverstrekkingen voor een patiënt op. In het opvraagbericht wordt de juiste contextcode opgenomen behorende bij “het opvragen van een overzicht van medicijnverstrekkingen”. De opvraging van het ziekenhuis is geadresseerd aan de ZIM. De ZIM is verantwoordelijk voor het opzoeken van de applicaties die deze gegevens kunnen leveren. Hiervoor zoekt de ZIM de juiste bouwstenen op behorende bij de context en bepaalt welke bouwsteeninteracties verzonden moeten worden naar welke bronsystemen. Het is mogelijk dat een bronsysteem meerdere bouwsteeninteracties ontvangt. Alle berichten zijn SOAP berichten. In SOAP client/server terminologie betekent dit het volgende. Het opvragende, initiërende zorg informatiesysteem is client, de ZIM fungeert als server voor dit systeem. Wanneer de ZIM de opvragingen doorstuurt is de ZIM client en zijn de reagerende zorg informatiesystemen die de gevraagde gegevens opleveren server. De ZIM is vervolgens weer, als server, verantwoordelijk voor het doorsturen van de antwoorden aan het initiërende ziekenhuis. Het doorsturen van de ontvangen antwoorden gebeurt met behulp van een batchwrapper. Alle antwoorden zijn hiermee opgenomen in één antwoordbericht.

4 XML, SOAP en HTTP

4.1 XML

Berichten worden uitgewisseld in XML 1.0.

UTF-8 is voor XML de meest gangbare Unicode-encoding. Encoding wordt aangegeven in een XML-prolog:

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

Berichten worden uitgewisseld en gelogd in UTF-8 encoding.

4.2 SOAP berichtformaat

Berichten worden verpakt volgens de SOAP 1.1 standaard.

Het HL7v3-bericht wordt opgenomen in de SOAP Body van de SOAP Envelope.

Hieronder staat een voorbeeld van een HL7v3-bericht dat is verpakt in een SOAP Envelope. Het betreft een opvraag van medicatieverstrekkingen. De voorbeelden in dit document gaan uit van een opvraag met behulp van een zorgtoepassing specifieke interactie. De opbouw van een generieke interactie is in grote lijnen hetzelfde. Eventuele afwijkingen ten opzichte van de voorbeelden worden in het hoofdstuk besproken.

```
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <QURX_IN990111NL xmlns="urn:hl7-org:v3">
      <id extension="0032616767" root="2.16.840.1.113883.2.4.6.2.451.12.21"/>
      <creationTime value="20040910170245"/>
      ... andere elementen van de Transmission Wrapper...
      <ControlActProcess moodCode="EVN">
        <subject>
          <QueryByParameterPayload>
          ...  verdere inhoud ...
          </QueryByParameterPayload>
        </subject>
      </ControlActProcess>
    </QURX_IN990111NL>
  </soapenv:Body>
</soapenv:Envelope>
```

Het voorbeeld is hieronder in stukken geknipt en van commentaar voorzien.

De prolog met UTF-8 declaratie.

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

SOAP Envelope met daarin namespace declaraties.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

SOAP Body met daarin het omsluitende element van de opvraging van medicatieverstrekkingen. Omdat dit element de HL7v3-namespace tot default namespace maakt, maken alle elementen daarin automatisch deel uit van de HL7v3-namespace.

```
  <soapenv:Body>
    <QURX_IN990111NL xmlns="urn:hl7-org:v3">
```

Elementen uit de Transmission Wrapper van HL7v3.

```
      <id extension="0032616767" root="2.16.840.1.113883.2.4.6.2.451.12.21"/>
      <creationTime value="20040910170245"/>
      ... andere elementen van de Transmission Wrapper...
```

De Control Act Wrapper.

```
      <ControlActProcess moodCode="RQO">
        <subject>
```

Het feitelijke bericht, een opvraging naar medicatie van een bepaald PatientId.

```
          <QueryByParameterPayload>
          ...  verdere inhoud ...
          </QueryByParameterPayload>
```

Afsluiting van de onderdelen.

```
        </subject>
      </ControlActProcess>
    </QURX_IN990111NL>
  </soapenv:Body>
</soapenv:Envelope>
```

4.3 SOAP Header attributen

Er zijn verschillende soorten SOAP Headers mogelijk. De beveiliging implementatiehandleidingen [IH BA DigiD]
[IH BA PKIO-pas][IH BA Transactietoken][IH BA Mandaattoken][IH BA Inschrijftoken]
[IH tokens generiek] beschrijven deze headers. Deze paragraaf beschrijft de mogelijke attributen van SOAP-headers.

SOAP 1.1 kent twee attributen voor headers, namelijk “actor” en “mustUnderstand”. De hierna volgende paragrafen beschrijven deze achtereenvolgens.

4.3.1 SOAP actor

SOAP headers kunnen verschillende bestemmingen hebben. AORTA kent de Zorg Informatie Makelaar (ZIM) en het Goed Beheerde Informatiesysteem (GBx), zie [Arch AORTA] voor een uitleg van deze begrippen.

Het attribuut soap:actor geeft aan welke "actor" een bepaalde header moet verwerken. Met de soap:actor wordt in AORTA een functie aangegeven: bijvoorbeeld de ZIM of het GBx dat als eindbestemming dient. De soap:actor geeft in AORTA dus geen adres van een specifiek GBx.

```
<soap:Header xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    ...
    <wss:Security xmlns:wss=
        "http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd"
        soap:actor="http://www.aortarelease.nl/actor/gbx" soap:mustUnderstand="1">
        ...
    </wss:Security>
</soap:Header>
```

Voor de headers voor tokenauthenticatie en andere headers gericht aan de ZIM wordt aangeraden de waarde http://www.aortarelease.nl/actor/zim te gebruiken. Voor headers gericht op een GBx is de waarde http://www.aortarelease.nl/actor/gbx verplicht.

```
<ao:authenticationTokens ... soap:actor="http://www.aortarelease.nl/actor/zim">
<wss:Security ... soap:actor="http://www.aortarelease.nl/actor/gbx">
```

Andere waarden voor actor zijn niet toegestaan. De ZIM dient een header gericht aan een GBx ongewijzigd door te geven aan het GBx van bestemming.

4.3.2 SOAP mustUnderstand

Het SOAP mustUnderstand attribuut geeft aan of de ontvanger de header moet verwerken. Dit geldt voor de bestemming (actor) van de header, zoals beschreven in voorgaande paragraaf 4.3.1. Het mustUnderstand attribuut werkt dus samen met het actor attribuut. Een ontvanger die niet de bestemming van de header vertegenwoordigt, dus niet de actor is, verwerkt de header sowieso niet. De ZIM moet een header zonder actor interpreteren alsof die voor de ZIM bedoeld is.

De waarde van het mustUnderstand attribuut kan “1” of “0” zijn. Afwezigheid van het mustUnderstand attribuut is gelijk aan waarde “0”.

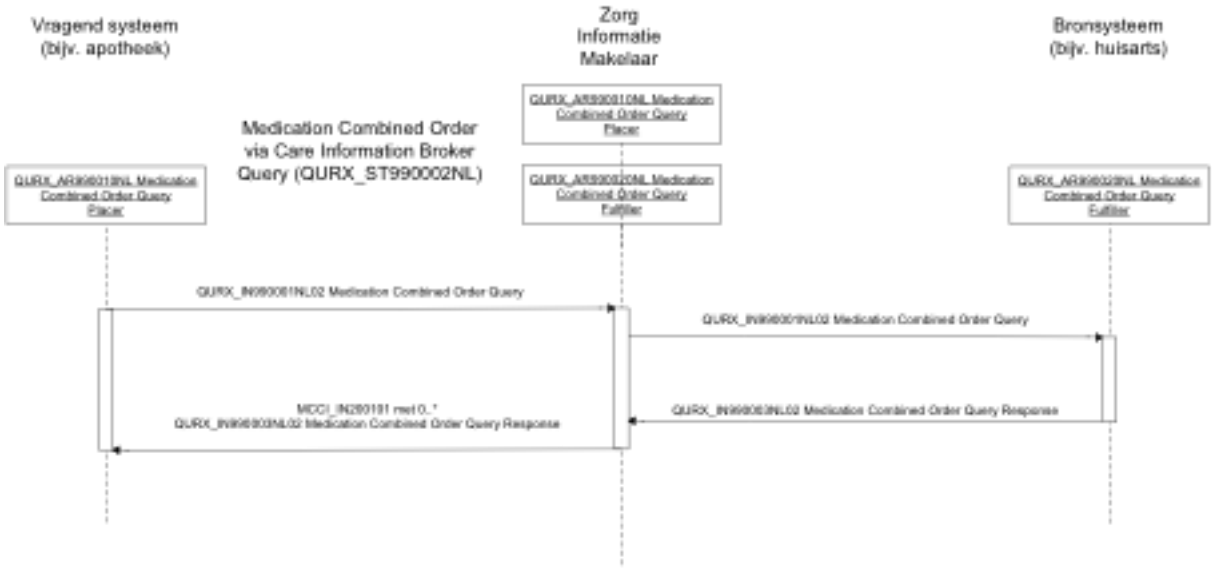
Als de header een SOAP mustUnderstand attribuut bevat met waarde “1”, dan is de actor verplicht om deze header correct te verwerken òf moet de actor het gehele bericht weigeren met een SOAP:MustUnderstand fout, zie paragraaf 4.5.2.

Als de header een SOAP mustUnderstand attribuut bevat met waarde “0”, dan mag de actor deze header verwerken, maar het hoeft niet. Het is dan niet toegestaan om een SOAP mustUnderstand fout te genereren ten gevolge van deze header.

4.4 HTTP Binding

HTTP voegt aan een te transporteren document – bijvoorbeeld een SOAP-document – een aantal HTTP Headers toe. HTTP uitwisseling vindt altijd in paren plaats – een HTTP request wordt beantwoord met een HTTP response. HTTP kent een aantal verschillende methoden – zoals GET (opvragen van een resource), PUT (afleveren van een resource) en POST (wijzigen van een resource). Voor SOAP is alleen POST van belang.

SOAP 1.1 definieert een binding op HTTP waarbij SOAP requests en responses met HTTP POST worden uitgewisseld. Daarbij wordt het SOAP request als HTTP request met de method POST verzonden naar een server, die (behoudens fouten) een SOAP response, verpakt in een HTTP response, terugzendt als antwoord op de POST. Hieronder een voorbeeld van een SOAP request – response over HTTP. Dit voorbeeld is gebaseerd op het volgende interactiediagram, een fragment van "medicatieverstrekkingen opvragen".



Figuur 6 – Interactiediagram "medicatieverstrekkingen opvragen"

Hieronder het verzoek, dus zowel het HTTP als het SOAP request, als de HL7v3-interactie QURX_IN990111NL (Medication Dispense List Query).

```
POST /VerstrekkingsLijstquery HTTP/1.1
Host: www.zim.nl
SOAPAction: "urn:hl7-org:v3/VerstrekkingsLijstquery_QueryResponse"
Connection: Close
Content-Length: 1874
Content-Type: text/xml; charset=UTF-8

<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <QURX_IN990111NL xmlns="urn:hl7-org:v3">

        ... inhoud ...
```

```
    </QURX_IN990111NL>
  </soapenv:Body>
</soapenv:Envelope>
```

Eerst wordt de HTTP methode genoemd (POST), en de versie, dan volgen een aantal headers met HTTP gegevens. SOAP 1.1 verplicht het noemen van "text/xml" als Content-Type.

Merk op: de HTTP Header "Content-Type" moet waarde "text/xml" bevatten.

SOAP 1.1 verplicht het gebruik van een SOAPAction HTTP Header bij SOAP over HTTP. De quotes eromheen zijn verplicht volgens het WS-I Basic Profile, om te voorkomen dat een header met en zonder quotes als verschillend geïnterpreteerd kunnen worden. Een HTTP Server kan deze SOAPAction gebruiken om het verkeer binnen een Goed Beheerd Zorgsysteem (GBZ), Goed Beheerd Patiëntenportaal (GBP), Goed Beheerd Klantenloket (GBK) of ZIM verder te routeren naar de verwerkende applicatie zonder het SOAP bericht zelf te hoeven openen.

Hieronder een voorbeeld van de HTTP response op de POST methode. Dit is tevens de SOAP response, en de HL7v3-interactie QURX_IN990113NL (Medication Dispense List Query Response):

```
HTTP/1.1 200 OK
Content-Length: 7904
Content-Type: text/xml;charset=utf-8
Date: Wed, 03 Feb 2010 12:59:18 GMT
Server: Apache-Coyote/1.1

<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <QURX_IN990113NL xmlns="urn:hl7-org:v3">

    ... inhoud ...

    </QURX_IN990113NL>
  </soapenv:Body>
</soapenv:Envelope>
```

Eerst wordt de HTTP-versie aangegeven en de status (200 OK). Vervolgens weer enige HTTP Headers en dan de SOAP Envelope.

Het is belangrijk in te zien dat er requests / responses zijn op drie verschillende niveaus:

- een HTTP request / response paar
- een SOAP request / response paar
- een HL7v3-interactie en een HL7v3-antwoordinteractie.

Deze drie niveaus zijn in principe onafhankelijk van elkaar. In dit voorbeeld is er voor gekozen het HL7v3-antwoord in de SOAP response te verpakken, en die weer in de HTTP response. Dit is echter niet noodzakelijk. Het is bijvoorbeeld heel goed mogelijk de HL7v3-interactie in een apart HTTP request/response paar te verpakken, en het HL7v3-antwoord in een tweede HTTP request/response paar. De HTTP Binding van SOAP bepaalt de manier waarop een SOAP request en response op HTTP afgebeeld worden. De SOAP Binding van HL7v3 bepaalt de wijze waarmee een HL7v3-request en -response (bijvoorbeeld een QueryContinuation en QueryResponse) afgebeeld worden op SOAP.

In AORTA is er, behoudens HTTP- en SOAP-fouten, altijd een HTTP Response met daarin een SOAP Body die een HL7v3-interactie bevat. Een SOAP request/response paar wordt dus gebonden op één HTTP request/response paar en er wordt voor deze response altijd een HL7v3-interactie gemodelleerd.

4.5 Bij HTTP- en SOAP-fouten kan niet altijd een HL7v3-bericht en/of SOAP response worden geconstrueerd.HTTP errors en SOAP Faults

Foutafhandeling kan plaatsvinden op diverse niveaus. Fouten op TCP-niveau of lager in de stack (IP tot en met fysiek medium) worden hier niet besproken. Relevant zijn fouten op de volgende niveaus:

- HTTP;
- SOAP;
- HL7v3.

Bij alle fouten moet onderscheid gemaakt worden tussen foutsituaties waarbij de fout ligt bij het bericht dat ingezonden wordt (client fouten) en de server (server fouten). Wanneer er iets mis is met het bericht (volgens de ontvanger), is het niet zinvol hetzelfde bericht later nogmaals te sturen: dat zal tot dezelfde fout leiden. Wanneer er iets mis is met de server, is dat vaak tijdelijk. Dan is later opnieuw proberen wel zinvol.

Protocol	Client fouten	Server fouten
HTTP	HTTP statuscode in de range 4xx	HTTP statuscode in de range 5xx
SOAP	soap:Client faults	soap:Server faults
HL7v3	Error met typecode CE (Commit Error)	Error met typecode CR (Commit Reject)

Bij alle fouten heeft het de voorkeur in beschikbare tekst elementen (zoals SOAP:detail of HTTP Body) meer informatie op te nemen omtrent de fout.

4.5.1 HTTP-foutsituaties

AORTA volgt de door WS-I Basic Profile gedefinieerde richtlijnen voor HTTP-fouten bij gebruik van SOAP.

De meest relevante richtlijnen zijn opgenomen in onderstaande tabel.

Code	Basic Profile tekst	Toelichting
R1140	A MESSAGE SHOULD be sent using HTTP/1.1.	
R1107	A RECEIVER MUST interpret SOAP messages	Een applicatie mag dus niet alleen naar HTTP

	containing only a soapenv:Fault element as a Fault.	200 OK kijken voor het bepalen van het succes. Hetzelfde geldt voor HL7v3-fouten: als het HL7v3-bericht een foutsituatie aangeeft, is het een fout, ook al is de HTTP status 200 OK.
R1111	An INSTANCE SHOULD use a "200 OK" HTTP status code for responses that contain a SOAP message that is not a SOAP fault.	Aanbevolen wordt om alleen HTTP 200 OK te gebruiken als HTTP succes code.
R1124	An INSTANCE MUST use a 2xx HTTP status code for responses that indicate a successful outcome of a request.	Ontvangers dienen HTTP responses met andere 2xx succes statussen wel te kunnen accepteren.
R1125	An INSTANCE MUST use a 4xx HTTP status code for responses that indicate a problem with the format of the request.	
R1113	An INSTANCE SHOULD use a "400 Bad Request "HTTP status code, if the request message is a malformed HTTP request, or not well-formed XML.	
R1126	An INSTANCE MUST use a "500 Internal Server Error" HTTP status code if the response message is a SOAP Fault.	

De ontvanger van een HTTP response dient de HTTP status te controleren, en alle geldige HTTP statussen te accepteren en behandelen. Met name de HTTP "408 Request Timeout" dient behandeld te worden, zie hoofdstuk 6.

4.5.2 SOAP Faults

SOAP kent een Fault element dat onderdeel is van de SOAP Body en daar maar één keer mag voorkomen. Het SOAP Fault element heeft vier subelementen, hieronder kort toegelicht.

faultcode

Een verplicht element. De waarde van dit element moet namespace gekwalificeerd zijn. SOAP kent zelf vier foutcodes die verderop in deze paragraaf genoemd worden.

faultstring

Een verplicht element die een voor mensen leesbare uitleg bevat over de aard van de opgetreden fout.

faultactor

Dit element is bedoeld om informatie te geven over wie de fout gegenereerd heeft. Het lijkt op het SOAP actor attribuut, maar geeft de bron van de fout aan in plaats van het einddoel van de header. Applicaties die niet het eindpunt (ultimate destination) zijn van het SOAP bericht, moeten het faultactor element opnemen. Applicaties die wel het einddoel zijn, mogen het faultactor element opnemen.

Voor fouten gegenereerd door de ZIM is de waarde "http://www.aortarelease.nl/actor/lsp" verplicht (NB: deze waarde komt niet overeen met die van de SOAP actor, zie par 4.3.1). Voor fouten gegenereerd door de GBx wordt aangeraden de waarde "http://www.aortarelease.nl/actor/gbx" te gebruiken. Andere waarden voor faultactor zijn niet toegestaan.

detail

Het element detail is bedoeld voor applicatie specifieke foutinformatie die gerelateerd is aan het Body element van het SOAP request. Het is verplicht wanneer de inhoud van het Body element niet succesvol verwerkt kon worden. Het mag niet gebruikt worden voor foutinformatie over de header. Wanneer het element detail aanwezig is, betekent dit dat de fout gerelateerd is aan de

verwerking van het element Body. Wanneer het element detail niet aanwezig is, dan is de fout niet gerelateerd aan de verwerking van het element Body.

SOAP kent de faultcodes zoals opgesomd in onderstaande tabel. Deze faultcodes kunnen worden opgenomen in het element faultcode en moeten dan ook namespace gekwalificeerd zijn.

Naam	Toelichting
VersionMismatch	De SOAP Envelope heeft een ongeldige namespace, wat gebruik van een SOAP-versie die niet ondersteund wordt kan aangeven. Wanneer de SOAP Envelope een andere namespace heeft dan ""http://schemas.xmlsoap.org/soap/envelope/" MOET deze Fault gegenereerd worden.
MustUnderstand	Er is een SOAP Header aanwezig die niet bekend is, maar met mustUnderstand = "1". In dit geval MOET deze Fault gegenereerd worden.
Client	Deze Faults kunnen gebruikt worden bij syntaxfouten in het SOAP bericht of HL7v3-payload. Het gaat daarbij bijvoorbeeld om berichten die niet voldoen aan de wsdl. Voor semantische fouten in de HL7v3-payload gebruikt AORTA HL7v3-foutberichten (Accept Acknowledgements). Een applicatie is niet verplicht zelf SOAP Client Faults te gebruiken, maar dient ze wel te accepteren.
Server	Deze Fault mag alleen gebruikt worden wanneer de ontvangende applicatie niet in staat is de HL7v3-payload te verwerken omdat de lokale HL7v3-applicatie niet beschikbaar is. Een applicatie is niet verplicht zelf SOAP Server Faults te gebruiken, maar dient ze wel te accepteren.

Het WS-I Basic Profile definieert een aantal richtlijnen voor het gebruik van SOAP Faults. De meest relevante zijn opgenomen in onderstaande tabel.

Code	Basic Profile tekst	Toelichting
R1000	When a MESSAGE contains a `soapenv:Fault` element, that element MUST NOT have element children other than `faultcode`, `faultstring`, `faultactor` and `detail`	SOAP staat dit wel toe, maar WS-I Basic Profile dus niet. Het child element `detail`, mag zelf wel child elementen hebben, zie ook R1002.
R1001	When a MESSAGE contains a `soapenv:Fault` element its element children MUST be unqualified	Dus `faultcode`, `faultstring`, `faultactor` and `detail moeten unqualified zijn`.
R1002	A RECEIVER MUST accept fault messages that have any number of elements, including zero, appearing as children of the `detail` element. Such children can be qualified or unqualified.	AORTA laat het gebruik van het `detail` element in principe vrij. Echter, het gebruik van de gekwalificeerde child elementen code en text wordt aanbevolen. Zie hieronder voor de voorbeelden.
R1003	A RECEIVER MUST accept fault messages that have any number of qualified or unqualified attributes, including zero, appearing on the detail element. The namespace of qualified attributes can be anything other than "http://schemas.xmlsoap.org/soap/envelope/".	AORTA laat de te gebruiken namespaces voor de child elementen van het `detail` element vrij. Echter, het gebruik van de volgende namespace is een suggestie: http://www.aortarelease.nl/actor/[specificpart]/soapFault/detail. Bijvoorbeeld voor de ZIM: http:// www.aortarelease.nl/actor/lsp/soapFault/detail en bijvoorbeeld voor het GBK: http:// www.aortarelease.nl/actor/gbk/soapFault/detail.
R1016	A RECEIVER MUST accept fault messages that carry an `xml:lang` attribute on the `faultstring` element.	Het attribuut `xml:lang` geeft aan in welke taal de faultstring staat. In AORTA is alleen nederlands of engels toegestaan en is het gebruik van het attribuut `xml:lang` voor de

		zender optioneel.
R1004	When a MESSAGE contains a `faultcode` element the content of that element SHOULD be one of the fault codes defined in SOAP 1.1 or a namespace qualified fault code.	Een gekwalificeerde foutcode is toegestaan, dus bijvoorbeeld: zim:AUTERR Waarbij de prefix zim in een bijbehorend xmlns:zim attribuut correct gedefinieerd moet worden.
R1031	When a MESSAGE contains a `faultcode` element the content of that element SHOULD NOT use of the SOAP 1.1 "dot" notation to refine the meaning of the Fault.	AORTA staat het gebruik van de zogenaamde "dot" notatie niet toe.

Ter illustratie hieronder een voorbeeld van een correcte en een incorrecte SOAP Fault in AORTA.

```
CORRECT:
<soapenv:Fault xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/' >
    <faultcode>soapenv:Client</faultcode>
    <faultstring>Het input document is niet valide.</faultstring>
    <faultactor>http://www.aortarelease.nl/actor/lsp</faultactor>
    <detail>
        <lsp:code xmlns:lsp='http://www.aortarelease.nl/actor/lsp/soapFault/detail'>
            MissingMandatoryElement
        </lsp:code>
        <lsp:text xmlns:lsp='http://www.aortarelease.nl/actor/lsp/soapFault/detail'>
            Een verplicht element mist in het input document: …..
        </lsp:text>
    </detail>
</soapenv:Fault>
```

```
INCORRECT:
<soapenv:Fault xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/' >
    <!-- Onderstaande is fout omdat "dot" notatie niet is toegestaan -->
    <faultcode>soapenv:Client.ZIM.XSD_VALIDATION_ERR</faultcode>
    <!-- Onderstaande is fout omdat child elementen van soapenv:Fault
         niet namespace gekwalificeerd mogen zijn -->
    <soapenv:faultstring> Het input document is niet valide</soapenv:faultstring>
    <soapenv:faultactor>http://www.aortarelease.nl/actor/lsp</soapenv:faultactor>
    <detail>
        <code>MissingMandatoryElement</code>
        <text>Een verplicht element mist in het input document: …..</text>
        <lsp:extraInfo xmlns:lsp='http://www.zim.nl/soapFault/detail'>
              Het missende element is verplicht geworden in release….
       </lsp:extraInfo>
    </detail>
</soapenv:Fault>
```

Bij ontvangst van een bericht dat geen well-formed XML bevat, wordt bij voorkeur een HTTP response met status 400 'Bad Request' gegeven. Een applicatie dient ook rekening te houden met een SOAP Client Fault of een HL7v3 Accept Acknowledgement Error.

Dit omdat het niet exact te bepalen is welke fouten een applicatie het eerst dient te signaleren. Er is bijvoorbeeld geen verplichting eerst het hele XML-document te parsen

om op well-formedness te controleren voordat verdere verwerking plaatsvindt. Bij niet well-formed XML hoeft geen SOAP Fault of HL7v3-error gegeven te worden.

Een applicatie dient altijd op een bericht te reageren - uiteraard zolang de applicatie überhaupt tot reageren in staat is. Een uitzondering is wanneer een applicatie reden heeft aan te nemen dat er sprake is van een malicieuze actie, zoals een Denial Of Service aanval.

Wanneer een applicatie niet op HL7v3-niveau kan reageren, bijvoorbeeld omdat er geen zinnige HL7v3-interactie uit het bericht te destilleren valt, of omdat er geen geschikte HL7v3-reactie voorhanden is, heeft de volgende reactie de voorkeur:

- als er een fout is in de SOAP syntax, (bijvoorbeeld soapenv:Envelope zonder soapenv:Body of soapenv:Fault): een SOAP Client Fault, met HTTP status 500 'Internal Server Error';
- als er een andere fout is in de syntax, een reactie met HTTP status 400 'Bad Request';
- wanneer er geen fout is in de syntax, maar er serverproblemen zijn, een reactie met HTTP Status 500 'Internal Server Error', eventueel gevuld met een SOAP Server Fault.

Bij foutafhandeling dient het principe gevolgd te worden: "wees strikt in wat je zendt, en tolerant bij fouten die je ontvangt". De afhandeling als hier geschetst heeft de voorkeur, bij ontvangst van onverwachte fouten dient een applicatie echter adequaat te reageren, bijvoorbeeld door de fout te loggen voor menselijke afhandeling.

RFC 2616 (HTTP/1.1) beveelt het gebruik van een entity (HTTP berichtinhoud) aan bij een response met status 4xx of 5xx. "... Except when responding to a HEAD request, the server SHOULD include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition." Aanbevolen wordt uiteraard RFC 2616 hier te volgen. Deze specificatie geeft geen verdere richtlijnen voor de op te nemen boodschap. In de meeste gevallen zullen de toelichtingen bestemd zijn voor menselijk gebruik (systeembeheerders en applicatieontwikkelaars), omdat het ernstige fouten betreft.

4.5.3 HL7v3-fouten

HL7v3 kent eigen foutberichten. Deze vallen buiten de scope van dit document. Wel wordt in de tabel hieronder aangegeven welke HTTP status gebruikt dient te worden bij HL7v3-fouten.

Code	Naam	Toelichting (zie HL7v3 voor details)	HTTP Status
AA	Application Acknowledgement Accept	Succesvolle verwerking.	200 OK
AE	Application Acknowledgement Error	Fout in bericht (permanent).	200 OK
AR	Application Acknowledgement Reject	Fout, maar niet in inhoud of formaat van bericht (tijdelijk).	200 OK
CA	Accept Acknowledgement Commit Accept	Geaccepteerd.	200 OK

CE	Accept Acknowledgement Commit Error	Niet geaccepteerd (permanent).	200 OK[2]
CR	Accept Acknowledgement Commit Reject	Niet geaccepteerd (tijdelijk).	200 OK

HL7v3 Application en Accept Acknowledgements worden behandeld als succes op HTTP niveau.

4.5.4 Fout bij de ZIM als tussenstation

Wanneer een versturend GBx een bericht zendt aan een ontvangend GBx, kan het ontvangende GBx een foutmelding genereren. De ZIM dient daar als volgt mee om te gaan:

- HL7 foutmeldingen dienen net als een gewoon HL7 antwoord ongewijzigd teruggegeven te worden aan het versturende GBx;
- SOAP Faults dienen net als een gewoon SOAP antwoord ongewijzigd teruggegeven te worden aan het versturende GBx;
- HTTP fouten worden vertaald naar HL7 fouten, waarbij HTTP client fouten naar HL7 client fouten vertaald worden en HTTP server fouten naar HL7 server fouten:
    - HTTP fouten in de 4xx range worden vertaald naar een HL7 SYNGBX error uit LSP-tabel (OID 2.16.840.1.113883.2.4.6.6.1.1000) met typecode CE en als weergavenaam het applicatieId en de HTTP resultaatcode.
    - HTTP fouten in de 5xx range worden vertaald naar een HL7 RTEDEST error uit HL7-tabel (OID 2.16.840.1.113883.5.1100) met typecode CR en als weergavenaam het applicatieId en de HTTP resultaatcode.

In het geval van opvragen van patiëntgegevens door een initiërend GBx wordt een eventuele foutmelding gegenereerd door het reagerende GBX door de ZIM in het batchopleverbericht opgenomen zoals hierboven beschreven. De ZIM zal in het uiteindelijk opleverbericht een extra foutmelding opnemen over de aard van de aanwezige foutmeldingen in het batchopleverbericht.

4.6 HTTP Redirects

Het gebruik van HTTP redirects is niet toegestaan.

In de internationale standaarden die AORTA voorschrijft, met name [Basic Profile], staat dat voor HTTP Redirects alleen 307 gebruikt mag worden. De betekenis van 307 is "temporary redirect". De rest is in [Basic Profile] expliciet verboden vanwege interoperabiliteitsproblemen. In AORTA zijn ook de 307 redirects verboden.

[2] In versie 1.1 en eerder was dit "400: Bad Request". Het gebruik van een HTTP status anders dan 2xx voor SOAP berichten die geen SOAP Fault zijn, staat echter op gespannen voet met SOAP 1.1 en WS-I Basic Profile 1.0, en deze beide standaarden worden gevolgd in AORTA. Er was dus sprake van een inconsistentie.

5 AORTA en WSDL

Een WSDL in AORTA dient als documentatie en contract, omdat de WSDL precies beschrijft aan wat voor service de gegevensleverancier zich gecommitteerd heeft.

Beschrijvingen van webservices worden opgesteld in WSDL 1.1.

5.1 Inhoud WSDL

Deze paragraaf beschrijft de inhoud van een AORTA WSDL stap voor stap.

De uitwerking maakt gebruik van een voorbeeldtransactie: VerstrekkingsLijstquery. Volgens het interactiediagram bestaat deze transactie uit:

- de vraag QURX_IN990111NL (Medication Dispense List Query);
- het antwoord QURX_IN990113NL (Medication Dispense List Query Response).

Ieder bericht is in HL7v3 gedefinieerd met een XML Schema. Zo is er een XML Schema QURX_IN990111NL.xsd, QURX_IN990113NL.xsd, etcetera. In de XML Schema's ligt de structuur van het bericht vast, inclusief de HL7v3-Transmission en -Control Act Wrappers.

In de WSDL worden allereerst benodigde XML Schema's gedeclareerd als WSDL Types. Hieronder volgt een uitleg aan de hand van een voorbeeld.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns="http://schemas.xmlsoap.org/wsdl/"
             xmlns:hl7="urn:hl7-org:v3"
             targetNamespace="urn:hl7-org:v3"
             name="VerstrekkingsLijstquery">
   <documentation> WSDL implementatie van VerstrekkingsLijstquery
   </documentation>
   <types>
      <xsd:schema targetNamespace="urn:hl7-org:v3" elementFormDefault="qualified">
         <xsd:include schemaLocation="../schemas_codeGen/QURX_IN990111NL.xsd"/>
      </xsd:schema>
      <xsd:schema targetNamespace="urn:hl7-org:v3" elementFormDefault="qualified">
         <xsd:include schemaLocation="../schemas_codeGen/QURX_IN990113NL.xsd"/>
      </xsd:schema>
   </types>
```

Eerst is er een inleidend deel, daarna worden schema’s gekoppeld. De schema's worden opgenomen met een schema-include om een en ander beknopt, leesbaar en onderhoudbaar te houden. Er zijn in dit geval twee schema's:

- QURX_IN990111NL.xsd (Medication Dispense Event List Query)
- QURX_IN990113NL.xsd (Medication Dispense Event List Query Response)

Vervolgens worden de nodige berichten gedeclareerd als WSDL Message. De berichten krijgen een unieke naam: die van de interactie.

```
  <message name="QURX_IN990111NL">
     <part name="body" element="hl7:QURX_IN990111NL"/>
   </message>
   <message name="QURX_IN990113NL">
     <part name="body" element="hl7:QURX IN990113NL"/>
   </message>
```

Er zijn ook twee verschillende berichten:

- QURX_IN990111NL (Medication Dispense List Query)
- QURX_IN990113NL (Medication Dispense List Query Response)

De verschillende samenhangende interacties uit het Storyboard worden vertaald naar WSDL Port Types. Hierin zijn bij elkaar horende in- en outputs gekoppeld tot WSDL operations.

```
   <portType name="VerstrekkingsLijstquery_PortType">
      <operation name="VerstrekkingsLijstquery_QueryResponse">
         <input message="hl7:QURX_IN990111NL"/>
         <output message="hl7:QURX_IN990113NL"/>
      </operation>
   </portType>
```

Hierboven is een operation getoond volgens het synchrone model van berichtenuitwisseling: het antwoord op de query komt synchroon binnen over hetzelfde SOAP en HTTP request/response paar. Dit levert de volgende operation op: VerstrekkingsLijstquery_QueryResponse. Deze bestaat weer uit de eerder gedefinieerde berichten, die als input en output message zijn gedefinieerd.

In sommige gevallen is bij een HL7v3-interactie die verzonden wordt meer dan één interactie als antwoord mogelijk. WSDL staat dit niet toe: iedere WSDL-operation mag maximaal één input en één output hebben. De oplossing is een samengesteld xsd element te maken waarin alle HL7v3-antwoordinteracties mogelijk zijn. In onderstaand fragment is deze stijl te zien. Er wordt aan het schema één extra element toegevoegd, met een keuze tussen beide HL7v3-interacties. Dit element heet [request-interaction-id]Response. Dit element wordt vervolgens gebruikt in de message definitie en de operation, zodat de operation weer een enkele output heeft.

```
<types>
  <xsd:schema targetNamespace="urn:hl7-org:v3" elementFormDefault="qualified"
              xmlns:hl7="urn:hl7-org:v3" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    ...
    <xsd:include schemaLocation="../schemas/COMT_IN800310.xsd" />
  </xsd:schema>
  <xsd:schema targetNamespace="urn:hl7-org:v3" elementFormDefault="qualified"
              xmlns:hl7="urn:hl7-org:v3" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:include schemaLocation="../schemas/COMT_IN800320.xsd" />
  </xsd:schema>
  <xsd:schema targetNamespace="urn:hl7-org:v3" elementFormDefault="qualified"
              xmlns:hl7="urn:hl7-org:v3" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

```
    <xsd:element name="COMT_IN800300Response">
      <xsd:complexType>
        <xsd:choice>
          <xsd:element ref="hl7:COMT_IN800310"/>
          <xsd:element ref="hl7:COMT_IN800320"/>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>
</types>
...
<message name="COMT_IN800300Response">
  <part name="body" element="hl7:COMT_IN800300Response" />
</message>
<portType name="OverdrachtVerantwoordelijkheid_PortType">
   ...
  <operation name="OverdrachtVerantwoordelijkheid_VerzoekOverdrachtVervallen">
    <input message="hl7:COMT_IN800300" />
    <output message="hl7:COMT_IN800300Response" />
  </operation>
</portType>
```

De WSDL Port Types worden gekoppeld aan WSDL Bindings waar meer details over de SOAP-interactie gegeven worden. Hier wordt gespecificeerd dat de stijl van gegevensuitwisseling ”document/literal” is.

In de Binding wordt ook de waarde van SOAPAction bepaald. Deze SOAPAction komt terug in de HTTP Header, zie paragraaf 4.4.

```
   <binding type="hl7:VerstrekkingsLijstquery_PortType"
            name="VerstrekkingsLijstquery_Binding">
      <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="VerstrekkingsLijstquery_QueryResponse">
         <soap:operation soapAction="urn:hl7-org:v3/VerstrekkingsLijstquery_QueryResponse"/>
         <input>
            <soap:body use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
      </operation>
   </binding>
```

Ten slotte wordt van de WSDL Bindings een WSDL Service gemaakt, die het webadres van een specifieke webservice geeft.

```
   <service name="VerstrekkingsLijstquery_Service">
      <port binding="hl7:VerstrekkingsLijstquery_Binding"
            name="VerstrekkingsLijstquery_Port">
         <!--Deze service location URI verwijst niet naar een echt bestaande webservice. -->
         <soap:address location="http://www.xis.nl/VerstrekkingsLijstquery"/>
      </port>
   </service>
</definitions>
```

De waarde van "location" is hier nog fictief. Zie paragraaf 5.2 voor meer details.

5.2 Locatie van een webservice

De ZIM of een GBx vervullen in HL7v3-terminologie zogenaamde Application Roles. Een WSDL beschrijft de webservice voor iedere Application Role. Nictiz levert WSDL's waarin alleen de locatie (de url) van de service nog niet definitief is. Ieder GBx die een webservice aanbiedt in AORTA, biedt ook de echte locatie van de webservice.

Voor het opvragen van medicatie is er bijvoorbeeld een "VerstrekkingsLijstquery.wsdl" bestand waarin de locatie van de webservice nog niet definitief is ingevuld. Als locatie staat er:

```
<!--Deze service location URI verwijst niet naar een echt bestaande webservice. -->
<soap:address location="http://www.gbx.nl/VerstrekkingsLijstquery" />
```

Voor ieder GBx komt er dan een specifieke invulling van de locatie. Stel dat er een GBx is met het (fictieve) adres http://www.gbx1.nl, dan kan de ZIM deze webservice aanroepen met onderstaande webservice locatie.

```
<soap:address location="http://www.gbx1.nl/VerstrekkingsLijstquery" />
```

De padnamen zijn verplicht: http://www.gbx1.nl/VerstrekkingsLijstquery is goed, http://www.gbx1.nl/VerstrekkingsLijstquery**.py** of
http://www.gbx1.nl/**Mijn**VerstrekkingsLijstquery is dat niet.

Een GBx-applicatie mag maar één hostname gebruiken voor alle webservices die de GBx-applicatie aanbiedt. Dus in bovenstaand voorbeeld wordt de webservice locatie van een Voorschriftquery van dezelfde GBx-applicatie:

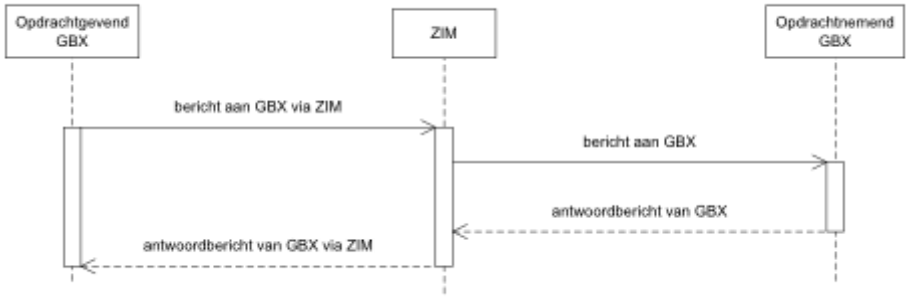
```
<soap:address location="http://www.gbx1.nl/Voorschriftquery" />
```

Dit om te voorkomen dat de ZIM een registratie moet bijhouden van alle webservices per GBx, met de bijbehorende plicht voor GBx'en om deze aan te melden en wijzigingen door te geven.[3]

5.3 Opvragen patiëntgegevens en "batch" wsdl's

Opvragen patiëntgegevens betreft een berichtuitwisselingspatroon in AORTA, waarbij een initiërend GBx een opvraging doet via de ZIM aan nul of meer reagerende GBZ'en. De ZIM bundelt de antwoorden van de reagerende systemen in een zogenaamde batch wrapper en retourneert dit batch bericht aan het initiërende systeem. Zie onderstaand sequentiediagram in Figuur 7 voor een illustratie hiervan met twee reagerende GBZ'en bij een VerstrekkingsLijstQuery (zie [Ontw OPV] voor een volledig overzicht).

[3] In AORTA kunnen IP-adressen gebruikt worden voor adressering. In de voorbeelden worden URI's gebruikt. In beide gevallen gelden de verplichte padnamen.



Figuur 7 - sequentie diagram opvragen patiëntgegevens: VerstrekkingsLijstQuery

Dit berichtuitwisselingspatroon vindt zijn weerslag in de WSDL's. Immers, het request/response paar tussen een initiërend GBx en de ZIM kent een ander antwoordbericht (namelijk een batch bericht) dan het request/response paar tussen ZIM en reagerend GBZ. Bij opvragen patiëntgegevens zijn er dan ook twee WSDL-ports gedefinieerd, een batch versie voor de webservice-port tussen initiërend GBx en de ZIM en een 'normale' versie voor de webservice-port tussen ZIM en reagerend GBZ.

Een voorbeeld hiervan is de VerstrekkingsLijstQuery, deze kent de volgende twee WSDL-service ports:

- VerstrekkingsLijstQueryBatch_Port (aangeroepen door initiërend GBx bij ZIM),
- VerstrekkingsLijstQuery_Port (aangeroepen door ZIM bij reagerend GBZ).

Paragraaf 5.5 bevat illustraties van de html-documentatie van deze webservice in Figuur 8 en Figuur 9.

Merk op dat reagerende GBZ'en niet allemaal dezelfde AORTA release hoeven te ondersteunen en dat een batch antwoordbericht van de ZIM derhalve berichten met verschillende versies kan bevatten. Eén batch antwoordbericht kan dus zowel AORTA 6 als AORTA 7 berichten bevatten.

5.4 Versionering van webservices

Het kan voorkomen dat een webservice wijzigt bij een nieuwe AORTA publicatie, bijvoorbeeld omdat:

- een input of output bericht wijzigt;
- het pad van de webservice locatie wijzigt;
- een operation wijzigt of toegevoegd of verwijderd wordt;
- de soap action wijzigt.

De meest waarschijnlijke wijziging is overigens bij de eerstgenoemde: het input of output bericht.

Als er iets wijzigt in de webservice, krijgt de webservice een (nieuw) versienummer, dat zich als volgt manifesteert:

- in de waarde van het `definitions/service/@name` attribuut, bijvoorbeeld: "VerstrekkingsLijstquery_02_Service";

- in de waarde van het `definitions/service/port/soap:address/@location` attribuut, bijvoorbeeld: “http://www.gbx.nl/02/VerstrekkingsLijstquery”;
- in de bestandsnaam van de wsdl, bijvoorbeeld “VerstrekkingsLijstquery-02.wsdl”;
- in de waarde van het `definitions/@name` attribuut, bijvoorbeeld “VerstrekkingsLijstquery_02”;
- in de waarde van het `definitions/documentation` element, bijvoorbeeld “WSDL implementatie van VerstrekkingsLijstquery_02”.

De eerste twee genoemde wijziging zijn het belangrijkst: een nieuwe servicenaam en een nieuwe (verplichte) padnaam voor de webservice locatie. Deze padnaam bevat het versienummer direct na de hostnaam van het GBx en/of ZIM.

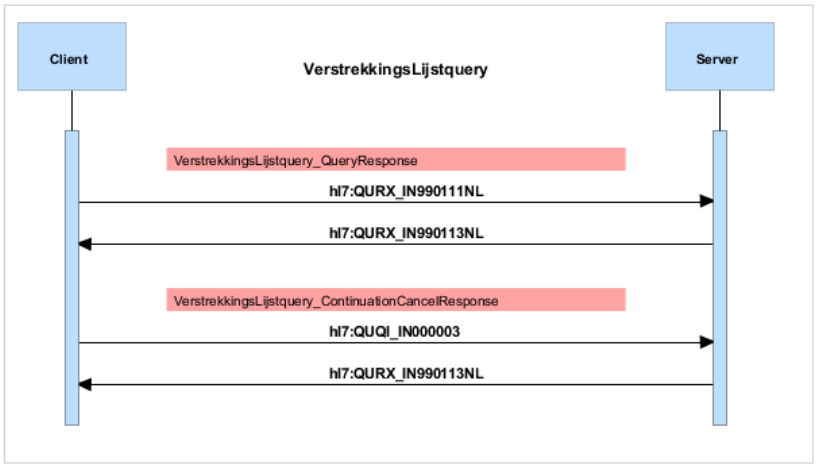
5.5 WSDL documentatie

Bij ieder WSDL bestand wordt ook documentatie in HTML formaat geleverd. Deze bevat een grafische weergave van de communicatie op HTTP-niveau en een overzicht van de relevante parameters uit de WSDL. Figuur 8 en Figuur 9 bevatten een illustratie van deze HTML-documentatie voor VerstrekkingsLijstQuery.wsdl.

VerstrekkingsLijstquery

Betere zorg door betere informatie Nictiz

WSDL	VerstrekkingsLijstquery.wsdl
Service	VerstrekkingsLijstquery_Service
Service Location	http://www.xis.nl/VerstrekkingsLijstquery
Diagram	Communicatie op HTTP-niveau: Iedere pijl van client naar server is een HTTP-request, iedere pijl terug van server naar client daaronder is de bijbehorende HTTP response.

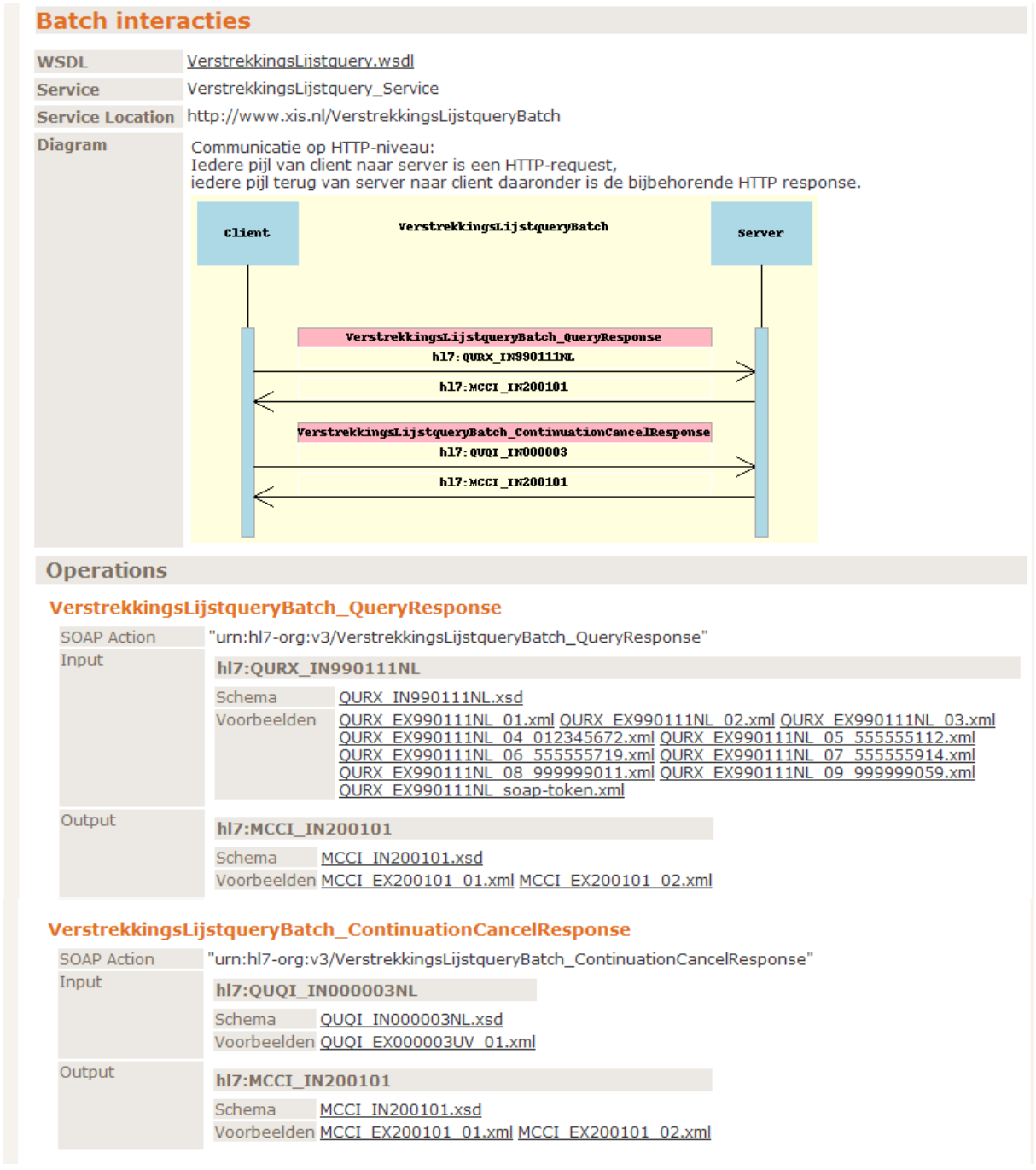


Operations

VerstrekkingsLijstquery_QueryResponse

SOAP Action	"urn:hl7-org:v3/VerstrekkingsLijstquery_QueryResponse"
Input	**hl7:QURX_IN990111NL**
Schema	QURX_IN990111NL.xsd
Voorbeelden	QURX_EX990111NL_01.xml QURX_EX990111NL_02.xml QURX_EX990111NL_03.xml QURX_EX990111NL_04_012345672.xml QURX_EX990111NL_05_555555112.xml QURX_EX990111NL_06_555555719.xml QURX_EX990111NL_07_555555914.xml QURX_EX990111NL_08_999999011.xml QURX_EX990111NL_09_999999059.xml QURX_EX990111NL_soap-token.xml
Output	**hl7:QURX_IN990113NL**
Schema	QURX_IN990113NL.xsd
Voorbeelden	QURX_EX990113NL_01.xml QURX_EX990113NL_02_012345672.xml QURX_EX990113NL_03_555555112.xml QURX_EX990113NL_04_555555719.xml QURX_EX990113NL_05_555555914.xml QURX_EX990113NL_06_999999011.xml QURX_EX990113NL_07_999999059.xml QURX_EX990113NL_INVALMEDCODE.xml

Figuur 8 - Schermafdruk WSDL-documentatie VerstrekkingsLijstQuery



Figuur 9 - Schermafdruk WSDL-documentatie VerstrekkingsLijstQueryBatch

6 Betrouwbaar transport

Dit hoofdstuk beschrijft betrouwbare uitwisseling van berichten in AORTA. De eisen aan een GBx voor betrouwbaar transport zijn beschreven in [PvE GBx Rollen]. Dit hoofdstuk dient vooral om deze eisen nader toe te lichten.

Betrouwbaar transport stelt – in het algemeen - de volgende eisen:

- een bericht wordt eenmaal verwerkt (duplicaatdetectie en –verwijdering);
- berichten worden afgeleverd in de volgorde waarin ze verzonden zijn;
- er is garantie dat een bericht ofwel afgeleverd wordt, ofwel dat het falen gemeld wordt.

Voor betrouwbaar transport moet ieder systeem voorzieningen treffen tegen het falen van systemen. Een voorbeeld hiervan is dat een ontvanger een bericht persistent opslaat, voordat een antwoord verzonden wordt. Daarnaast is het van belang dat ieder bericht een unieke identificatie bevat zodat een ontvanger duplicaten kan detecteren. Betrouwbare aflevering kan dan worden ingevuld op de volgende wijze:

- een verzender zendt een bericht aan een ontvanger;
- de ontvanger slaat het bericht persistent op en retourneert een antwoordbericht;
- wanneer de verzender een antwoordbericht ontvangt, is de status van de verzending bekend bij deze verzender;
- wanneer de verzender geen antwoordbericht ontvangt, mag deze een nieuwe poging doen;
- de ontvanger detecteert of het bericht al eerder verwerkt is; is dit het geval, dan verwerkt de ontvanger het bericht deze keer niet en informeert de verzender met een antwoordbericht;
- wanneer na herhaalde pogingen geen antwoordbericht komt, wordt de gebruiker (of diens beheerder) van het verzendende systeem gewaarschuwd.

6.1 Herhaald uitvoeren

Herhaald uitvoeren is een essentieel onderdeel van betrouwbaar transport. Het is mogelijk dat ergens in de keten een bericht niet verwerkt is zoals het door de verzender bedoeld was. De verzender zal een foutmelding of een waarschuwingsmelding ontvangen. De verzender kan hierdoor getriggerd worden om hetzelfde verzoek nogmaals te verzenden. Er zijn verschillende manieren om een actie herhaald uit te voeren. AORTA onderscheidt de volgende mogelijkheden:

- Herhalen met een duplicaat bericht. Een duplicaat is exact hetzelfde bericht als het origineel, inclusief wrappers. Dit betekent dus dat het bericht nogmaals verzonden wordt met een identiek message-id en bij tokenauthenticatie een identiek transactietoken.
- Herhalen met een nieuw bericht. Een nieuw bericht bevat weliswaar hetzelfde functionele verzoek, maar heeft (minimaal) een nieuw message id en bij tokenauthenticatie ook een nieuw transactietoken.

Wanneer de ZIM een bericht gaat verwerken, wordt het transactietoken op “gebruikt” gezet. Dit token kan dan niet nogmaals gebruikt worden. Bij tokenauthenticatie is een nieuwe poging met een duplicaat bericht dan ook alleen zinvol wanneer het originele bericht niet bij de ZIM is aangekomen. Het is dus sterk afhankelijk van waar in de keten zich een uitzonderingssituatie voordoet.

6.2 Duplicaatdetectie

Als gevolg van het toepassen van herhaald uitvoeren wordt duplicaatdetectie van belang. AORTA onderscheidt de volgende vormen:

- Duplicaatdetectie op transportniveau. Hierbij worden duplicaat berichten gedetecteerd aan de hand van een identieke transmission wrapper, meer specifiek op basis van het message id.
- Duplicaatdetectie op applicatieniveau. Hierbij worden duplicaten gedeteceerd aan de hand van de HL7v3-payload, meer specifiek op basis van het patiëntstuk-id (in HL7v3 ingevuld door "Act id"). Het betreft hier niet per definitie duplicaat berichten, er kan ook sprake zijn van nieuwe berichten met duplicaat payload.

Het kan zijn dat een bericht niet als duplicaat geldt op transportniveau, maar wel een duplicaat blijkt te zijn op applicatieniveau. Het betreft dan een nieuw bericht met een nieuw message id, maar met een duplicaat payload. Andersom kan dit niet: een duplicaat op transportniveau is altijd ook een duplicaat op applicatieniveau. Dit laatste is overigens de verantwoordelijkheid van de zender: een duplicaatbericht moet identiek zijn aan het originele bericht. De ontvanger mag hiervan uitgaan en hoeft de payload dus niet te evalueren wanneer een duplicaat gedetecteerd wordt op transportniveau.

Merk op dat het ook mogelijk is dat er nieuwe berichten zijn met hetzelfde patiëntstuk-id en nieuwe, bijgewerkte payload. Als dit mogelijk is, zal duplicaatdetectie op applicatieniveau niet van toepassing zijn.

De ZIM voert duplicaatdetectie op transportniveau uit bij tokenauthenticatie: duplicaat transactietokens worden geweigerd. Dit geldt niet in het geval van het GBP waar een DigiD-SAML-token kan worden hergebruikt. Inschrijf- en mandaattokens kunnen ook hergebruikt worden. Programma's van eisen stellen duplicaatdetectie op applicatieniveau verplicht voor specifieke berichten.

6.3 Beveiliging en betrouwbaarheid

Omdat herhaald uitvoeren onderdeel is van betrouwbaar transport, ontstaat een aanvullend risico voor aanvallen, zoals een "Denial of Service" (DoS) aanval of bij tokenauthenticatie een "Replay Attack".

Een DoS aanval is een poging met een grote hoeveelheid berichten een server zo te belasten dat deze het normale, valide verkeer niet meer aankan. De verplichting om een bericht te beantwoorden geldt niet bij een vermoeden van een DoS aanval. (Duplicaat) berichten hoeven niet verwerkt te worden wanneer er een vermoeden is van een DoS aanval.

Bij een replay attack steelt een aanvaller een bericht en verzendt dit opnieuw. De ZIM weigert duplicaat transactietokens en pareert zo dergelijke replay attacks.

6.4 Toepassing betrouwbaar transport in AORTA

Deze paragraaf beschrijft welke algemene aspecten van betrouwbaar transport van toepassing zijn voor AORTA. Daarom hier eerst nogmaals de eisen die geïntroduceerd zijn in het begin van dit hoofdstuk:

- een bericht wordt eenmaal verwerkt (duplicaatdetectie en –verwijdering);
- berichten worden afgeleverd in de volgorde waarin ze verzonden zijn;

- er is garantie dat een bericht ofwel afgeleverd wordt, ofwel dat het falen gemeld wordt.

AORTA ondersteunt eenmalige verwerking en aflevergarantie, en niet volgordelijkheid.

Het garanderen van volgordelijkheid van berichten is geen vereiste omdat er (nog) geen use case is om dit aan de *ontvangende* kant te garanderen. Sommige berichten moeten wel in de goede volgorde afgehandeld worden, bijvoorbeeld: een heraanmelding verwijsindex moet na een aanmelding plaatsvinden en een afmelding moet na een aanmelding plaatsvinden. De verantwoordelijkheid voor volgordelijkheid ligt hier echter bij de verzender. Indien volgordelijkheid van belang is mag het GBx een tweede bericht pas verzenden, wanneer het eerste bericht succesvol afgehandeld is.

Aflevering van berichten in de volgorde waarin ze verzonden zijn wordt niet ondersteund.

De aflevergarantie dwingt af dat er altijd gepoogd wordt om een antwoordbericht te versturen.

Ieder HL7v3-bericht in AORTA moet beantwoord worden met een antwoordbericht, behalve wanneer er een vermoeden is van een DoS aanval.

In AORTA gelden de volgende uitgangspunten:

- De verzender is verantwoordelijk om te achterhalen hoe het bericht verwerkt is.
- Bij ontvangst van een antwoordbericht kan de verzender zeker weten dat het bericht ten minste één keer is afgeleverd.
- Bij het niet ontvangen van een antwoordbericht weet de verzender niets zeker over de aflevering van het bericht. Normaliter zal de verzender dan (automatisch) opnieuw proberen en wanneer dit blijvend niet lukt een beheerder of gebruiker alarmeren alsnog actie te ondernemen.
- De ontvanger is verantwoordelijk om te voorkomen dat een actie onterecht een tweede keer wordt verwerkt. De ontvanger moet dan duplicaatdetectie toepassen.

6.5 Uitwisselingspatronen

AORTA maakt onderscheid tussen de volgende primaire uitwisselingspatronen:

- Opvragen patiëntgegevens. Opvragen van GBX1 via ZIM aan GBX2.
- Versturen patiëntgegevens. Versturen van GBX1 via ZIM naar GBX2.

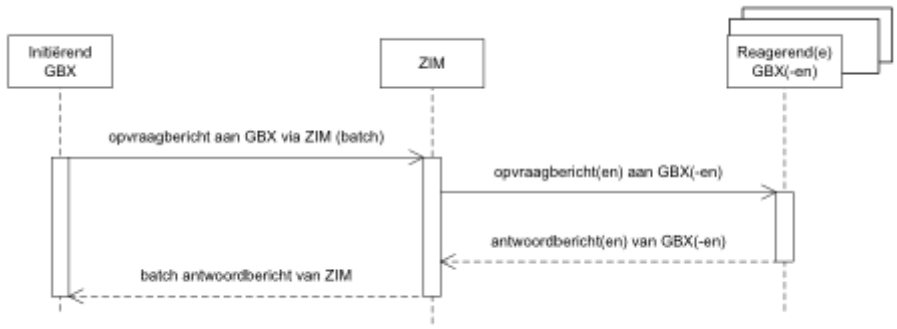
AORTA maakt onderscheid tussen de volgende ondersteunende uitwisselingspatronen:

- Opvragen van gegevens. Opvragen van GBX1 aan ZIM.
- Versturen van gegevens. Versturen van GBX1 naar ZIM.

6.5.1 Primaire uitwisselingspatronen

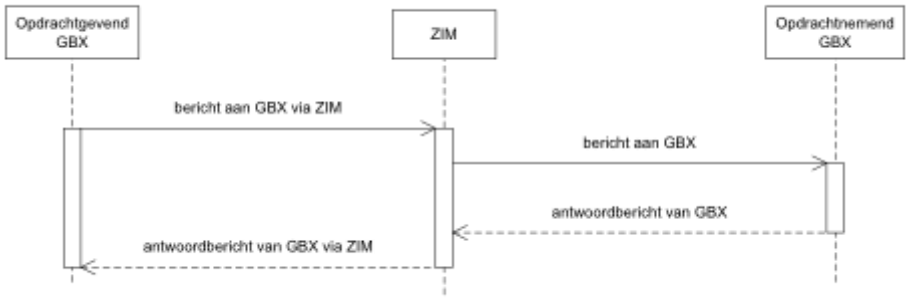
Opvragen patiëntgegevens betreft opvragingen van een initiërend GBx aan één of meerdere reagerende GBx-en via de ZIM. De vraag wordt gesteld aan de ZIM, de ZIM divergeert de vraag naar de juiste GBx(-en). In het geval er sprake is van een generieke

opvraag met context, stuurt de ZIM een specifiek bouwsteen opvraagbericht naar de juiste GBx-en. Het is mogelijk dat een GBx meerdere bouwsteen opvraagberichten te verwerken krijgt. Deze reagerende GBx-en sturen hun antwoord naar de ZIM. De ZIM bundelt deze antwoorden in een batch antwoord voor het initiërende GBx. Onderstaande figuur illustreert dit uitwisselingspatroon.



Figuur 10 - Opvragen patiëntgegevens

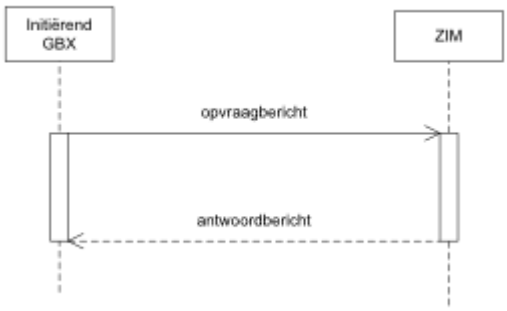
Versturen patiëntgegevens betreft berichten van een initiërend GBx aan een reagerend GBx via de ZIM. De ZIM stuurt dit bericht door naar het ontvangende, reagerende GBx. Het antwoord van dit reagerende GBx wordt ontvangen door de ZIM en doorgestuurd naar het initiërende GBx. Onderstaande figuur illustreert dit uitwisselingspatroon.



Figuur 11 - Versturen patiëntgegevens

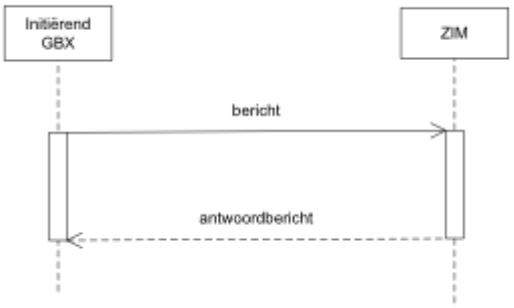
6.5.2 Ondersteunende uitwisselingspatronen

Opvragen van gegevens betreft opvragingen van een GBx aan de ZIM. Onderstaande figuur illustreert dit uitwisselingspatroon.



Figuur 12 - Opvragen gegevens

Versturen van gegevens betreft berichten van een GBx aan de ZIM. Onderstaande figuur illustreert dit uitwisselingspatroon.



Figuur 13 - Versturen gegevens

De invulling van betrouwbaar transport verschilt tussen deze uitwisselingspatronen en dan met name tussen opvragen en versturen. De volgende twee paragrafen lichten toe hoe AORTA betrouwbaar transport invult voor opvragen patiëntgegevens en opvragen gegevens respectievelijk versturen patiëntgegevens en versturen gegevens.

6.5.3 Opvragen patiëntgegevens en opvragen gegevens

Bij opvragen patientgegevens en opvragen gegevens gaat het om het opvragen van informatie. Er wordt dus niets gewijzigd in de geadresseerde applicatie. Het staat het initiërende GBx daarom vrij om nieuwe pogingen te ondernemen. Het verdient de voorkeur om mislukte opvragingen altijd terug te melden aan de gebruiker en deze gebruiker vrij te laten om nieuwe pogingen uit te voeren. Een nieuwe poging wordt daarmee altijd gedaan met een nieuw bericht.

6.5.4 Versturen patiëntgegevens en versturen gegevens

Bij versturen patiëntgegevens en versturen gegevens gaat het om een systeem dat een bericht verstuurt aan een ander systeem. [PvE GBx Rollen] specificeert algemene eisen voor betrouwbaar transport. De verschillende programma's van eisen definiëren per specifiek bericht welke eisen met betrekking tot betrouwbaar transport van toepassing zijn.

Bij het opstellen van deze programma's van eisen geldt de richtlijn dat wanneer het niet schadelijk is om een bericht een tweede keer te versturen, dit dan ook toegestaan is.

Bij versturen gegevens geldt dat de ZIM duplicaatdetectie op applicatieniveau toepast. Het is daarom toegestaan en zelfs aanbevolen om een retry-mechanisme in te bouwen voor versturen, bijvoorbeeld het (her)aanmelden en afmelden van gegevens in de verwijsindex.

Op het moment van schrijven van deze implementatiehandleiding zijn er twee interacties voor versturen patiëntgegevens bekend:

- Waarneemretourbericht (bedoeld voor rapportage van HAP naar huisarts).

- Voorschriftbericht.

Hoewel het voor de hand ligt om bij het Waarneemretourbericht duplicaatdetectie op applicatieniveau te implementeren, is dit niet expliciet geëist van het reagerende GBZ. Het opnieuw verwerken van hetzelfde bericht is hooguit lastig voor de zorgverlener, maar niet gevaarlijk voor de patiënt. Het is daarom ook voor dit bericht toegestaan om een retry mechanisme te gebruiken.

Op het moment van schrijven van de specificaties voor AORTA 2013, zijn er nog geen gekwalificeerde systemen voor het voorschriftbericht. In AORTA 2013 is duplicaatdetectie op applicatieniveau geëist voor het voorschriftbericht. Ook bij het versturen van het voorschriftbericht is het daarom toegestaan om een retry-mechanisme in te bouwen.

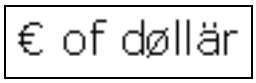
Overigens blijven de specificaties in de programma's van eisen leidend voor de vereiste implementatie van betrouwbaar transport. De beschrijving in deze gids dient slechts als toelichting.

Referenties

Referentie	Document	Versie
[Basic Profile]	Basic Profile Version 1.0 www.ws-i.org	1.0
[SOAP]	Simple Object Access Protocol (SOAP) 1.1 http://www.w3.org/TR/SOAP	1.1
[WSDL]	Web Services Description Language (WSDL) 1.1, W3C Note, 15 March 2001 http://www.w3.org/TR/wsdl	15-mrt-2001
[XML]	Extensible Markup Language (XML) 1.0, W3C Recommendation, Fourth Edition, 16 August 2007 http://www.w3.org/TR/xml	16-aug-2007
[PvE GBx Rollen]	Programma van eisen infrastructurele systeemrollen	8.2.0.0
[IH BA Mandaattoken]	Implementatiehandleiding Berichtauthenticatie Mandaattoken	8.2.0.0
[IH tokens generiek]	Implementatiehandleiding security tokens generiek	8.2.0.0
[IH BA DigiD]	Implementatiehandleiding berichtauthenticatie met DigiD	8.2.0.0
[IH BA PKIO-pas]	Implementatiehandleiding berichtauthenticatie met PKIO-pas	8.2.0.0
[IH BA Transactietoken]	Implementatiehandleiding berichtauthenticatie transactietoken	8.2.0.0
[IH BA Inschrijftoken]	Implementatiehandleiding berichtauthenticatie inschrijftoken	8.2.0.0
[Arch AORTA]	Architectuur AORTA	8.2.0.0

Diakritische tekens in UTF-8

Het is van belang te verifiëren of diakritische en andere tekens goed gecodeerd zijn. Dat kan bijvoorbeeld met de volgende tekst:



Noot: afhankelijk van de instellingen van de PC kunnen diakritische tekens anders overkomen. Daarom is deze tekst grafisch opgenomen. Wanneer deze tekens hieronder er anders uitzien, gebruik de tekens uit het plaatje.

Deze tekst dient exact zo in het bronsysteem van de zender ingevoerd te worden in een veld dat vrije tekst toestaat. Bij de ontvanger dient de tekst in het doelsysteem weer zo leesbaar te zijn. Wanneer het doelsysteem één of meer van deze tekens niet ondersteunt, dient de ontvangende partij in de XML te verifiëren dat de tekst ongeschonden is overgekomen. De tekst is zo gekozen omdat ze de verschillen tussen de meest gebruikte tekstcoderingen, UTF-8, ISO-8859-1, ISO8859-15 en Windows-1252 blootlegt.

Hexadecimale representatie van enkele diakritische tekens			
Encoding	**Euroteken (€)**	**Kleine letter o met forward slash (ø)**	**Kleine a met trema (ä)**
UTF-8	E2 82 AC	C3 B8	C3 A4
ISO-8859-1	bestaat niet	F8	E4
ISO-8859-15	A4	F8	E4
Windows-1252	80	F8	E4

Een geschikte plaats voor deze tekst is (alleen in testomgevingen) het optionele element <softwareName> in de Transmission Wrapper. Dit komt in alle berichten voor, maar wordt veelal niet gebruikt in testen.

```
<?xml version="1.0" encoding="utf-8"?>
...
<sender>
    <device>
        <id extension="01234567" root="2.16.840.1.113883.2.4.6.6"/>
        <softwareName>€ of døllär</softwareName>
    </device>
</sender>
...
```