

Implementatiehandleiding berichttransport



Betere zorg door betere informatie

postadres: Postbus 19121, 2500 CC Den Haag
bezoekadres: Oude Middenweg 55, 2491 AC Den Haag
telefoon: (070) 317 34 50; fax: (070) 320 74 37; e-mail: servicedesk@infoEPD.nl
www.nictiz.nl

Versie : 6.0.0.0
Datum : 31 oktober 2008

Inhoudsopgave

1	Inleiding	4
1.1	Doel en doelgroep	4
1.2	Versie, status en wijzigingshistorie	4
1.3	Achtergrond	5
1.4	Reikwijdte	5
1.5	Structuur	5
1.6	Samenhang met andere documenten	5
2	Uitgangspunten	6
2.1	Normatieve referenties	6
2.2	Informatieve referenties	6
2.3	Afkortingen en begrippen	6
3	Berichtuitwisselingsstandaarden	8
3.1	SOAP	8
3.2	WSDL	9
3.3	WS-I Basic Profile	10
3.4	HL7v3	10
3.5	Overzicht van standaarden	11
4	Informatiestromen	12
4.1	AORTA en SOAP / WSDL gegevensuitwisseling	12
4.2	Globaal overzicht informatiestroom	12
4.3	SOAP en de ZIM	13
5	XML en SOAP	15
5.1	UTF-8 encoding	15
5.2	SOAP berichtformaat	16
5.3	HTTP Binding	17
5.4	Betrouwbare aflevering over SOAP en HTTP	20
5.5	HTTP errors en SOAP Faults	26
5.6	HTTP Redirects	29
6	WSDL	31
6.1	AORTA en WSDL	31
6.2	HL7v3 Application Roles, berichten en WSDL	31
7	Bijlage: Voorbeelden	38
7.1	Statische voorbeelden	38
7.2	Dynamische voorbeelden	58

1 Inleiding

1.1 Doel en doelgroep

In 2003 heeft Nictiz een standaard voor het gebruik van SOAP voor HL7v3 berichten ontwikkeld. Parallel daaraan werd ook binnen HL7 gewerkt aan een integratie van SOAP en HL7. Het werk wat binnen Nictiz is verricht is vervolgens geïntegreerd met het werk wat binnen HL7 al was gedaan, om zo tot een gezamenlijke SOAP / HL7 standaard te komen, geheten "Web Services Profile". De standaard is opgenomen in Ballot 7 van HL7v3 met status DSTU (Draft Standard for Trial Use). Deze standaard wordt gezien als voldoende stabiel om als basis voor proefimplementaties te dienen. Dit document is geen vervanging voor het lezen van het HL7v3 Web Services Profile.

Dit document heeft tot doel het gebruik van SOAP in de Nederlandse zorgsector nader te specificeren. Het is met name een uitwerking van het transport over HTTP zoals dat in AORTA wordt toegepast.

Dit document is vooral bedoeld voor softwareontwikkelaars van zorgapplicaties en zorg-infrastructurele applicaties, die op grond van de HL7 versie 3 communicatiestandaard en op grond van dit document berichten over Internet willen transporteren.

De lezer wordt verondersteld kennis te hebben van [XML], [SOAP] en [WSDL]. Lezing van de [IH berichtwrappers], en (overige relevante delen van) de AORTA-documentatie in samenhang met dit document wordt ten zeerste aanbevolen.

1.2 Versie, status en wijzigingshistorie

1.2.1 Versie

Dit is versie 6.0.0.0 van het document "Implementatiehandleiding berichttransport".

1.2.2 Status

De status van deze versie is "Definitief".

1.2.3 Wijzigingshistorie

Wijzigingen in versie 6.0.0.0 ten opzichte van versie 2.0:

- Wijziging 1390: Paragraaf over WSDL bij HL7v3 interactie met meerdere antwoordinteracties opgenomen.
- WSDL voorbeeld <types> sectie gefixt, er was tekst weggefallen.
- Hoofdstuk 7 "Versiebeheermechanismen" opgenomen.
- Spellingscontrole met nictizv2.dic gedaan.

1.3 Achtergrond

Dit document is geschreven in het kader van het AORTA-programma van Nictiz. Het ministerie van VWS, CIBG en Nictiz werken samen met partijen in het veld aan de invoering van een landelijk Elektronische Patiëntdossier (EPD). Het landelijk EPD is gebaseerd op open Internetstandaarden.

1.4 Reikwijdte

Dit document specificeert het berichttransport voor het landelijk EPD over het Internet. Het vormt daarbij de verbinding tussen de onderliggende HTTP(S)-laag, en de er bovenliggende HL7 berichten. In dit document wordt gespecificeerd hoe het berichttransport er op dit niveau uit ziet, gebruikmakend van standaarden als SOAP en WSDL.

Dit document beschrijft een drietal zaken:

- vormgeven van een HL7v3 bericht als webservice met SOAP en WSDL;
- transport van een HL7v3 bericht over HTTP(S);
- betrouwbaarheidsaspecten van dat transport;.

Dit document dient als toevoeging op de internationale HL7 versie 3 materialen. In geval van tegenstrijdigheden tussen de internationale standaard en dit document geldt hetgeen bepaald is in dit document.

1.5 Structuur

Dit document is als volgt opgebouwd:

- 1) Inleiding (niet normatief)
- 2) Uitgangspunten
- 3) Berichtuitwisselingsstandaarden (niet normatief): Een inleiding in de relevante gebruikte standaarden.
- 4) Informatiestromen (niet normatief): Een overzicht van berichtuitwisseling zoals gebruikt bij het landelijk EPD.
- 5) XML en SOAP (normatief): richtlijnen voor het gebruik van XML en SOAP.
- 6) WSDL (normatief): algemene toelichting bij de WSDL bestanden.

1.6 Samenhang met andere documenten

Dit document maakt deel uit van de documentatieset AORTA-basisinfrastructuur. Die documentatieset, nader beschreven het [Documentatieoverzicht], definieert de basisinfrastructuur voor berichtuitwisseling in de Zorg.

Dit document is een van de vier implementatiehandleidingen, die zijn afgeleid van het architectuurontwerp en daarvan met name de technische architectuur. Waar dit document zich richt op het transportmechanisme voor het uitwisselen van digitale berichten tussen zorgverleners, richten de overige drie implementatiehandleidingen zich vooral op die berichten zelf.

2 Uitgangspunten

2.1 Normatieve referenties

De onderstaande documenten zijn beschouwd als leidend voor dit document:

Identificatie	Titel	Bron	Versie Datum
[Basic Profile]	Basic Profile Version 1.0	www.ws-i.org	1.0
[HL7v3]	HL7 Version 3 Standard	www.hl7.org	
[HTTP]	RFC 2616 Hypertext Transfer Protocol -- HTTP/1.1	http://www.ietf.org	
[Namespaces]	Namespaces in XML 1.0 (Second Edition)	www.w3.org/TR/xml-names/	1.0
[Technische architectuur]	Technische architectuur AORTA	Nictiz	6.0.0.0 31 oktober 2008
[SOAP]	Simple Object Access Protocol (SOAP) 1.1	http://www.w3.org/TR/SOAP	
[WSDL]	Web Services Description Language (WSDL) 1.1, W3C Note, 15 March 2001	http://www.w3.org/TR/wsdl	15 March 2001
[XML]	Extensible Markup Language (XML) 1.0, W3C Recommendation, Fourth Edition, 16 August 2007	http://www.w3.org/TR/xml	16 August 2007

2.2 Informatieve referenties

De onderstaande documenten hebben gediend als bron voor dit document:

Identificatie	Titel	Bron	Versie Datum
[Documentatieoverzicht]	Documentatieoverzicht AORTA-basisinfrastructuur	Nictiz	6.0.0.0 31 oktober 2008
[Verklarende woordenlijst]	Verklarende woordenlijst AORTA	Nictiz	6.0.0.0 31 oktober 2008
[IH berichtwrappers]	Implementatiehandleiding berichtwrappers	Nictiz	6.0.0.0 31 oktober 2008
[IH Generieke berichten]	Implementatiehandleiding generieke berichten	Nictiz	6.0.0.0 31 oktober 2008
[IH HL7v3 Medicatiegegevens]	Implementatiehandleiding HL7v3 Medicatiegegevens	Nictiz	6.0.0.0 31 oktober 2008

2.3 Afkortingen en begrippen

Alle afkortingen die relevant zijn voor het voorliggende document zijn opgenomen in de [Verklarende woordenlijst].

Algemene begrippen die relevant zijn voor het voorliggende document zijn opgenomen in de [Verklarende woordenlijst].

Overal in dit document waar de voornaamwoorden "hij", "hem" of "zijn" staan, wordt "hij of zij" resp. "hem of haar" resp. "zijn of haar" bedoeld.

3 Berichtuitwisselingsstandaarden

Het transport van zorgdocumenten volgens AORTA is gebaseerd op een aantal standaarden: met name SOAP, WSDL en relevante delen van de HL7 standaard (met name Transmission Infrastructure en Query Infrastructure).

3.1 SOAP

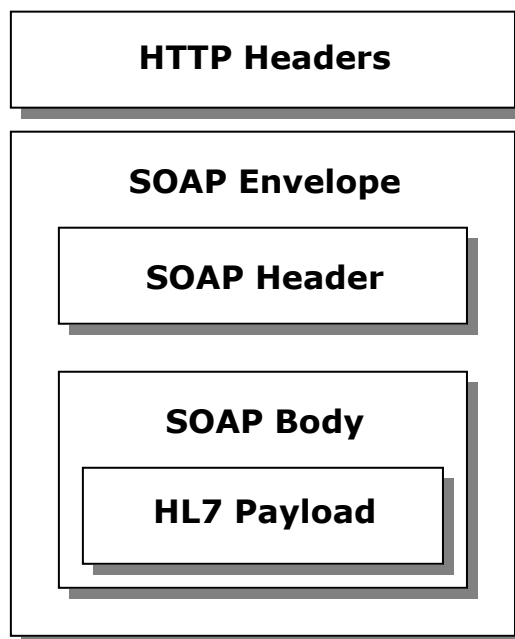
SOAP 1.1 – Simple Object Access Protocol – is een veelgebruikte standaard voor uitwisseling van gegevens over het Internet. SOAP is ontstaan uit een samenwerking van IBM, Microsoft en enkele andere partijen. SOAP 1.1 is op 8 mei 2000 gepubliceerd als Note door het World Wide Web Consortium (W3C). Een Note heeft geen officiële status, en is geen W3C-standaard. Desalniettemin is SOAP 1.1 door veel partijen geïmplementeerd en is het een "de facto standaard". SOAP 1.1 biedt een mechanisme om eenvoudig in XML verpakte informatie uit te kunnen wisselen, onder andere via het HTTP-protocol. SOAP 1.1 dient als de basis voor de uitwisseling van gegevens in de zorg via AORTA.

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.
3. De SOAP **Body** is het eigenlijke bericht. De Body bevat dus meestal de "data", de Header meestal bijbehorende "metadata".

SOAP kent nog een hoofdstuk "Encoding" waarin een mechanisme wordt beschreven om RPC-parameters te vertalen naar XML. Deze SOAP Encoding wordt hier niet gebruikt. Tenslotte kent SOAP nog een mapping op HTTP, waarbij HTTP als transportmechanisme gebruikt wordt om een SOAP document te transporteren.

Hieronder is een overzicht van een SOAP bericht gegeven.



De verschillende onderdelen zien er als volgt uit:

```
<?xml version="1.0" encoding="utf-8"?>
<SOAP-ENV:Envelope ... namespace declaraties ... >
  <SOAP-ENV:Header>
    ... headers ...
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    ... payload ...
  </SOAP-ENV:Body>
</SOAP-ENV: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. De Body bevat de "payload", de eigenlijke gegevens die verzonden worden. SOAP Headers worden meestal gebruikt voor meta-informatie die te maken heeft met verzending en aflevering, waaronder encryptie, signering en dergelijke.

SOAP 1.2 is op 24 juni 2003 een W3C Recommendation, en dus een officiële W3C standaard. SOAP 1.2 wordt niet breed genoeg gebruikt. De AORTA-uitwisseling is daarom niet op SOAP 1.2 gebaseerd. De beschreven uitwisseling is wel zo opgezet dat overgang naar SOAP 1.2 niet nodeloos gecompliceerd wordt.

3.2 WSDL

WSDL 1.1 – Web Services Description Language – is een taal om Web Services te beschrijven. WSDL dient twee 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.

In de praktijk is WSDL een design-time hulpmiddel wat het bouwen van interoperabele Web Services vereenvoudigt. WSDL 1.1 is een W3C Note, die echter als "de facto" standaard geldt. WSDL 1.2 was op het moment van opstellen van deze specificaties een W3C Working Draft – deze documenten zijn lang niet altijd stabiel. (Inmiddels is WSDL 1.2 een W3C Recommendation.) Er is in dit document gewerkt op basis van WSDL 1.1.

Een WSDL document bestaat uit de volgende onderdelen:

- 1) **Types**. Hier worden abstracte XML-structuren beschreven in XML **Schema** formaat.
- 2) **Messages**. Met de elementen en elementtypen uit de Types worden Messages samengesteld – dit zijn generieke berichten gedefinieerd in XML.
- 3) **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.
- 4) **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. Ook wordt aangegeven op welke wijze het SOAP bericht er uit dient te zien (document versus rpc, literal versus encoded).
- 5) **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.

3.3 WS-I Basic Profile

WS-I Basic Profile 1.0 is een specificatie gepubliceerd door WS-I, een wat ad hoc consortium van voornamelijk softwareleveranciers is. Initiatiefnemers zijn IBM en Microsoft. SUN neemt inmiddels ook deel. Het Basic Profile is een zeer nuttig werkstuk. Het beschrijft hoe SOAP 1.1 en WSDL 1.1 gebruikt kunnen worden om interoperabele Web Services te bouwen en geeft veel 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. Het Basic Profile wordt gevolgd tenzij anders aangegeven (aanbeveling R2710 wordt niet gevolgd, zie hieronder voor details).

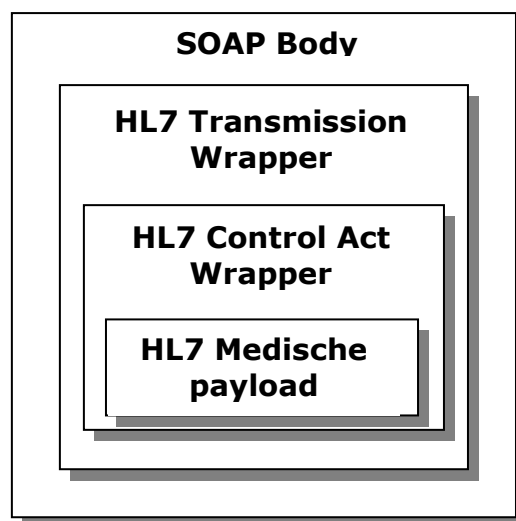
3.4 HL7v3

HL7v3 kent een – mede door Nictiz ontwikkelde – SOAP/WSDL-standaard "Web Services Profile" geheten in de HLv3 Ballot 6. Deze standaard wordt geheel nagevolgd.

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

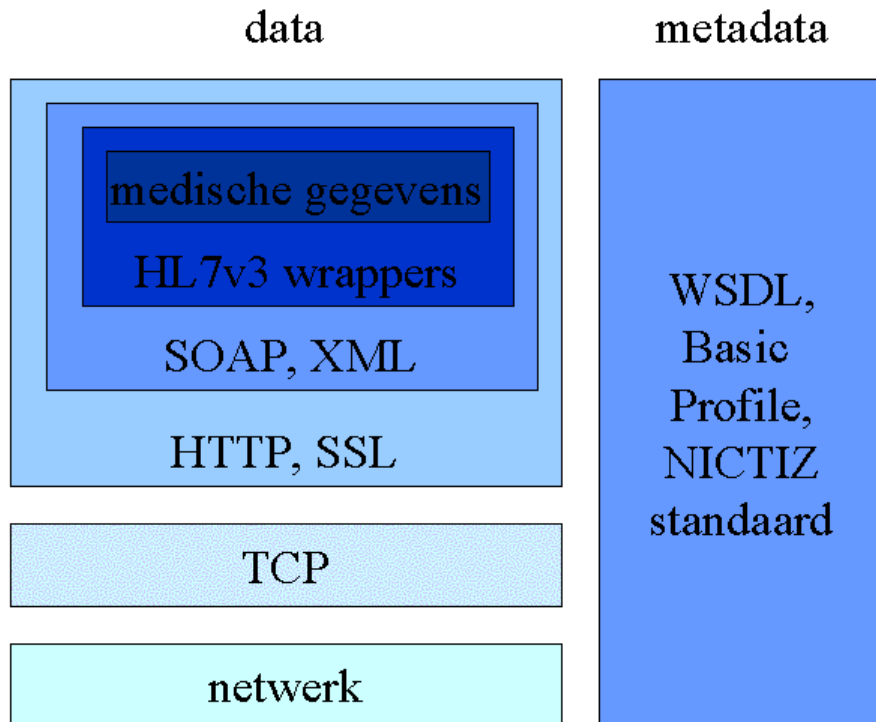
1. In de Transmission **Wrapper** worden metagegevens opgeslagen die te maken hebben met het bericht zelf: id, datum waarop het is aangemaakt, afzender, geadresseerde en dergelijke.
2. De **Trigger Event Control Act Wrapper** bevat gegevens die te maken hebben met de verwerking van het bericht, bijvoorbeeld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

Voor details, zie [IH berichtwrappers]. Tezamen met de eigenlijke medische gegevens, de payload, vormen deze Wrappers een XML document. De inhoud van de SOAP Body is dus als volgt:



3.5 Overzicht van standaarden

In onderstaand schema zijn de relevante standaarden in onderlinge relatie weergegeven (waarbij de benaming data en metadata slechts een onderscheid aangeeft tussen een interactie en een beschrijving ervan):



4 Informatiestromen

4.1 AORTA en SOAP / WSDL gegevensuitwisseling

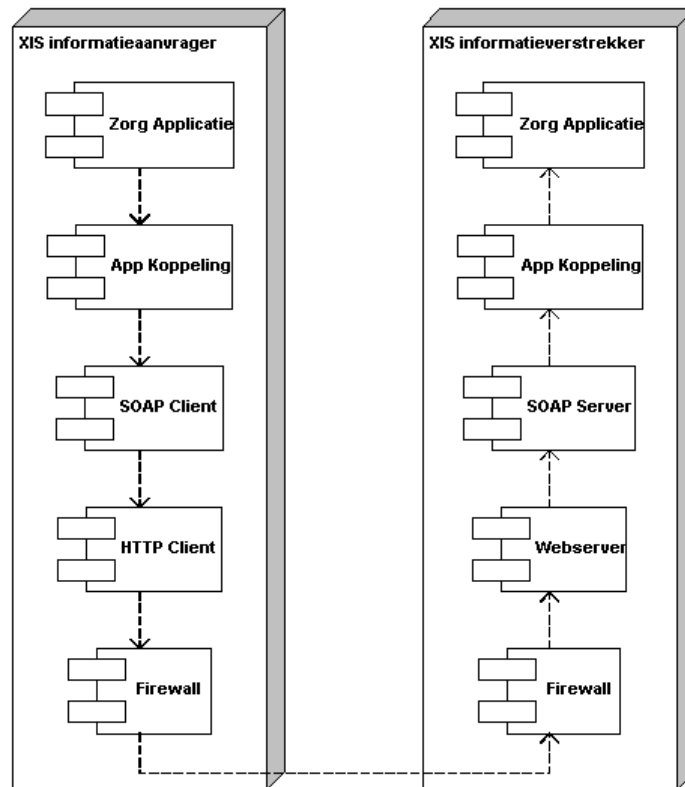
Nictiz heeft in AORTA gekozen voor gegevensuitwisseling op basis van SOAP en (optioneel) WSDL, zoals beschreven in de (mede door Nictiz opgestelde) HL7v3 standaard "Web Services Profile". SOAP is een XML berichtformaat wat o.a. over HTTP verzonden kan worden. De keuze voor SOAP is voor de hand liggend maar zeker niet triviaal. Er zijn alternatieven zoals HL7 direct met HTTP verzenden. De voornaamste redenen om voor SOAP te kiezen zijn:

- Er is uitgebreide ondersteuning van ontwikkelhulpmiddelen om gegevens via het Web uit te wisselen met SOAP en WSDL. Dat geldt ook voor HTTP zonder SOAP, maar de ondersteuning daarvoor is niet specifiek op berichtuitwisseling gebaseerd.
- Naar verwachting zullen in de nabije toekomst standaarden voor encryptie, authenticatie en dergelijke ontstaan die SOAP Headers gebruiken. Door de keuze voor SOAP nu wordt het "inpluggen" van dergelijke standaarden later eenvoudiger.

WSDL is, strikt gesproken, optioneel. Met WSDL wordt een web service beschreven. Volgens de Nictiz standaard zal WSDL alleen op "design time" gebruikt worden, dus als beschrijving van een te bouwen web service. Veel tools genereren op basis van WSDL een deel van de code van zo'n web service. Het is echter ook mogelijk de web service te bouwen zonder WSDL, en in die zin is gebruik van WSDL optioneel.

4.2 Globaal overzicht informatiestroom

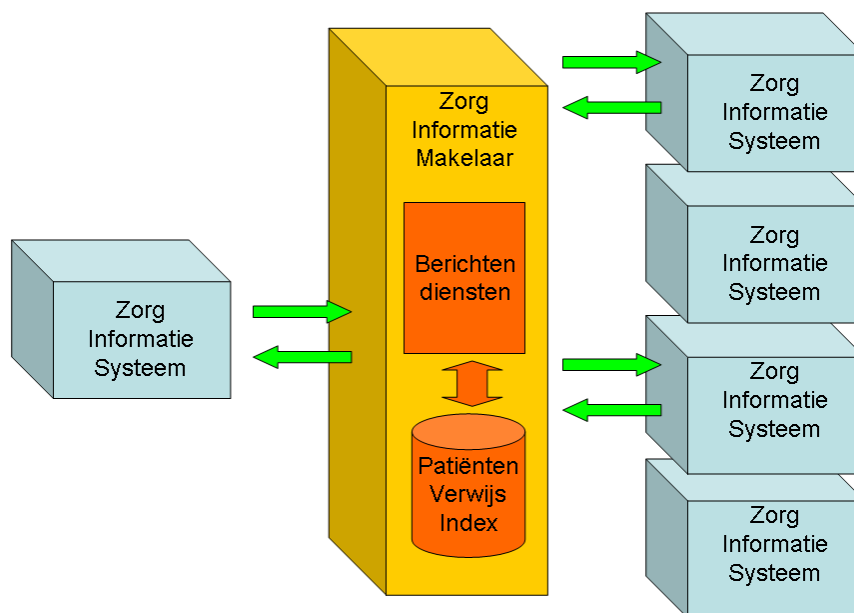
Een SOAP interactie kent een cliënt en een server. De SOAP cliënt stuurt een request naar de server via HTTP, en ontvangt het antwoord via HTTP.



Dit betekent dat heel andere eisen aan de cliënt en de server gesteld worden. De cliënt hoeft alleen een Internetverbinding te hebben en beschikbaar te zijn op het moment dat de aanroep gedaan wordt. De server zal echter continue moeten luisteren op een poort of er een SOAP aanroep binnenkomt. De meeste SOAP-implementaties die een SOAP-server ondersteunen maken dan ook gebruik van een (lichtgewicht) webserver, veelal een die ook als gewone webserver HTML pagina's via HTTP kan leveren. Een firewall zal zo geconfigureerd moeten worden dat de webserver SOAP requests en responses door mag laten. De pijlen geven het verloop van een SOAP request aan; een response volgt de omgekeerde weg. Als alternatieve configuratie kunnen de onderste vier componenten (firewall – webserver – SOAP cliënt/server – applicatiekoppeling) op een aparte machine ondergebracht worden die via het lokale netwerk met de zorgapplicatie verbonden is.

4.3 SOAP en de ZIM

In bovenstaande voorbeelden is alleen gesproken over uitwisseling tussen een cliënt en een server. Dit is echter een versimpeling. AORTA gaat uit van berichten die uitgewisseld worden tussen een cliënt en server via de ZIM (Zorg Informatie Makelaar).



Een gebruikersapplicatie van een zorgverlener doet een gegevensopvraag. De ZIM is verantwoordelijk voor het opzoeken van de applicaties die deze gegevens kunnen leveren. Dit kan de ZIM zelf zijn (bijvoorbeeld als het om een overzicht van de aangemelde gegevens gaat), maar veelal zullen dit applicaties bij andere zorgverleners zijn. In het laatste geval zal de ZIM de opvraag doorzenden. De ZIM is dan ook verantwoordelijk voor het doorsturen van de antwoorden. Al deze berichten zijn SOAP berichten. De ZIM zal (de adressering van) de berichten aanpassen; bijvoorbeeld een opvraag van een ziekenhuis van een overzicht van medicijnverstrekking aan een patiënt zal geadresseerd zijn aan de Zorg Informatie Makelaar; de opvragen die doorgezonden worden, zullen geadresseerd zijn aan de zorgverleners die over deze medicatiegegevens beschikken.

5 XML en SOAP

BT-01 Berichten worden uitgewisseld in XML 1.0.

5.1 UTF-8 encoding

Een van de meest elementaire problemen bij digitaal uitwisselen van gegevens is de gebruikte tekenset. In de tekenset worden binaire codes toegekend aan leesbare tekens. Voorbeelden zijn:

- ASCII, waarin onder andere de normale letters en cijfers en veelgebruikte interpunctietekens staan
- ISO Latin, of ISO-8859-1, wat ASCII is met een uitbreiding met West-Europese diakritische tekens,
- ISO-8859-15, wat in de medische wereld ook regelmatig gebruikt wordt, en wat een paar wijzigingen t.o.v. ISO-8859-1 bevat,
- Windows Code Page 1252, welke gebaseerd is op ISO Latin, maar meer tekens bevat, o.a. de Euro, en veelgebruikt wordt op Windows-platforms,
- DOS Code Page 850, ook met diakritische tekens
- Unicode, met nagenoeg alle tekens uit alle talen die nu gesproken worden.

Naast de tekenset is er een encoding. Unicode kent meerdere encodings. De belangrijkste zijn UTF-16, waarbij ieder karakter in minimaal twee bytes wordt opgeslagen, en UTF-8, waarbij de eerste 128 karakters hetzelfde zijn gecodeerd als in ASCII.

Bij de uitwisseling van gegevens kan nu het probleem optreden dat een gegevensleverancier tekens gebruikt (b.v. diakritische tekens) die het systeem van de ontvanger niet ondersteunt. Er zijn twee oplossingen denkbaar: de gegevensleverancier past de gegevens aan voor verzending, of de ontvanger doet dit na ontvangst. De tweede optie is uiteraard de meest wenselijke: een gegevensleverancier hoeft dan niet vooraf te weten welke tekensets het systeem van de ontvanger ondersteunt. Dit houdt in dat gepoogd wordt in het bericht dat verzonden wordt zoveel mogelijk tekens te ondersteunen: iedere gegevensleverancier kan dan de eigen tekens altijd kwijt. Dit betekent dat er voor Unicode gekozen wordt.

Een XML document is altijd in een bepaalde encoding. Dit wordt bij voorkeur ook aangegeven in een XML-prolog:

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

BT-02 Voor de uitwisseling volgens AORTA is gekozen om altijd een document in UTF-8 formaat uit te wisselen.

UTF-8 is voor XML de meest gangbare Unicode-encoding en wordt zeer goed ondersteund. Gebruik van UTF-8 garandeert dat alle nodige inhoud zoals diakritische tekens en het Euroteken ook daadwerkelijk gecodeerd kunnen worden. Ook gegevens die in een document gecodeerd dienen te zijn, vallen binnen het UTF-8 formaat (een beperkt datatype is immers te beschouwen als een subset van UTF-8). Wanneer de applicatie van de ontvanger geen volledig Unicode kan verwerken zal daar gecontroleerd moeten worden of er daadwerkelijk karakters in het bericht zitten die niet in de applicatie ondersteund worden, en de nodige actie ondernomen moeten worden (b.v. vervangen door andere te-

kens, of handmatige verwerking). Dit geldt alleen voor de eindbestemming, wanneer bijvoorbeeld de database alleen opslag in ISO-Latin of Windows Code Page 1252 kent.

BT-03 "In transit" en bij logging, dus vóór het eindpunt moet het bericht als UTF-8 bewaard en/of doorgezonden worden.

5.2 SOAP berichtformaat

BT-04 Berichten worden verpakt volgens de SOAP 1.1 standaard.

Hieronder staat een voorbeeld van een HL7v3 bericht wat volgens de HL7v3 standaard in een SOAP Envelope is verpakt. Het betreft een opvraag van medicijngebruik. Het voorbeeld is in stukken geknipt en van commentaar voorzien.

De prolog met UTF-8 declaratie.

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

SOAP Envelope met daarin alle namespace declaraties. De "soap" namespace bevat de namen uit de SOAP standaard zelf.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:hl7="urn:hl7-org:v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

SOAP Body met daarin het omsluitende element van de medicatieopvraag. Omdat dit element de HL7 namespace tot default namespace maakt, maken alle elementen daarin automatisch deel uit van de HL7 namespace. Er is gekozen om het bericht op te slaan in de SOAP Body zelf, en niet als attachment omdat de standaarden voor SOAP attachments nog niet uitgekristalliseerd zijn. Wanneer een XML bericht in de SOAP Body opgenomen wordt, dient de encoding van het ingekapselde XML bericht en die van het omsluitende SOAP bericht uiteraard dezelfde te zijn. Omdat Nictiz er voor gekozen heeft altijd UTF-8 uit te wisselen mag dit geen probleem zijn.

```
<soap:Body>
  <QURX_IN990011NL
    xsi:schemaLocation="urn:hl7-org:v3 ../schemas/QURX_IN990011NL.xsd"
    xmlns="urn:hl7-org:v3">
```

Elementen uit de Transmission Wrapper van HL7.

```
<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 opvraag naar medicatie van een bepaald PatientId.

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

Afsluiting van de onderdelen.

```
</subject>
</ControlActProcess>
</QURX_IN990011NL>
</soap:Body>
</soap:Envelope>
```

De inbedding van een HL7v3 XML bericht in SOAP is dus redelijk eenvoudig.

BT-05 Het XML bericht wordt als zodanig opgenomen in de SOAP Body in een SOAP Envelope.

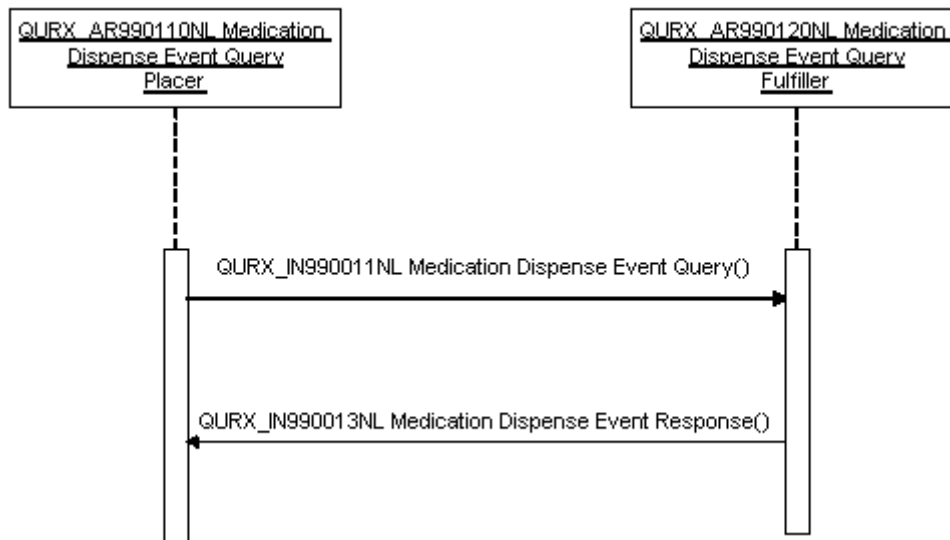
Twee zaken waar op gelet moet worden: het geheel moet in de juiste encoding (UTF-8) zijn, en de namespaces moeten kloppen. De meeste XML-ontwikkeltools zullen hier overigens zorg voor dragen.

5.3 HTTP Binding

SOAP is op zich een transport-neutraal formaat, maar in de praktijk is de koppeling (Binding) met HTTP (HyperText Transfer Protocol) de standaard werkwijze. HTTP is uiteraard welbekend als het transportmechanisme van het Web. HTTP voegt aan een te transporteren document – bijvoorbeeld in SOAP document – een aantal HTTP Headers toe. De headers worden afgesloten met een dubbele nieuwe regel. 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. In HTTP wordt POST normaliter gebruikt om gegevens aan een resource (website) toe te voegen.

SOAP is oorspronkelijk opgezet als een mechanisme om objecten aan te kunnen roepen middels een Remote Procedure Call – RPC over het Internet. Dit uitgangspunt is uitgebreid tot een meer generiek mechanisme om XML-documenten uit te wisselen, maar het is zinvol hier toch nog naar de oorspronkelijke RPC-natuur te kijken. Bij RPC over het Internet is er sprake van een object-aanroep (request) en een antwoord (response) van het object. In zowel de aanroep als het antwoord zitten een aantal parameters.

SOAP 1.1 definieert een Binding op HTTP waarbij SOAP requests en responses met HTTP POST worden uitgewisseld over het Internet. Daarbij wordt de 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 met de POST-methode. Dit voorbeeld is gebaseerd op het volgende interactiediagram, een fragment van de Verstrekingsquery, ontleend aan [Medicatieberichten], hoofdstuk 4.6.



Hieronder het verzoek, dus zowel de HTTP als de SOAP request, als de HL7v3 interactie QURX_IN990011NL (Medication Dispense Event Query):

```

POST /Verstrekkingsquery.asmx HTTP/1.1
Content-Length: 1874
SOAPAction: "urn:hl7-org:v3/Verstrekkingsquery_QueryResponse"
Host: acceptatie.webservice.testtool.nl
Content-Type: text/xml; charset=utf-8
Accept-Encoding:

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <QURX_IN990011NL xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:hl7-org:v3 ../schemas/QURX_IN990011NL.xsd" xmlns="urn:hl7-
      org:v3">
      ... inhoud ...
    </QURX_IN990011NL>
  </soap:Body>
</soap: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.

BT-07 De HTTP Header "Content-Type" moet waarde "text/xml" bevatten.

SOAP kent ook een HTTP header SOAPAction. Deze kan gebruikt worden door HTTP servers "en route" om intelligent het SOAP verkeer verder te leiden. Binnen de HL7v3 stan-

daard wordt dit veld gevuld met een namespace en het HL7 interactie-id. Een HTTP Server kan deze waarde gebruiken om het verkeer binnen een Goed Beheerd Zorgsysteem (GBZ) 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 HTTP en SOAP response, en de HL7v3 interactie QURX_IN990013NL (Medication Dispense Event Response):

```
HTTP/1.1 200 OK
Date: Mon, 20 Feb 2006 14:28:42 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
Set-Cookie: ASP.NET_SessionId=gora1p55lvhuet551sq0fqu3; path=/; HttpOnly
Cache-Control: private, max-age=0
Content-Type: text/xml; charset=utf-8
Content-Length: 7904

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <QURX_IN990013NL xsi:schemaLocation="urn:hl7-org:v3
../schemas/QURX_IN990013NL.xsd" xmlns="urn:hl7-org:v3" xmlns:code="urn::tabel25-code"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

      ... inhoud ...

    </QURX_IN990013NL>
  </soap:Body>
</soap:Envelope>
```

Eerst wordt de HTTP versie aangegeven en de status (200 OK, dat is succesvolle verwerking op HTTP niveau). 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 bericht en een HL7v3 antwoordbericht.

Deze drie niveaus zijn onafhankelijk van elkaar. In dit voorbeeld is er voor gekozen het HL7v3 antwoordbericht in de SOAP response te verpakken, en die weer in de HTTP response, maar dat is geen noodzaak. Het is bijvoorbeeld heel goed mogelijk het HL7v3 bericht in een apart HTTP request/response paar te verpakken, en het HL7v3 antwoord in een tweede HTTP request/response paar. De manier waarop een SOAP request en response op HTTP afgebeeld worden, heet de **HTTP Binding van SOAP**. En uiteraard kan een niveau hoger gesproken worden van de **SOAP Binding van HL7v3**, wanneer we praten over de wijze waarmee een HL7v3 request en response (b.v. een QueryContinuation en QueryResponse) afgebeeld worden op SOAP. In de volgende paragraaf over betrouwbare aflevering wordt toegelicht wanneer welke Binding gekozen dient te worden.

5.4 Betrouwbare aflevering over SOAP en HTTP

Aflevering van gegevens op basis van HTTP is op zichzelf niet betrouwbaar. Wanneer een verzender een bericht verstuurt, weet deze niet zeker of dit aangekomen is. Het antwoord op het verzonden bericht levert deze zekerheid wel. Wanneer b.v. een bericht verzonden wordt met HTTP POST, komt daar – behoudens fouten – een HTTP response met een antwoord (b.v. een SOAP Envelope met een HL7v3 XML bericht) op. Deze respons is uiteraard een bewijs dat de oorspronkelijke request is aangekomen. Voor betrouwbare aflevering is dat echter niet voldoende. Protocollen voor betrouwbare aflevering stellen normaliter als eis dat er voorzieningen zijn getroffen tegen het falen van systemen – bijvoorbeeld dat een ontvanger een bericht opslaat op een (redelijk) persistent medium zoals een harde schijf, voordat een antwoord verzonden wordt. Daarnaast wordt geëist dat ieder bericht een unieke identificatie bevat en dat een ontvanger duplicaten detecteert in de ontvangen berichten. Betrouwbare aflevering wordt dan gegarandeerd op de volgende wijze:

- verzender zendt een bericht aan ontvanger;
- ontvanger slaat het bericht persistent op, verwerkt het bericht en retourneert een ontvangstbevestiging;
- wanneer verzender deze ontvangt, is de verzending succesvol afgesloten;
- wanneer verzender deze niet ontvangt, zendt deze hetzelfde bericht met dezelfde unieke identificatie opnieuw;
- ontvanger ziet aan de identificatie dat het bericht eerder al verwerkt is, verwerkt het deze keer niet (tenminste niet voor zover dit reeds aangebrachte wijzigingen betreft) maar zendt wel de ontvangstbevestiging opnieuw terug;
- wanneer na herhaalde pogingen geen ontvangstbevestiging komt, wordt de gebruiker (of diens systeembeheerder) gewaarschuwd.

Een HTTP request wordt bij succesvolle verwerking beantwoord met een HTTP response die begint met de status: "200 OK". Deze status zegt alleen iets over het HTTP-verkeer: deze bevestigt dus dat de HTTP server de HTTP request ontvangen heeft. De HTTP status zegt op zich niets over persistente opslag. Daarom wordt er bij AORTA een aanvullende eis gesteld.

BT-08 Wanneer een HL7v3 verzoekbericht beantwoord wordt met een HL7v3 antwoordbericht, mag dat antwoordbericht pas verzonden worden wanneer de ontvanger in staat is dit identieke bericht opnieuw te genereren op een later tijdstip bij een binnenkomend identiek verzoekbericht.

Bijvoorbeeld in bovenstaand diagram mag de interactie QURX_IN990011NL (Medication Dispense Event Query) met een unieke identificatie Message.id met waarde X pas beantwoord worden met een QURX_IN990013NL (Medication Dispense Event Response) met een Message.id met waarde Y, wanneer later, bij binnenkomst van een QueryContinuati-on met wederom Message.id X, dezelfde Response met wederom Message.id Y terugge-stuurd kan worden.

BT-09 Ook bij een HL7v3 Accept Acknowledgement wordt geëist dat de ontvanger in staat is dit identieke bericht opnieuw te genereren.

Het ligt voor de hand dat de ontvanger de berichten op de harde schijf opslaat; dit is echter geen eis, zolang de ontvanger maar in staat is de berichten desgewenst opnieuw te genereren.

BT-10 De ontvangende partij dient een bericht met een bepaald Message.id slechts eenmaal te verwerken. Ook bij queries dient iedere keer hetzelfde antwoord geleverd te worden op een verzoek met een bepaald Message.id.

Wanneer bijvoorbeeld de fictieve query "Geef me de huidige tijd" komt met Message.id X, en het antwoord luidt "22:43", en om 23:10 komt de query met Message.id X opnieuw binnen, dan zal het antwoord weer "22:43" dienen te luiden.)

BT-11 Duplicaatdetectie dient tot 48 uur na eerste ontvangst van een bericht gegarandeerd te worden.

De ontvangende partij moet de Message.id's dus tenminste 48 uur bewaren en gebruiken om duplicaten te detecteren (maar dit mag ook veel langer, want duplicaatdetectie kan nooit kwaad).

BT-12 De ontvangende partij dient ook te verifiëren dat het duplicaat Message.id afkomstig is van de zender van het originele Message.id om misbruik door het opgeven van valse Message.id's te voorkomen. Dit kan bijvoorbeeld door ook het applicatie id te bewaren.

Voor betrouwbare aflevering is gekozen zoveel mogelijk gebruik te maken van HL7v3 Accept Acknowledgements. Accept Acknowledgements doen een uitspraak over de syntactische verwerkbaarheid van een bericht, en vormen een impliciete bevestiging van de ontvangst (en opslag) van een bericht. De functionaliteit van Accept Acknowledgements is dus breder dan de voor transportdoeleinden benodigde ontvangstbevestiging.

BT-13 De standaard transportmethode (methode 1) voor alle interacties is als volgt:

Voorbeeld 1: een HL7v3 Accept Acknowledgement voor alle HL7 berichten

HL7 niveau: Zender verzendt QueryByParameter naar Ontvanger
Ontvanger verzendt QueryResponse naar Zender

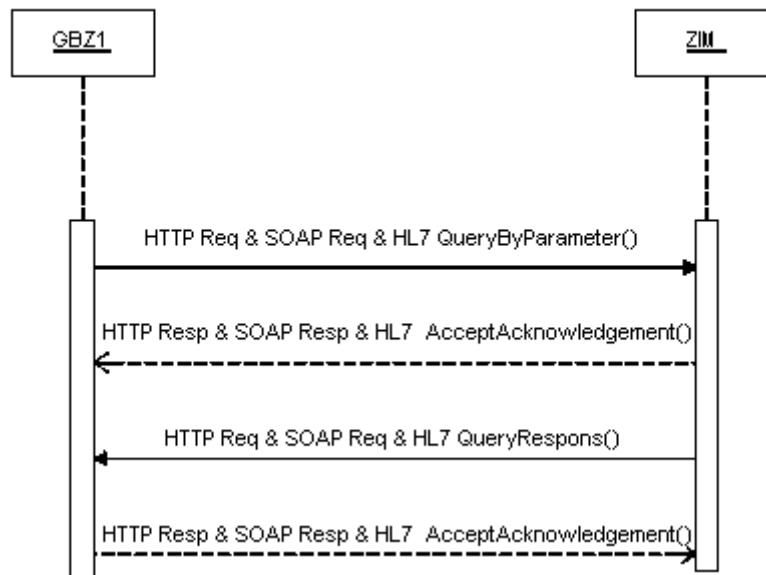
SOAP niveau: SOAP Request 1 (met QueryByParameter)
SOAP Response 1 (met Accept Acknowledgement)

SOAP Request 2 (met QueryResponse)
SOAP Response 2 (met Accept Acknowledgement)

HTTP niveau: HTTP Request 1 (met SOAP Envelope (met QueryByParameter))
HTTP Response 1 (met SOAP Envelope (met Accept Acknowledgement))

HTTP Request 2 (met SOAP Envelope (met QueryResponse))
HTTP Response 2 (met SOAP Envelope (met Accept Acknowledgement))

(Onderstaand plaatje laat dit zien voor een Query met QueryByParameter.
responsePriorityCode "Deferred", zie verderop voor toelichting.)



In dit voorbeeld is er op SOAP operation en HTTP-niveau sprake van asynchrone communicatie (niet op SOAP Binding niveau). Bij deze asynchrone methode zal een Zender bij moeten houden welke HL7 Response bij welk HL7 Request hoort. Omdat de communicatie asynchroon is, kunnen er immers meerdere HL7 Request tegelijkertijd uitstaan, en er is geen garantie dat de antwoorden binnenkomen in de volgorde waarin de Requests gedaan zijn.

In HL7 interactiediagrammen wordt dit niet beschreven; accept acknowledgements worden daar niet in opgenomen, en het wel of niet voorkomen van een accept acknowledgement wordt bepaald door de waarde van het HL7-element acceptAck.

BT-14 Alle uitgaande HL7v3 interacties behalve queries worden beantwoord met een HL7v3 Accept Acknowledgement.

Bovenstaande methode is toe te passen voor alle interacties. Voor 1 specifieke categorie interacties is een geoptimaliseerde methode (methode 2) van toepassing. Deze synchrone methode is voor deze specifieke categorie efficiënter en eenvoudiger te implementeren. Er is maar sprake van één HTTP Request-Response paar. De vragende applicatie blokkeert tot het antwoord ontvangen is (eventueel in meerdere parallele threads), wat de implementatie eenvoudiger maakt.

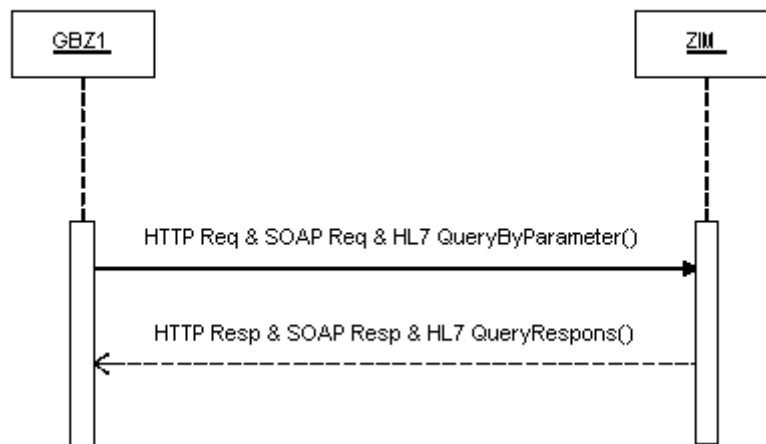
Voorbeeld 2: Direct antwoord bij queries

HL7 niveau: Zender verzendt QueryByParameter naar Ontvanger
 Ontvanger verzendt QueryResponse naar Zender

SOAP niveau: SOAP Request (met QueryByParameter)
 SOAP Response (met QueryResponse)

HTTP niveau: HTTP Request (met SOAP Envelope (met QueryByParameter))
 HTTP Response (met SOAP Envelope (met QueryResponse))

(Onderstaand plaatje laat dit zien voor een Query met QueryByParameter. responsePriorityCode "Immediate" en Message.acceptAckCode "NE", zie verderop voor toelichting.)



Dit is een betrouwbare aflevering over het Internet: als er geen HTTP 200 OK antwoord ontvangen wordt, weet de Zender dat de communicatie gefaald heeft, en kan hij het opnieuw proberen. Er zijn geen HL7 Accept Acknowledgements (berichtontvangstbevestigingen) nodig. Deze werkwijze gaat alleen goed bij **idempotente** berichten: berichten waarop bij herhaling van het bericht telkens hetzelfde antwoord komt, en die geen neveneffecten (zoals een gewijzigde toestand) op de server hebben. Het antwoordbericht komt synchron terug op het verzoekbericht.

BT-15 Methode 2 **moet** worden toegepast indien aan alle onderstaande criteria voldaan wordt:

- a) de interactiecode van de te verzenden interactie ongelijk is aan MCCI_IN200100 (Batch)¹
- b) er een QueryByParameter XML-Tag in de interactie aanwezig is (het is een query),
- c) de waarde van QueryByParameter.responsePriorityCode "I" (Immediate) is.
- a) de waarde van Message.acceptAckCode "NE" (Never) is.

BT-16 Methode 2 **moet** ook worden toegepast wanneer:

- a) er een QueryContinuation XML-Tag in de interactie aanwezig is (het is een queryvervolgvrage).
- b) voor de bijbehorende QueryByParameter (QueryByParameter.queryId = QueryContinuation.queryId) vraag waren alle 4 bovenstaande punten van toepassing.

¹ Er zijn op het moment van schrijven geen batch-interacties MCCI_IN200100 in AORTA (er zijn alleen batch antwoorden, MCCI_IN200101), deze inperking is dus alleen voor de toekomst.

c) de waarde van `Message.acceptAckCode` "NE" (Never) is.

BT-17 Een `QueryByParameter` en alle bijbehorende `QueryContinuation` vragen dienen alle dezelfde methode te volgen.

De `QueryContinuation` zelf bevat niet de waarde van `reponsePriorityCode`. De ontvanger kan dit bijhouden in een tabel (dat zal toch moeten gebeuren voor de beoordeling van de betrouwbaarheid). Voor het afleveren van een antwoord mag de ontvanger ook gewoon kijken naar de waarde van `Message.acceptAckCode`, en wanneer deze "NE" is, synchroon antwoorden. Deze waarde mag ook alleen "NE" zijn wanneer de oorspronkelijke query Immediate was.

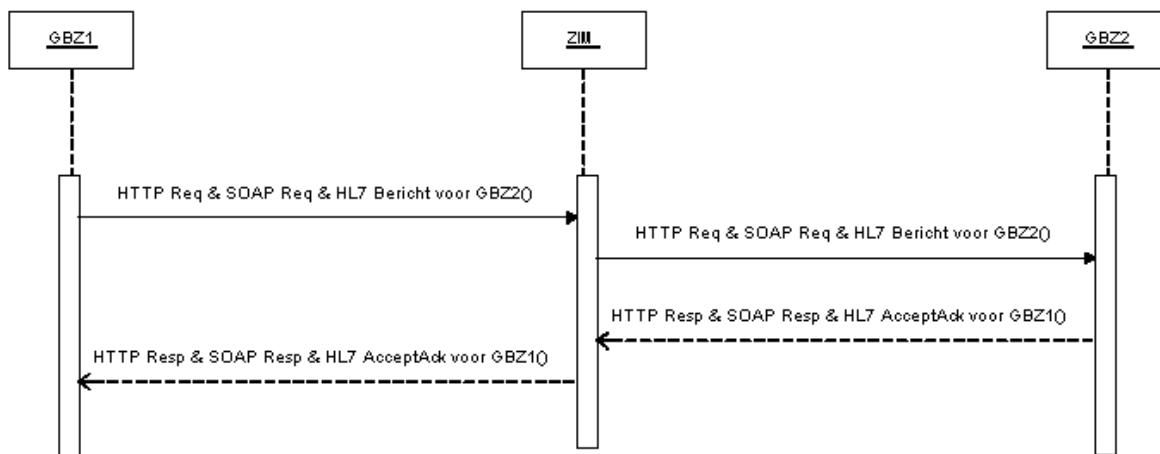
BT-18 Antwoordberichten op synchrone queries (zoals in het bovenstaande voorbeeld) mogen zelf nooit om een accept acknowledgement vragen.

Er is immers geen transport over een HTTP response mogelijk, omdat het antwoordbericht zelf al een HTTP response is. Dat betekent dat de waarde van `Message.acceptAckCode` "NE" moet zijn. (Dit betekent tevens dat fouten op synchrone antwoordberichten niet met een accept acknowledgement gedaan kunnen worden, en dat bijvoorbeeld syntactische problemen niet op deze wijze teruggemeld kunnen worden. Deze situatie is parallel aan die van accept acknowledgements zelf, die ook geen accept acknowledgement vragen, en waarvan syntactische issues dus ook niet op deze wijze teruggemeld kunnen worden. Bij synchrone vragen dient de partij die de vraag gesteld heeft actie te ondernemen wanneer er onverwachte syntactische issues met het antwoord zijn.)

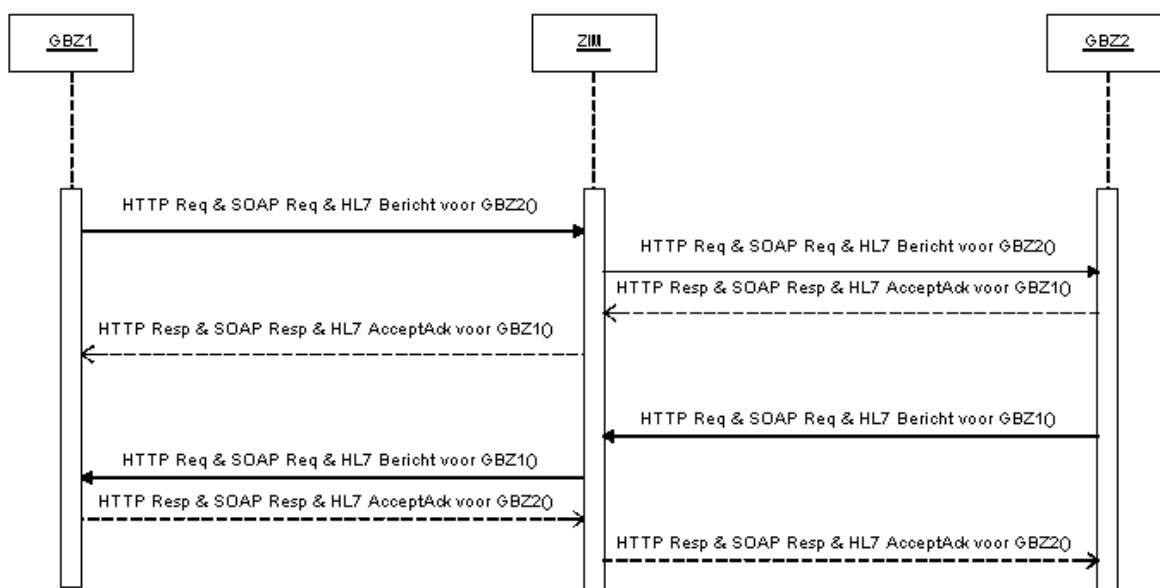
Dit zijn tevens de enige combinaties waarbinnen gebruik van `Message.acceptAckCode` "NE" binnen AORTA is toegestaan. Voor alle andere interacties dient methode 1 gebruikt te worden, en dus `Message.acceptAckCode` "AL". (Het is wel toegestaan `QueryByParameter.reponsePriorityCode` "I" een `Message.acceptAckCode` "AL" te gebruiken. Dit levert een interactie volgens Methode 1 op. Aanbevolen wordt echter Immediate queries volgens Methode 2 te implementeren.)

Ieder uitgaand bericht krijgt dus óf direct een Accept Acknowledgement óf direct een Application Response (een inhoudelijk antwoord). Het bericht en het antwoord worden weergegeven als HTTP Request – Response paar. Dit houdt in dat alle berichten een expliciete of impliciete Accept Acknowledgement hebben (een Application Response is een impliciete Accept Acknowledgement). Wanneer een GBZ met een ZIM communiceert, zal de Accept Acknowledgement redelijk snel moeten kunnen komen. En wanneer de ZIM niets ontvangt, zal het GBZ dat toch minimaal willen weten. Op deze wijze kunnen Immediate berichten uitgevoerd worden als een simpel "HTTP Request – Response paar" wat voordelen heeft in de snelheid van de uitvoering en de eenvoud van de interactie.

De partij die de Accept Acknowledgement levert is de geadresseerde (Receiver) zoals gespecificeerd in de Transmission Wrapper. Dat kan de ZIM zijn (zoals in de voorbeelden boven), maar het kan ook een ander GBZ zijn. Dit laatste is het geval bij berichten die niet aan de ZIM gericht zijn, maar aan dat andere GBZ.



Wanneer er een HL7 bericht verzonden wordt waarop een HL7 antwoord komt van een ander GBZ, ziet de totale interactie er als volgt uit.



Het voornaamste nadeel is dat de interacties anders lopen voor Immediate en Deferred Queries. Wanneer er dus één en dezelfde query is die zowel een Immediate als Deferred antwoord kan krijgen, zal deze in twee verschillende HTTP interacties op kunnen opleveren. De ontvanger van een verzoekbericht zal bij queries dus de waarde van de Message.acceptAckCode moeten inspecteren om te kunnen zien of met een antwoordbericht (bij Message.acceptAckCode = "NE") of een Accept Acknowledgement (bij Message.acceptAckCode = "AL") moet worden geantwoord.

Bij het maken van deze specificatie is er voor gekozen voor queries beide varianten te ondersteunen.

Het protocol biedt de volgende zekerheden (respons wordt gebruikt voor alle soorten antwoorden, zowel applicatie-antwoorden als ontvangstbevestigingen):

- Bij ontvangst van de respons weet de verzender 100% zeker dat het bericht tenminste 1 keer is afgeleverd.
- Bij het niet ontvangen van de respons weet de verzender niets zeker over de aflevering van het bericht, maar kan hij opnieuw proberen het bericht te verzenden (automatisch / telefoon / papieren bericht zenden o.i.d.). Wanneer de verzender het niet opnieuw probeert:
 - mag de verzender er niet van uitgaan dat de ontvanger het bericht ontvangen heeft (de verzender mag niet rekenen op bepaalde acties aan de zijde van de ontvanger);
 - moet de verzender er wel rekening mee houden dat de ontvanger het bericht ontvangen heeft (de verzender mag er niet op rekenen dat de ontvanger bepaalde acties achterwege laat).

Niet opnieuw proberen is acceptabel voor alle interacties die geen inhoudelijke gevolgen aan de kant van de ontvanger hebben, bijvoorbeeld het opvragen van gegevens.

- Bij ontvangst van het bericht weet de ontvanger 100% zeker dat het is ontvangen (tautologie).
- De ontvanger weet nooit zeker of de verzender de respons heeft ontvangen (dit kan door ontvangstbevestiging op respons te vragen, waarvoor niet gekozen is). Wanneer er geen kopieberichten komen, mag de ontvanger er van uitgaan dat de verzender weet dat de ontvanger het bericht heeft ontvangen (het initiatief ligt bij de verzender wanneer deze geen respons ontvangt).

5.5 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 dus fouten op de volgende niveaus:

- HL7v3 fouten
- SOAP fouten
- HTTP fouten

5.5.1 HTTP foutsituaties

BT-19 Het WS-I Basic Profile definieert een aantal richtlijnen voor HTTP fouten bij gebruik van SOAP. AORTA volgt deze.

De meest relevante zijn:

Code	Basic Profile tekst	Toelichting
R1140	A MESSAGE SHOULD be sent using HTTP/1.1.	
R1141	A MESSAGE MUST be sent using either HTTP/1.1 or HTTP/1.0	Bij gebruik van andere HTTP versies dient een HTTP error 505 "HTTP Version not supported" geretourneerd te worden.
R1107	A RECEIVER MUST interpret SOAP messages containing only a soap:Fault element as a Fault.	Een applicatie mag dus niet alleen naar HTTP 200 OK kijken voor het bepalen van het succes. Hetzelfde geldt voor

Code	Basic Profile tekst	Toelichting
		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 alleen HTTP 200 OK te gebruiken bij succesvolle verwerking; ontvangers dienen echter HTTP responses met andere succes statussen te accepteren.
R1124	An INSTANCE MUST use a 2xx HTTP status code for responses that indicate a successful outcome of a request.	
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 te behandelen. Met name de HTTP "408 Request Timeout" dient behandeld te worden (door een hernieuwde poging of een foutmelding aan de gebruiker, afhankelijk van de context, zoals aantal reeds gedane pogingen). Verdere behandeling van HTTP verkeer valt buiten de scope van dit document.

5.5.2 SOAP Faults

SOAP onderkent de volgende Faults:

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 <code>http://schemas.xmlsoap.org/soap/envelope/</code> MOET deze Fault gegenereerd worden.
MustUnderstand	Er is een SOAP Header aanwezig die niet bekend is, maar met <code>mustUnderstand = "1"</code> . In dit geval MOET deze Fault gegenereerd worden.
Client	Deze Faults worden alleen gebruikt bij syntaxfouten in het SOAP bericht die geen syntaxfouten zijn in de HL7 payload. Voor syntaxfouten in de HL7v3 payload gebruikt AORTA HL7v3 foutberichten (Accept Acknowledgements). Een voorbeeld is een SOAP Envelope waar de SOAP Body ontbreekt. 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 ap-

Naam	Toelichting
	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.

SOAP "VersionMismatch" en "MustUnderstand" Faults zouden in de eerste release van AORTA niet voor mogen komen, omdat de SOAP versie altijd 1.1 is, en er geen SOAP Headers gebruikt worden.

BT-20 Om de uitbreidbaarheid van SOAP te ondersteunen moeten de Faults "VersionMismatch" en "MustUnderstand" wel gegenereerd kunnen worden door een GBZ.

Alleen op deze wijze kan bij een eventuele toekomstige migratie naar een andere SOAP versie of het gebruik van Headers gegarandeerd worden dat de huidige systemen, die dat niet ondersteunen, een juiste foutmelding geven aan het verzendende systeem dat zo'n nieuwe versie wel ondersteunt. Omdat de Faults in de praktijk voorlopig niet op zullen treden, moet het genereren van deze Faults in testscenario's meegenomen worden.

BT-21 Bij een 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 echter 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. Het is bijvoorbeeld geen verplichting eerst het hele XML document te parsen om op wellformedness te controleren voordat verdere verwerking plaatsvindt. Bij niet well-formed XML hoeft geen SOAP Fault of HL7v3 error gegeven te worden.

BT-22 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.

BT-23 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 HL7 reactie voorhanden is, heeft de volgende reactie de voorkeur:

- als er een fout is in de SOAP syntax, (b.v. soap:Envelope zonder soap:Body of soap: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 problemen zijn die te wijten zijn aan de server, een reactie met HTTP Status 500 'Internal Server Error', eventueel gevuld met een SOAP Server Fault.

BT-24 Bij foutafhandeling dient het principe gevolgd te worden: "ben strikt in wat je zendt, en tolerant in wat je ontvangt". De afhandeling als hier geschetst heeft

de voorkeur, bij ontvangst van onverwachte fouten dient een applicatie echter adequaat te reageren, b.v. door de fout te loggen voor menselijke afhandeling.

RFC 2616 (HTTP/1.1) beveelt het gebruik van een entity (HTTP berichtinhoud) toe bij een respons 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.

5.5.3 HL7v3 fouten

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

BT-25 HL7v3 Application en Accept Acknowledgements worden behandeld als succes op HTTP niveau.

Code	Naam	Toelichting (zie HL7v3 voor details)	HTTP Status
AA	Application Acknowledgement Accept	Succesvolle verwerking.	200 OK
AE	Application Acknowledgement Error	Fout in bericht.	200 OK
AR	Application Acknowledgement Reject	Fout, maar niet in inhoud of formaat van bericht.	200 OK
CA	Accept Acknowledgement Commit Accept	Geaccepteerd.	200 OK
CE	Accept Acknowledgement Commit Error	Niet geaccepteerd.	200 OK ²
CR	Accept Acknowledgement Commit Reject	Niet geaccepteerd.	200 OK

5.6 HTTP Redirects

BT-26 Het gebruik van HTTP redirects is niet toegestaan.

In de internationale standaarden die we voorschrijven, met name [Basic Profile], staat dat alleen 307 gebruikt mag worden, en de betekenis daarvan is "temporary redirect". De rest is expliciet verboden. De achterliggende reden is dat er interoperabiliteitsproblemen zijn met veel producten. In het geval van AORTA is er voor gekozen ook geen 307 redirects te gebruiken.

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 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.

In het [Basic Profile] staat:

"4.3.7 HTTP Redirect Status Codes

There are interoperability problems with using many of the HTTP redirect status codes, generally surrounding whether to use the original method, or GET. The Profile mandates "307 Temporary Redirect", which has the semantic of redirection with the same HTTP method, as the correct status code for redirection. For more information, see the 3xx status code descriptions in RFC2616."

En in [HTTP] staat:

"10.3.8 307 Temporary Redirect

The requested resource resides temporarily under a different URI. Since the redirection MAY be altered on occasion, the client SHOULD continue to use the Request-URI for future requests."

Een GBZ mag dus standaard zeker geen andere 3xx redirects gebruiken. In AORTA is ervoor gekozen ook geen 307-redirects te gebruiken. Er lijkt geen goede reden te zijn om dit wel te doen. Van een SOAP-implementatie mag verwacht worden dat de URI's waarop de service te bereiken is zo te configureren zijn dat ze voldoen aan de eisen die AORTA daar aan stelt. Daarnaast is het mogelijk aan serverzijde zelf de binnenkomende requests door te routeren naar iedere gewenste fysieke of logische locatie. Redirects zijn feitelijk een manier om dit door de client te laten doen. Er is wel een goede reden het niet toe te staan: er wordt nergens van SOAP-implementaties geëist dat ze 307-redirects ondersteunen. Dat betekent dat deze niet gebruikt kunnen worden zonder dat eerst de ZIM en alle GBZ'en deze ondersteunen op een voor de gebruiker transparante wijze.

6 WSDL

BT-27 Beschrijvingen van berichttransport worden opgesteld in WSDL 1.1. Deze WSDL is normatief.

6.1 AORTA en WSDL

In de AORTA context komt er uiteindelijk een WSDL beschrijving van iedere aanbieder die een bepaalde Application Role vervult. Van de WSDL uit voorgaande paragraaf zal dus een versie komen voor een ZIM die het opvragen van medicijnen ondersteunt, met daarin het webadres van die webservice op de ZIM, en een versie voor iedere zorgleverancier die het opvragen van medicijnen ondersteunt. Deze WSDL beschrijvingen zullen gelijk zijn in alles behalve het webadres in de service. (Door gebruik van WSDL modules kan uiteraard het generieke deel van de webservice door iedereen gedeeld worden.) Voor AORTA zal dus een "template" WSDL beschrijving gemaakt worden van iedere HL7 Application Role die in het repertoire van in Nederland gebruikte uitwisselingen past. Voor het opvragen van medicatie komt er dus een "Verstrekingsquery.wsdl" bestand waarin delen leeg zijn, met name het webadres van de service. Voor de ZIM en voor ieder GBZ komt er dan een specifieke invulling van deze WSDL waarin het webadres van deze ZIM is toegevoegd. Stel dus dat er een ZIM met het (fictieve) adres <http://www.zim.nl> komt, dan zal deze ZIM een document "Verstrekingsquery.wsdl" publiceren met daarin het eigen webadres voor opvragingen. Implementaties die deze Application Role bij de ZIM willen aanroepen, b.v. applicaties voor behandelend specialisten die medicijngebruik van een patiënt willen opvragen, kunnen dan deze WSDL gebruiken om de code te ontwerpen (deels te genereren) om de ZIM aan te roepen. Iedere implementatie die de Application Role vervult, dus bijvoorbeeld een applicatie voor apothekers, zal een eigen "Verstrekingsquery.wsdl" publiceren, met het webadres van de service van die apotheker. De ZIM gebruikt die WSDL dan om deze service aan te roepen. In de praktijk zal de ZIM een tabel gebruiken met daarin de geleverde services en het webadres van een gegevensleverancier, en zal de WSDL als documentatie en contract dienen, omdat de WSDL precies beschrijft aan wat voor service de gegevensleverancier zich gecommitteerd heeft. Bij ieder WSDL bestand is ook specifieke documentatie in HTML aanwezig.

BT-28 Ieder GBZ, en de ZIM, dienen een eigen variant op de WSDL te maken met daarin als enige aanpassing de "location" van de service.

6.2 HL7v3 Application Roles, berichten en WSDL

De uitwerking maakt gebruik van dezelfde voorbeeldtransactie als hierboven gebruikt, Verstrekingsquery. Volgens het interactiediagram bestaat deze transactie uit:

- de query QURX_IN990011NL (Medication Dispense Event Query)
- het antwoord QURX_IN990013NL (Medication Dispense Event Response)
- de vervolgvraag QUQI_IN000003 (General Query Activate Query Continue)
- de ontvangstbevestiging MCCI_IN200101 (Accept Acknowledgement) (deze worden niet getoond in de interactiediagrammen).

Een vervolgvraag met QueryContinuation. continuationQuantity > 0 is een queryContinuation verzoek: de vraag om meer antwoorden, een vervolgvraag met

QueryContinuation. continuationQuantity = 0 is een queryCancel, een afbreking van de logische query sessie.

De verschillende berichten krijgen in HL7 een eigen XML Schema. Zo is er een XML Schema QURX_IN990011NL.XSD, QURX_IN990013NL.XSD et cetera. In de XML Schema's ligt de precieze structuur van het bericht vast, inclusief de HL7 Transmission en Control Act Wrappers.

Sommige Application Roles krijgen in de HL7 standaard een WSDL-beschrijving. Deze WSDL beschrijving is een beschrijving van de Web Service. Dit gebeurt voor alle Application Roles die als server functioneren, in dit geval Verstrekkingquery, de Medication Dispense Event Query Fulfiller, ofwel de instantie die opvragingen naar medicijngebruik kan beantwoorden (dit kan de ZIM of een GBZ zijn). In deze WSDL beschrijving worden allereerst benodigde XML Schema's gedeclareerd als WSDL Types. (Hieronder volgt een verkorte uitleg. Dit is een voorbeeld: de WSDL die bij AORTA wordt gepubliceerd is mogelijk recenter, en leidend.)

```
<?xml version="1.0" encoding="utf-8"?>
<definitions targetNamespace="urn:hl7-org:v3"
  name="Verstrekkingquery"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <documentation> WSDL implementatie van Verstrekkingquery </documentation>
  <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/QURX_IN990011NL.xsd'/>
      <xsd:include schemaLocation='../schemas/QURX_IN990013NL.xsd'/>
      <xsd:include schemaLocation='../schemas/QUQI_IN000003.xsd'/>
      <xsd:include schemaLocation='../schemas/MCCI_IN200101.xsd'/>
    </xsd:schema>
  </types>
```

Eerst is er een inleidend deel, daarna wordt ieder Schema gedeclareerd. De Schema's worden opgenomen met een Schema-include om een en ander beknopt en leesbaar te houden. Er zijn 4 Schema's:

- QURX_IN990011NL.xsd (Medication Dispense Event Query)
- QURX_IN990013NL.xsd (Medication Dispense Event Response)
- QUQI_IN000003.xsd (General Query Activate Query Continue)
- MCCI_IN200101.xsd (Batch Wrapper)

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

```
<message name='QURX_IN990011NL'>
  <part name='body' element='hl7:QURX_IN990011NL'/>
</message>
<message name='QURX_IN990013NL'>
  <part name='body' element='hl7:QURX_IN990013NL'/>
</message>
<message name='QUQI_IN000003'>
```

```

    <part name='body' element='hl7:QUQI_IN000003' />
  </message>
  <message name='MCCI_IN200101'>
    <part name='body' element='hl7:MCCI_IN200101' />
  </message>

```

Er zijn ook vier verschillende berichten:

- QURX_IN990011NL.xsd (Medication Dispense Event Query)
- QURX_IN990013NL.xsd (Medication Dispense Event Response)
- QUQI_IN000003.xsd (General Query Activate Query Continue)
- MCCI_IN200101.xsd (Batch Wrapper)

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='Verstrekkingquery_PortType'>
  <operation name='Verstrekkingquery_QueryResponse'>
    <input message='hl7:QURX_IN990011NL' />
    <output message='hl7:QURX_IN990013NL' />
  </operation>
  <operation name='Verstrekkingquery_ContinuationCancelResponse'>
    <input message='hl7:QUQI_IN000003' />
    <output message='hl7:QURX_IN990013NL' />
  </operation>
</portType>
<portType name='VerstrekkingqueryBatch_PortType'>
  <operation name='VerstrekkingqueryBatch_QueryResponse'>
    <input message='hl7:QURX_IN990011NL' />
    <output message='hl7:MCCI_IN200101' />
  </operation>
  <operation name='VerstrekkingqueryBatch_ContinuationCancelResponse'>
    <input message='hl7:QUQI_IN000003' />
    <output message='hl7:MCCI_IN200101' />
  </operation>
</portType>

```

Hierboven zijn de twee operations getoond volgens het synchrone model van betrouwbaarheid: het antwoord op de query komt synchroon binnen over hetzelfde SOAP en HTTP request/response paar. Dit levert twee operations op:

- Verstrekkingquery_QueryResponse
- Verstrekkingquery_ContinuationCancelResponse

(Batch interacties hebben een eigen portType en operations.)

Deze bestaan weer uit de eerder gedefinieerde berichten.

In sommige gevallen is bij een HL7v3 interactie die verzonden wordt meer dan een interactie als antwoord mogelijk. WSDL staat dit niet toe: Iedere WSDL-operation mag maximaal een input en een output hebben. Wanneer de HL7v3 antwoorden later verzonden worden, is dit geen probleem: de antwoorden worden dan aparte, losstaande WSDL operations. Wanneer echter de antwoorden direct komen, als HTTP respons op een HTTP request, kan dit niet meer met WSDL. De oplossing is een samengesteld document 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:include schemaLocation="./schemas/COMT_IN800320.xsd" />
    <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. Met name wordt hier ook gespecificeerd dat er een "document/literal" stijl van gegevensuitwisseling gebruikt wordt. In SOAP bestaat de keuze tussen "document", waarbij de SOAP Body direct het topelement van een XML document bevat, en "rpc", waarbij de SOAP Body een element bevat wat correspondeert met een "method", en pas daaronder gegevens. Het verschil tussen "document" en "rpc" is niet erg groot. Daarnaast kan gekozen worden voor "literal", waarbij de XML-elementen gegevens bevatten die buiten de SOAP, b.v. in een XML Schema beschreven worden. Het alternatief is "encoded", waarbij de XML elementen volgens bepaalde regels geserialiseerd worden. Vroege SOAP applicaties gebruikten meestal "rpc/encoded", met name omdat XML Schema toen nog niet bestond. Een "rpc/encoded" aanpak past beter bij een web service die het karakter heeft van een functieaanroep, b.v. het opvragen van de koers van een beursfonds. Een "document/literal" aanpak past beter bij het uitwisselen van documenten, wat ook bij AORTA het geval is. Het verschil tussen "rpc/encoded" en "document/literal" is een verschil van stijl en eenvoud, het is niet zo dat met de ene aanpak zaken kunnen die met de andere aanpak niet kunnen. In de Binding wordt ook de waarde van SOAPAction bepaald. Dit is een HTTP Header. De waarde is bedoeld als indicatie van de inhoud. Deze wordt op de interactie gezet. Bij grotere zorginstellingen kan de HTTP Server deze waarde gebruiken om het bericht door te routeren naar het juiste verwerkende systeem zonder het bericht te hoeven openen en de XML te parsen.

```
<binding type='hl7:Verstrekingsquery_PortType' name='Verstrekingsquery_Binding'>
  <soap:binding style='document' transport='http://schemas.xmlsoap.org/soap/http' />
  <operation name='Verstrekingsquery_QueryResponse'>
```

```

    <soap:operation soapAction='urn:hl7-org:v3/Verstrekkingquery_QueryResponse'/>
    <input>
      <soap:body use='literal'/>
    </input>
    <output>
      <soap:body use='literal'/>
    </output>
  </operation>
  <operation name='Verstrekkingquery_ContinuationCancelResponse'>
    <soap:operation soapAction='urn:hl7-
org:v3/Verstrekkingquery_ContinuationCancelResponse'/>
    <input>
      <soap:body use='literal'/>
    </input>
    <output>
      <soap:body use='literal'/>
    </output>
  </operation>
</binding>
<binding type='hl7:VerstrekkingqueryBatch_PortType'
name='VerstrekkingqueryBatch_Binding'>
  <soap:binding style='document' transport='http://schemas.xmlsoap.org/soap/http'/>
  <operation name='VerstrekkingqueryBatch_QueryResponse'>
    <soap:operation soapAction='urn:hl7-org:v3/VerstrekkingqueryBatch_QueryResponse'/>
    <input>
      <soap:body use='literal'/>
    </input>
    <output>
      <soap:body use='literal'/>
    </output>
  </operation>
  <operation name='VerstrekkingqueryBatch_ContinuationCancelResponse'>
    <soap:operation soapAction='urn:hl7-
org:v3/VerstrekkingqueryBatch_ContinuationCancelResponse'/>
    <input>
      <soap:body use='literal'/>
    </input>
    <output>
      <soap:body use='literal'/>
    </output>
  </operation>
</binding>

```

Tenslotte wordt van de WSDL Bindings een WSDL Service gemaakt, die het webadres van een specifieke webservice geeft:

```

<service name='Verstrekkingquery_Service'>
  <port binding='hl7:Verstrekkingquery_Binding' name='Verstrekkingquery_Port'>
    <!-- Deze service location URI verwijst niet naar een echt bestaande webservice. -->
    <soap:address location='http://www.xis.nl/Verstrekkingquery'/>
  </port>
  <port binding='hl7:VerstrekkingqueryBatch_Binding' name='VerstrekkingqueryBatch_Port'>
    <!-- Deze service location URI verwijst niet naar een echt bestaande webservice. -->
    <soap:address location='http://www.xis.nl/VerstrekkingqueryBatch'/>
  </port>
</service>
</definitions>

```

De waarde van "location" is hier nog fictief. Iedere ZIM en GBZ die de webservice Verstrekkingsquery implementeert, zal hier een eigen waarde in moeten vullen. De padnamen zijn verplicht: <http://www.gbz.nl/Verstrekkingsquery> is goed, <http://www.gbz.nl/Verstrekkingsquery.py> of <http://www.gbz.nl/MijnVerstrekkingsquery> is dat niet. Dit om te voorkomen dat de ZIM een registratie moet bijhouden van alle padnamen, met de bijbehorende plicht voor GBZ'en om deze aan te melden en wijzigingen door te geven.³

BT-29 De waarden van SOAPAction zijn verplicht.

De WSDL beschrijft hier bijvoorbeeld waarden als:

```
soapAction="urn:hl7-org:v3/Verstrekkingsquery_ContinuationCancelResponse"
```

Deze waarde is verplicht: ze worden letterlijk in de WSDL genoemd, een GBZ en de ZIM dienen deze waarden zo te implementeren. 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.

De verschillende URI's en SOAPActions zijn bewust gekozen zodat een ontvangende webserver bij een GBZ met meerdere fysieke achterliggende systemen eventuele routing en scheduling kan doen zonder in de XML te hoeven kijken, dus zonder de kosten van parsing. Het is een optie, geen plicht. Een GBZ of de ZIM mag de binnenkomende SOAPAction header dus negeren. Bij uitgaande berichten moet wel een correcte SOAPAction header geplaatst worden.

Het WS-I Basic Profile bevat een aanbeveling:

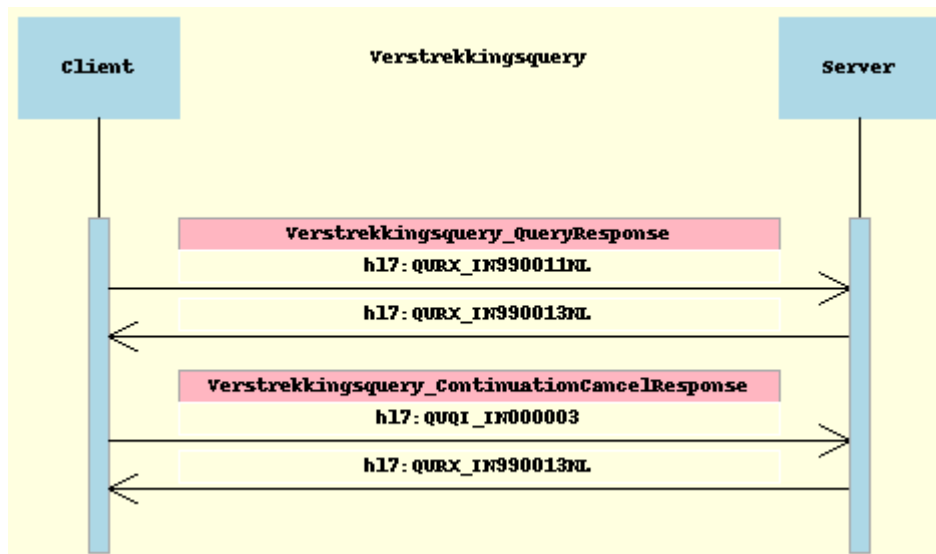
"R2710: The operations in a wsdl:binding in a DESCRIPTION MUST result in wire signatures that are different from one another. The Profile defines the "wire signature" of an operation in a wsdl:binding to be the fully qualified name of the child element of the soap:Body of the SOAP input message it describes."

Dit betekent dat iedere operation een input moet hebben met een unieke naam voor het eerste element in de SOAP Body. HL7v3 voldoet niet aan deze voorwaarde.

BT-30 AORTA volgt aanbeveling R2710 uit het Basic Profile niet.

Bij ieder WSDL bestand wordt ook documentatie in HTML formaat geleverd. Dit bevat een grafische weergave van de communicatie op HTTP-niveau:

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



en een overzicht in tabelvorm van alle relevante parameters:

Parameter	Waarde
WSDL:	Verstrekkingsquery.wsdl
Service:	Verstrekkingsquery_Service
XSD:	QURX_IN990011NL.xsd
	QURX_IN990013NL.xsd
	QUQI_IN000003.xsd
	MCCI_IN200101.xsd
Service Location	http://www.xis.nl/Verstrekkingsquery
Operation	Verstrekkingsquery_QueryResponse
Input	h17:QURX_IN990011NL
Output	h17:QURX_IN990013NL
SOAP Action	"urn:hl7-org:v3/Verstrekkingsquery_QueryResponse"
Operation	Verstrekkingsquery_ContinuationCancelResponse
Input	h17:QUQI_IN000003
Output	h17:QURX_IN990013NL
SOAP Action	"urn:hl7-org:v3/Verstrekkingsquery_ContinuationCancelResponse"

7 Bijlage: Voorbeelden

Deze bijlage beschrijft de voorbeelden voor transport conform de AORTA-infrastructuur. De voorbeelden in deze bijlage zijn niet normatief; waar er conflict is tussen de voorbeelden en de tekst in de implementatiehandleiding, heeft de tekst voorrang.

De voorbeelden voor het transport zijn gesplitst in twee delen.

Statische voorbeelden. Deze beschrijven de voorbeelden die gesteld worden aan losse AORTA-artefacten op HTTP, XML, SOAP en betrouwbaarheid. Alles wat binnen HL7v3 Transmission Wrapper niveau gebeurt heet vanuit de optiek van dit document "lading", en de inhoud daarvan wordt niet beschouwd (met uitzondering van voorbeelden die aan het XML document *als geheel* worden gesteld).

Dynamische voorbeelden. Deze voorbeelden beschrijven de voorbeelden die gesteld worden aan een applicatie die volgens de AORTA-infrastructuur communiceert met een andere applicatie. Als zodanig heeft dit deel van de voorbeelden meestal betrekking op meerdere AORTA-artefacten, bijvoorbeeld een verzoek- en een antwoordbericht. Dynamische voorbeelden beschrijven dus een interactie tussen twee volgens AORTA communicerende systemen.

Er zijn dus ook maar twee rollen: de agerende applicatie en de reagerende applicatie. Omdat de voorbeelden op diversie niveaus spelen: HTTP, XML, SOAP en HL7, zijn de rollen hiermee verfijnd. Voorbeelden worden gesteld aan één of meer rollen.

Onderscheiden rollen zijn:

HTTP agerende applicatie
HTTP reagerende applicatie
XML agerende applicatie
XML reagerende applicatie
SOAP agerende applicatie
SOAP reagerende applicatie
HL7 agerende applicatie
HL7 reagerende applicatie

Met agerende applicatie wordt telkens bedoeld de initiërende HL7 applicatie, ook in gevallen waar er geen HL7-respons komt. Niet alle voorbeelden zijn relevant voor alle rollen, per voorbeeld is aangegeven welke rol hier aan moet voldoen.

7.1 Statische voorbeelden

De statische voorbeelden van het transport worden gesplitst naar HTTP(S), XML, SOAP en HL7v3 niveau. Er worden geen voorbeelden van lagere niveaus (TCP, IP, et cetera) gemaakt. De statische voorbeelden zijn beschreven als fragmenten van AORTA-artefacten. B.v. voor voorbeelden op HTTP-niveau wordt niet de hele XML getoond, voor voorbeelden op HL7 niveau zijn de HTTP-Headers niet relevant. In alle voorbeelden is daarom het eerste deel dan wel het laatste deel van een bestand dan wel beide weggelaten. Complete voorbeelden met HL7 inhoud zijn veel te lang; HTTP Headers zijn meestal

niet relevant. Alle voorbeelden zijn dus fragmenten die uit een geheel aantal complete regels bestaan. Drie puntjes (...) geven aan waar nog meer tekst moet volgen.

De voorbeelden kunnen niet gelezen worden als bit-voor-bit of karakter-voor-karakter inputdocumenten. HTTP, XML en SOAP staan een aantal vrijheden toe, b.v.:

- er kunnen meer HTTP Headers voorkomen
- HTTP regeleinden bestaande uit een enkele linefeed moeten geaccepteerd worden, ook al wordt carriage return & linefeed aanbevolen
- volgorde van attributen in XML is niet relevant
- whitespace tussen elementen in XML is veelal niet relevant
- namespace afkortingen kunnen variëren per document.

Een ontwikkelaar die een bepaalde HTTP server, XML ontwikkelomgeving of SOAP toolkit gebruikt heeft vaak helemaal niet de vrijheid hier zelf keuzes in te maken. Daarom moet bij gebruik van de voorbeelden alleen gelet worden op de relevante zaken zoals in de voorbeeld geformuleerd. Kennis van HTTP, XML en SOAP is noodzakelijk.

De statische voorbeelden zijn weergegeven als goede en foute varianten van AORTA-artefacten. Hierdoor is het eenvoudig de relevante fouten te zien: wanneer het bijvoorbeeld gaat om de juiste HTTP status (200 OK voor een verwachte respons), is de precieze waarde van het OID van Sender niet zo belangrijk. Het verschil tussen de foute en goede antwoorden geeft precies aan waar het in dit voorbeeldscenario om gaat, hoewel er uiteraard veel meer goede en foute varianten zijn.

7.1.1 HTTP

Voorbeelden op HTTP niveau. De input voor deze voorbeelden is een document zoals dat over een secure TCP verbinding (SSL/TLS) wordt afgeleverd.

7.1.1.1 De artefact moet een geldig HTTP 1.1 document zijn

Rol	HTTP agerende applicatie en reagerende applicatie
-----	---

FOUT

```
POST / HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

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

FOUT

```
HTTP/2.0 200 OK
Content-type: text/xml; charset="utf-8"

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

GOED

```
POST / HTTP/1.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

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

GOED

```
HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-8"

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

Noot: Aanwezigheid van andere HTTP headers is voor dit voorbeeld niet relevant.

7.1.1.2 De HTTP method moet POST zijn

Rol | HTTP agerende applicatie

FOUT

```
PUT / HTTP/1.1  
Content-Type: text/xml; charset=utf-8  
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"
```

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

FOUT

```
GET / HTTP/1.1  
Content-Type: text/xml; charset=utf-8  
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"
```

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

GOED

```
POST / HTTP/1.1  
Content-Type: text/xml; charset=utf-8  
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"
```

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

Noot: Aanwezigheid van andere HTTP headers is voor dit voorbeeld niet relevant.

7.1.1.3 Content moet in UTF-8 formaat zijn

Rol | HTTP agerende applicatie en reagerende applicatie

FOUT

```
POST / HTTP/1.1
Content-Type: text/xml; charset=iso-8859-1
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

<?xml version="1.0" encoding="iso-8859-1"?>
...
```

FOUT

```
HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-16"

<?xml version="1.0" encoding="utf-16"?>
...
```

GOED

```
POST / HTTP/1.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

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

GOED

```
HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-8"

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

Noot: Aanwezigheid van andere HTTP headers is voor dit voorbeeld niet relevant.

7.1.1.4 Diakritische tekens moeten in UTF-8 geëncodeerd zijn

Rol | HTTP agerende applicatie en reagerende applicatie

Ook moet getest worden of diakritische en andere tekens goed overgebracht worden. Dat kan met de volgende tekst:

€ of døllär

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

Deze dient exact zo in het bronsysteem van de zender ingevoerd te worden in een veld wat 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. Dat kan veelal niet met een tekstgebaseerde editor, maar moet met een tool dat XML en Unicode ondersteunt, bijvoorbeeld een recente versie van Firefox of Internet Explorer. De tekst is zo gekozen omdat ze de verschillen tussen de meestgebruikte encodings, UTF-8, ISO-8859-1, ISO8859-15 en Windows-1252 blootlegt.

Hexadecimale representatie van enkele diakritische tekens in diverse encodings			
	Euroteken (€)	Kleine letter o met 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) in het optionele element <softwareName> in de Transmission Wrapper. Dit komt in alle berichten voor, maar wordt veelal niet gebruikt in testen.

GOED

```
<?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>
...
```

7.1.1.5 Geldige HTTP 1.1 statussen dienen geaccepteerd te worden

Rol | HTTP reagerende applicatie

Zie RFC 2616 (<http://www.w3.org/Protocols/rfc2616/rfc2616.txt>) voor een overzicht van HTTP statussen. HTTP Redirect statussen zijn niet toegestaan.

FOUT

```
HTTP/1.1 9999 OK
Content-type: text/xml; charset="utf-8"

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

FOUT

```
HTTP/1.1 307 Temporary Redirect
Content-type: text/xml; charset="utf-8"

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

FOUT

```
HTTP/1.1 302 Found
Content-type: text/xml; charset="utf-8"

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

GOED

```
HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-8"

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

GOED

```
HTTP/1.1 400 Bad Request
Content-type: text/xml; charset="utf-8"

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

GOED

```
HTTP/1.1 500 Internal Server Error
Content-type: text/xml; charset="utf-8"

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

Noot: Aanwezigheid van andere HTTP headers is voor dit voorbeeld niet relevant.

7.1.1.6 Een HL7 respons krijgt HTTP status 200 OK

Rol | HTTP reagerende applicatie

FOUT

```
HTTP/1.1 400 Bad Request
Content-type: text/xml; charset="utf-8"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<COMT_IN229229 xmlns="urn:hl7-org:v3"
  ...
</COMT_IN229229>
</soap:Body>
</soap:Envelope>
```

Bovenstaande is een verwachte HL7 respons (Ping response).

GOED

```
HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-8"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<COMT_IN229229 xmlns="urn:hl7-org:v3"
  ...
</COMT_IN229229>
</soap:Body>
</soap:Envelope>
```

Bovenstaande is een verwachte HL7 respons.

GOED

```
HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-8"

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
```

```
<MCCI_IN000002 xmlns="urn:hl7-org:v3"
...
<acknowledgement typeCode="CA">
  <targetMessage>
    <id extension="34597476" root="2.16.528.1.1007.3.2.400575.7"/>
  </targetMessage>
</acknowledgement>
...
</MCCI_IN000002>
</soap:Body>
</soap:Envelope>
```

Bovenstaande is een Accept Acknowledgement met een Commit Ack.

7.1.1.7 Accept Acknowledgement fouten krijgen HTTP status 200 OK

Rol | HTTP reagerende applicatie, HL7 reagerende applicatie

Als de inhoud een HL7v3 Accept Ack met waarde 'CE' of 'CR' is, is HTTP status 200 OK.

Dit is een Accept Acknowledgement Commit Error.

[..\XML\soap\SOAP MCCI_IN000002_01_NAK.xml](#)

FOUT

HTTP/1.1 400 Bad Request

Content-type: text/xml; charset="utf-8"

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<MCCI_IN000002 xmlns="urn:hl7-org:v3"
  ...
  <acknowledgement typeCode="CE">
    <acknowledgementDetail typeCode="E">
      <code code="NS200" displayName="Unsupported InteractionID"
        codeSystem="2.16.840.1.113883.5.1100"/>
    </acknowledgementDetail>
    <targetMessage>
      <id extension="34597474" root="2.16.528.1.1007.3.2.400575.7"/>
    </targetMessage>
  </acknowledgement>
  ...
</MCCI_IN000002>
</soap:Body>
</soap:Envelope>
```

GOED

HTTP/1.1 200 OK

Content-type: text/xml; charset="utf-8"

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Body>
<MCCI_IN000002 xmlns="urn:hl7-org:v3"
  ...
  <acknowledgement typeCode="CE">
    <acknowledgementDetail typeCode="E">
      <code code="NS200" displayName="Unsupported InteractionID"
        codeSystem="2.16.840.1.113883.5.1100"/>
    </acknowledgementDetail>
```

```
<targetMessage>  
  <id extension="34597474" root="2.16.528.1.1007.3.2.400575.7"/>  
</targetMessage>  
</acknowledgement>  
...  
</MCCI_IN000002>  
</soap:Body>  
</soap:Envelope>
```

7.1.1.8 SOAP Faults leveren HTTP 500 Internal Server Error

Rol | SOAP reagerende applicatie

SOAP Faults dienen alleen voor upward compatibiliteit. In voorbeeldscenario's mogen (moeten) ze voorkomen, runtime niet. De Faults VersionMismatch en MustUnderstand mogen voorkomen in voorbeeldscenario's.

FOUT

```
HTTP/1.1 HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-8"

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

```

GOED

```
HTTP/1.1 500 Internal Server Error
Content-type: text/xml; charset="utf-8"

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

```

7.1.2 XML

Voorbeelden op XML niveau. De input voor deze voorbeelden is een document zoals dat verpakt is in het HTTP bericht.

7.1.2.1 Er moet een XML-prolog aanwezig zijn

Rol	XML agerende applicatie en reagerende applicatie
-----	--

FOUT

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
...
```

GOED

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

7.1.2.2 De encoding in de XML-prolog moet UTF-8 zijn

Rol | XML agerende applicatie en reagerende applicatie

FOUT

```
<?xml version="1.0" encoding="iso-8859-1"?>  
...
```

GOED

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

GOED

```
<?xml version="1.0"?>  
...
```

7.1.3 SOAP

Invoer voor deze voorbeelden is dezelfde als hierboven, alleen is nu zeker dat het een XML document is wat is afgeleverd.

7.1.3.1 Het document (root) element moet een SOAP 1.1 Envelope zijn

Rol	SOAP agerende applicatie en reagerende applicatie
-----	---

Root moet SOAP Envelope volgens SOAP 1.1 (<http://schemas.xmlsoap.org/soap/envelope/>) zijn.

FOUT

```
<?xml version="1.0" encoding="utf-8"?>
<COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

FOUT

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

Bovenstaande is een SOAP 1.2 Envelope.

GOED

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

7.1.3.2 Namespaces voor SOAP, HL7 en XML Schema moeten gedeclareerd zijn

Rol | SOAP agerende applicatie en reagerende applicatie

FOUT

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

FOUT

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hl7="urn:hl7-org:v3">
<soap:Body>
<COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

HL7 content in soap Envelope ([..\XML\soap\SOAP_MFMT_EX002101-1.xml](#))

GOED

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <MFMT_IN002101 xmlns="urn:hl7-org:v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:hl7-org:v3 ../schemas/MFMT_IN002101.xsd">
      <id extension="34234" root="2.16.528.1.1007.3.2.700222.1"/>
      <creationTime value="20040417161000"/>
      <versionCode code="NICTIZED2005-Okt"/>
      ...
    </MFMT_IN002101>
  </soap:Body>
</soap:Envelope>

```

Met embedded default namespaces. Geen fraaie XML, maar dient wel geaccepteerd te worden ([..\XML\soap\SOAP_MFMT_EX002101-2.xml](#))

GOED

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

```

<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Body>
    <MFMT_IN002101 xmlns="urn:hl7-org:v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:hl7-org:v3 ../schemas/MFMT_IN002101.xsd">
      <id extension="34234" root="2.16.528.1.1007.3.2.700222.1"/>
      <creationTime value="20040417161000"/>
      <versionCode code="NICTIZEd2005-Okt"/>
      ...
    </MFMT_IN002101>
  </Body>
</Envelope>

```

Zonder default namespaces, alles gekwalificeerd met prefixes (..\XML\soap\SOAP_MFMT_EX002101-3.xml)

GOED

```

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <hl7:MFMT_IN002101 xmlns:hl7="urn:hl7-org:v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:hl7-org:v3 ../schemas/MFMT_IN002101.xsd">
      <hl7:id extension="34234" root="2.16.528.1.1007.3.2.700222.1"/>
      <hl7:creationTime value="20040417161000"/>
      <hl7:versionCode code="NICTIZEd2005-Okt"/>
      ...
    </hl7:MFMT_IN002101>
  </soap:Body>
</soap:Envelope>

```

Ongebruikelijke prefixes mogen (..\XML\soap\SOAP_MFMT_EX002101-4.xml)

GOED

```

<?xml version="1.0" encoding="UTF-8"?>
<foo:Envelope xmlns:foo="http://schemas.xmlsoap.org/soap/envelope/">
  <foo:Body>
    <bar:MFMT_IN002101 xmlns:bar="urn:hl7-org:v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:hl7-org:v3 ../schemas/MFMT_IN002101.xsd">
      <bar:id extension="34234" root="2.16.528.1.1007.3.2.700222.1"/>
      <bar:creationTime value="20040417161000"/>
      <bar:versionCode code="NICTIZEd2005-Okt"/>
      ...
    </bar:MFMT_IN002101>
  </foo:Body>
</foo:Envelope>

```

7.1.3.3 Er zijn geen SOAP Headers aanwezig

Rol | SOAP agerende applicatie en reagerende applicatie

In toekomstige versies van AORTA komen waarschijnlijk wel SOAP Headers, in deze niet. Ze mogen dus niet gegenereerd worden.

FOUT

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap=http://schemas.xmlsoap.org/soap/envelope/
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Header>
    <foo soap:mustUnderstand="1">bar</foo>
  </soap:Header>
  <soap:Body/>
</soap:Envelope>
<COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

FOUT

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap=http://schemas.xmlsoap.org/soap/envelope/
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Header>
    <foo>bar</foo>
  </soap:Header>
  <soap:Body/>
</soap:Envelope>
<COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

GOED

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hl7="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
  <COMT_IN118118 xmlns="urn:hl7-org:v3"
...

```

7.1.4 Betrouwbaarheid

7.1.4.1 Message.acceptAckCode moet "NE" (Never) of "AL" (Always) zijn

Rol	HL7 agerende applicatie
-----	-------------------------

FOUT

```
...
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <hl7:MCCI_IN000002
      ...
      <hl7:acceptAckCode code="ER" />
    ...
  </soap:Body>
</soap:Envelope>
```

GOED

```
...
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <hl7:MCCI_IN000002
      ...
      <hl7:acceptAckCode code="NE" />
    ...
  </soap:Body>
</soap:Envelope>
```

GOED

```
...
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <MFMT_IN002101 xmlns="urn:hl7-org:v3"
      ...
      <acceptAckCode code="AL"/>
    ...
  </soap:Body>
</soap:Envelope>
```

7.2 Dynamische voorbeelden

De dynamische voorbeelden zijn vormgegeven met een input en output, en alleen goede varianten (inclusief een correcte afhandeling van foutsituaties) wordt weergegeven.

7.2.1 HTTP

7.2.1.1 Succesvolle verwerking beantwoorden met HTTP 200 OK

Rol	HTTP reagerende applicatie
-----	----------------------------

Een HTTP request wordt bij succesvolle verwerking beantwoord met een HTTP response die begint met de status: "200 OK".

Voorbeeldscenario:

- Agerende applicatie levert valide HL7v3 document af
- Reagerende applicatie levert respons met 200 OK

AGERENDE APPLICATIE

```
POST / HTTP/1.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <COMT_IN118118 xmlns="urn:hl7-org:v3"
      ...
```

REAGERENDE APPLICATIE

```
HTTP/1.1 200 OK
Content-type: text/xml; charset="utf-8"

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <COMT_IN229229 xmlns="urn:hl7-org:v3"
      ...
```

7.2.1.2 HTTP 1.1 is verplicht

Rol | HTTP agerende applicatie, HTTP reagerende applicatie

Bij gebruik van andere HTTP versies dient een HTTP error 505 "HTTP Version not supported" geretourneerd te worden.

Voorbeeldscenario:

- Agerende applicatie levert HTTP 1.0 document af.
- Reagerende applicatie levert respons met 505 HTTP Version not supported

AGERENDE APPLICATIE

```
POST / HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <COMT_IN118118 xmlns="urn:hl7-org:v3"
      ...
    ...
```

REAGERENDE APPLICATIE

```
HTTP/1.1 505 HTTP Version not supported
```

Bij een respons is eventuele overige inhoud niet relevant.

7.2.1.3 Geen valide inhoud levert HTTP 400 Bad Request

Rol | HTTP reagerende applicatie

Voorbeeldscenario:

- Agerende applicatie levert niet well-formed XML document af
- Reagerende applicatie levert respons met 400 Bad Request

AGERENDE APPLICATIE

POST / HTTP/1.1

Content-Type: text/xml; charset=utf-8

SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    </soap:Envelope>
    <COMT_IN118118 xmlns="urn:hl7-org:v3"
      ...
```

REAGERENDE APPLICATIE

HTTP/1.1 400 Bad Request

Bij respons is eventuele overige inhoud niet relevant. De standaard staat toe dat een andere foutmelding gegeven wordt, b.v. een HL7 Accept Acknowledgment met een fout erin.

7.2.2 XML

Er zijn geen dynamische voorbeelden op XML niveau.

7.2.3 SOAP

7.2.3.1 VersionMismatch Fault

Rol	SOAP reagerende applicatie
-----	----------------------------

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. Hoewel dit in de huidige release van AORTA niet voor mag komen, is dit essentieel om een overgang naar eventuele komende versies te vergemakkelijken.

Voorbeeldscenario:

- Agerende applicatie levert SOAP 1.2 ("http://www.w3.org/2003/05/soap-envelope") document
- Reagerende applicatie levert respons met SOAP VersionMismatch Fault

AGERENDE APPLICATIE

```
POST / HTTP/1.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  <soap:Body>
</soap:Envelope>
  <COMT_IN118118 xmlns="urn:hl7-org:v3"
  ...
```

REAGERENDE APPLICATIE

```
HTTP/1.1 500 Internal Server Error
Content-type: text/xml; charset="utf-8"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns="">
      <faultcode>soap:VersionMismatch</faultcode>
      <faultstring>The processing party found an invalid namespace for the SOAP
      Envelope element</faultstring>
```

```
</soap:Fault>  
</soap:Body>  
</soap:Envelope>
```

De precieze inhoud van faultstring mag variëren.

7.2.3.2 MustUnderstand Fault

Rol | SOAP reagerende applicatie

Er is een SOAP Header aanwezig die niet bekend is, maar met mustUnderstand = "1". In dit geval MOET deze Fault gegenereerd worden.

Voorbeeldscenario:

- Agerende applicatie levert SOAP document met Header met mustUnderstand = "1"
- Reagerende applicatie levert respons met SOAP MustUnderstand Fault

AGERENDE APPLICATIE

```
POST / HTTP/1.1
Content-Type: text/xml; charset=utf-8
SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  <soap:Header>
    <foo soap:mustUnderstand="1">bar</foo>
  </soap:Header>
  <soap:Body>
</soap:Envelope>
  <COMT_IN118118 xmlns="urn:hl7-org:v3"
  ...
```

REAGERENDE APPLICATIE

```
HTTP/1.1 500 Internal Server Error
Content-type: text/xml; charset="utf-8"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns="">
      <faultcode>soap:MustUnderstand</faultcode>
      <faultstring>An immediate child element of the SOAP Header element that was
        either not understood or not obeyed by the processing party contained a SOAP
        mustUnderstand attribute with a value of "1".</faultstring>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

De precieze inhoud van faultstring mag variëren.

7.2.4 Betrouwbaarheid

7.2.4.1 Bij een asynchrone interactie volgt op een bericht een Accept Ack

Rol	HL7 reagerende applicatie
-----	---------------------------

Op een bericht uit een asynchrone interactie (te herkennen aan Message.acceptAckCode="AL") volgt een HL7v3 Accept Acknowledgement.

Voorbeeldscenario:

- Agerende applicatie levert valide HL7v3 verzoekbericht af met Message.acceptAckCode="AL" [..\XML\soap\SOAP_COMT_IN113113NL.xml](#)
- Reagerende applicatie levert valide HL7v3 Accept Acknowledgement af [..\XML\soap\SOAP_MCCI_IN000002_03_ACK.xml](#)

AGERENDE APPLICATIE

POST / HTTP/1.1

Content-Type: text/xml; charset=utf-8

SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <COMT_IN113113NL xmlns="urn:hl7-org:v3"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:hl7-org:v3 ../schemas/COMT_IN113113NL.xsd">
      <id extension="200104" root="2.16.528.1.1007.3.3.112233.1"/>
      ...
      <acceptAckCode code="AL"/>
      ...
    </COMT_IN113113NL>
  </soap:Body>
</soap:Envelope>
```

REAGERENDE APPLICATIE

HTTP/1.1 200 OK

Content-type: text/xml; charset="utf-8"

```
<?xml version="1.0" encoding="UTF-8"?>
<MCCI_IN000002 xmlns="urn:hl7-org:v3"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:hl7-org:v3 ../schemas/MCCI_IN000002.xsd">
  <id extension="34236" root="2.16.528.1.1007.3.3.700844.1"/>
  ...
  <acceptAckCode code="NE"/>
  <acknowledgement typeCode="CA">
    <targetMessage>
      <id extension="200104" root="2.16.528.1.1007.3.3.112233.1"/>
    </targetMessage>
  </acknowledgement>
</MCCI_IN000002>
```

</acknowledgement>

...

7.2.4.2 Bij een synchrone interactie volgt op een bericht een respons

Rol | HL7 reagerende applicatie

Op een bericht uit een synchrone interactie (te herkennen aan Message.acceptAckCode="NE") volgt een HL7v3 inhoudelijke respons.

Voorbeeldscenario:

- Agerende applicatie levert valide HL7v3 bericht af met Message.acceptAckCode="NE" [..\XML\soap\SOAP_COMT_IN118118.xml](#)
- Reagerende applicatie levert valide HL7v3 antwoordbericht af [..\XML\soap\SOAP_COMT_IN229229.xml](#)

AGERENDE APPLICATIE

POST / HTTP/1.1

Content-Type: text/xml; charset=utf-8

SOAPAction: "urn:hl7-org:v3/Ping_PingPong"

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <COMT_IN118118 xmlns="urn:hl7-org:v3"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:hl7-org:v3 ../schemas/COMT_IN118118.xsd">
      <id extension="200103" root="2.16.528.1.1007.3.3.112233.1"/>
      ...
      <acceptAckCode code="NE"/>
      ...
    </COMT_IN118118>
  </soap:Body>
</soap:Envelope>
```

REAGERENDE APPLICATIE

HTTP/1.1 200 OK

Content-type: text/xml; charset="utf-8"

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <COMT_IN229229 xmlns="urn:hl7-org:v3"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:hl7-org:v3 ../schemas/COMT_IN229229.xsd">
      <id extension="20060417161000972" root="2.16.528.1.1007.3.3.884654.1"/>
      ...
      <!-- acceptAckCode = NE in HL7 Pong interacties -->
      <acceptAckCode code="NE"/>
      <!-- AA = applicatie-level ack -->
      <acknowledgement typeCode="AA">
        <targetMessage>
          <id extension="200103" root="2.16.528.1.1007.3.3.112233.1"/>
        </targetMessage>
      </acknowledgement>
    </COMT_IN229229>
  </soap:Body>
</soap:Envelope>
```

```
</targetMessage>  
</acknowledgement>
```

```
...
```