Infrastructuur documentuitwisseling zorginstellingen

Auteur: Marc de Graauw
Versie: 4, 14 april 2015

Inhoudsopgave

1. Inleiding 2
2. Web Services 3
2.1 Gebruikte standaarden 3
2.2 ProvideDocument 3
2.2.1 SOAP bericht 3
2.2.2 Metadata 4
2.2.3 Foutafhandeling 5
2.2.4 Ping 6
3. Payload 7
3.1 HL7v3 CDA documenten 7
3.1.1 Versionering van CDA documenten 7
3.1.2 CDA sections 9

1. Inleiding

Dit document beschrijft de infrastructurele aspecten van documentuitwisseling tussen zorginstellingen. Deze uitwisseling is oorspronkelijk opgezet voor het RIVM ten behoeve van de landelijke darmkanker screening, maar ook breder toepasbaar.

Dit document dient in samenhang gelezen te worden met:

- Beveiliging documentuitwisseling zorginstellingen

Per implementatie dienen er extra documenten te zijn:

- Configuratie infrastructuur [naam uitwisseling]
- Parameters document [naam verslag en uitwisseling]

Versie	Datum	Wijzigingen
1	10 september 2013	Groter voorgaand document gesplitst.
2	10 maart 2014	Foutmelding voor ongeldige metadata toegevoegd.
3	23 april 2014	SOAP Faults en http fouten beschreven.
4	14 april 2015	Toegevoegd hoe lege extension in metadata komt.

2. Web Services

Voor de uitwisseling van HL7v3 CDA documenten worden Web Services gebruikt.

2.1 Gebruikte standaarden

De gebruikte SOAP stack is:

- SOAP 1.1
- WSDL 1.1
- WS-I Basic Profile 1.0
- HTTP 1.1
- TLS 1.0
- TCP
- IPv4

We gebruiken voorlopig geen MTOM en WS-Addressing 1.0, al zouden beide wel eenvoudig pluggable moeten zijn. Deze stack is dezelfde als die in Aorta gebruikt wordt.

2.2 ProvideDocument

2.2.1 SOAP bericht

Er wordt een Web Service gedefinieerd: ProvideDocument. De semantiek is het zenden van een HL7v3 CDA document.

Alle berichten worden afgehandeld in een enkele HTTP request en response paar. Dat betekent gebruik van een anonymous WS-Addressing replyTo adres, en de ack komt dus altijd in de HTTP response op een bericht terug. De komt CDA in payload base 64 encoded zodat de XML parser die ontvangt deze als tekst kan behandelen en niet als XML. Er worden metadata toegevoegd, zodat voor het parsen van het CDA document al enkele gegevens bekend zijn. Dit maakt het trouwens ook eenvoudig om signing van metadata toe te voegen, mocht dat ooit wenselijk worden. Een SOAPAction HTTP header is niet verplicht. Als deze SOAP client deze toevoegt, wordt deze genegeerd.

Aan de ProvideDocument payload worden een aantal metadata toegevoegd. Deze worden letterlijk gekopieerd uit het CDA document. Het doel hiervan is de ontvanger in staat te stellen al enige validaties te verrichten vóór de Base64 decoding plaatsvindt. Tenslotte volgt het CDA document zelf, Base 64 encoded (MIME Base 64 encoding als gedefinieerd in RFC 2045: http://www.ietf.org/rfc/rfc2045.txt).

Een voorbeeld:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <docws:ProvideDocument
            xmlns:docws="urn:oid:2.16.840.1.113883.2.4.3.46.10.1">
            <docws:DocumentMetaData>
                <!-- Verslag identificatie -->
                <docws:ClinicalDocument.id>
                    <docws:root>2.16.840.1.113883.2.4.99.3.22</docws:root>
                    <docws:extension>3266473876378237</docws:extension>
                </docws:ClinicalDocument.id>
                <docws:ClinicalDocument.setId>
                    <docws:root>2.16.840.1.113883.2.4.99.3.22</docws:root>
                    <docws:extension>s3266473</docws:extension>
                </docws:ClinicalDocument.setId>
                <docws:ClinicalDocument.versionNumber>1
                </docws:ClinicalDocument.versionNumber>
                <!-- Document typering -->
                <docws:ClinicalDocument.code>
                    <docws:codeSystem>2.16.840.1.113883.6.1</docws:codeSystem>
```

```
                    <docws:code>18746-8</docws:code>
                </docws:ClinicalDocument.code>
                <!-- Burger Service Nummer -->
                <docws:patientId>
                    <docws:root>2.16.840.1.113883.2.4.6.3</docws:root>
                    <docws:extension>228454128</docws:extension>
                </docws:patientId>
                <!-- Beheerder -->
                <docws:custodian>
                    <docws:root>2.16.528.1.1007.3.3</docws:root>
                    <docws:extension>67823221</docws:extension>
                </docws:custodian>
                <!-- DECOR version, project/@id and project/version/@date-->
                <docws:project>
                    <docws:id>2.16.840.1.113883.2.4.3.36.77.0.1</docws:id>
                    <docws:version>2013-03-23T00:00:00</docws:version>
                </docws:project>
            </docws:DocumentMetaData>
            <docws:Document>PD94b...base64 encoded CDA doc...Pg==</docws:Document>
        </docws:ProvideDocument>
    </soap:Body>
</soap:Envelope>
```

Een voorbeeld van een antwoord:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <docws:ProvideDocumentResponse
xmlns:docws="urn:oid:2.16.840.1.113883.2.4.3.46.10.1">
            <docws:Success>false</docws:Success>
            <docws:Code>PATUNK</docws:Code>
            <docws:Text>Patiënt niet bekend</docws:Text>
        </docws:ProvideDocumentResponse>
    </soap:Body>
</soap:Envelope>
```

Een volledige set specificaties, met WSDL en XML Schema is ook gepubliceerd.

In gevallen waar een id wel een root heeft maar geen extension wordt de volgende notatie gebruikt:

```
<docws:custodian>
    <docws:root>2.16.528.1.1007.3.3</docws:root>
    <docws:extension>67823221</docws:extension>
</docws:custodian>
```

2.2.2 Metadata

De volgende metadata worden in het bericht opgenomen:

Metagegeven	**Verplicht**	**Overnemen uit**
ClinicalDocument.id	j	CDA document
ClinicalDocument.setId	j	CDA document
ClinicalDocument.versionNumber	j	CDA document
ClinicalDocument.code	j	CDA document
patientId	j	CDA document
custodian	j	CDA document
projectVersion	j/n	Vaste waarden per release van de specificatie.

Dit bevat de identificatie van het DECOR project (is in dit project altijd '2.16.840.1.113883.2.4.3.36.77.0.1') en de version, die varieert per release van de

DECOR specificaties. De waarde van version mag dus niet gekopieerd worden uit dit voorbeeld. De DECOR specificaties kunnen immers wijzigen zonder nieuwe release van deze specificatie.

2.2.3 Foutafhandeling

2.2.3.1 HTTP(S) Fouten

Een zender moet altijd rekening houden met fouten op HTTP of TLS niveau. Deze kunnen optreden als de service niet beschikbaar is, of als er problemen met de certificaten zijn. Deze fouten worden verder niet in dit document uitgewerkt, zie documentatie van HTTP.

Een voorbeeld uit Colonis, ingeval van onvoldoende authorisatie:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
    <title>401 Authorization Required</title>
</head><body>
    <h1>Authorization Required</h1>
    <p>This server could not verify that you
        are authorized to access the document
        requested.  Either you supplied the wrong
        credentials (e.g., bad password), or your
        browser doesn't understand how to supply
        the credentials required.</p>
</body></html>
```

Dit betreft de HTTP Body van het antwoord, de HTTP statuscode is: ‘401 Authorization Required’.

2.2.3.2 SOAP Faults

Een zender moet altijd rekening houden met SOAP Faults.

Een voorbeeld uit Colonis, ingeval van fouten tegen het schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <soap:Fault>
            <faultcode>soap:Client</faultcode>
            <faultstring>Unmarshalling Error: cvc-complex-type.2.4.a: Invalid
content was found starting with element 'docws:Fout'. One of
'{"urn:oid:2.16.840.1.113883.2.4.3.46.10.1":DocumentMetaData,
"urn:oid:2.16.840.1.113883.2.4.3.46.10.1":Document,
"urn:oid:2.16.840.1.113883.2.4.3.46.10.1":Ping}' is expected.</faultstring>
            <detail>
                <stackTrace xmlns="http://cxf.apache.org/fault">...</stackTrace>
            </detail>
        </soap:Fault>
    </soap:Body>
</soap:Envelope>
```

Colonis geeft ook SOAP Faults terug in geval van de volgende foutsituaties genoemd in 2.2.3.3:

- VERSION_UNKNOWN
- METADATA_INVALID

2.2.3.3 ProvideDocument fouten

Fouten in de metadata van ProvideDocument worden afgehandeld met een responsebericht met HTTP statuscode '200 OK' (het is immers succesvol op HTTP niveau).

Succesvol verwerkte berichten worden voorzien van een ack met succes status. Er zijn drie velden voor status en foutafhandeling in de ack:

Success	Code	Text
true	OK	("OK")
true	REEDS_CORRECT_VERWERKT	("Bericht met id %1 is al eerder ontvangen en succesvol verwerkt."), -> %1 = bericht ID
false	CLIENT_UNK	("Client met bsn %1 is niet bekend."), -> %1 = bsn
false	ONGELDIGE_VERSIE	("Van het bericht met setId %1 is reeds een versie >=%2 ontvangen."), -> %1 = setID, %2 = versie
false	SYSTEM_ERROR	("Er is een fout opgetreden in de broker bij verwerken van bericht: %1"), -> %1 = fout
false	CDA_SOAP_INCONSISTENT	("%1 (%2) in SOAP is niet gelijk aan %3 (%4) in CDA."); -> %1 = waarde van %2 veld, %2 = naam veld in SOAP, %3 = waarde van %4 veld, %4 = veld in CDA bericht
true	PING_OK	("Ping succesvol")
false	VERSION_UNKNOWN	("Versie %1 van project %2 is niet bekend."), -> %1 = project version, %2 = project id
false	BEZWAAR_GEMAAKT	("Patiënt heeft bezwaar gemaakt tegen delen gegevens.")
false	METADATA_INVALID	ProvideDocument metadata zijn niet (schema-)valide.

2.2.4 Ping

Voor het kunnen testen van de connectie is een 'Ping' bericht gedefinieerd:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <docws:ProvideDocument
xmlns:docws="urn:oid:2.16.840.1.113883.2.4.3.46.10.1">
            <docws:Ping/>
        </docws:ProvideDocument>
    </soap:Body>
</soap:Envelope>
```

Dit bericht kan naar de ontvanger gestuurd worden, en er zal een normaal Response bericht terug komen - met Succes=true als alles goed is. Het Ping bericht mag geen metadata en CDA payload bevatten. Het dient om bij problemen in de productionele omgeving te kunnen verifiëren of de connectiviteit in orde is. Gebruik van Ping is voor leveranciers optioneel, maar wel aan te bevelen. Vaststellen wat de oorzaak is van een verstoring in een productionele omgeving kan ingewikkeld zijn, en met de Ping zijn veel mogelijke oorzaken uit te sluiten.

3. Payload

In de payload van het bericht zit een document.

3.1 HL7v3 CDA documenten

3.1.1 Versionering van CDA documenten

CDA staat als versionering append en replace toe. We ondersteunen alleen replace. Dat wil zeggen dat een MDL- of PALGA verslag voor de darmkanker screening ofwel een origineel verslag moet zijn, ofwel een verslag dat het vorige verslag vervangt. (CDA append is bedoeld wanneer er een origineel verslag is, waarbij een addendum wordt toegevoegd, dat tesamen met het originele verslag gelezen dient te worden. Die situatie ondersteunen we niet: een inzender moet, wanneer er wijzigingen op het originele verslag zijn, een nieuw zelfstandig leesbaar en compleet verslag inzenden.)

Wanneer een verslag is ingezonden maar het vervalt om een of andere reden, wordt dit offline afgehandeld: dus via de support desk van de screeningsorganisatie. Alternatief is het inrichten van een aparte Web Service met "Cancel" logica. De verwachting is dat het schrappen van een verslag een zeldzame gebeurtenis zal zijn, en dat de kosten van het incidenteel afhandelen van Cancel aanvragen het bouwen van een aparte Web Service niet rechtvaaardigen.

Voor versionering zijn enkele attributen van een CDA document van belang:

ClinicalDocument.id	Unieke identifier voor iedere versie van een CDA document. Verplicht element. Ieder CDA document moet een wereldwijde unieke identificatie hebben. De meest voor de hand liggende methode is een unieke root per document-producerende installatie te hebben, en in de extension een volgnummer of -string te zetten die uniek is binnen die document-producerende installatie. Daartoe is een eigen OID nodig. Wanneer de leverancier die nog niet heeft, kan er bij Stichting HL7 Nederland een aangevraagd worden. Onder de eigen OID kan bijvoorbeeld een tak aangemaakt worden voor MDL verslagen, en daaronder weer een tak voor iedere document-producerende installatie. Voorbeeld: Root OID van Marc de Graauw IT: 2.16.840.1.113883.2.4.3.46 MDL verslagen Marc de Graauw IT: 2.16.840.1.113883.2.4.3.46.99.5.6.1 Coloscopiecentrum Amsterdam van Marc de Graauw IT 2.16.840.1.113883.2.4.3.46.99.5.6.1.1
ClinicalDocument .setId	Unieke identifier een set CDA documenten. Voor opeenvolgende versies van hetzelfde origineel moet setId dezelfde waarde hebben. Volgens het CDA schema is setId niet verplicht, maar omdat opvolgende versies van verslagen mogelijk zijn, moet het altijd gevuld worden.
ClinicalDocument .versionNumber	Oplopend volgnummer voor opeenvolgende versies. Waarde moet "1" zijn voor het origineel, en hoger voor iedere

	volgende versie. Evenals setId moet dit voor darmkanker screening altijd gevuld zijn.
ClinicalDocument .relatedDocument .parentDocument	Bevat een verwijzing naar de vorige versie van het document. Bij 'replace' is deze verwijzing redundant: op basis van setId kan de collectie van vorige versies worden bepaald, op basis van versionNumber de volgorde daarvan. Het vullen van relatedDocument is niet verplicht volgens het CDA schema. Het is geen fout als het aanwezig is, maar de relatie tussen documenten wordt bepaald op basis van setId en versionNumber.

Voor versionering van original en replace wordt een eenvoudige logica genomen:

i. een original document moet altijd versionNumber 1 hebben;
ii. opeenvolgende replaces krijgen een oplopend versionNumber 2, 3, 4...
iii. de verzender zorgt:
    a. dat berichten in de goede volgorde verzonden worden;
    b. verifieert dat een ack ontvangen wordt na een verzonden bericht;
    c. wordt geen ack ontvangen, b.v. omdat de verbinding uitgevallen is, dan wordt hetzelfde bericht (een replica) net zolang ingezonden tot een ack ontvangen is.
iv. de ontvanger controleert bij een inkomend bericht:
    a. of het ClinicalDocument.id al bestaat: in dat geval gaat het om een replica, en die mag, maar hoeft niet opnieuw verwerkt te worden. Er wordt een ack teruggestuurd: 'success=true' als het oorspronkelijke bericht succesvol was, en fout ('success=false', ofwel een nack) als dat niet het geval was. Wanneer een bericht een nack oplevert, zal het niet opgeslagen worden, en zal een replica dus opnieuw verwerkt worden, wat weer zal falen en tot een nack leiden. Wordt een bericht wel correct verwerkt en opgeslagen, dan zal bij inzenden van een replica de ontvanger dat moeten signaleren en een ack met status 'success=true' moeten genereren.
    b. of het setId al bestaat: is dat het geval, en heeft het opgeslagen bericht een versionNumber < versionNumber van het ingezonden bericht, dan is het ingezonden bericht de nieuwere versie, die opgeslagen wordt. Heeft het opgeslagen bericht een versionNumber > versionNumber van het ingezonden bericht, dan is het ingezonden bericht een oudere versie die niet opgeslagen hoeft te worden. Bij inzending op de juiste volgorde, waarbij de verzender wacht op een ack, zal dit niet gebeuren, maar als het toch gebeurt zal er niets mis gaan.
    c. ook als het original ontbreekt wordt een latere versie gewoon opgeslagen, dus wanneer het eerst ontvangen bericht van een set versionNumber 2 of hoger heeft, wordt het opgeslagen. Ook dit zal niet voorkomen wanneer zender alle berichten in de goede volgorde inzendt, maar de service is zo voldoende robuust om simpele fouten te doorstaan.
v. en de ontvanger retourneert een ack met succes- of foutmelding.

Betrouwbaar transport wordt zo gegarandeerd. De verzender blijft net zolang berichten zenden totdat een ack met status succes of fout is ontvangen. De eerste keer het bericht zelf, de volgende keren een exacte replica. Mislukt de communicatie op de heenweg (het bericht komt niet aan), dan krijgt de ontvanger dus later een replica die alsnog verwerkt wordt. Mislukt de communicatie op de terugweg, dan krijg de ontvanger alsnog een ack met status succes of fout. Het is daarbij wel van belang dat de ontvanger op een replica dezelfde ack stuurt als op het oorspronkelijke bericht. De verzender heeft immers het oorspronkelijke antwoord niet gehad, en weet dus niet of het oorspronkelijke bericht goed of niet goed is verwerkt. De ontvanger mag ervan uitgaan dat een bericht met een reeds ontvangen ClinicalDocument.id een replica is van het vorige bericht: het is de verantwoordelijkheid van de verzender ervoor te zorgen dat een ClinicalDocument.id nooit opnieuw gebruikt wordt voor een ander bericht. De ontvanger hoeft voor het

behandelen van replica's dus alleen het ClinicalDocument.id te controleren tegen de reeds ontvangen en succesvol verwerkte id's.

De trigger van het maken van een 'replace' document bij zender is de hernieuwde autorisatie van de gegevens door de zorgverlener.

3.1.2 CDA sections

CDA kent een verslagdeel in vrije tekst in de sections (XHTML-lookalike markup) en gestructureerde verslaglegging in de entries. De tekst in de sections zal middels de generieke CDA stylesheet gebruikt worden door de regionaal coördinerend MDL-arts en patholoog om de verslagen in te zien. Leveranciers dienen dus te borgen gestructureerde en tekstuele delen dezelfde informatie-inhoud bevatten.