

# **Implementeren van HL7v3 Web Services**

- Marc de Graauw -

# SOAP & WSDL

# SOAP & WSDL

- Intro
- WSDL & code generation
- Dynamic response, “wrapped” style
- Generic WSDL
- Reliability issues
- Wire signature

# Web Services

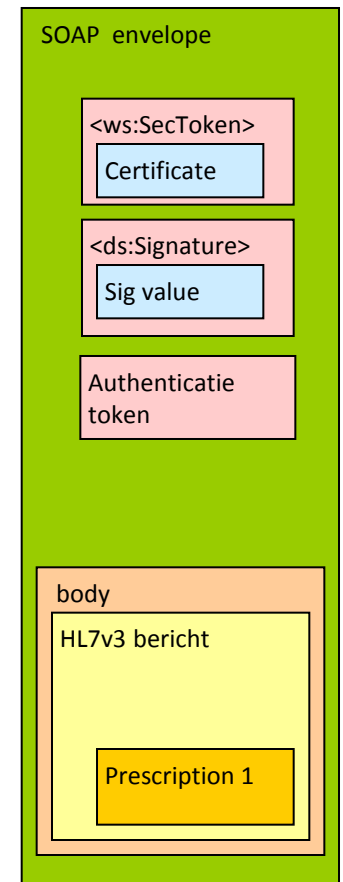
- WSDL
- historie
- opsplitsen WSDL
- IHE, OMG/HL7

# Historie XML

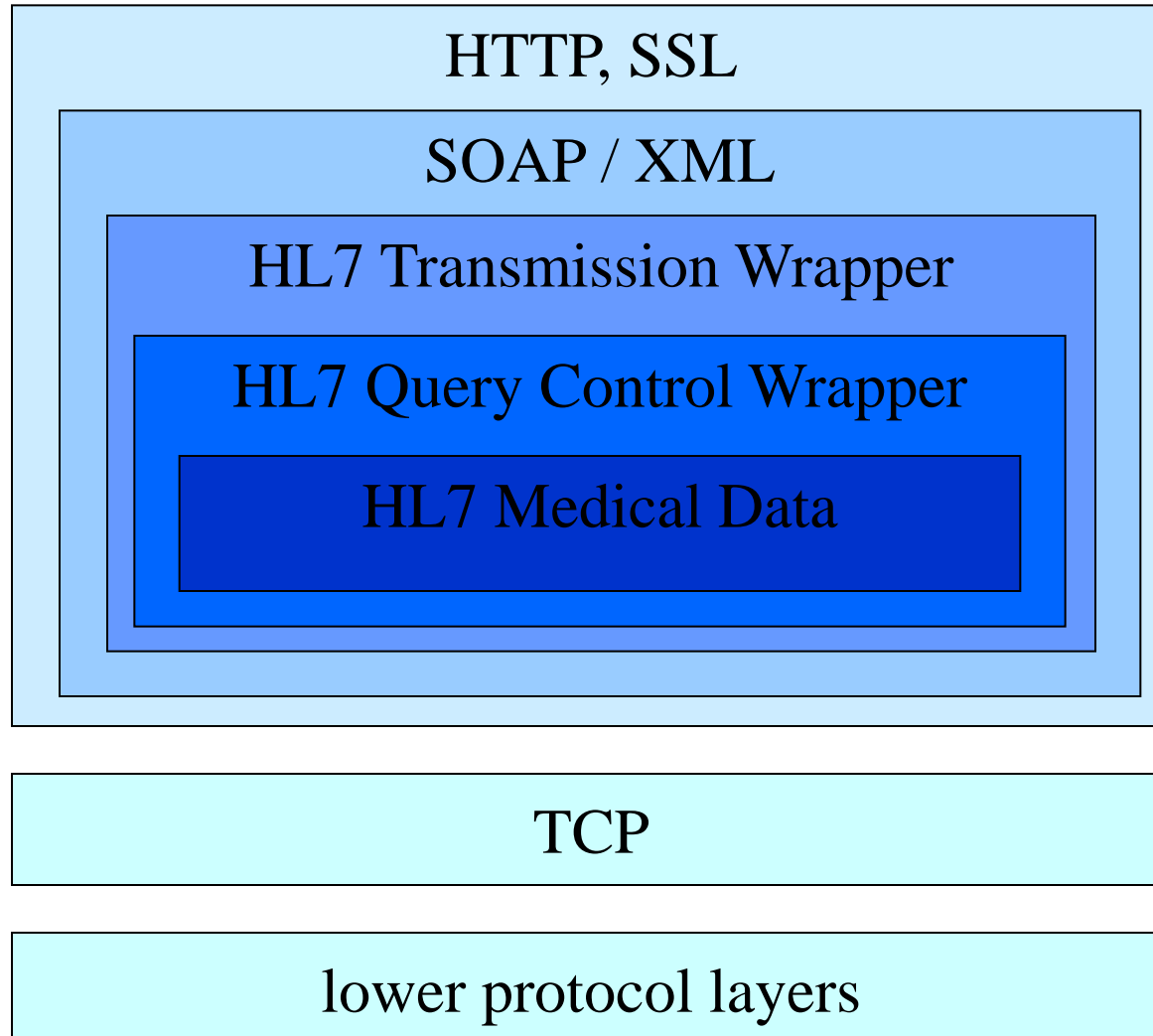
- SGML
  - Standard Generalized Markup Language
  - IBM: back to the sixties...
  - Markup: structuur, niet processing
- HTML: SGML spinoff
- 1998: XML
  - SGML voor het Web, zonder ballast
- 2000: SOAP 1.1, WSDL 1.0
- 2001: XML Schema

# SOAP

- SOAP
  - Envelope, Header, Body
  - transport: HTTP POST



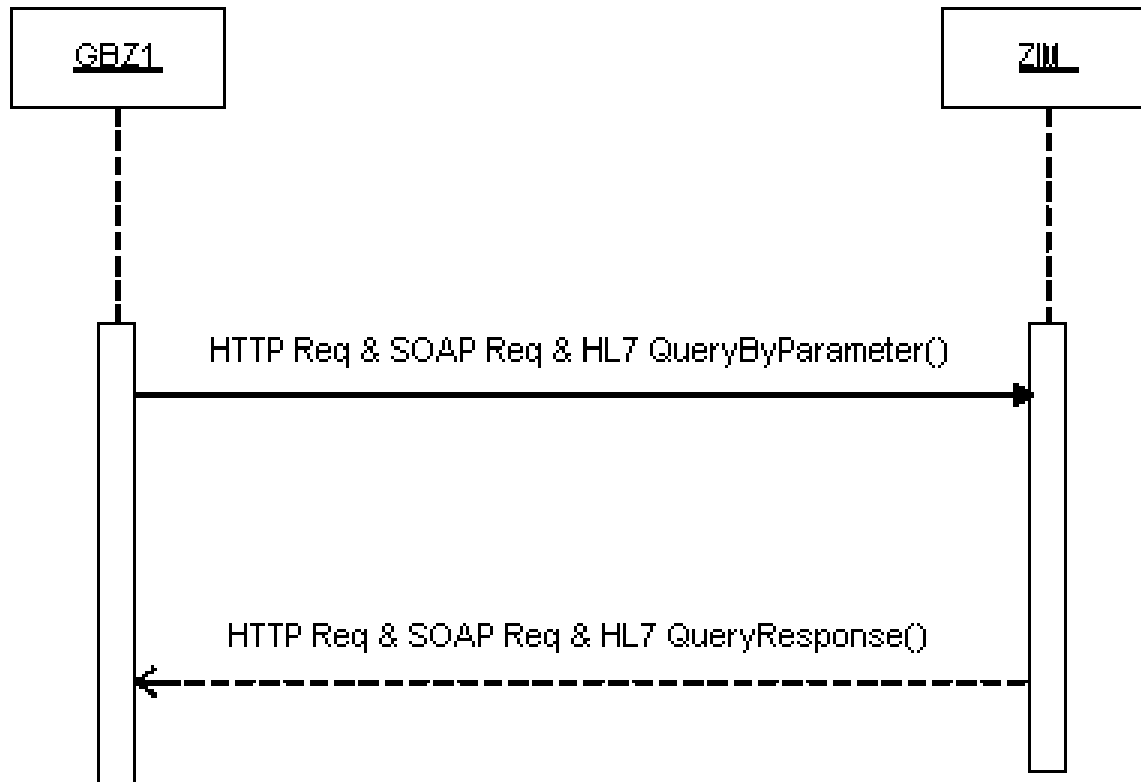
# HL7v3 Layered Model



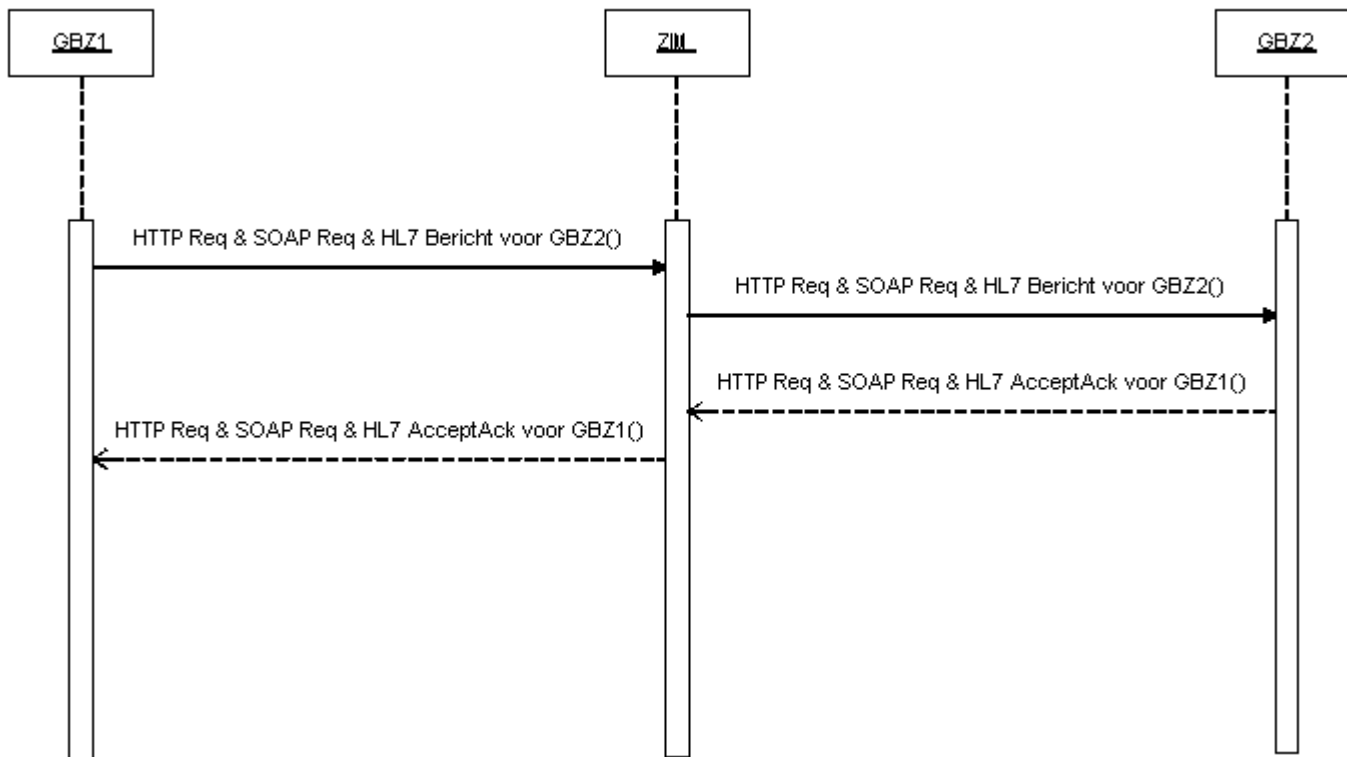
# AORTA Message Patronen

- Queries (van GBZ aan ZIM)
  - asynchroon
  - synchroon
- Andere berichten
  - met respons
  - zonder respons

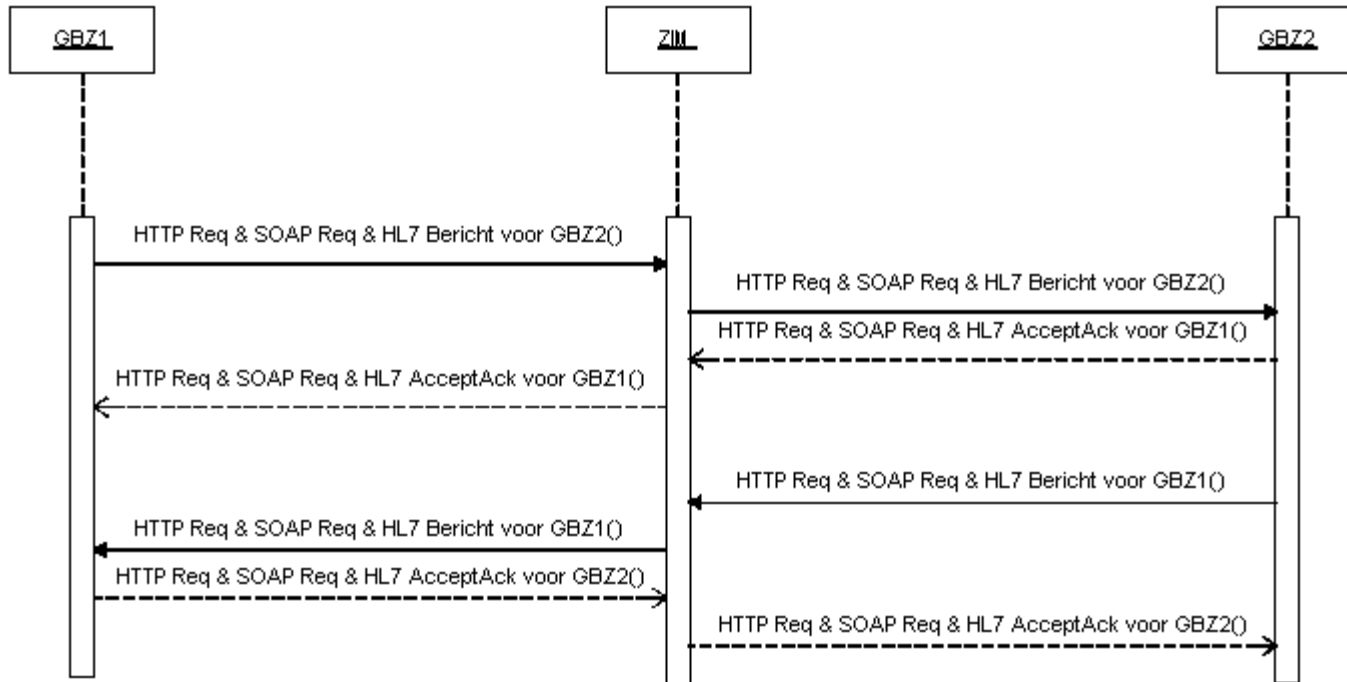
# Query met synchroon antwoord



# Bericht zonder respons



# Bericht met asynchrone respons



# SOAP in one slide

- Envelope, Headers, Body

```
<soap:Envelope ... namespaces ... >  
  <soap:Header mustUnderstand = '1'>  
    ... headers ...  
  </soap:Header>  
  <soap:Body>  
    ... payload ...  
  </soap:Body>  
</soap:Envelope>
```

- HTTP Binding

POST / HTTP/1.1

bla bla...

SOAPAction: "urn:hl7-org:v3/QURX\_AR990120NL"

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

```
<soap:Envelope ... namespaces ... >
```

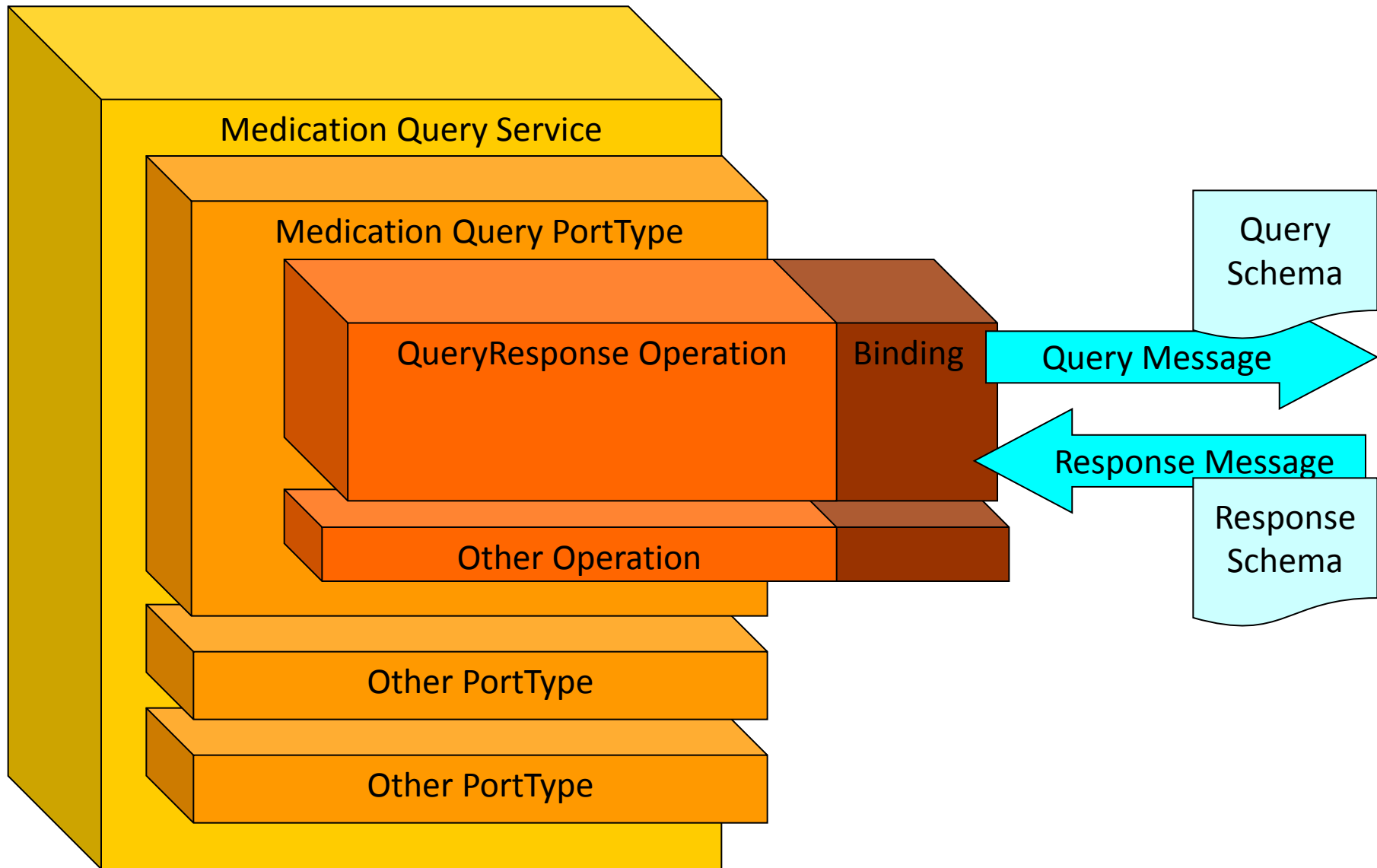
- SOAP Encoding: legacy, forget it

# 'Real world' complexity

- HL7v3 XML Schema <include> stacks of 10 – 15 schemas
- HL7v3 typing system (mapped onto XSD datatypes)
- HL7v3 vocabulary
- Layered 'wrapper' approach
- SOAP (Transmission (Query (Medical Data)))

# WSDL in one slide

- WSDL defines a web service
- Which schemas are used?
- Which messages are used & which schemas are involved?
- Which operations are used & which messages go in and which go out?
- How do operations assemble to make a web service (PortType, Service)?
- Binding to SOAP and HTTP



# 'Real world' complexity (cont.)

- WSDL is a:
  - description of a web service
  - generate WSDL from code?
  - generate code from WSDL?
- WSDL code generation
  - map XML to programming object
    - `<birthdate>19610306</birthdate>`
    - maps to: date
    - `<name><first>Marc</first><last>de Graauw</last></name>`
    - maps to: struct of string, string
    - `<gender>M</gender>`
    - maps to: char(1) or: enum('M', 'F') or: GenderType
  - map operations, HTTP Binding et cetera

# 'Real world' complexity (cont.)

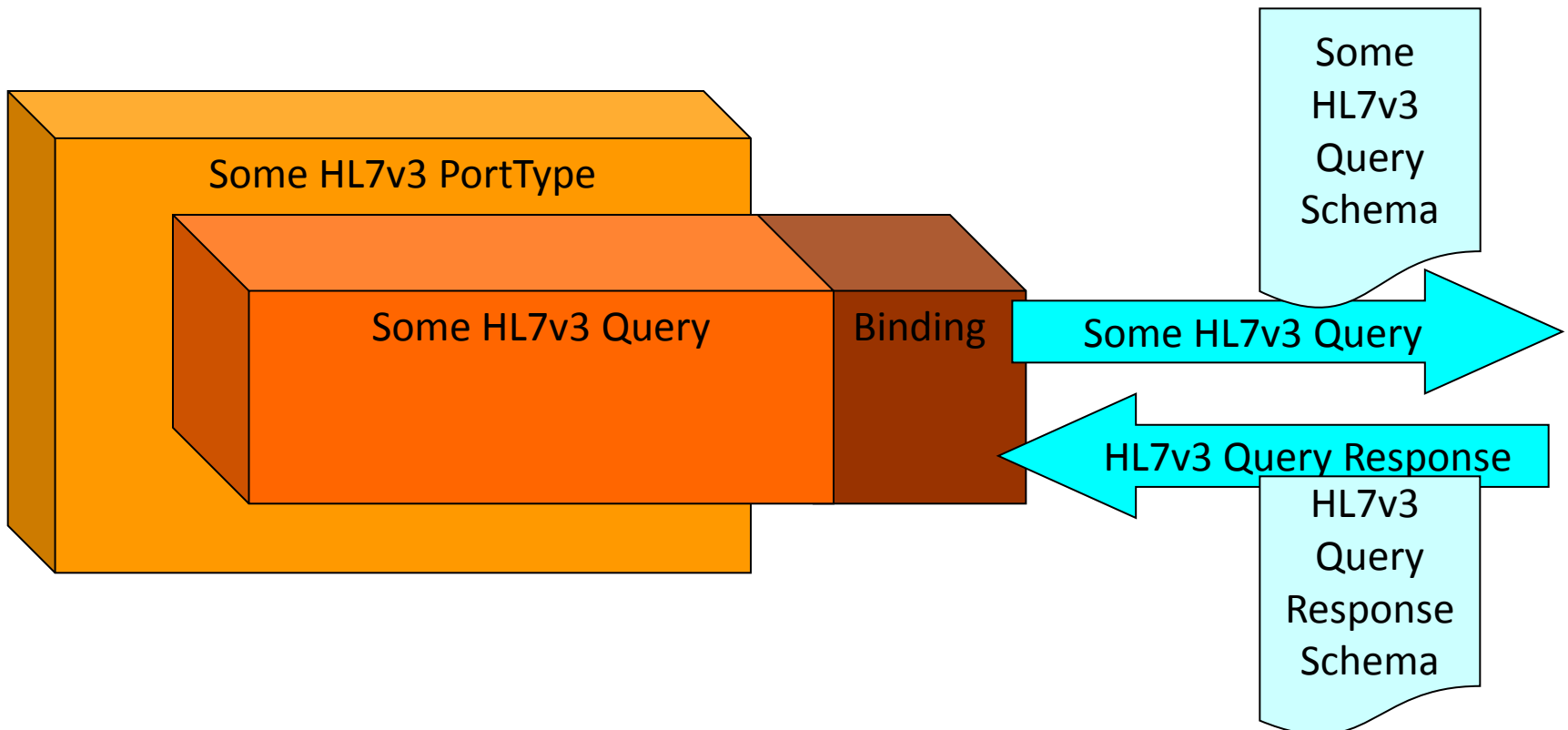
- WSDL code generation:
  - reserved word clashes
  - creates object for each XML construct
  - 15 schemas -> Gargantuan objects!
  - all of vocabulary.xsd
  - all objects in one module
- “Out of the box” cCode generation: fine for
  - float FahrenheitToCelsius(float)
  - currency StockQuote(string)

# Dynamic response types

- WSDL: operation with defined message types with defined Schemas
- HL7v3 has attributes where content co-determines response Schema

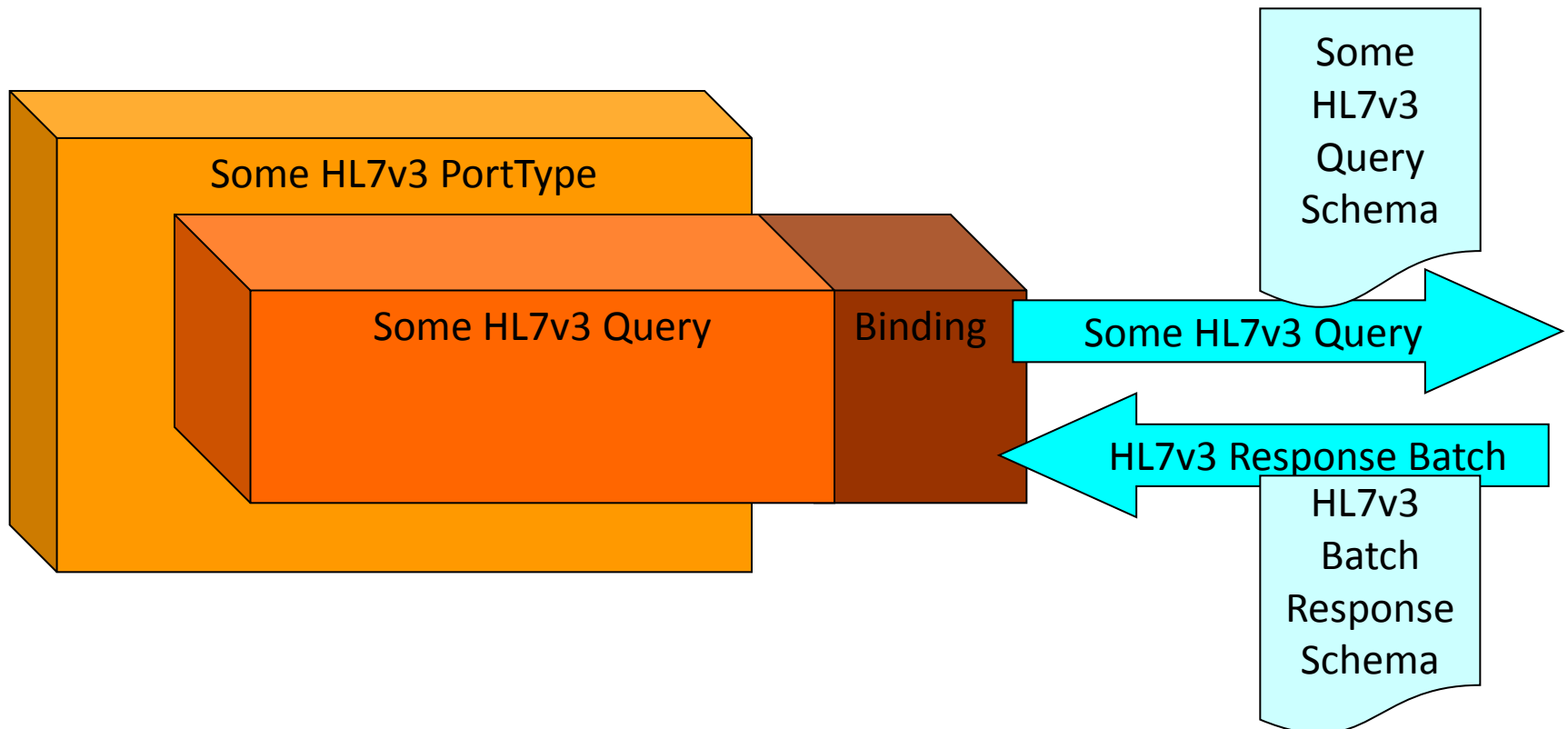
# Dynamic response

- HLv3 responseModalityCode = "R"



# Dynamic response (cont.)

- HLv3 responseModalityCode = "B"

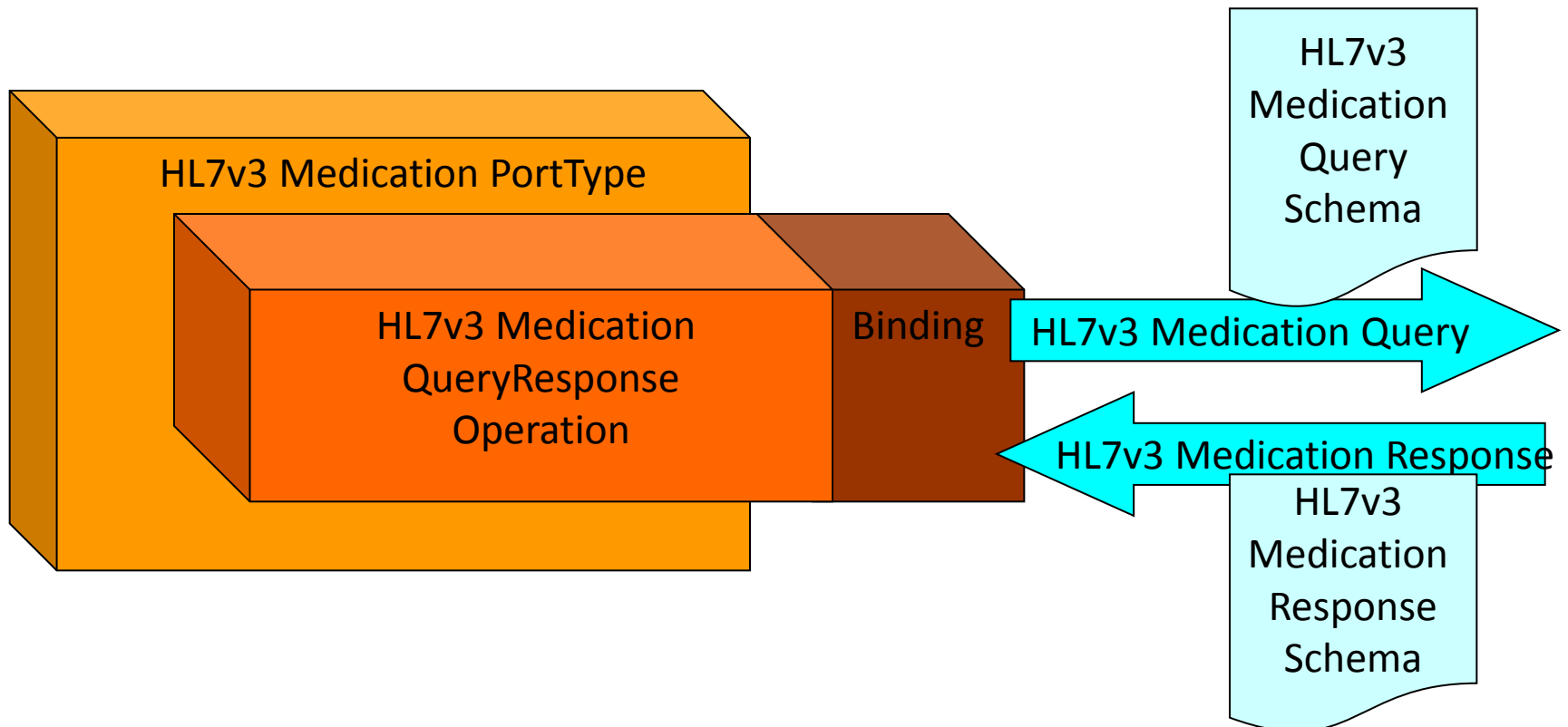


# Dynamic response (cont.)

- More HL7v3 attributes (=XML elements)
  - acceptAckCode
  - responsePriority
  - continuationQuantity
- content co-determines response Schema
- Solutions:
  - multiple PortTypes = clutter, bad design
  - <choice> in Schema = undescriptive, hard to read

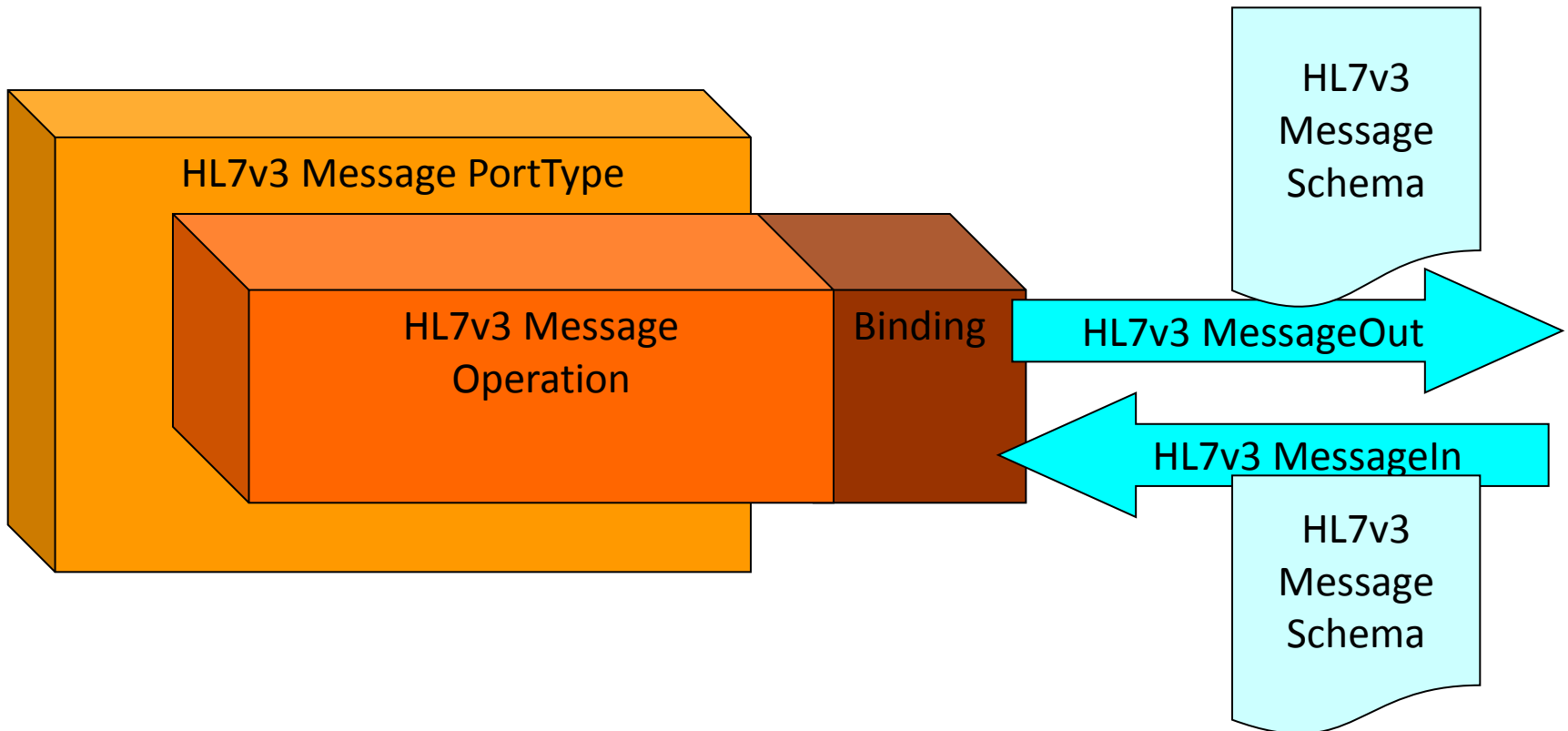
# Generic vs. specific WSDL

- Specific WSDL:



# Generic vs. specific WSDL

- Generic WSDL:

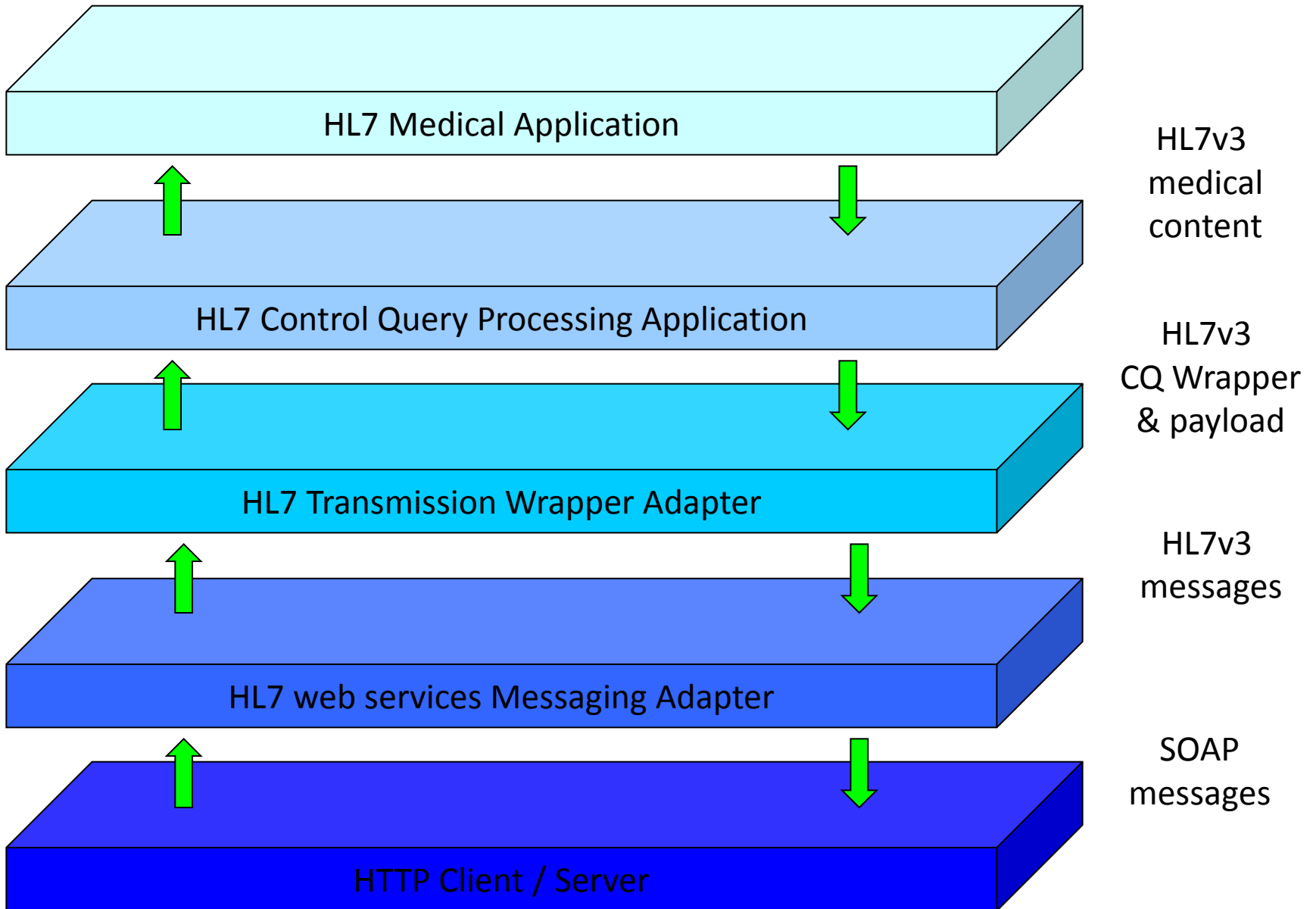


# Generic vs. specific WSDL

- Generic WSDL uses generic Schemas

```
<xs:schema targetNamespace="urn:hl7-org:v3">
  <xs:element name="hl7Message">
    <xs:complexType>
      <xs:sequence>
        <xs:any/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

- Possibly with Transmission Wrapper



# Generic WSDL

- Pro:
  - eases separation of layers
  - no problems with dynamic response
  - code generation is useful again
- Con:
  - does not describe actual web service neatly

# Basic Profile

- 2004: aanvullingen en correcties op SOAP/WSDL
- veel tooling ondersteunt dit
- belangrijk

# HL7 Web Services Profile

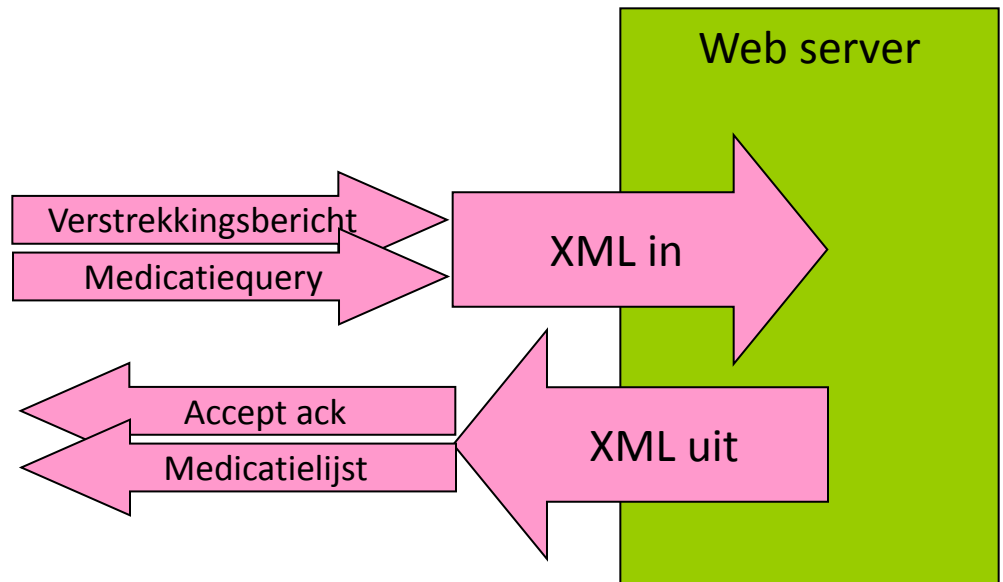
- HL7 Web Services Profile DSTU 1
  - SOAP, WSDL, WS-I Basic Profile 1.0
  - naamgeving WSDL-componenten
  - gladstrijken oneffenheden HL7 – SOA
- HL7 Web Services Profile DSTU 2
  - WS-Addressing, WS-Security, WS-ReliableMessaging
  - tamelijk dunne toevoegingen, veel nader in te vullen
- Daarna: stilstand, DSTU 2 is verlopen

# Web Services en HL7

- Abstract Transport Specification
- ebXML Messaging
- HL7/OMG
- IHE

# Transport vs. SOA

- HL7
  - RIM based model
  - serialisatie in XML
- Transport
  - agnostic
    - MIME
    - ebXML
  - alles kan erin



- Send XML / Receive XML
- Send something / Receive something

# Transport vs. SOA

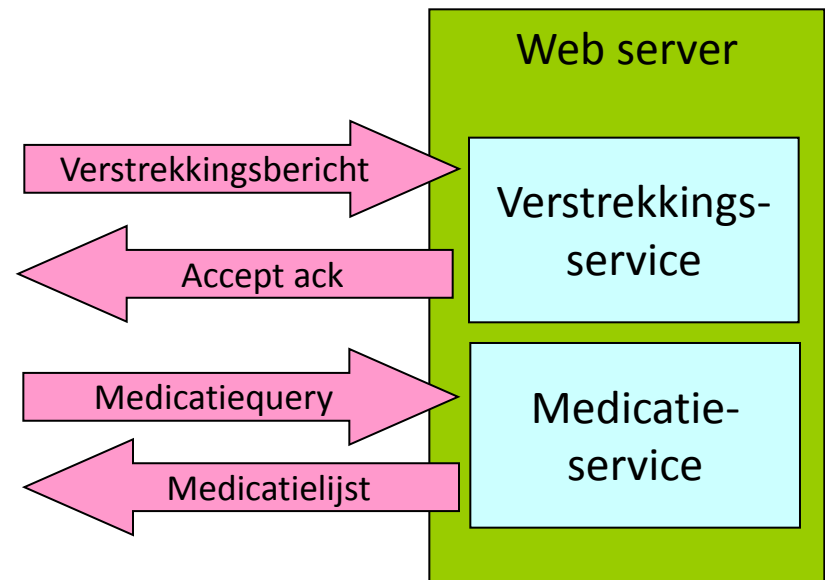
- SOA

- Services

- FindPatientsByTrait
    - DispenseMedication

- Expliciete service

- Namen van operations vastgelegd
    - Content model (schema) expliciet gemaakt
    - Blootstellen van services
    - Implementatie service is niet zichtbaar



# HL7

- Transmission wrapper
- Control Act Wrapper

# OMG/HL7

- Retrieve, Locate, and Update (RLUS) Service
- Entity Identification Service (EIS)
- HL7 Wrappers dropped

# IHE

- XCPD (Cross-Community Patient Discovery)
- Patient Identifier Cross-Reference HL7 V3 (PIXV3) and Patient Demographic Query HL7 V3 (PDQV3)
- Cross-Community Access (XCA)
- ebXML Registry / Repository
- MTOM

# 'Design time' versionering

- Design time versionering gaat over versies van dingen die gebruikt worden bij het implementeren
- Versies van:
  - documentatie (Aorta v6.0.0.0)
  - architectuur
  - implementatiehandleidingen
  - XML Schema
  - WSDL
  - Schematron
  - (gegenereerde) programmacode
  - database schema
- Op basis van wat bouw je?

# 'Run time' versionering

- Run time versionering gaat over versies van dingen die je tegenkomt in productie
- Versies van:
  - XML instances en HL7 artefacten daarin (datgene wat een client stuurt)
  - Web Services endpoints (datgene waarnaar het verzonden wordt)
- De versie van een ingestuurde XML instance kun je potentieel herkennen door middel van:
  - HTTP Header: SOAPAction
  - SOAP Headers in de SOAP Envelope
  - Top element in SOAP Body == HL7 interactionId (b.v. QURX\_IN990011NL)
  - HL7 namespace (urn:hl7-org:v3)
  - andere namespaces (b.v.: <http://www.aortarelease.nl/805/>)
  - profileId (<profileId root="2.16...11.1" extension="810"/>)
- De versie van een Web Service is te herkennen aan de URI
- Wat gebeurt er “on the wire”?