

# **IMPLEMENTIERUNGSLEITFADEN**

## **HL7 VERSION 3**

### **DATENTYPEN**

### **CMETS**

**für das Deutsche Gesundheitswesen**

**– Spezifikationen –**  
**Version 1.21**

Dokumenten-OID: 2.16.840.1.113883.2.6.7.51



HL7-Benutzergruppe in Deutschland e. V.

in Zusammenarbeit mit  
HL7 Niederlande (Stichting HL7 Nederland) und dem  
Nationalen IT Institut für das  
Gesundheitswesen der Niederlande (NICTIZ)

Geschäftsstelle Köln  
An der Schanz 1 • 50735 Köln  
Tel.: +49 700 77776761 • Fax: +49 700 7777 6767  
E-Mail: [info@hl7.de](mailto:info@hl7.de)  
[www.hl7.de](http://www.hl7.de)

## **IMPLEMENTIERUNGSLEITFADEN**

### **HL7 VERSION 3 DATENTYPEN CMETS**

### **FÜR DAS DEUTSCHE GESUNDHEITSWESEN**

vorgelegt von der

#### **HL7 Benutzergruppe in Deutschland e. V.**

Geschäftsstelle Köln  
An der Schanz 1  
50735 Köln

Erstellt durch Stichting HL7 Nederland, Technische Stuur Commissie, Veenendaal (Niederlande) und NICTIZ, Nationaal ICT Instituut in de Zorg, Leidschendam

Übersetzt aus dem Niederländischen und übertragen an die deutschen Gegebenheiten durch die HL7 Benutzergruppe in Deutschland e. V., Köln

Mit freundlicher Unterstützung des Verband der Hersteller von IT-Lösungen für das Gesundheitswesen, e.V. (VHitG), Berlin

#### *Ansprechpartner*

Kai U. Heitmann (email: hl7@kheitmann.nl)  
HL7-Benutzergruppe in Deutschland e.V.  
Heitmann Consulting and Services  
Sciphox Arbeitsgemeinschaft GbR mbH

Der Inhalt dieses Dokumentes ist öffentlich. Zu beachten ist, dass Teile dieses Dokuments auf dem Abstimmungspaket 11 und der Normative Edition 2005 von HL7 Version 3 beruhen, für die © HL7 Inc gilt.

#### Disclaimer

Obwohl diese Publikation mit größter Sorgfalt erstellt wurde, kann die HL7 Benutzergruppe in Deutschland keinerlei Haftung für direkten oder indirekten Schaden übernehmen, die durch den Inhalt dieser Spezifikation entstehen könnten.

# Dokumentinformation

Von diesem Dokument besteht eine elektronische und eine Druckversion.

## 1.1 Status

|                         |                |
|-------------------------|----------------|
| Erste Konzeptversion    | April 2005     |
| Letzte Konzeptversion   | November 2005  |
| Erste komplette Version | November 2005  |
| Korrigierte Version     | April/Mai 2006 |
| Publikationsdokument    |                |

## 1.2 Datum

## 1.3 Revisionsliste (NL)

| Version | Autoren   | Inhalt  | Datum      |
|---------|-----------|---|------------|
| 0.70    | KH        | Erster Ansatz Implementierungsleitfaden, teilweise basierend auf früheren Bearbeitungen von RS und TdJ    | 2005-05-09 |
| 0.71    | KH        | Anmerkungen erste und zweite Besprechung verarbeitet  | 2005-05-27 |
| 0.72    | MT        | Anmerkungen Einleitung  | 2005-06-08 |
| 0.73    | KH        | Anmerkungen dritte Besprechung verarbeitet  | 2005-06-24 |
| 0.80    | KH        | Korrekte Sequenz, Bearbeitungen RS, TdJ, KH verarbeitet   | 2005-07-16 |
| 0.81    | KH        | Bearbeitungen GB, TdJ, KH verarbeitet   | 2005-08-26 |
| 0.82    | TdJ       | Letzte Bearbeitungen TdJ verarbeitet (Leseversion für die DT-CMET Besprechung von HL7 NL am 6. September) | 2005-09-04 |
| 0.90    | KH, TdJ   | Verarbeitung der Anmerkungen des Besprechungstages und E-Mail-Kommentare                                  | 2005-10-05 |
| 0.91    | KH, TdJ   | Hinzufügen der noch fehlenden Elemente und Bearbeitung weiterer Anmerkungen                               | 2005-10-08 |
| 0.92    | TdJ       | Bearbeitung offener Punkte anlässlich des HL7 NL Tages  | 2005-10-18 |
| 1.00    | TdJ       | Publikation NICTIZ (HTML)   | 2005-11-30 |
| 1.20    | -Autoren- | Abstimmungskommentare Niederlande verarbeitet   | 2006-04-05 |

## 1.4 Revisionsliste (DE)

| Version | Autoren | Inhalt  | Datum      |
|---------|---------|---|------------|
| 1.20    | KH      | Übertragung in Deutsche auf der Basis der Übersetzung und der Diskussionen innerhalb des deutschen HL7 Version 3 Technischen Komitees | 2006-04-09 |
| 1.21    | KH      | Erste Kommentare verarbeitet  | 2006-05-07 |

## **Editoren**

Kai U. Heitmann (KH), Heitmann Consulting & Services (Purmerend) und HL7 Benutzergruppe in Deutschland e. V.

## **Autoren**

|                      |  |
|----------------------|--|
| Gerrit Boers (GB)    | Universität Maastricht (Maastricht)        |
| Kai U. Heitmann (KH) | Heitmann Consulting & Services (Purmerend) |
| Tom de Jong (TdJ)    | Nova Pro (Purmerend)                       |
| René Spronk (RS)     | Ringholm (Zeist)                           |
| Michael Tan (MT)     | NICTIZ (Leidschendam)                      |

## **Mit Beiträgen von**

|                 |                      |
|-----------------|----------------------|
| Christof Geßner | Optimal Systems GmbH |
|-----------------|----------------------|

# Inhaltsverzeichnis

|  |               |
|--|---------------|
| <b>Dokumentinformation</b>   | <b>4</b>      |
| 1.1 Status   | 4             |
| 1.2 Datum  | 4             |
| 1.3 Revisionsliste (NL)  | 4             |
| 1.4 Revisionsliste (DE)  | 4             |
| Editoren   | 5             |
| Autoren  | 5             |
| Mit Beiträgen von  | 5             |
| <br><b>2 Einleitung</b>  | <br><b>8</b>  |
| 2.1 Ziel dieses Dokuments  | 9             |
| 2.2 Aufbau des Dokuments   | 9             |
| 2.3 Relation zu den übrigen HL7v3 Implementierungsleitfäden        | 10            |
| 2.4 Status dieses Dokuments  | 10            |
| <br><b>3 Datentypen</b>  | <br><b>11</b> |
| 3.1 Einleitung   | 12            |
| 3.2 Allgemeine Erläuterungen zur Benutzung von XML                 | 12            |
| 3.2.1 Zeichensatz (Character Set)                                  | 12            |
| 3.2.2 Spezielle Zeichen  | 12            |
| 3.2.3 Allgemeiner Aufbau einer HL7 Version 3 Nachricht             | 13            |
| 3.2.4 Transport der XML Nachrichten                                | 13            |
| 3.3 XML Repräsentation von Klassen-Attributen                      | 13            |
| 3.3.1 Klassen-Attribute und Datentypen                             | 13            |
| 3.3.2 Ausnahmen mit Informationen als Element Content              | 14            |
| 3.3.3 Zusammengesetzte Datentypen                                  | 15            |
| 3.3.4 Strukturelle Attribute                                       | 15            |
| 3.4 Kardinalitäten, "mandatory" und "required"                     | 16            |
| 3.5 Fehlende Informationen: nullFlavors                            | 16            |
| 3.6 Der generische (ANY) Datentyp                                  | 19            |
| 3.7 BL (Boolean)   | 20            |
| 3.8 BIN (Binäre Daten – binary data)                               | 21            |
| 3.9 ED (Eingekapselte Daten – encapsulated data)                   | 22            |
| 3.9.1 Attribute von ED   | 22            |
| 3.9.2 Kindelemente von ED  | 23            |
| 3.10 ST (Zeichenkette – Character String)                          | 25            |
| 3.11 SC (Zeichenkette mit Code – Character String with code)       | 26            |
| 3.12 CD (Konzeptdeskriptor – Concept Descriptor)                   | 27            |
| 3.13 CE (Kodierte Daten mit Äquivalenten – Coded with Equivalents) | 28            |
| 3.14 CV (Kodierte Werte – Coded Value)                             | 31            |
| 3.15 CO (Sortierbarer kodierter Wert – Coded Ordinal)              | 32            |
| 3.16 CS (Einfacher Code – Coded Simple)                            | 33            |
| 3.17 II (Objekt Identifikation – Instance Identifier)              | 34            |
| 3.18 URL (Universal Resource Locator)                              | 36            |
| 3.19 TEL (Telekommunikationskontakt – Telecommunication Address)   | 37            |
| 3.20 AD (Adresse – Postal Address)                                 | 38            |
| 3.21 PN (Personenname – Person Name)                               | 40            |

|   |    |
|---|----|
| 3.22 ON (Organisationsname – Organization Name)                                     | 54 |
| 3.23 QTY (Quantität – Quantity)   | 57 |
| 3.24 INT (Ganze Zahl – Integer number)  | 58 |
| 3.25 REAL (Zahl mit Dezimalen – Real number)  | 59 |
| 3.26 PQ (Quantitative Angabe physikalischer Größen – Physical Quantity)             | 60 |
| 3.27 MO (Geldbetrag – monetary amount)  | 62 |
| 3.28 TS (Zeitpunkt – timestamp)   | 63 |
| 3.29 BAG Bag  | 64 |
| 3.30 SET Set  | 65 |
| 3.31 SXCM Set Component   | 66 |
| 3.32 IVL Intervall  | 67 |
| 3.33 IVL_TS (Zeitintervall – Interval of Timestamps)                                | 68 |
| 3.34 IVL_INT (Intervall von Integer – Interval of Integers)                         | 70 |
| 3.35 IVL_PQ (Intervall bei physikalischen Mengen – Interval of Physical Quantities) | 71 |
| 3.36 RTO (Verhältnisangaben zweier Daten – ratio)                                   | 72 |
| 3.37 GTS – Der generische Zeit Datentyp   | 73 |
| 3.38 PIVL (Periodisches Zeitintervall – Periodic Interval of Time)                  | 75 |

## **4 CMETS 88**

## **5 Referenzen 90**

## **6 Anhang 92**

|   |    |
|---|----|
| 6.1 HL7   | 93 |
| 6.2 Hinweise zur Vergabe und Verwendung von Object Identifiern (OIDs) | 93 |
| 6.2.1 Identifikationen von Objekten                                   | 93 |
| 6.2.2 Identifikationen von Codesystemen                               | 94 |

# 2

## Einleitung

---



*Note: This document contains a description of the constraints applicable to either Datatypes or CMETs in a German Context of Use (a.k.a. Realm). This document is available in German only. This Dutch version document was created under assignment by NICTIZ, the National IT institute for Healthcare, by Stichting HL7 Nederland, the Dutch affiliate, and is subject to the Dutch affiliate balloting process. This document has been translated and adapted to the German Healthcare requirements by HL7 Germany (HL7 Benutzergruppe in Deutschland e. V.) and is subject to the German affiliate balloting process.*

## **2.1 Ziel dieses Dokuments**

Die HL7 Version 3 Standard beschreibt unter anderem eine Reihe von Artefakts, die in vielen Nachrichten benutzt werden. CMETs und Datentypen werden in allen HL7v3 Nachrichten benutzt.

Die diversen Implementierungsleitfäden der HL7 Version enthalten bis jetzt jeweils eine Beschreibung der darin verwendeten Datentypen und CMETs. Um Wiederholungen zu vermeiden, aber auch aus Gründen der Konsistenz, wurde in mehreren Ländern beschlossen, für dieses Thema einen eigenen Implementierungsleitfaden zu erstellen.

Dieser Leitfaden ist eine Übersetzung des Leitfadens, den die niederländischen Kollegen erstellt haben. Er wurde an die deutschen Gegebenheiten und Erfordernisse angepasst.

Zweck dieses Dokuments ist es, die meistgebrauchten Datentypen und CMETs ausführlicher zu beschreiben und für die Anwendung im deutschen Gesundheitswesen zu spezifizieren. Dieses Dokument muss zusammen mit der HL7 Version 3 Standard Materie gelesen werden.

Dieses Dokument richtet sich vor allem auf Softwareentwickler von IT Anwendungen im Gesundheitswesen und auf das Gesundheitswesen bezogene infrastrukturelle Applikationen, die anhand des HL7 Version 3 Kommunikationsstandards und anhand dieses Dokuments ihre Nachrichtenschemas und Nachrichten bzw. Dokumente definieren wollen.

## **2.2 Aufbau des Dokuments**

Dieses Dokument enthält eine Beschreibung der wichtigsten Datentypen und CMETs, wobei ausschließlich die Aspekte beschrieben werden, die auf die deutschen Situation zutreffen. Falls es spezielle Anhaltspunkte dafür gibt, wie etwas (möglicherweise abweichend vom internationalen Standard) in Deutschland implementiert werden muss, wird dies im Text angegeben.

HL7 Artefakts werden in diesem Dokument mit ihrer offiziellen Identifikation konform mit der HL7 Version 3 Ballot #7 vom März 2004 angegeben. Diese Artefakts werden in diesem Dokument nicht im Detail besprochen. Dazu wird auf den HL7 Version 3 Standard selbst verwiesen.

Einige der in diesem Implementierungsleitfaden beschriebenen HL7 Artefakts sind (noch) nicht im internationalen HL7 Standard aufgenommen. Diese Artefakts werden in diesem Dokument detailliert beschrieben, da sie nicht in der HL7 Materie dokumentiert sind. Als Vorbereitung auf eine eventuelle spätere Aufnahme in den weltweiten Standard sind die Namen dieser HL7 Artefakts in Englisch abgefasst. Neue Artefakts haben eine Identifikation konform mit dem HL7 v3 Kennzeichnungs- und Identifikationsabkommen erhalten. Die (vorläufig) Deutschland-spezifischen Artefakts sind mit einem "DE" Code gekennzeichnet. Alle neuen Artefakts werden der internationalen HL7 Organisation zur Beurteilung vorgelegt, bevor sie in den internationalen Standard aufgenommen werden können. Wenn sie einmal aufgenommen sind, verfällt der "DE" Code.

## 2.3 Relation zu den übrigen HL7v3 Implementierungsleitfäden

Diverse Veröffentlichungen zu IT Spezifikation im Rahmen von HL7 Version 3 in Deutschland enthalten Beschreibungen von Datentypen und CMETs. Dieses Dokument ist eine detailliertere, nicht kontextabhängige Bearbeitung der Datentypen und CMETs. Dieses Dokument wurde zudem an neue Erkenntnisse aus vorigen Projekten und Spezifikationen angepasst.

Bereits publizierte Implementierungsleitfäden behalten ihre Relation zu den CMETs und Datentypen, die in den Implementierungsleitfäden enthalten waren. Allerdings werden künftige Publikationen an eine Version des Implementierungsleitfadens für CMETs und Datentypen gekoppelt.

## 2.4 Status dieses Dokuments

Dieses Dokument enthält eine Reihe deutscher Angaben zur Anwendung der HL7 Version 3. Dieses Dokument ist auf Ballot 7 des HL7v3 Standards basiert.

Dieses Dokument ist eine Erweiterung (und kein Ersatz) für die internationale HL7 Version 3 Materie. Bei Differenzen zwischen dem internationalen Standard und diesem Dokument gilt dasjenige, das in dieser Richtlinie festgelegt ist.

Dieses Dokument schreibt einzelne Einschränkungen der Freiheiten vor, die der internationale Standard bietet; Es ist ein "conformance profile". Parteien, welche die Version 3 auf der Basis des internationale HL7 Version 3 Standards implementieren, entsprechen damit also nicht dem "HL7 Deutschland conformance profile". Anbieter von Anwendungen im Gesundheitswesen und Lieferanten, die Daten über die kommende nationale Infrastruktur austauschen wollen, müssen dem "HL7 Deutschland conformance profile" entsprechen.

FAQ: An den Stellen, an denen dieses Dokument zusätzliche deutsche Anforderungen im Hinblick auf den internationalen Standard stellt, wird dies im Text angegeben mit "In Deutschland..." . Der internationale Standard beschreibt Objektklassen, die in diesem Dokument überhaupt nicht wiedergegeben oder beschrieben sind. Als HL7 Deutschland sehen wir für diese Klassen keine Anwendungsmöglichkeiten, obwohl die Benutzung dieser Klassen im Prinzip zugelassen ist.

Wenn nicht ausdrücklich anders angegeben wurde, ist in Deutschland das Vokabular (Code-Tabellen) anwendbar, das im internationalen Standard beschrieben ist.

Wenn Sie in der deutschen Situation eine HL7 Version 3 Datentyp oder CMETs benutzen wollen, die

1. im Widerspruch zu den deutschen Richtlinien ist, und die
2. nicht im Widerspruch zum internationalen HL7 Version 3 Standard ist,

können Sie ein Beispiel-Nutzungsszenario (use-case) bei HL7 Deutschland einreichen und einen Antrag zur Anpassung dieses Dokuments anmelden.

# 3

## Datentypen

---

### 3.1 Einleitung

In diesem Abschnitt werden die wichtigsten Datentypen beschrieben, wobei ausschließlich die Aspekte beschrieben werden, die auf die deutsche Situation zutreffen. Falls es spezielle Anhaltspunkte dafür gibt, wie etwas (möglicherweise abweichend vom internationalen Standard) in Deutschland implementiert werden muss, wird dies im Text angegeben.

Nur die innerhalb der Modelle dieses Leitfadens verwendeten Datentypen werden erläutert. Weitere Information finden Sie in der HL7 V3 Standard/Ballot Materie, namentlich in den Kapiteln "abstract data types" und "XML ITS data types".

### 3.2 Allgemeine Erläuterungen zur Benutzung von XML

Das Austauschformat für HL7 Version 3 Nachrichten ist (zum heutigen Zeitpunkt) Extensible Markup Language (XML, siehe [XML]). Es ist nicht Zweck dieses Leitfadens, allgemeine Erläuterungen über XML zu geben, allerdings werden einige Kennzeichen und Anforderungen hier trotzdem erwähnt.

#### 3.2.1 Zeichensatz (Character Set)

Das Encoding im XML-Prolog einer HL7 Version 3 Nachricht muss UTF-8 sein.

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

*oder (als default)*

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

#### 3.2.2 Spezielle Zeichen

In XML sind bestimmte Zeichen in Informationen durch Zeichenkombinationen zu ersetzen (character entities), weil sie im XML Format eine andere und spezielle Bedeutung haben. Eine XML Character Entity ist ein "&" gefolgt von einer Zeichenkette und einem abschließenden ";". Diese Sequenz wird vom XML Prozessor wieder ersetzt durch das spezielle Zeichen.

Die nachstehende Tabelle zeigt eine (unvollständige) Übersicht dieser Zeichen. Für mehr Informationen siehe die XML Spezifikation ([XML]).

| Spezielle Zeichen in Datei | Zu ersetzen durch | Anmerkungen  |
|----------------------------|-------------------|--|
| &                          | &amp;             |  |
| <                          | &lt;              |  |
| >                          | &gt;              |  |
| '                          | &apos;            |  |
| "                          | &quot;            | Ein " ist bei innerhalb von Attributwerten nicht zugelassen und muss ersetzt werden. |

#### XML Beispiel

```
<text>Werte &gt; 5 mg/l sind pathologisch</text>
```

→ XML Prozessor liefert: Werte > 5 mg/l sind pathologisch

```
<name>Apotheke Leermans &amp; Söhne</name>
```

→ XML Prozessor liefert: Apotheke Leermans & Söhne

### 3.2.3 Allgemeiner Aufbau einer HL7 Version 3 Nachricht

HL7 Version 3 Nachrichten haben als Root-Element immer den Namen der dazugehörigen Interaktion. So beginnt eine HL7 Version 3 Nachricht auf der Basis der Interaktion "PORX\_IN123456" mit einem XML Root-Element PORX\_IN123456. Die HL7 V3 Interaktionsschemas (allgemeine Erläuterungen W3C Schema Sprache siehe [XMLSC]) haben im allgemeinen den gleichen Namen wie das Root-Element.

```
<?xml version="1.0"? encoding="utf-8">
<PORX_IN123456 ...
  xmlns="urn:hl7-org:v3"
  xmlns:voc="urn:hl7-org:v3/voc"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  ... HL7 Version 3 Nachricht inklusive Wrapper und Payload
</PORX_IN123456>
```

Der Sender kann ein schemaLocation zwar hinzufügen, aber der Empfänger darf die Information in schemaLocation nicht benutzen.

Bei einem HL7 Message Batch hat das Root-Element den Namen der Batch-Interaktion und die verschiedenen Interaktionen (Teilberichte) sind darin aufgenommen. Für weitere Informationen siehe [Wrapper-Leitfaden].

### 3.2.4 Transport der XML Nachrichten

In diesem Leitfaden finden Sie keine Informationen über den eigentlichen Transport der Nachrichten. Hierzu erscheinen separate Leitfäden.

## 3.3 XML Repräsentation von Klassen-Attributen

### 3.3.1 Klassen-Attribute und Datentypen

Die Attribute von Klassen in den HL7 Modellen werden auf eine bestimmte, fest vordefinierte Weise in die XML Repräsentation umgesetzt.

Beispiel:

#### Patient

**classCode\*:** <= PAT

**id\*:** II [1..1]

**addr:** AD [0..1]

**telecom:** TEL [0..\*]

In der vorstehenden Klasse "Patient" ist eine Reihe von Attributen aufgenommen, darunter der classCode (mit dem festen Wert "PAT"), id, addr und telecom.

Jedes dieser Klassen-Attribute hat unter anderem

- einen Namen, beispielsweise id,
- einen Datentyp, beispielsweise AD,
- sowie eine Kardinalität, beispielsweise [0..\*]

Mit Ausnahme der strukturellen Attribute (siehe nachstehend) werden die Modell- Attribute als XML Elemente wiedergegeben. Die Elemente tragen den im Modell angegebenen Namen. Wenn z.B. ein Attribut den Namen *id* hat, ist der Name des XML Elements <id/>.

```
<id ... />
<addr ... />
<telecom ... />
```

In dem XML Schema ist die Kardinalität des Modellattributs aufgenommen. In dem vorstehenden Beispiel ist `<id>` Pflicht [1..1], `<addr>` [0..1] und `<telecom>` [0..\*] sind optional.<sup>1</sup>

Die XML Elemente haben XML Attribute, die durch den Datentyp selbst bestimmt werden. Die hier verwendete Notationsform für die Datentyp-Attribute lautet:

*Name Attribut .....Beschreibung / Erklärung / Bedeutung (Datentype /Pattern)*

Für die Datentypen wurden Attribute definiert, welche die Daten tragen. So hat in dem Beispiel das `id` Modellattribut den Datentype II (Instance Identifier). Die zu diesem Datentyp gehörenden Attribute sind beispielsweise `root` und `extension`, die als XML Attribute beim Element wiedergegeben werden.

```
<id root="..." extension="..." />
```

Nähere Erläuterungen zu den Datentypen und den dazugehörenden Attributen sind in den folgenden Kapiteln pro Datentyp aufgenommen.

### 3.3.2 Ausnahmen mit Informationen als Element Content

Bei den meisten Datentypen werden die faktischen Informationen in den XML Attributen wiedergegeben. Es gibt allerdings ein paar Ausnahmen. Bei den Datentypen *Binary*, *Encapsulated Data*, *Entity Name*, *Person Name*, *Organization Name*, *Trivial Name*, *Address* und *Character String* werden die Informationen als Element Content wiedergegeben.

Beispiel: Die Komponenten einer Adresse werden durch Kindelemente des `addr` Elements mit den Informationen als Element Content (im Beispiel rot markiert) aufgenommen.

```
<addr>
  <streetName>Große Bleichen</streetName>
  <houseNumber>23-25</houseNumber>
  <postalCode>20354 </postalCode>
  <city>Hamburg</city>
</addr>
```

<sup>1</sup> Achtung: Im heutigen XML Schema Sprache kann die Kardinalität eines XML Elements festgelegt werden, aber ob der "Inhalt", also die Informationen gemäß der Datentyp-Spezifikation vorhanden sind oder nicht, oder ob sie die korrekte Zusammensetzung haben, ist mit den jetzt vorhandenen Schemas nicht validierbar. Hier ist aber ein zweiter Validierungsschritt erforderlich, wenn garantiert werden muss, dass beispielsweise ein II Datentyp immer mit einem `root/extension` Attributpaar ausgefüllt ist oder ein `nullFlavor` Attribut enthält. Um dieses Validierungsniveau möglich zu machen, sind weitere Validierungsschritte mit so genannten *constraint* Sprachen erforderlich (beispielsweise OCL, Schematron).

### 3.3.3 Zusammengesetzte Datentypen

Es gibt eine Reihe zusammengesetzter Datentypen, die eine Sammlung von Datenelementen darstellen. So wird beispielsweise in einem Intervall die Unter- und Obergrenze angegeben und eine Ratio stellt ein Verhältnis zweier Werte dar.

Bei zusammengesetzten Datentypen wird die Substruktur (beispielsweise <low> und <high> bei einem Intervall) immer als Kindelement des betreffenden Datentyp-Elements wiedergegeben.

```
<effectiveTime>
  <low value="20040507"/>
  <high value="20040909"/>
</effectiveTime>
```

In den Beschreibungen der zusammengesetzten Datentypen werden die Kindelemente folgendermaßen notiert:

<Name Kindelement> .....Beschreibung /Erklärung /Bedeutung (Datentyp)

### 3.3.4 Strukturelle Attribute

Es gibt eine Liste von Klassen-Attributen, die nicht als separate Elemente präsentiert werden, sondern als XML Attribute der Klassenelemente.

| RIM Klasse      | Attribut             |
|-----------------|----------------------|
| Act             | moodCode             |
|                 | classCode            |
|                 | negationInd          |
|                 | levelCode            |
| ActRelationship | typeCode             |
|                 | inversionInd         |
|                 | contextControlCode   |
|                 | contextConductionInd |
|                 | negationInd          |
| Entity          | classCode            |
|                 | determinerCode       |
| Participation   | typeCode             |
|                 | contextControlCode   |
| Role            | classCode            |
|                 | negationInd          |
| RoleLink        | typeCode             |

Beispiel: In der Klasse *Observation* gibt es zwei Attribute, *classCode* und *moodCode*, die als strukturelle Attribute nicht als separate XML Elemente auftreten, sondern als XML Attribute des Klassenelements.

```
<Observation classCode="OBS" moodCode="EVN"/>
```

### 3.4 Kardinalitäten, "mandatory" und "required"

In den HL7 Version 3 Nachrichten können bei den Klassen-Attributen weitere Eigenschaften genutzt werden, wie z.B. die Kardinalität des Attributs. Die nachstehende Tabelle enthält eine Übersicht dieser zusätzlichen Eigenschaften.

| Begriff      | Erklärung / Anmerkungen  |
|--------------|--|
| Kardinalität | Spezifiziert die minimale und maximale Anzahl des Vorkommens eines Elements (Feld oder Assoziation) in einer XML Instanz. Beispielsweise 1..* bedeutet, dass das Element mindestens 1 Mal vorkommen muss und dass maximal eine unbeschränkte Anzahl Elemente zugelassen ist.   |
| Mandatory    | Ein "Mandatory" Feld/Assoziation <b>muss</b> in einer XML Instanz vorkommen, ansonsten ist die Nachricht ungültig. Für "Mandatory" Elemente ist die Mindestanzahl (Kardinalität) auf 1 (eins) festgelegt.  |
| Conformance  | Hier wird ein Unterschied gemacht zwischen R (required, Pflicht), NP (not permitted, nicht zugelassen) und optional.<br><br><i>R = Required</i> bedeutet, dass das sendende Anwenderprogramm dieses Feld oder diese Assoziation unterstützen muss. Wenn Daten verfügbar sind, muss dieses Feld/diese Assoziation in einer Nachricht vorhanden sein. Wenn die Mindest-Kardinalität 0 ist und wenn keine Daten verfügbar sind, darf das Element in einer XML Instanz fehlen und ist die Nachricht immer noch gültig. Wenn die Mindest-Kardinalität 1 ist und keine Daten verfügbar sind, muss dies mit einem nullFlavor (siehe an anderer Stelle in diesem Leitfaden) angegeben werden.<br><br><i>NP = not permitted</i> (nicht zugelassen) bedeutet, dass das Feld / die Assoziation nicht in einer Nachricht vorkommen darf (und auch nicht in dem zugrunde liegenden Schema vorhanden ist).<br><br><i>Optional</i> (optional) bedeutet, dass ein Element nicht oder wohl vorhanden sein darf und dass Unterstützung durch das sendende Anwenderprogramm auch nicht verpflichtend ist. |
| NullFlavor   | Für Felder/Assoziationen mit einer Mindest-Kardinalität von 1 muss ein nullFlavor angegeben werden, wenn in einem sendenden Anwenderprogramm keine Informationen für dieses Element verfügbar sind. Beispiele von nullFlavor sind "keine Information" (NI – no information), "unbekannt" (UKN – unknown) usw. Für nähere Erläuterungen siehe unter nullFlavor in diesem Leitfaden.   |

### 3.5 Fehlende Informationen: nullFlavors

Jeder Datentyp und jede Assoziation besitzt das Attribut *nullFlavor*, mit dem angegeben wird, dass der betreffende Wert fehlt, inklusive einer möglichen Erklärung für das Fehlen.

Dies wird verwendet, wenn eine Information in einer HL7v3 Nachricht verpflichtend ist (Kardinalität 1..), das sendende Anwenderprogramm aber einfach keinen Wert verfügbar hat. Es handelt sich dabei um Informationen, von denen die Conformance *required* ist, denn wenn die Conformance *mandatory* ist, muss grundsätzlich ein Nicht-Null-Wert zugewiesen werden.

Der Datentyp ist CS (Coded Simple Value) mit dem Vokabular der Domäne *NullFlavor*:

| NullFlavor |  |               |   |
|------------|--|---------------|---|
| Lvl        | Type, Domänenname und/oder Mnemonic code | Druckname     | Definition/Beschreibung   |
| 1          | <b>S: NoInformation</b><br>(NI)          | NoInformation | Aus der Verwendung von diesem nullFlavor darf keinerlei Information abgeleitet werden. Es bedeutet lediglich, dass die betreffende Information fehlt, |



|   |                                    |                         |  |
|---|------------------------------------|-------------------------|--|
|   |                                    |                         | ohne dass dafür ein Grund angegeben wird.  |
| 2 | L: (NA)                            | not applicable          | Es gibt keinen zutreffenden Wert in diesem Kontext. (z.B. Datum der letzten Menstruation bei einem männlichen Patienten).  |
| 2 | <b>S: Unknown</b> (UNK)            | Unknown                 | Es gibt zwar einen zutreffenden Wert, aber dieser ist beim Versender nicht bekannt (diverse Spezialisierungen sind möglich).   |
| 3 | L: (NASK)                          | not asked               | Die Information wurde nicht 'gesucht' (z.B. wenn dem Patienten eine bestimmte Frage nicht gestellt wurde) und ist dadurch nicht bekannt.   |
| 3 | <b>S: Asked But Unknown</b> (ASKU) | AskedButUnknown         | Die Information wurde zwar 'gesucht', aber nicht 'gefunden' (z.B. wenn der Patient zwar befragt wurde, es aber nicht wusste).  |
| 4 | L: (NAV)                           | Temporarily unavailable | Die Information ist momentan noch nicht vorhanden, aber man erwartet, dass sie zu einem späteren Zeitpunkt doch noch verfügbar sein wird.  |
| 3 | L: (TRC)                           | Trace                   | Es handelt sich um eine Menge die größer ist als 0, die aber zu klein ist, um zu quantifizieren. Dieser nullFlavor wird nur bei dem Datentyp PQ (Physical Quantity) verwendet.   |
| 2 | <b>S: Other</b> (OTH)              | Other                   | Es ist kein brauchbarer Wert in der Domäne verfügbar, der für die entsprechende Information zutreffend ist (z.B. vorgeschriebenes Vocabulary Domain für einen Code).   |
| 3 | L: (PINF)                          | positive infinity       | Ein numerischer Wert, der (positiv) endlos ist.  |
| 3 | L: (NINF)                          | negative infinity       | Ein numerischer Wert, der (negativ) endlos ist.  |
| 2 | L: (MSK)                           | Masked                  | Es ist zwar Information über dieses Item verfügbar, aber der Sender (oder eine Gateway) hat diese aus Gründen der Sicherheit, Privatsphäre oder anderweitig abgesichert. Es gibt evt. einen zusätzlichen Mechanismus um die Information zu erhalten. |

### XML Beispiel

Da keinerlei Information in einer HL7v3 Nachricht de facto den Datentyp ANY haben kann, wird hier die Verwendung des Attributs nullFlavor anhand einiger spezieller Datentypen erläutert (die das Attribut nullFlavor also von ANY erben). Bei den Beschreibungen im weiteren Verlauf dieses Implementierungsleitfadens wird im übrigen von Nicht-Null-Werten ausgegangen.

1) Der Anfrage/Verordnungszeitpunkt ist in einer HL7 v3 Nachricht Pflicht, aber das sendende Anwenderprogramm weiß einfach nicht, wann die Anfrage oder Verordnung ausgeschrieben wurde (auch wenn das Konzept unterstützt wird, weil es required ist).

```
<author>
  <time nullFlavor="UNK"/>
</author>
```

2) Ein Anwenderprogramm hat die Möglichkeit, Daten von Patienten zu registrieren, deren Burger Service Number (eindeutige Identifikationsnummer der Niederlande) (noch)

nicht bekannt ist, und diese später doch noch an die registrierten Daten zu koppeln. In diesem Fall kann die Personenidentifikation (wenn sie verpflichtend in der betreffenden HL7 v3 Nachricht vorhanden ist), folgendermaßen wiedergegeben werden:

```
<Patient>
  ...
  <Person>
    <id nullFlavor="NAV"/>
    ...
  </Person>
</Patient>
```

3) In bestimmten Situationen wird bei den Spezifikationen ein festes Vocabulary Domain angegeben, ohne die Möglichkeit, extra Werte hinzuzufügen. Das sendende Anwenderprogramm kann allerdings keinen dieser Werte in der betreffenden Domäne nutzen (ein Beispiel dieser Situation ist die Übermittlung von Arzneimittel-Rezepturen).

```
<medicationKind>
  <code nullFlavor="OTH"/>
</medicationKind>
```

4) Bei der Spezifikation der Inhaltsstoffe eines bestimmten Arzneimittels muss angegeben werden, dass es Spuren von Eisen enthält, obwohl die exakte Menge nicht bekannt ist (und angesichts der geringen Menge nicht relevant). In diesem Fall könnte angegeben werden, dass es sich um eine TRC (trace) Menge von unbekannter Größe handelt.

```
<ingredient>
  <quantity nullFlavor="TRC"/>
</ingredient>
```

5) Das Privacy Filter eines bestimmten Anwenderprogramms (oder ein zwischengelagertes Gateway) hat beschlossen, dass eine bestimmte Information nicht übermittelt werden darf. Der betreffende Wert wird durch ein nullFlavor des Typen MSK (masked) ersetzt, um ihn abzuschirmen. Achten Sie darauf, dass der Empfänger trotzdem darüber informiert wird, dass die Information verfügbar war!

```
<subject>
  <Patient>
    ...
    <addr nullFlavor="MSK"/>
    ...
  </Patient>
</subject>
```



Achten Sie darauf, dass die Anwendung des `nullFlavor` Attributs in welcher Information einer HL7 v3 Nachricht auch immer, bedeutet, dass grundsätzlich kein anderes Attribut oder Element der betreffenden Information vorhanden sein darf. Eine Ausnahme hierzu ist zum Beispiel die Nutzung von *originalText* in Kombination mit `nullFlavor="OTH"` im Datentyp CD.



Innerhalb von HL7 ist leider noch keine Klarheit darüber, wie mit *xsi:nil* in Assoziationen umgegangen werden muss. An sich sind *nullFlavor* und *xsi:nil* äquivalent. Die XML Prozessoren aber verlangen, dass neben *nullFlavor* auch *xsi:nil* angegeben wird.

### 3.6 Der generische (ANY) Datentyp

Dieser abstrakte Datentyp ist die Basis für alle anderen Datentypen. Kein einziger Wert in einer HL7 v3 Nachricht hat de facto den Datentyp ANY, obwohl jeder Datentyp innerhalb HL7 v3 eine Spezialisierung von ANY ist. Das bedeutet auch, dass jeder andere Datentyp die Attribute von ANY mittels Vererbung übernimmt (siehe "Fehlende Daten: `nullFlavor`").

Der ANY Type kommt hin und wieder in den HL7 Modellen vor, wobei es sich meistens um den Wert klinischer Befunde oder Verordnungen handelt. Der ANY Typ kommt jedoch in XML Nachrichten nicht vor, da ANY immer durch einen bestimmten Datentyp in einer Nachricht ersetzt wird.

Der Datentyp des Attributs *value* ist beispielsweise bei einer Observation "ANY", denn es ist vorab (im Modell) nicht deutlich, welcher Typ für den Wert zutreffend ist. Dies wird jedoch erst durch die faktische Instanziierung festgelegt. Der Datentyp für *value* muss also immer über die *xsi:type* Instruktion festgelegt werden. Wenn man einen ANY Typ in einer Instanziierung nicht begrenzt, kann eine XML Nachricht nicht validiert werden.

Attribute eines Elements mit diesem Datentyp sind:

*@nullFlavor.....fehlender Wert (CS)*

Nachstehend sind einige Beispiele aufgeführt, aber für alle Datentypen gilt, dass ein `nullFlavor` angegeben werden kann (außer wenn das Element mandatory ist).

#### XML Beispiele

1) ANY umgeformt in CE

```
<value xsi:type="CE" code="N11.9" codeSystem="2.16.840.1.113883.6.3"/>
```

2) ANY umgeformt in CD

```
<value xsi:type="CD" code="C1" codeSystem="2.16.528.1.1003.99.100"
  displayName="Warnung: die gefundenen Namensdaten
  weichen ab von den Namensdaten in der Anfrage."/>
```

3) ANY umgeformt in PQ

```
<value xsi:type="PQ" value="12" unit="ml"/>
```

4) ANY umgeformt in ED

```
<value xsi:type="ED">dies ist der Text mit der Begründung</value>
```

### 3.7 BL (Boolean)

Der Datentyp BL (Boolean) bezieht sich auf die so genannte Zwei-Werte-Logik. Eine Information dieses Typs kann lediglich die Werte "true" oder "false" enthalten oder ein nullFlavor, falls die Conformance dies zulässt (also wenn die Information nicht mandatory ist).

Jeder Wert (oder zwei Werte) des Typen Boolean kennt die nachstehenden Bearbeitungen:

| Tabelle für Boole'sche Logik<br>(ein nullFlavor bedeutet, dass ein Wert fehlt) |       |  |       |       |       |       |  |       |      |       |      |
|--|-------|--|-------|-------|-------|-------|--|-------|------|-------|------|
| NOT  |       |  | AND   | true  | false | NULL  |  | OR    | true | false | NULL |
| true   | false |  | true  | true  | false | NULL  |  | true  | true | true  | true |
| false  | true  |  | false | false | false | false |  | false | true | false | NULL |
| NULL   | NULL  |  | NULL  | NULL  | false | NULL  |  | NULL  | true | NULL  | NULL |

Eine Information mit dem Datentyp BL hat (wenn es Nicht-Null ist) das Attribut *value*. Die möglichen Werte sind "true" oder "false", womit angegeben wird, ob die Information richtig oder falsch ist.

*value*..... Wert (true|false)

#### XML Beispiele

1) Eine Person (oder ein anderes 'living subject') ist gestorben.

```
<livingSubject>
  ...
  <deceasedInd value="true"/>
</livingSubject>
```

2) Eine Assoziation ist non-conductive (das heißt: gibt keinen Kontext an).

```
<support2 contextConductionInd="false">
  ...
</support2>
```

In der vorstehenden Situation ist der Datentyp BL nicht zutreffend auf ein XML Element, sondern auf ein Attribut. In diesem Fall erhält das betreffende Attribut direkt den Boole'schen Wert (Es übernimmt also praktisch die Rolle, die das Attribut *value* normalerweise hat).

### **3.8 BIN (Binäre Daten – binary data)**

BIN ist der Supertyp für Multimedia-Daten. Der BIN-Typ kommt nicht selbständig in Nachrichten vor. Wenn Multimedia-Daten in einer Nachricht aufgenommen werden müssen, muss dazu der Encapsulated Data (ED) Typ verwendet werden.

### 3.9 ED (Eingekapselte Daten – encapsulated data)

Dies ist ein allgemeiner Typ für allerlei Multimedia-Daten. In Deutschland wird dieser Typ vorläufig für Texte mit oder ohne (einfachem) Layout verwendet.

ED ist ein komplexer Typ, der Elemente und Attribute enthält. Die Daten (Text, Bilder) befinden sich im ED-Element in der Form, die das 'encoding' Attribut spezifiziert hat.

Der ED-Typ kennt zweierlei Nutzungsformen:

1. *Inline data*  
In diesem Typ werden die vollständigen Daten gesendet. Diese Form wird überwiegend für Texte verwendet.
2. *By reference*  
Eine verkleinerte Version der Daten wird in einem "thumbnail" erfasst, wobei mit einer "reference" auf die vollständigen Daten verwiesen wird. Diese Form wird in Deutschland vorläufig noch nicht verwendet.

#### 3.9.1 Attribute von ED

*encoding* ..... *Kodierung (cs)*

Benutzung dieses Attributs ist optional. Dabei sind zwei Kodierungstypen möglich.

| Tabelle der Attributwerte für encoding |   |
|--|---|
| Code                                   | Definition  |
| TXT                                    | Für Textdaten. Das ist der Standardtyp. Wenn kein Encoding angegeben ist, wird von TXT ausgegangen. |
| B64                                    | Die Base 64 Kodierung wird für alle anderen Multimedia Daten benutzt                                |


*mediaType* ..... *Datenart (cs)*

Dieses Attribut zeigt die Art der Daten an. In Deutschland werden vorläufig nur die verpflichteten Datenarten unterstützt, die in der nachstehenden Tabelle wiedergegeben sind:

| Tabelle mediaType Attribut Werte (Datenarten): |                    |               |   |
|--|--------------------|---------------|---|
| Code   | Name               | Status        | Definition  |
| text/plain                                     | Plain Text         | verpflichtend | Für willkürliche Texte. Dies ist der 'default' Typ. In dieser Form ist er identisch mit dem ST Type.  |
| text/html                                      | HTML Text          | empfohlen     | Bestimmt für formatierte Texte in HTML Format. HTML reicht für die meisten Anwendungen aus, bei denen Layout erwünscht ist. HTML ist Plattform-unabhängig und weit verbreitet.      |
| audio/basic                                    | Basic Audio        | verpflichtend | 1- Kanal Audioformat für Sprache. Obwohl Unterstützung dieses Formats verpflichtend ist, wird es in den Niederlanden kaum benutzt werden.   |
| audio/mpeg                                     | MPEG audio layer 3 | verpflichtend | MPEG-1 Audio layer-3 (auch bekannt als MP3) ist momentan der Standard für komprimierte Audiodaten.  |
| image/png                                      | PNG Image          | verpflichtend | Portable Network Graphics (PNG) [http://www.cdrom.com/pub/png] ist eine verlustfreie Komprimierung von Bilddateien. Wo vorher GIF benutzt wurde, muss jetzt PNG unterstützt werden. |
| image/jpeg                                     | JPEG Image         | verpflichtend | Dieses Format wird benutzt für hohe Resolutionen bei Fotos und anderen Bilddateien. Die Komprimierung verläuft nicht ohne Verluste, wodurch dieses Format nicht                     |

|                 |            |               |   |
|-----------------|------------|---------------|---|
|                 |            |               | immer geeignet ist für diagnostische Zwecke. JPEG wird vorwiegend benutzt werden für 'thumbnails' von großen (DICOM) Dateien.   |
| video/mpeg      | MPEG Video | verpflichtend | MPEG ist ein internationaler Standard für Videobilder. Er ist weit verbreitet und Open-Source-Implementierungen sind verfügbar. |
| text/rtf        | RTF        | empfohlen     | RTF-Format  |
| application/pdf | PDF        | verpflichtend | PDF-Format  |

Zugelassene *mediaTypes* in einer faktischen Implementierung können weiter eingeschränkt werden.



Es wird ausdrücklich keine Begrenzung der Größe von den Attributwerten vorgegeben. Auch bei spezifischen Elementen einer HL7 Nachricht geschieht dies nicht, weil der Standard die maximale Größe prinzipiell nicht einschränkt.

Vereinbarungen hierüber werden auf Implementierungsniveau gemacht. Ohne spezifische Vereinbarungen muss der Empfänger in der Lage sein, Encapsulated Data (z. B. Freitext) von beliebiger Größe zu verarbeiten.

### 3.9.2 Kindelemente von ED

*reference* ..... *Verweis (TEL)*

Dieses Element wird nur in Kombination mit dem 'thumbnail' Element benutzt. Es enthält den Hinweis auf die vollständigen Daten in der Form eines TEL (Telecom) Typs.

*thumbnail* ..... *verkleinerte Wiedergabe (ED)*

Dieses Element wird benutzt wenn die Datei zu groß ist für den ED-Typ. Das 'reference' Element enthält den Hinweis auf die originale Datei. Das 'thumbnail' Element kann beispielsweise benutzt werden, um JPEG Versionen von DICOM Dateien weiterzuleiten.

Da 'thumbnail' ein ED-Typ ist, können alle hiervoor genannten Datenarten darin untergebracht werden.

### XML Beispiele

Einfacher Freitext:

```
<text xsi:type="ED" mediaType="text/plain">
Die häusliche Situation des Patienten ist schwierig, da die Kinder weit
weg wohnen.
</text>
```

Beispiel einer Anwendung von Thumbnail und Reference:

```
<value xsi:type="ED" mediaType="image/png" encoding="B64">
  <reference value="http://radiology.iuinc.edu/xrays/128s.png">
    <useablePeriod xsi:type="IVL_TS">
      <low value="200007200845" />
      <high value="200008200845" />
    </useablePeriod>
  </reference>
</value>
```

```
</useablePeriod>
</reference>
<thumbnail mediaType="image/jpeg" representation="B64">
MNYD83jmMdomSJUEdmde9j44zmMir6edjzMMIjdMDSsWdIJdksIJR3373jeu83
6edjzMMIjdMDSsWdIJdksIJR3373jeu83MNYD83jmMdomSJUEdmde9j44zmMir
...
omSJUEdmde9j44zmMiromSJUEdmde9j44zmMirdMDSsWdIJdksIJR3373jeu83
4zmMir6edjzMMIjdMDSsWdIJdksIJR3373jeu83==
</thumbnail>
</value>
```



### 3.10 ST (Zeichenkette – Character String)

Dieser Typ ist für Freitext in der einfachsten Form gedacht. ST ist eine Spezialisierung des ED-Typs. Der *mediaType* ist festgelegt mit 'text/plain' und der *encoding* Typ ist 'TXT'. Die anderen Attribute und Elemente von ED dürfen beim ST-Typ nicht benutzt werden.

Der ST-Typ wird vorwiegend in anderen Datentypen, wie AD und PN verwendet. Im allgemeinen gilt jedoch, dass man für Texte besser den ED-Typ benutzen kann, weil dieser ja auf einen ST-Typ reduziert werden kann, wenn die Attribute korrekt ausgefüllt werden.

#### ***XML Beispiel***

Dieses Beispiel ist nahezu gleich mit dem ersten Beispiel bei ED. Funktional sind beide identisch.

```
<text xsi:type="ST">
```

```
Die häusliche Situation des Patienten ist schwierig, da die Kinder weit  
weg wohnen.
```

```
</text>
```

### **3.11 SC (Zeichenkette mit Code – Character String with code)**

Dieser Typ ist eine Erweiterung des ST-Typen, ergänzt mit Attributen des Datentyps CV. Hierdurch ist es möglich, den Text über einen Code näher zu spezifizieren. Momentan wird der SC-Typ noch nicht benutzt. In einem folgenden Ballot wird er ein Bestandteil des AD (Adresse) Typen werden. Es wird dann möglich sein, um beispielsweise dem Land, neben dem freien Text, auch einen Code zuzuordnen.

### 3.12 CD (Konzeptdeskriptor – Concept Descriptor)

Dieser Datentyp spezifiziert ein Konzept über einen Code und das Codesystem (die Tabelle) aus dem der Code stammt, und enthält optional eine oder mehrere Übersetzungen dieses Codes mit Hilfe anderer Kodiersysteme.

Der einzige Unterschied zwischen dem CE-Datentyp und dem CD-Datentyp ist der Fakt, dass CD ein *qualifier* Kindelement besitzen kann. Die weiteren Attribute, mit Ausnahme von *qualifier*, sind bei der Beschreibung von CE für die Benutzung des CE-Datentyps in Deutschland zu finden.

*<qualifier>.....Ergänzung zum Code (CR)*

Optional. Enthält eine nähere Präzisierung des unter Code Attribut beschriebenen Konzepts. In dieser Version des Implementierungsleitfadens wird dieses Attribut und der verwendete CR-Datentyp nicht näher ausgearbeitet. Momentan sind in der deutschen Situation keine Terminologien bekannt, die dieses Attribut verwenden. Die Benutzung komplexer Terminologien (z.B. SNOMED CT) ist ausschließlich in Kombination mit diesem Attribut möglich.

### 3.13 CE (Kodierte Daten mit Äquivalenten – Coded with Equivalents)

Dieser Datentyp spezifiziert ein Konzept über einen Code und das Codesystem (die Tabelle) aus dem der Code stammt und enthält optional eines oder mehrere Äquivalente dieses Codes mit Hilfe anderer Codiersysteme.

Beispiel: Das Konzept "Männlich" wird abhängig von dem verwendeten Codiersystem identifiziert, beispielsweise durch ein "M" oder den Code "1". Der Empfänger ist nur dann in der Lage, einen Code eindeutig zu interpretieren, wenn neben dem Code auch das verwendete Codiersystem identifiziert wird. Die Kombinationen ("M", HL7 v3 Tabelle *AdministrativeGender*) oder deren Übersetzung ("1", ABC-KIS System Geschlechtscodetabelle) sind eindeutig zu interpretieren.

Attribute eines Elements mit diesem Datentyp sind:

**@code** ..... *Code (string)*

Verpflichtend. Enthält den Code (mnemonic), eine Identifikation des Konzepts, das in dem unter *codeSystem* angegebenen Codiersystem beschrieben ist.

**@codeSystem** ..... *Codiersystem (OID)*

Verpflichtend. Enthält die Identifikation des Codiersystems (Tabelle, Terminologie). Zur Identifikation wird ein OID verwendet.

Ein ISO Object Identifier (OID) ist ein weltweit eindeutiger String (Zeichenkette), der aus Zahlen und Punkten besteht (beispielsweise "2.16.840.1.113883.3.1"). Laut ISO Definition bestehen OIDs aus Pfaden mit einer Baumstruktur, wobei die äußerst links situierte Zahl die *root* ist und die am meisten rechts situierte Zahl ein *leaf* (ein Blatt als Endpunkt) markiert. Die Nummer ist garantiert weltweit eindeutig, weil sie auf dem System der delegierten Verantwortlichkeit basiert. Jeder Zweig unter einer Wurzel in der Baumstruktur korrespondiert mit einer Domäne, in der eine Organisation die Abgabe von OIDs verwaltet. Die zentralen Vergabestelle für OIDs im Gesundheitswesen in Deutschland publiziert eine Tabelle mit OIDs. Informieren Sie sich bei Zweifel bei der zentralen Vergabestelle, welche (eventuell neu zu registrierende) OID benutzt werden muss.

Im *OID Konzept für das Deutsche Gesundheitswesen [oidk]* finden Sie nähere Informationen über Codiersysteme und OIDs.

Grundsätzlich sollte erwähnt werden, dass die Verwendung von Codesystemen in einem bestimmten Kontext z. B. durch Implementierungsleitfäden eingeschränkt wird.

**@displayName** ..... *Konzeptbeschreibung (string)*

Optional. Eine textliche Beschreibung des Konzepts. Diese Beschreibung legt das sendende System seinen Benutzern vor und auf Basis dieser Beschreibung selektiert ein Pflegeanbieter den Code. Dem Wert dieses Attributs darf keine Bedeutung zugemessen werden, außer dass es dem Benutzer vorgelegt wird, um den Hintergrund des Codes zu verdeutlichen. Ein *displayName* darf niemals ohne zugehörigen *code* vorkommen und muss dieselbe Bedeutung haben.



Es darf nicht erwartet werden, dass der Empfänger den *displayName* auf Konsistenz mit dem angegebenen Code prüft. Der Empfänger kann selbst bestimmen, ob er eine eventuell vorhandene eigene Beschreibung oder den *displayName* des Senders für eine Wiedergabe benutzt. Bei der Benutzung eines nullFlavors darf kein *displayName*, wohl aber das Kindelement *originalText* genutzt werden, um nicht-kodierte Beschreibungen anzugeben.

**<originalText>** ..... *ursprünglicher Text (string)*

Optional. Eine textliche Beschreibung des Konzepts. Dies stellt den Text dar, worauf das sendende System den Code zugewiesen hat. Dies ist also gerade umgekehrt zu *displayName* zu sehen, wo eine Beschreibung auf Basis des Codes bestimmt wird. Das bedeutet auch, dass *originalText* gerade auch ausdrücklich in den Fällen vorkommen kann, wo kein Code zugewiesen ist oder gefunden werden kann. In diesem Falle biete *originalText* die Möglichkeit, Text anzugeben, der anscheinend (noch) nicht in einen Code umzusetzen war.



Es darf nicht erwartet werden, dass der Empfänger den *originalText* auf Konsistenz mit dem eventuell abgeleiteten Code prüft. Bei der Nutzung von *null-Flavor* in diesem Datentyp kann der *originalText* als Alternative zum codierten Wert verwendet werden (siehe auch Beispiele unten).

#### *@codeSystemName ..... Name des Codiersystems (string)*

Optional. Eine Textform des Namens des Codiersystems, das den Code enthält. Dem Wert dieses Attributs darf keine Bedeutung zugemessen werden, außer dass es dem Benutzer vorgelegt wird, um den Hintergrund des Codes zu verdeutlichen. Für die Lesbarkeit von Nachrichten wird empfohlen, das *codeSystemName* mitzusenden.

#### *@codeSystemVersion ..... Version des Codiersystems (string)*

Optional. Eine Textform der Version des Codiersystems, das den Code enthält. Dem Wert dieses Attributs darf keine Bedeutung zugemessen werden, außer dass es dem Benutzer vorgelegt wird, um den Hintergrund des Codes zu verdeutlichen.

Verschiedene Versionen eines Codiersystems können über einzelne OIDs im *codeSystem* Attribut identifiziert werden. Das empfangende System darf deshalb niemals den Wert des *codeSystemVersion* benutzen, um den Code zu interpretieren. Ob eine neue Version eines Codesystems ein neue OID bekommt, hängt von verschiedenen Faktoren ab, die hier nicht näher erläutert werden sollen. So sehen die verschiedenen ICD-Schlüssel in Deutschland für jede Version eine eigene OID vor, verschiedene Versionen des LOINC-Codes beispielsweise hingegen haben konstant eine OID.

#### *<translation> ..... Übersetzungen des Codes (CD)*

Optional. Null oder mehr Übersetzungen des Konzepts mit Hilfe alternativer Codiersysteme.

Alle Übersetzungen müssen auf ein und dasselbe Konzept hinweisen. Dabei ist es zugelassen, dass die Übersetzungen "weniger nuanciert/detailliert" sind als das originale Konzept.

Beispiel: Das Konzept "Granny Smith" kann, wenn ein Codiersystem nicht das entsprechende Niveau an Details enthält, übersetzt werden in das Konzept "Apfel". Das Konzept "Apfel" darf allerdings niemals übersetzt werden in das detaillierte Konzept "Granny Smith". Das Konzept "Apfel" darf ebenfalls nicht übersetzt werden in das Konzept "Grün": Es handelt sich dabei vielleicht um verwandte Konzepte, aber inhaltlich ist es keine Übersetzung des originalen Konzepts.

### **XML Beispiele**

```
<administrativeGenderCode code="M" codeSystem="2.16.840.1.113883.5.1"/>
```

```
<value xsi:type="CE" code="Z94.0" codeSystem="2.16.840.1.113883.6.3"
  displayName="Lungenentzündung" codeSystemName="ICD10"/>
```

```
<code code="ERY" codeSystem="2.16.840.1.113883.2.4.4.13"/>
```

```
<code code="GSMITH" codeSystem="2.16.840.1.22.3"
      codeSystemName="US Fruit Category" displayName="Granny Smith">

  <translation code="100" codeSystem="2.16.840.1.113883.2.4.4.99"
    displayName="Apfel"
    codeSystemName="Deutscher Verband für Nahrungsmittel" />

</code>
```

```
<code code="13715119" codeSystem="2.16.840.1.113883.2.4.4.8"
      codeSystemName="G-Standard Artikel"
      displayName=" ABSORIN UNTERLAGE 60X90CM 120G FLUFF 60920">

  <translation code="753696" codeSystem="2.16.840.1.113883.2.4.4.7"/>

</code>
```

```
<code nullFlavor="OTH" originalText="Een zelfgemaakt medicijn"/>
```

### 3.14 CV (Kodierte Werte – Coded Value)

Dieser Datentyp spezifiziert ein Konzept über einen Code und das Codesystem (die Tabelle) aus dem der Code stammt.

Der einzige Unterschied zwischen dem CV-Datentyp und dem CE-Datentyp ist der Fakt, dass CE Übersetzungen aus einem Konzept enthalten kann und ein *qualifier* Attribut besitzt. Siehe Beschreibung von CE – mit Ausnahme von *qualifier* und *translation* – für die Verwendung des CV-Datentyps in Deutschland.

#### ***XML Beispiel***

```
<code code="SF36" codeSystem="2.16.840.1.113883.2.4.4.13"/>  
  
<value xsi:type="CV" code="Z94.0" codeSystem="2.16.840.1.113883.6.3"  
  displayName="Lungenentzündung" codeSystemName="ICD10"/>
```

### 3.15 CO (Sortierbarer kodierter Wert – Coded Ordinal)

Dieser Datentyp spezifiziert ein Konzept über einen Code und das Codesystem (die Tabelle) aus dem der Code stammt.

Der einzige Unterschied zwischen dem CO-Datentyp und dem CV-Datentyp ist der Fakt, dass die Konzepte, die im CO Datentyp aufgenommen sind, eine bestimmte Sequenz haben und sortiert werden können. Siehe Beschreibung von CV, für eine Beschreibung der Verwendung des CO-Datentyps in Deutschland.

#### ***XML Beispiele***

```
<value xsi:type="CO" code="4" codeSystem=.../>
```

```
<value xsi:type="CO" code="+++" codeSystem=.../>
```



### 3.16 CS (Einfacher Code – Coded Simple)

Dieser Datentyp spezifiziert ein Konzept über einen Code aus einem vordefinierten Codesystem (die Tabelle), aus dem der Code stammt. Dieser Datentyp wird häufig verwendet, wenn feste, HL7 definierte Tabellen, benutzt werden.

*@code ..... Code (string)*

Verpflichtend. Enthält den Code (mnemonic), eine Identifikation des Konzepts, das in dem vordefinierten Codiersystem beschrieben ist.

#### **XML Beispiele**

```
<Observation classCode="OBS" moodCode="EVN"/>
```

```
<statusCode code="active"/>
```

### 3.17 II (Objekt Identifikation – Instance Identifier)

Dieser Datentyp spezifiziert Identifikationen von Objekten. Dazu gehören beispielsweise Identifikationen für Organisationen oder Personen. Ein Attribut des II Datentyp enthält eine weltweit eindeutige Identifikation eines Objekts.

Attribute eines Elements mit diesem Datentyp sind:

**@root .....Identifikationssystem (OID)**

In Deutschland ist dies ein Pflichtattribut. Es enthält einen eindeutigen Identifikator (in Deutschland: eine OID) für das Identifikationssystem, in dem die Extension generiert (und eindeutig) ist.

Ein Identifikationssystem wird verwendet, um Personen, Systeme, Institutionen und andere materielle Sachen identifizieren zu können. Einige (*deutsche*) Beispiele sind: 'Personalausweisnummer', unveränderlicher Teil des Personenkennzahl auf der Versicherungskarte, Krankenhausnummer des St. Josef Krankenhauses, die eindeutige Identifikationsnummer eines Anbieters von Software im Gesundheitswesen, das Institutskennzeichen (IK-Nummer).

Ein ISO Object Identifier (OID) ist ein weltweit eindeutiger String, der aus Zahlen und Punkten besteht (beispielsweise "2.16.840.1.113883.3.1"). Laut ISO Definition bestehen OIDs aus Pfaden mit einer Baumstruktur, wobei die äußerst links situierte Zahl als *root* und die äußerst rechts situierte Zahl als *leaf* (ein Blatt als Endpunkt) bezeichnet werden. Die Nummer ist garantiert weltweit eindeutig, weil das Ausgabesystem auf dem System der delegierten Verantwortlichkeit basiert. Jeder Zweig unter einem *root* in der Baumstruktur korrespondiert mit einer Domäne, in der eine Organisation die Abgabe von OIDs verwaltet. Die zentralen Vergabestelle für OIDs im Gesundheitswesen in Deutschland publiziert eine Tabelle mit OIDs. Informieren Sie sich bei Zweifel bei der zentralen Vergabestelle, welche (eventuell neu zu registrierende) OID benutzt werden muss.

Im *OID Konzept für das Deutsche Gesundheitswesen [oidk]* finden Sie nähere Informationen über Codiersysteme und OIDs.

**@extension .....Identifikation (string)**

Verpflichtend. Eine eindeutige Zeichenkette im Kontext des Identifikationssystems, das definiert wird in der *root*.

Ein Attribut des Datentyps II muss in der deutschen Situation aus einer Kombination von *root* und *extension* bestehen (z.B. *root* = "1.2.528.4.5" mit *extension* "22"). Diese Kombination ist verpflichtend für die Identifikation von allen Objekten.

Die Länge des *extension* String und die Benutzung von eventuellen Vorlaufnullen in der Extension, sowie deren Anzahl wird vom Verwalter des Identifikationssystem festgelegt.

**@assigningAuthorityName .....Name der ausgebenden Organisation (String)**

Optionales Attribut. Eine Textform der so genannten 'assigning authority', der Organisation, die die Identifikation festgelegt hat (und meistens das entsprechende Identifikationssystem verwaltet). Dem Wert des Attributs darf keine Bedeutung zugemessen werden, außer dass es einem Benutzer vorgelegt wird, um den Hintergrund der Identifikation zu verdeutlichen. Für die Lesbarkeit der Nachrichten wird empfohlen, den *assigning-AuthorityName* mitzusenden.

***XML Beispiele***

```
<id extension="13234453645" root="2.16.840.1.113883.2.4.15.3.427.1"/>
<id root="2.16.840.1.113883.2.4.15.3.427.1. 13234453645"/>
<id extension="1234567890" root="2.16.840.1.113883.2.4.6.3"
    assigningAuthorityName="Innenministerium"/>
<id extension="JANS2" root="2.16.840.1.113883.2.4.7.33"
    assigningAuthorityName="Alfa Krankenhaus"/>
```

### 3.18 URL (Universal Resource Locator)

Dies ist eine Telekommunikationsadresse, die nach dem Internet Standard "RFC 1738 [<http://www.ietf.org/rfc/rfc1738.txt>]" spezifiziert ist. Die URL gibt ein Protokoll und einen Kontaktpunkt an, die für dieses Protokoll definiert sind. Bekannte Beispiele sind Telefon- und Faxnummern, E-Mail-Adressen, Hyperlink Referenzen, Datenübertragungsprotokolle (FTP) Referenzen usw.

URLs haben eine Standard-Repräsentationsform wie Strings im Format *scheme:address*; Die bekanntesten Schemen sind in der folgenden Tabelle aufgenommen.

Im Bereich *address* einer URL ist ein String, dessen Format durch die URL *scheme* bestimmt wird.

| Tabelle Domäne URLScheme: |                                  |  |
|---------------------------|----------------------------------|--|
| <i>code</i>               | <i>Name</i>                      | <i>Definition</i>  |
| tel                       | Telefon                          | Telefonnummer [draft-antti-telephony-url-11.txt].  |
| fax                       | Fax                              | Eine Nummer für ein Faxgerät [draft-antti-telephony-url-11.txt].   |
| mailto                    | Mailto                           | Elektronische Mailadresse [RFC 2368].  |
| http                      | HTTP                             | Hypertext Transfer Protocol [RFC 2068].  |
| ftp                       | FTP                              | File Transfer Protocol (FTP) [RFC 1738].   |
| mllp                      | HL7 Minimal Lower Layer Protocol | Das traditionelle HL7 Minimal Lower Layer Protokoll. Die URL hat die Form einer IP URL, beispielsweise mllp://<host>:<port>/ mit <host> als IP Adresse oder DNS Hostname und <port> als port Nummer, wo der MLLP Dienst erreichbar ist.          |
| file                      | File                             | Computerspezifische, lokale Dateinamen [RFC 1738]. Diese Schemen funktionieren nur für lokale Dateien. Wird kaum verwendet, weil der Empfänger bei einem Sender / Empfänger Szenario mit einem lokalen Dateinamen meistens nichts anfangen kann. |
| nfs                       | NFS                              | Network File System Protokoll [RFC 2224]. Wird für NFS Server zur gemeinsamen Benutzung von Dateien verwendet.   |
| telnet                    | Telnet                           | Referiert an eine interaktive Session [RFC 1738].  |
| modem                     | Modem                            | Eine Telefonnummer mit einem Modem [draft-antti-telephony-url-11.txt].   |
| x-hl7-applicatie          | Programm-Identifikation          | OID die ein HL7 Nachrichten sendendes Programm eindeutig identifiziert.  |

### 3.19 TEL (Telekommunikationskontakt – Telecommunication Address)

Es gibt keinen separaten Datentyp für Telefonnummern. Dies sind lediglich die URLs, die sich auf Telekommunikationsanlagen beziehen.

Details über die Definition von Telefonnummern sind definiert in den Internet "RFC 2806 [<http://www.ietf.org/rfc/rfc2806.txt>] URLs for Telephone Calls". Beispielsweise ist "tel:+49(221)6754-63" eine Telefonnummer und "fax:+49(221)6571412-3" eine Faxnummer. Es wird bevorzugt, die globalen, absoluten Telefonnummern mit einem "+" und der Länderkennung anzugeben. Trennungszeichen können hinzugefügt werden (zur besseren Lesbarkeit) haben aber keine Bedeutung für die Nummern. "tel:+49221675463" und "fax:+4922165714123" sind also identisch mit den vorstehenden Beispielen.

Die Nummern müssen im Falle von internationalen Telefonnummern mit einem „+“ beginnen. Die Angaben dürfen nur Ziffernzeichen 0 bis 9 nutzen sowie als visuelle Separatorenzeichen nur Bindestrich –, Punkte . oder Klammern ( ) verwenden.

Elemente mit diesem Datentype haben ein Attribut,

*value*..... Wert (TEL)

das verschiedene Zeichenkombinationen enthalten kann, die Telekommunikationskontakte wie Telefon (tel:), Fax (fax:), E-Mail (mailto:) usw. wiedergeben.

*use*.....Benutzungshinweis (cs)

Mit diesem *use* Attribut können einer oder mehrere Codes angegeben werden, um für ein System oder einen Benutzer den besten Telekommunikationskontakt für den jeweiligen Zweck anzudeuten.

| Tabelle: Domäne TelecommunicationAddressUse Attribut Werte |   |   |
|--|---|---|
| Code   | Name  | Definition  |
| HP   | primary tel (Wohn- / Aufenthaltsadresse)          | Der primäre Telekommunikationskontakt um eine Person zu erreichen; kann maximal ein Mal vorkommen.  |
| HV   | vacation tel (Urlaubs- Telekommunikationskontakt) | Ein Ferienhaus, wo eine Person im Urlaub zu erreichen ist.  |
| WP   | work place (Arbeit)                               | Ein Telekommunikationskontakt am Arbeitsplatz. Erste Wahl für arbeitsbezogene Kontakte. Ist für Organisationen und Leistungsanbieter der primäre Telekommunikationskontakt. |
| AS   | Beantwortungsservice                              | Eine Person oder ein Service zum Hinterlassen von Nachrichten.  |
| EC   | Notfallkontakt                                    | Ein Telekommunikationskontakt für Notfälle.   |
| PG   | Pager   | Ein Pager (Funkmeldeempfänger) mit dem man um einen Rückruf fragen oder eine kurze Nachricht hinterlassen kann.   |
| MC   | Mobilkontakt                                      | Ein Handy oder ein Gerät, das der Besitzer immer bei sich trägt, darf andere use Codes enthalten.   |

*@useablePeriod* ..... Gültigkeitsdauer (TS)

Es ist möglich, die Gültigkeitsdauer eines Telekommunikationskontakts zu begrenzen. Das Element *usablePeriod* kann zur Angabe eines Zeitintervalls benutzt werden.

#### XML Beispiel

```
<telecom value="tel:+31.10.4765342"/>
<telecom value="mailto:dialyse@centrum-rotterdam.nl"/>
```

### 3.20 AD (Adresse – Postal Address)

Elemente dieses Datentyps haben eine Substruktur mit verschiedenen Elementen, die in einer Adresse vorkommen können.

Eine Adresse wird in der HL7 Version 3 in einer Serie von *Address Name Parts* wiedergegeben, beispielsweise Straßename, Stadt usw.<sup>2</sup> Der Datentyp für alle *Address Name Parts* ist SC. Momentan ist Country das einzige Element, bei dem wir Codes zulassen.

| <b>Tabelle: Domäne AddressNamePartType Elementnamen</b> |  |
|---|--|
| <i>Element Name</i>                                     | <i>Definition</i>  |
| delimiter   | Trennzeichen [delimiters] werden ohne Leerstellen ausgedruckt [framing]. Wenn keine Wertkomponente geliefert wird, erscheint das Trennzeichen als Zeilenumbruch [line break].  |
| country   | Das Land, z.B. "Deutschland". Wenn das Land codiert übermittelt wird, geschieht dies konform mit der ISO 3166 Länderkennung mit zwei Buchstaben. Die OID ist 1.3.6.1.4.1.1466.115.121.1.11.  |
| county  | In Deutschland wird dieses Element benutzt, um die Bundesländer anzugeben (in anderen Ländern kann es sich um eine andere administrative Einheit von Staat/Provinz handeln).   |
| city  | Der Name einer Stadt, eines Dorfes, eines anderen Wohngebietes oder Zustellzentrums. Bitte beachten: dies ist der Wohnort und nicht die eventuelle Gemeinde, zu der dieser Wohnort gehört. Beispiel: <i>Fleestedt</i> , Gemeinde Hiltfeld.   |
| postalCode  | Eine Postleitzahl, für deutsche Postleitzahlen, ohne Leerzeichen oder vorangestellten Ländercode, #####.   |
| houseNumber   | Die Nummer eines Gebäudes, Hauses oder Grundstücks an der Straße. Wird auch manchmal "primary street number" genannt. Hierbei wird nicht die Straße nummeriert, sondern das einzelne Haus, und zwar mit der vollständigen Hausnummer, als Adresse für die Postzustellung durch den externen Postboten. Eine alphanumerische Zufügung wie "14a" wird auch bei houseNumber mitgesendet.  |
| streetName  | Straßename   |
| streetAddressLine                                       | Eine Kombination von Straße und Hausnummer. Dieses Element sollte nur benutzt werden, wenn der Sender Hausnummer und Straße nicht getrennt speichert oder nicht herleiten kann. Der getrennten Angabe von Straße und Hausnummer ist der Vorzug zu geben.   |
| additionalLocator                                       | Zusätzliche Standortandeutung als Ergänzung zur Postadresse. Dabei kann es sich um die Bezeichnung einer Wohneinheit (Unit) handeln, wie die Nummer eines Appartements, einer Suite oder einer Etage. Es können mehrere Unit-Bezeichnungen in einer Adresse vorkommen (z.B. "3e Etage, Appartement 342"). Außer einem kleineren Unit innerhalb eines größeren Units kann auch ein abweichender Standort spezifiziert werden, wie beispielsweise "g.ü.", womit ein gegenüberliegender Standort von Wohnbooten an der Straße angegeben wird. |
| postBox   | Eine Postfach-Angabe. Darf nicht in Kombination mit dem PostalAddressUse Code PHYS benutzt werden, da es sich bei einem Postfach nicht um eine Besuchsadresse handelt.   |

Die Codes für Postadressen werden definiert von der HL7 Domäne PostalAddressUse, angegeben im "use" Attribut des *addr* Mutterelements (siehe Beispiele).#

<sup>2</sup> Im HL7 Standard ist der Datentyp AD auch als so genannter "mixed content" zugelassen, was bedeutet, dass einige Teile der Daten von einem *partType* Element umschlossen werden und andere Teile nicht (Mischung aus Text und XML Elementen). Dies ist in Deutschland nicht zugelassen.

| <b>Tabelle: Domäne PostalAddressUse Attribut-Werte</b> |  |  |
|--|--|--|
| <i>Code</i>  | <i>Name</i>                              | <i>Definition</i>  |
| PHYS   | visit address (Wohn- / Aufenthaltsort)   | Eine physische Adresse; Wird an erster Stelle benutzt, um den Adressaten zu besuchen. Kann in den Niederlanden benutzt werden, um eine Adresse durchzugeben, die von der offiziellen Adresse abweicht. |
| PST  | postal address (Postanschrift, Postfach) | Adresse für die Postzustellung   |
| HP   | primary home (offizielle Adresse)        | Die Adresse, die in den offiziellen Registern, z.B. im deutschen Einwohnermeldeamt festgelegt ist; kann maximal ein Mal vorkommen; ist für Personen die primäre Adresse.                               |
| HV   | vacation home (Ferienhaus)               | Ein Ferienhaus, wo man eine Person im Urlaub erreichen kann.   |
| WP   | work place (Arbeit)                      | Eine Adresse am Arbeitsplatz. Erste Wahl für arbeitsbezogene Kontakte. Ist für Organisationen und Pflegeanbieter die primäre Adresse.  |

Für Adressdaten von Patienten sind die Attributwerte HP, WP, PST, PHYS zugelassen, für Organisationen WP, PHYS, PST und für Ärzte WP.

### **XML Beispiele**

```
<streetName>An der Garnbleiche</streetName>
```

```
<addr>
  <postalCode>52349</postalCode>
</addr>
```

```
<addr use="WP">
  <streetName>An der Garnbleiche</streetName>
  <houseNumber>16</houseNumber>
  <postalCode>52349</postalCode>
  <city>Düren</city>
</addr>
```

```
<addr use="HP">
  <streetAddressLine>Burgweg 42</streetAddressLine>
  <postalCode>51371</postalCode>
  <city>Leverkusen</city>
</addr>
```

```
<addr>
  <country code="DE" codeSystem="2.16.1">Deutschland</country>
</addr>
```

### 3.21 PN (Personenname – Person Name)

Ein Name einer Person.

Attribut:

*@use.....Benutzertyp(en) (SET<CS>)*

Elemente:

*validTime..... Gültigkeitsperiode (IVL<TS>)*

*delimiter..... Trennzeichen (PNXP)*

*family..... Nachname (PNXP)*

*given..... Vorname (PNXP)*

*prefix..... Präfix (PNXP)*

*suffix..... Suffix (PNXP)*

Struktur: Der Daten-Typ PN ist eine Extension des Daten-Typs EN (*Entity Name*) und besitzt folglich einen sogenannten 'mixed content', wobei im Prinzip Freitext mit *name parts* kombiniert werden kann. Für Deutschland gilt, dass die Verwendung von 'mixed content' bei Personennamen begrenzt ist. Zugelassen sind:

- Der vollständige Personenname als Freitext (es gibt also keine *person name parts*), wenn es dem Sender nicht möglich ist, Teile des Namen zu benennen.
- Alle Teileinheiten sind als *person name part* definiert. In einem solchen Fall darf also kein Text vorkommen, der nicht von einem der nachstehend beschriebenen Tags begleitet wird.

**Die Sequenz der *person name parts* ist relevant!** Als Richtlinie gilt, dass diese in der 'natürlichen' Sequenz der Benutzung des Namens angegeben werden. Die angegebene Sequenz ist besonders in den folgenden Fällen wichtig:

- Präfixe (prefix) müssen immer vor dem Namen stehen, zu dem sie gehören.
- Suffixe (suffix) müssen immer hinter dem Namen stehen, zu dem sie gehören.
- Vornamen (given) müssen immer in der offiziellen (gesetzlichen) Sequenz stehen.
- Nachnamen (family) und ein eventuelles Trennzeichen (meistens '-') müssen in der offiziellen Sequenz stehen, abhängig von der Wahl bei der Eheschließung.

Nähere Erläuterungen finden Sie bei den verschiedenen *person name parts*.

#### XML Beispiel

Personennamen als Freitext

```
<name>
  Jan Meier
</name>
```

Der Name wurde ohne interne Struktur übermittelt.



Personenname mit name parts:

```
<name>
  <given>Jan</given>
  <family>Meier</family>
</name>
```

Die beiden Bestandteile des Namens werden benannt und erhalten einen qualifier: "Jan" ist ein 'Name, der bei der Geburt gegeben wurde', bzw. ein Vorname (voll ausgeschrieben). "Meier" ist ein 'Familiennamen, der bei der Geburt empfangen wurde', bzw. der eigene Nachname.

Ungültiger Personenname:

```
<name>
  Jan <family>Meier</family>
</name>
```

Dies ist kein gültiger Personenname, da Freitext mit einem name part kombiniert wurde.

*@use.....Namenverwendungstyp(en)*

Im Prinzip kann von jedem *Person Name* angegeben werden, in welcher Situation dieser verwendet werden kann. Für Deutschland wurde beschlossen, dass die folgenden Verwendungstypen für Namen zugelassen sind:

| Tabelle: Domäne EntityNameUse Attribut Werte |                               |  |
|--|-------------------------------|--|
| Code   | Name                          | Definition   |
| L  | Regulärer Name                | Der Name, den die Person (Entität) führt. Die Abkürzung 'L' stand ursprünglich für Legal (gesetzlich), Tatsache ist aber, dass in dem Namen auch Komponenten vorkommen dürfen (z.B. ein Rufname), die nicht gesetzlich festgelegt sind. Dieser Namenverwendungstyp ist der <b>Default</b> , wenn kein Typ durchgegeben wird. |
| A  | Pseudonym                     | Ein Künstlernamen, 'Deckname' oder zeitlicher Name für eine Person (Entität). Dieser weicht also von dem regulär geführten Namen ab und wird z.B. benutzt, um die Identität einer Person zu tarnen (Privacy) oder als temporärer Name, wenn der echte unbekannt ist ('John Doe').  |
| OR   | Gesetzlich registrierter Name | Der Name mit den exakten Komponenten, der im Einwohnermeldeamt des betreffenden Landes registriert ist.  |



Beachten Sie, dass ein Name auch zwei *use* Attribute haben darf. Das kann z.B. vorkommen, wenn der gesetzlich registrierte Name ('OR') auch als regulärer Name geführt wird (siehe nachstehendes Beispiel). Achten Sie aber darauf, dass dann keine einzige Komponente vorkommen darf (wie Rufname oder ein Partnername) die nicht Teil des gesetzlich registrierten Namens ist.



Der Name use code 'OR' ist noch nicht im offiziellen HL7 Standard aufgenommen. Er wurde aber, vorauslaufend auf die internationale Harmonisierung, bereits hinzugefügt.

### XML voorbeelden

Der reguläre Name als default, also kein *use* Attribut

```
<name>
    <!-- Name, unterverteilt in Komponenten -->
</name>
```

Ein Pseudonym eines Patienten

```
<name use="A">
    <!-- Name, unterverteilt in Komponenten -->
</name>
```

Der gesetzlich registrierte Name

```
<name use="OR">
    <!-- Name, unterverteilt in Komponenten -->
</name>
```

Der geführte Name stimmt exakt mit dem gesetzlich registrierten Namen überein

```
<name use="OR L">
    <!-- Name, unterverteilt in Komponenten -->
</name>
```

**<validTime>** ..... Gültigkeitszeitraum

Dies ist ein **optionales** XML Element innerhalb *Person Name*, welches die Periode angibt, in der dieser Name für die betreffende Person 'in Gebrauch'/gültig war. Die Optionen sind:

- Es gibt kein *validTime* Element: Der betreffende Name ist im Prinzip unbegrenzt gültig.
- Es gibt eine Unter- und Obergrenze: Der Name war in der angegebenen Periode gültig.
- Es gibt nur eine Untergrenze: Der Name ist seit dem angegebenen Datum gültig.
- Es gibt nur eine Obergrenze: Der Name war bis einschl. angegebenem Datum gültig.

Dieses Element von *Person Name* kann verwendet werden, um anzugeben, dass im Leben einer Person einmal oder mehrmals eine Namensänderung stattgefunden hat. Dies geschieht u.a bei:

- Adoption eines Babys, das den Nachnamen der Adoptiveltern erhält.
- Eheschließung, wobei der Name des Partners an den eigenen Namen angefügt wird.
- Ehescheidung, wobei ein vorher angenommener Name wieder abgelegt wird.
- Personen, die aus anderen Gründen ihren Vor- oder Nachnamen ändern.



In allen Situationen, in denen ein oder mehrere *Person Names* durchgegeben werden, muss minimal der Name angegeben werden, der zum Zeitpunkt des Versendens gültig/aktuell ist. Nicht mehr gültige Namen können also nur angegeben werden, wenn das betreffende Nachrichtenelement wiederholt benutzt wird (also mit einer Kardinalität > 1).

Zu beachten ist, dass viele Patientenerfassungssysteme keine wirkliche Historie (mit Anfangsdatum) der Patientennamen führen. Wohl wird häufig ein allgemeines 'audit trail' (Änderungshistorie) der Patientendaten geführt. Im Bedarfsfall könnte daraus die Historie des Personennamens abgeleitet werden, obwohl es natürlich auch möglich ist, nur den aktuellen Namen durchzugeben (also kein *validTime* zu verwenden).



Im Gegensatz zu der Situation bei *Organization Name* ist es nicht zugelassen, dass die Unter- oder Obergrenze einer *validTime* Angabe bei *Person Name* in die Zukunft liegt. Es kann also kein 'geplanter' neuer Name oder ein 'geplantes Verfallsdatum' des heutigen Namens für Personennamen durchgegeben werden.

Der aktuelle Name ist gültig seit dem 12. Juli 2005

```
<name>
  <validTime>
    <low value="20050712"/>
  </validTime>
  <!-- Name, unterverteilt in Komponenten -->
</name>
```

Obenstehende Situation kann z.B. bei einem System vorkommen, das nur den aktuellen Namen übermittelt, aber auch die Historie führt. Die oben genannte Person kann z.B. am 12. Juli geheiratet haben und dabei den Namen des Partners angenommen haben.

Alte Namen plus aktueller Name

```
<name>
  <validTime>
    <high value="19850412"/>
  </validTime>
  <!-- "Nicole de Vries" als Name des Babys vor der Adoption -->
</name>
```

```

<name>
  <validTime>
    <low value="19850413"/>
    <high value="20050824"/>
  </validTime>

  <!-- "Nicolette Scheick" als Name nach der Adoption, aber vor Eheschließung -->
</name>
<name>
  <validTime>
    <low value="20050825"/>
  </validTime>
  <!-- "Nicolette Scheick-Jansen" als Name nach der Eheschließung -->
</name>

```

In vorstehendem Beispiel wird das Baby Nicole de Vries von der Familie Scheick adoptiert, wobei sich also ihr Nachname ändert. Weil den Adoptiveltern dieser Name besser gefällt, wird auch ihr Vorname (oder auf jeden Fall ihr Rufname) geändert in Nicolette. Nach ihrer Eheschließung nimmt sie den Nachnamen ihres Partners (Jansen) an. Das sendende System sendet in diesem Fall die gesamte Namenshistorie mit.

*<delimiter>..... Trennzeichen*

Ein *delimiter* hat keine spezielle Bedeutung als Bestandteil eines *Person Name*, im Gegensatz zur Übermittlung eines (Stückchens) wörtlichem Text, der in dem geschriebenen Namen vorkommt.

Ein *delimiter* muss immer an der Stelle in *Person Name* stehen, an der man auch den Text schreiben würde. Es gibt keine impliziten Leerstellen. Wenn man also normalerweise eine Leerstelle davor oder dahinter schreibt, muss diese explizit angegeben werden.

Beispiele von *delimiters* in *Person Names* sind:

- Der Bindestrich '-' zwischen dem eigenen Nachnamen und dem Partnernamen (oder umgekehrt).
- Das Komma plus Leerstelle ', ' zwischen dem Namen und bestimmten Nachsilben.
- Der Text ', geb. ' oder ', E.v. ' (Ehefrau von), der manchmal benutzt wird bei dem eigenen, bzw. Partnernamen.

Diese könnten folgendermaßen in einem *Person Name* XML Nachrichtenelement benutzt werden:

Trennung zwischen Partnername und Geburtsname

```

<name>
  <family qualifier="SP">Jansen</family>
  <delimiter>-</delimiter>
  <family qualifier="BR">Scheick</family>
</name>

```

Trennung zwischen Nachnamen und akademischem Titel

```
<name>
  <family>Jansen</family>
  <delimiter>,</delimiter>
  <title qualifier="AC">MSc</family>
</name>
```

Weitere allgemeine Beispiele finden Sie am Ende dieses Abschnitts.

Einige auf der Hand liegende Fragen (und Antworten) über *delimiters*:

**Frage:** Muss ein empfangendes System einen *delimiter* speichern?

**Antwort:** Ein empfangendes System, dass die einzelnen *person name parts* verarbeitet, wird *delimiters* praktisch nie in der eigenen Datenbank speichern.

**Frage:** Warum sollte ein sendendes System dann überhaupt *delimiters* mit-senden?

**Antwort:** Es kann vorkommen, dass empfangende Systeme keine oder nicht alle *person name parts* separat verarbeiten können (z.B. weil sie nur *ein* Feld für den Nachnamen haben). Sie können über *delimiters* einen voll ausgeschriebenen Namen übernehmen. (Ein Beispiel eines solch einfachen Empfängers ist ein Webinterface, das eingehende HL7 v3 Nachrichten in ein HTML Format umsetzt, zur direkten Wiedergabe auf dem Bildschirm).

*<family>* ..... *Nachname*

Attribut:

*@qualifier* ..... *Namensart (CS)*

Ein *person name part* des Typen *family* bezieht sich auf einen Teil des Namens, den eine Person durch Familienbände bekommen hat und der meistens als **Nachname** bezeichnet wird. Normalerweise bezieht sich dies also auf den eigenen Nachnamen (erhalten von den Eltern) und eventuell auf einen nach einer Heirat angenommenen Nachnamen. ('übernommen' vom Partner).

Einige Regeln in Bezug auf *person name parts* des Typen *family*:

- Diese müssen untereinander immer konform mit der offiziellen Schreibweise angeordnet sein (z.B. zuerst der eigene Nachname und dann der Nachname des Partners oder genau umgekehrt).
- Es gibt immer eine implizite Leerstelle als Zwischenraum mit dem darauf folgenden *name part*, außer bei einem *delimiter* oder einem *suffix* (siehe dort).
- Die Art des Nachnamens kann weiterhin durch die Verwendung des optionalen Attributs *qualifier* angegeben werden. Siehe Tabelle für die dabei zugelassenen Werte.
- Es darf nur *ein* Familienname pro Qualifier-Typ im Namen vorkommen!

| <i>qualifiers bei person name part family</i> |   |
|---|---|
| <i>Qualifier</i>                              | <i>Anwendung</i>  |
| BR  | <b>(birth) Geburtsname.</b> Ein Nachname, den man bei der Geburt erhalten hat. Normalerweise ist dies der Geburtsname eines (natürlichen) Elternteils, aber in einigen Kulturen kann es sich dabei auch um eine Kombination der Geburtsnamen beider Eltern handeln, oder um eine Ableitung des Vornamens eines Elternteils.   |
| CL  | <b>Rufname.</b> Ein Nachname, mit dem eine Person informell angesprochen wird und der meistens von einem der offiziellen Nachnamen abgeleitet ist.  |
| SP  | <b>(spouse) Partnername.</b> Ein Nachname, der von einem Partner bei einer Ehe 'übernommen' wurde (meistens der Geburtsname eines Partners). Dies sagt nichts aus über das Geschlecht des Namensträgers, da die Annahme von Namen zwischen beiden Partnern in den Deutschland gleichgestellt ist. Auch wenn ein Partnername fehlt, kann daraus keine Schlussfolgerung über den Ehestand gezogen werden, da jemand verheiratet sein kann, ohne den Namen des Partners zu führen. Es ist zu beachten, dass es bei der Verwendung eines Partnernamens nicht nur relevant ist, dass jemand verheiratet ist, sondern vor allem, ob die betreffende Person mit dem Namen des Partners angesprochen werden möchte oder nicht. Die Anwendung des Partnernamens im Personennamen ist unabhängig von einer eventuellen Benennung des Partners als Kontaktperson. In der heutigen deutschen Namensrecht ist nach einer Eheschließung jede Kombination von Eigennamen und/oder Partnername von beiden Partnern möglich. Natürlich ist das Geschlecht der beiden Partner dabei nicht relevant. |



Wenn ein *person name part* vom Typ *family* **ohne einen qualifier** benutzt wird, wird dies einfach als **Nachname** interpretiert. Wenn der Empfänger zwischen einer Speicherung als Geburtsname oder als Partnername wählen kann, muss in einem solchen Fall der **Geburtsname** gewählt werden.



Es muss noch entschieden werden, was mit einem Geburtsnamen geschehen soll, den jemand nach einer Eheschließung nicht mehr führt (weil ausschließlich der Partnername geführt wird). Der Geburtsname muss in einem solchen Fall trotzdem gespeichert (und übermittelt) werden, aber es könnte dabei angegeben werden, dass der Name **'unsichtbar'** sein muss. Eine Lösung könnte die Implementierung eines Attributs *invisible* sein. Es ist auf jeden Fall möglich, den Namen zweimal zu übermitteln: einmal mit dem use Attribut 'OR' (mit dem Geburtsnamen) und einmal mit einem 'L' (ohne).



Es muss im internationalen Rahmen noch untersucht werden, wie mit dem Nachnamen zu verfahren ist, der von den **Adoptiveltern** übernommen wurde. Nach der heutigen Definition ist der *qualifier* "BR" hierfür nicht geeignet, aber es ist klar, dass die *qualifiers* "SP" und "CL" ebenfalls nicht ausreichend sind. Vorläufig lautet die Empfehlung in solchen Fällen (falls überhaupt bekannt ist, dass es sich um einen Adoptivnamen handelt) keinen *qualifier* mitzusenden. In der Praxis wird ein derartiger Name allerdings meistens als 'eigener Nachname' (und daher mit *qualifier* "BR") benutzt werden.

**XML Beispiele**

Jan Meier

```

<name>
  <given>Jan</given>
  <family qualifier="BR">Meier</family>
</name>

```

Nicolette Jansen-Scheick

```

<name>
  <given>Nicolette</given>
  <family qualifier="SP">Jansen</family>
  <delimiter>-</delimiter>
  <family qualifier="BR">Scheick</family>
</name>

```

*<given>* ..... Vorname

Attribut:

*@qualifier* ..... Namensart (CS)

Ein *person name part* des Typen *given* bezieht sich auf den Teil des Namens, den eine Person meistens von den Eltern und meistens bei der Geburt erhält. In Deutschland wird dies meistens als **Vorname** bezeichnet, obwohl dieser nicht in allen Kulturen *vor* dem Familiennamen steht. Auch Namensbestandteile, die von dem Vornamen abgeleitet sind, z.B. die Initialen (Anfangsbuchstaben) und der Rufname (informeller Vorname) haben den Typ *given*.

Einige Regeln für *person name parts* des Typen *given*:

- Eine Person kann ohne weiteres mehrere Vornamen oder Initialen haben.
- Offizielle Vornamen und Initialen müssen immer in der richtigen Sequenz stehen.
- Es muss immer eine implizite Leerstelle als Zwischenraum zu dem darauf folgenden *name part* stehen, außer wenn es sich um ein *delimiter* oder ein *suffix* handelt (siehe dort).
- Die Datenart des gegebenen Namens kann zudem angedeutet werden, indem das optionale Attribut *qualifier* benutzt wird. Siehe Tabelle für die dabei zugelassenen Werte.

| <i>qualifiers</i> bei person name part <i>given</i> |   |
|---|---|
| <i>Qualifier</i>                                    | <i>Anwendung</i>  |
| BR  | <b>Offizieller Vorname.</b> Ein Vorname, der meistens bei der Geburt gegeben ist (meistens von den Eltern) und der offiziell registriert ist. Vornamen müssen voll ausgeschrieben werden und in der richtigen Sequenz stehen. Wenn eine Person mehrere Vornamen hat, muss auf jeden Fall der erste Name übermittelt werden (Die ausschließliche Benutzung des ersten Vornamens ist also zugelassen, die ausschließliche Benutzung des zweiten Vornamens dagegen nicht). Wenn es mehrere Vornamen gibt, können diese sowohl als separate <i>given</i> Elemente, als auch in <i>einem</i> Element (getrennt durch Leerstellen) übermittelt werden. Es kommt auch vor, dass der erste Vorname voll ausgeschrieben in Kombination mit den Initialen benutzt wird. |

|    |   |
|----|---|
| CL | <b>Rufname.</b> Ein Vorname, mit dem eine Person informell angesprochen wird und der meistens von einem der offiziellen Vornamen abgeleitet ist. Im Gegensatz zu den offiziellen Vornamen kann ein Rufname im Leben einer Person ohne weiteres variieren. Eine Person kann sogar mehrere Rufnamen gleichzeitig haben, je nachdem wie Menschen die Person kennen. In diesem Fall muss der am meisten zutreffende Rufname gesendet werden.  |
| IN | <b>Initialen.</b> Meistens eine Abkürzung des Vornamens. Dabei kann es sich um einen einzigen oder um mehrere Buchstaben handeln (z.B. 'Th.' für Thomas). Ein abschließender Punkt muss explizit angegeben werden. Initialen müssen in der korrekten Sequenz angegeben werden. Wenn es mehrere Initialen gibt, können diese sowohl als separate <i>given</i> Elemente, als auch in <i>einem</i> Element (getrennt durch Punkte) übermittelt werden. Der Vorteil dieser Methode ist, dass keine Leerstellen zwischen den einzelnen Initialen im Namen impliziert werden. |



Wenn ein *person name part* des Typen *given* **ohne einen *qualifier*** verwendet wird, wird dieses einfach als **Vorname** interpretiert. Wenn der Empfänger wählen muss, ob dies als offizieller Vorname oder als Rufname gespeichert werden soll, muss in einem solchen Fall der **offizielle Vorname** gewählt werden.



Die *qualifiers* "BR" und "CL" können auch gemeinsam vorkommen, um anzugeben, dass ein offizieller Vorname auch als Rufname fungiert. So kann vermieden werden, dass der gleiche Name zwei Mal mitgesendet werden muss. Wenn der Rufname allerdings von den offiziellen Vornamen abweicht, können beide übermittelt werden: zuerst die offiziellen Vornamen und danach der Rufname.



Wenn eine Kombination von offiziellem Vornamen und Initialen mitgesendet wird, dürfen diese sich in Deutschland nicht 'überlappen' dürfen, d.h. dass der erste Anfangsbuchstabe (falls erforderlich) von dem empfangenden System von dem/den ersten Buchstaben des offiziellen Vornamens abgeleitet werden muss (siehe nachstehendes Beispiel).



In Deutschland muss noch untersucht werden, wie man mit dem Vornamen, den die **Adoptiveltern** gegeben haben, zu verfahren hat. Nach der heutigen Definition ist der *qualifier* "BR" hierfür nicht geeignet, aber es ist nicht klar, ob der *qualifier* "CL" (Rufname) wohl ausreichend ist. Die Frage ist nämlich, ob ein derartiger Name einen offiziellen Charakter bekommt oder nur als Rufname fungiert.

### XML Beispiele

Hans Jansen, ohne *qualifier*

```
<name>
  <given>Hans</given>
  <family>Jansen</family>
</name>
```

Johannes Theodorus Cornelis Jansen, offizielle Vornamen



```
<name>
  <given qualifier="BR">Johannes Theodorus Cornelis</given>
  <family>Jansen</family>
</name>
```

Johannes Theodorus Cornelis Jansen, mit Rufname Hans

```
<name>
  <given qualifier="BR">Johannes Theodorus Cornelis</given>
  <given qualifier="CL">Hans</given>
  <family>Jansen</family>
</name>
```

Johannes Th. C. Jansen, ohne Duplizierung des Anfangsbuchstabens 'J'

```
<name>
  <given qualifier="BR">Johannes</given>
  <given qualifier="IN">Th.C.</given>
  <family>Jansen</family>
</name>
```

Kai Uwe Heitmann, Kai ist sowohl offizieller Name als auch Rufname

```
<name>
  <given qualifier="BR CL">Kai</given>
  <given qualifier="BR">Uwe</given>
  <family>Heitmann</family>
</name>
```

Das Fazit dieser Beispiele ist, dass diverse Kombinationen von offiziellen Vornamen, Rufnamen und Initialen möglich sind, abhängig von den Speicher- und Kommunikationskapazitäten des sendenden Systems. Die Bedeutung der einzelnen Bestandteile muss allerdings deutlich sein.

*<prefix> ..... Präfix*

Attribute:

*@code ..... Titelcode (CE), nicht bei qualifier "VV"*

*@qualifier ..... Namensart (CS)*

Ein *person name part* des Typen *prefix* bezieht sich auf einen Teil des Namens, der **zu einem oder mehreren anderen Namensbestandteilen gehört und vorne angehängt wird**. Im Prinzip gibt es zwei Arten von Vorsilben, nämlich **Vorsilben zu Nachnamen und zu Titeln/akademischen Graden** (die als Zufügung zum geschriebenen Namen aufgenommen werden).

Einige Regeln für *person name parts* des Typen *prefix*:

- Ein *prefix* muss immer direkt vor die Namensbestandteile platziert werden, auf die es sich bezieht (d.h. wo es normalerweise geschrieben wird).
- Es gibt **keine implizite Leerstelle** als Zwischenraum zu dem darauf folgenden *name part*, d.h. eine Leerstelle nach der Vorsilbe muss explizit im Text angegeben werden!

Die Art der Vorsilbe kann zudem angedeutet werden, indem man das optionale Attribut *qualifier* benutzt. Siehe Tabelle für die dabei zugelassenen Werte.



Im nächsten Release der Datentypen von HL7 v3 wird es möglich sein, diverse Elemente die jetzt ein 'string' sind, auch optional als Code anzugeben.

Dies kann u.a. zur kodierten Versendung von Titeln/akademischen Graden benutzt werden, wenn das sendende System diese als Code speichert. Darüber hinaus muss der Text aber immer gesendet werden, damit der Empfänger entscheiden kann, was benutzt wird.

| <i>qualifiers bei person name part prefix</i> |  |
|---|--|
| <i>Qualifier</i>                              | <i>Anwendung</i>   |
| VV  | <b>Vorsilbe zu Nachnamen.</b> Dabei handelt es sich um Namensbestandteile wie "von ", "den ", und "zu ", aber auch um Kombinationen wie "von der " usw. Eine Vorsilbe gehört immer zum <i>person name part</i> des Typen <i>family</i> , der immer direkt dahinter steht (siehe die XML Beispiele). Eine Vorsilbe kann als Bestandteil des Nachnamen aufgenommen werden, aber es ist gebräuchlich, sie einzeln zu speichern (und zu versenden), da Sortierungen (und daher auch das Zurücksuchen) immer auf Nachnamen ohne Vorsilbe stattfinden. |
| AC  | <b>Akademischer Grad.</b> Ein Grad (meistens in abgekürzter Form) den jemand aufgrund der Vollendung eines wissenschaftlichen Studiums oder eines anderen Studiums erworben hat, z.B.: "Dr. ", "Dipl.-Ing. ", "Ing. ", "Hr. ", "Dr. ", "Prof. ", aber auch "Prof. Dr. ", „Prof. Dr. med. “.  |
| NB  | <b>Adelstitel.</b> Ein Titel (meistens voll ausgeschrieben) der auf dem aristokratischen Status einer Person gründet. Beispiele sind "Baron ", "Graf ", usw.   |
| TITLE   | <b>Allgemeine Anrede (nicht Titel).</b> Diese wird im Prinzip als Anrede zu einem vollständigen Namen (als Briefanrede) verwendet, z.B. " Sehr geehrte Herr" , "Sehr geehrte Frau ", aber auch einfach "Frau ". Die Art einer solchen Anrede hängt also mit dem Geschlecht der Person und dem eventuellen akademischen Grad und/oder Adelstitel ab (mit Hilfe von Ableitungsregeln).   |



Wenn ein *person name part* des Typen *prefix* **ohne einen qualifier** benutzt wird, wird dies immer interpretiert als **Titel**. Dies geschieht, weil viele versendende Systeme zwar Titel unterstützen, aber keinen Unterschied machen zwischen akademischen Graden und Adelstiteln. Vorsilben bei Nachnamen und die allgemeine Anrede ("VV" bzw. "TITLE") müssen also immer explizit übermittelt werden.



Wenn nicht bekannt ist, ob ein Titel vor oder hinter dem Namen anzuhängen ist, muss dieser als *prefix* gesendet werden. Dies wird häufig der Fall sein bei Systemen, die keine Rangfolge im Textfeld bei einem Titel speichern. In solchen Fällen werden Titel also immer *vor* dem Namen im Datentyp *Person Name* positioniert.

Monique van Wijk

```
<name>
  <given>Monique</given>
  <prefix qualifier="VV">van </prefix>
  <family>Wijk</family>
</name>
```

Dr. Kai Heitmann

```
<name>
  <prefix qualifier="AC">Dr. </prefix>
  <given>Kai</given>
  <family>Heitmann</family>
</name>
```

In diesem Fall lautet die Regel also, dass der Grad ('Dr. ') vor dem kompletten Namen positioniert wird, weil dies die normale Schreibweise ist.

Berend-Jan Baron von Fürst zu Fürst

```
<name>
  <given>Berend-Jan</given>
  <prefix qualifier="NB">Baron </prefix>
  <prefix qualifier="VV">von </prefix>
  <family>Fürst zu Fürst</family>
</name>
```

In diesem (realen) Beispiel steht der Titel ('Baron ') zwischen dem Vornamen und dem Nachnamen, da dies die gebräuchliche Schreibweise ist. Zudem steht eine 'normale' Vorsilbe vor dem Nachnamen ('von '). Übrigens wird die Software häufig nicht in der Lage sein, einen (einzeln gespeicherten) Titel an die richtige Stelle zu setzen, wodurch 'Baron' auch vornan landen kann.

Sehr geehrter Herr Dr. Frank Düren

```
<name>
  <prefix qualifier="TITLE">Sehr geehrter Herr </prefix>
  <prefix qualifier="AC">Dr. </prefix>
  <given>Frank</given>
  <family>Düren</family>
</name>
```

In diesem Beispiel hat das versendende System offensichtlich den vollständigen Titel festgelegt (oder vom Titel ableitet) und als Teil des 'Korrespondenznamen' übermittelt. Aber auch ohne Titel kann das versendende System eine allgemeine Anrede mitsenden (siehe nachstehend).

Frau A. Jansen

```
<name>
  <prefix qualifier="TITLE">Frau </prefix>
  <given>A.</given>
  <family>Jansen</family>
</name>
```

<suffix> ..... Suffix

Attribut:

@code ..... Titelscode (CE)

@qualifier ..... Namensart (CS)

Ein *person name part* des Typen *Suffix* bezieht sich auf einen Teil des Namens, der **zu einem oder mehreren anderen Namensbestandteilen gehört und hinten angehängt wird**. In den Niederlanden sind Nachsilben ausschließlich bei **akademischen Graden** zugelassen.

Einige Regeln für *person name parts* des Typen *suffix*:

- Ein *suffix* muss immer direkt hinter die Namensbestandteile positioniert werden, auf die es sich bezieht (d.h. wo es normalerweise steht).
- Es gibt **keine implizite Leerstelle** als Zwischenraum zu dem davor stehenden *name part*, d.h. eine Leerstelle vor der Nachsilbe muss explizit angegeben werden!

Die Art der Nachsilbe kann ferner angegeben werden durch die Anwendung des optionalen Attributs *qualifier*. Siehe die Tabelle für die dabei zugelassenen Werte.



Im nächsten Release der Datentypen von HL7 v3 wird es möglich sein, diverse Elemente die jetzt 'string' sind, optional auch als Code anzudeuten.

Dies kann u.a. zur kodierten Versendung von Titeln benutzt werden, wenn das sendende System diese als Code speichert. Darüber hinaus muss aber immer der Text gesendet werden, damit der Empfänger entscheiden kann, was benutzt wird.

| qualifiers bei person name part suffix |   |
|--|---|
| Qualifier                              | Anwendung   |
| AC                                     | <b>Akademische Grade.</b> Ein Grad (meistens in abgekürzter Form) den jemand aufgrund der Vollendung eines wissenschaftlichen Studiums oder eines anderen Studiums erworben hat. Bei Nachsilben handelt es sich dabei meistens um internationale Termen wie "MSc", "PhD" oder "MD". |



Ein *person name part* des Typen *suffix*, der ohne *qualifier* benutzt wird, muss interpretiert werden als nicht näher bestimmtes Suffix. Auch die Verwendung von (vielfach amerikanischen) Termen wie 'Jr.', 'Sr.' of 'III' fällt unter diese Kategorie.

Ronald Cornet, MSc

```
<name>  
  <given>Ronald</given>  
  <family>Cornet</family>  
  <suffix qualifier="AC">, MSc</suffix>  
</name>
```

### 3.22 ON (Organisationsname – Organization Name)

Definition: Ein Name einer Organisation.

Attribut:

*@use..... Benutzertype(n) (SET<CS>), nicht verwenden in DE*

Element:

*validTime..... Gültigkeitszeitraum (IVL<TS>)*

Struktur: Der Datentyp ON ist eine Extension des Datentyps EN (Entity Name) und besteht demnach aus dem so genannten 'mixed content' Inhalt, in dem im Prinzip *name parts* angegeben werden können. Für Deutschland wurde vorläufig abgesprochen, keine *organization name parts* zu verwenden und den Namen immer als ein Text auszudrücken. Das heißt, dass ON in Deutschland faktisch identisch wird mit dem Datentyp TN (Trivial Name).

#### XML Beispiel

minimaal

```
<name>
  Universität zu Köln
</name>
```

Beachten Sie, dass der Text des Namens ohne Anführungszeichen übermittelt wird.

*@use..... Benutzungstype(n)*

Im Prinzip kann von jedem *Organization Name* angegeben werden, in welcher Situation dieser benutzt werden kann. Für die Niederlande wurde aber beschlossen, dass der Unterschied (noch) nicht relevant ist, so dass die Empfehlung lautet, das Attribut *use* nicht im XML-Tag zu benutzen.

*@validTime..... Gültigkeitszeitraum*

Dies ist ein **optionales** XML Element in *Organization Name*, das die Periode angibt, in der dieser Name für die betreffende Organisation 'in Gebrauch'/gültig war. Die Optionen sind:

- Es gibt kein *validTime* Element: Der betreffende Name ist im Prinzip universell gültig.
- Es gibt eine Unter- und Obergrenze: Der Name war in der angegebenen Periode gültig.
- Es gibt nur eine Untergrenze: Der Name ist seit dem angegebenen Datum gültig.
- Es gibt nur eine Obergrenze: Der Name war bis einschl. dem angegebenen Datum gültig.

## Unter- und Obergrenze

```
<name>
  <validTime>
    <low value="20030101"/>
    <high value="20041231"/>
  </validTime>
  Medizinische Einrichtungen der Universität zu Köln
  <!-- der alte Name der Organisation -->
</name>
```

Der Name war gültig vom 1.1.2003 bis einschl. 31.12.2004.

## Nur Untergrenze

```
<name>
  <validTime>
    <low value="20050113"/>
  </validTime>
  Klinikum der Universität zu Köln
</name>
```

Der Name ist gültig seit dem 1.1.2005.

## Nur Obergrenze

```
<name>
  <validTime>
    <high value="20051231"/>
  </validTime>
  Medizinische Einrichtungen der Universität zu Köln
</name>
```

Der Name ist gültig bis einschließlich 31.12.2005.

Sowohl die Untergrenze als auch die Obergrenze von *validTime* können in der Zukunft liegen, um einen geplanten neuen Namen oder ein geplantes Verfallsdatum für einen bestehenden Namens anzugeben.



In allen Situationen, in denen ein oder mehrere *Organization Names* übermittelt werden, muss minimal der Name angegeben werden, der zum Zeitpunkt des Versendens gültig/aktuell ist. Nicht mehr gültige Namen können also nur übermittelt werden, wenn das betreffende Nachrichtenelement wiederholt benutzt wird (also mit max. Kardinalität > 1).

## Wiederholter Name

```
<name>
  <validTime>
    <high value="20041231"/>
  </validTime>
  Altorganisation
</name>
<name>
  <validTime>
    <low value="20050101"/>
    <high value="20091231"/>
  </validTime>
  Jetztorganisation
</name>
<name>
  <validTime>
    <low value="20100101"/>
  </validTime>
  Zukunftsorganisation
</name>
```



### **3.23 QTY (Quantität – Quantity)**

Quantität ist ein abstrakter Datentyp. Er wird benutzt, um die Basiseigenschaften abgeleiteter Typen zu definieren. Diese Abstraktion ist erforderlich, um andere Datentypen definieren zu können, wie z.B. Integer, Physical Quantity etc.

### 3.24 INT (Ganze Zahl – Integer number)

Elemente dieses Datentyps haben ein Attribut,

*@value .....Wert (INT)*

das ganze Zahlen (Integer) annehmen kann. Integer können als solche in den Nachrichten vorkommen, beispielsweise die Anzahl der Wiederholungen einer Verordnung (repeatCount) und die sequenceNumber in einem wiederholbaren actRelationship. Im allgemeinen gilt, dass Integer zum Zählen, bzw. Ordnen verwendet wird.

#### ***XML Beispiel***

```
<number value="24"/>
```

### **3.25 REAL (Zahl mit Dezimalen – Real number)**

Der Datentyp Real kann eine Zahl mit Dezimalen als Wert annehmen. Die dezimale Wiedergabe mit einem Punkt „.“. Der Typ REAL kommt als solcher nicht in Nachrichten vor. Ein REAL ist immer Bestandteil eines anderen Typs, meistens Physical quantity (PQ).

### 3.26 PQ (Quantitative Angabe physikalischer Größen – Physical Quantity)

Elemente dieses Datentyps sind physikalische Größen, d.h. eine Menge, die einen messbaren (oder zählbaren) Wert aus dem physikalischen Umfeld wiedergibt. Das ist also etwas anderes als nur eine 'Anzahl', da es auch um eine (natürliche) Einheit geht, worin der festgelegte Wert ausgedrückt wird.

Beispiele:

- Eine abgenommene Blutmenge von 20 ml (Probengröße)
- Eine Schnittwunde mit einer Länge von 10 cm (Ergebnis Beobachtung)
- Eine Dosis von 200 Milligramm eines Medikaments (Dosiermenge)
- Lieferung von 60 Stück einer Tablette (erteilte Menge)

Elemente dieses Datentyps haben die folgende Struktur.

Das Attribut *value* trägt die Größe der Menge, ausgedrückt in der Einheit (*unit*).

*@value* ..... Größe der Menge (REAL)

In *unit* steht die Messeinheit, konform mit dem *Unified Code for Units of Measure (UCUM)*. Diese Tabelle enthält alle natürlichen Messeinheiten.

*@unit* ..... Messeinheit (CS)

Wenn *value* "nicht zählbar", "messbar" ist (z.B. ausgedrückt in Milligramm oder Liter) muss diese Einheit in diesem Attribut aufgenommen werden. Wenn *value* 'zählbar' ist (z. B. Tabletten), trägt dieses Attribut den Wert 1.

Eine vollständige Beschreibung der möglichen Einheiten finden Sie in der Spezifikation des *Unified Code for Units of Measure (UCUM)*, die in HL7 v3 aufgenommen ist. UCUM ist eine Sammlung von atomaren Einheiten und Präfixen mit der dazugehörigen Grammatik zum Aufbau gültiger Kombinationen.

Wenn ein *nullFlavor* Attribut vorhanden ist, dürfen *value* und *unit* nicht vorkommen; Wenn kein *nullFlavor* Attribut vorhanden ist, muss ein *value* und *unit* spezifiziert werden.

Eine alternative Repräsentation mit derselben physikalischen Menge, ausgedrückt in einer anderen Einheit, eventuell aus einem anderen System als UCUM und eventuell mit einem anderen zugehörigen Wert (*value*) kann unter *translation* wiedergegeben werden.

*<translation>* ..... alternative Repräsentation (PQR)

Weil die Messwerte meistens im Zusammenhang mit einer allgemeinen Beobachtung wiedergegeben werden (Observation), wobei der Datentyp für *value* ANY ist, muss auf jeden Fall der Datentyp für die Instantiierung über  *xsi:type* angegeben werden.

#### XML Beispiele

Messwert mit UCUM Einheit (165 cm)

```
<value value="165" unit="cm"/>
```

2) Messwert mit UCUM Einheit (92,1 kg)

```
<value value="92.1" unit="kg"/>
```

3) Messwert mit UCUM Einheit (Blutdruck 120 mm Hg)

```
<value value="120" unit="mm[Hg]"/>
```

Messwert z. B. Stück (5)

```
<value value="5" unit="1"/>
```

Messwert mit alternativer Repräsentation (1 [Stück], in HL7 V3 mit Einheit „1“, übersetzt in zwei alternative Einheiten [niederländische WCIA Tabelle 25 Dosiereinheit und G-Standard Thesaurus 002 Dosiereinheit])

```
<doseQuantity>
  <center value="1">
    <translation value="1" code="245"
      codeSystem="2.16.840.1.113883.2.4.4.1.900.2"
      displayName="Stück"/>
    <translation value="1" code="100"
      codeSystem="2.16.840.1.113883.2.4.4.1.361"
      displayName="Tablette"/>
  </center>
</doseQuantity>
```

### 3.27 MO (Geldbetrag – monetary amount)

Elemente dieses Datentyps haben zwei Attribute.

*@value ..... Wert (REAL)*

*@currency ..... Währung (CS)*

Dieser Datentyp wird benutzt, um einen Geldbetrag (value) in einer bestimmten Währung (currency) anzugeben. MO hat viel Ähnlichkeit mit dem PQ Typ, aber es gibt einen essentiellen Unterschied: Währungskurse sind im Gegensatz zur Umrechnung physikalischer Einheiten variabel, wodurch Geld nicht in einem PQ Typ definiert werden kann.

Währungen werden codiert nach ISO 4217.

| Tabelle: Codes für das currency Attribut (wichtigste Codes) |                       |                        |
|---|-----------------------|------------------------|
| Code  | Name                  | Land                   |
| EUR   | Euro                  | Europäische Union      |
| GBP   | Britisches Pfund      | Vereinigtes Königreich |
| USD   | Amerikanischer Dollar | Vereinigte Staaten     |

#### XML Beispiel

```
<value xsi:type="MO" currency="EUR" value="97.32"/>
```

### 3.28 TS (Zeitpunkt – timestamp)

Elemente dieses Datentyps haben ein Attribut

*@value ..... Wert (TS)*

Ein Datum kann mit dem folgenden Format festgelegt werden: YYYYMMDDHHMM. Wenn ein System zu wenig Information hat (beispielsweise: nur das Jahr ist bekannt), kann die Kette von rechts nach links abgebrochen werden.

#### **XML Beispiele**

14. Juni 1970

```
<effectiveTime value="19700614"/>
```

12. April 2004, 12:45 Uhr

```
<effectiveTime value="200409091245"/>
```

Das Jahr 1996

```
<effectiveTime value="1996"/>
```

24. September 2005, 17:32:45 Uhr

```
<effectiveTime value="20050924173245"/>
```

Mai 2000

```
<effectiveTime value="200005"/>
```

### **3.29 BAG Bag**

Einige Attribute tragen den Datentyp BAG<T> mit T als diskreten oder kontinuierlichen Datentyp. Dies ist eine ungeordnete Sammlung von Werten, wobei Werte auch mehr als einmal vorkommen können.

In der XML Repräsentation wird ein BAG<T> mit einer Kardinalität von n..m umgesetzt in ein wiederholbares XML Element mit der Kardinalität n..\*. Eine explizite Darstellung von BAG in XML ist also nicht vorhanden.



### **3.30 SET Set**

Einige Attribute tragen den Datentyp SET<T> mit T als diskreten oder kontinuierlichen Datentyp. Dies ist eine ungeordnete Sammlung von Werten, wobei Werte maximal einmal vorkommen können.

In der XML Repräsentation wird ein SET<T> mit einer Kardinalität von n..m umgesetzt in ein wiederholbares XML Element mit der Kardinalität n..\*. Eine explizite Darstellung von SET in XML ist also nicht vorhanden.

### **3.31 SXCM Set Component**

Es gibt ein generisches Set Component, das eine Komponente eines diskreten oder kontinuierlichen Datentyps repräsentiert und als Teil einer Menge fungiert. In den Niederlanden wird dies momentan ausschließlich beim Datentyp GTS angewendet (siehe GTS).

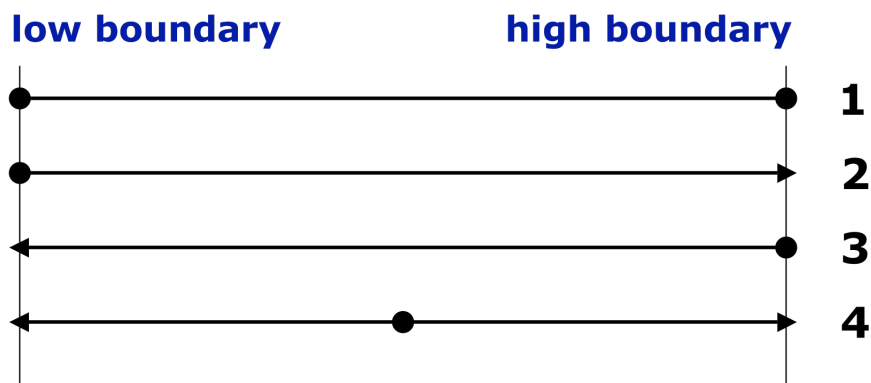
### 3.32 IVL Intervall

In der deutschen Situation werden momentan die folgenden Intervalldefinitionen benutzt.

Intervalle sind nur für Datentypen mit einem Ordinal- oder Intervall-Skalentyp definiert. Es gibt verschiedene Möglichkeiten, ein Intervall festzulegen (siehe nachstehende Abbildung): Hier wird beispielsweise

- eine Untergrenze <low> und eine Obergrenze <high> (1),
- eine Untergrenze <low> und eine Breite <width> (2)
- eine Obergrenze <high> und eine Breite <width> (3)
- oder ein Mittelpunkt <center> eines Intervalls und eine Breite <width> (4)

angegeben.



Es spricht für sich, dass es noch mehr Möglichkeiten gibt. Ein Intervall muss auch nicht immer vollständig spezifiziert werden (offene Intervalle). So kann beispielsweise mit nur einer Obergrenze eine maximale Dosis angegeben werden.

Aus Implementierungsgründen sind in der deutschen Situation ausschließlich die nachstehenden Kombinationen zur Angabe eines Intervalls zugelassen:

- eine Untergrenze <low> und eine Obergrenze <high>
- nur eine Untergrenze <low>
- nur eine Obergrenze <high>
- nur der Mittelpunkt des Intervalls <center>
- nur die Breite eines Intervalls <width>

Für die folgenden Datentypen sind Intervallspezifizierungen definiert.

### 3.33 IVL\_TS (Zeitintervall – Interval of Timestamps)

Elemente dieses Datentyps geben Zeitintervalle an. Dabei wird normalerweise eine Ober- und eine Untergrenze als Zeitpunkt angegeben. Bei offenen Zeitintervallen (beispielsweise ab Zeitpunkt X, oder gültig bis Zeitpunkt Y) wird lediglich eines der beiden Element angegeben.

*<low> ..... Untergrenze (TS)*

Enthält den Anfangszeitpunkt eines Zeitintervalls. In Deutschland gibt es nur zwei Möglichkeiten, *low* anzuwenden: entweder individuell (wenn der Endzeitpunkt des Intervalls unbekannt ist), oder zusammen mit *high*. *Low* wird in Deutschland nie zusammen mit *width* oder *center* benutzt.

Der bei *low* angegebene Wert wird standardmäßig interpretiert als "ein Anfangszeitpunkt später als, oder gleich mit". Wenn man explizit *inclusive="false"* angibt, kann dies verändert werden in "ein Anfangszeitpunkt später als". Zu beachten ist, dass die Interpretation u.a. von der Präzision des Zeitpunkts abhängt: "nach 2004" bedeutet "ab 1. Januar 2005", "nach 20041201" bedeutet "ab 2. Dezember 2004 00:00 Uhr", und "nach 200412011200" bedeutet "ab 1. Dezember 2004 12:01".

*<high> ..... Obergrenze (TS)*

Enthält den Endzeitpunkt eines Zeitintervalls. In Deutschland gibt es nur zwei Möglichkeiten, *high* anzuwenden: entweder individuell (wenn der Anfangszeitpunkt des Intervalls unbekannt ist), oder zusammen mit *low*. *High* wird in Deutschland nie in zusammen mit *width* oder *center* benutzt.

Der bei *high* angegebene Wert wird standardmäßig als "ein Endzeitpunkt vor oder gleich mit" definiert. Indem explizit angegeben wird, dass die Obergrenze nicht inklusiv ist, kann dies verändert werden in "ein Zeitpunkt vor". Dies wird mit dem Attribut

*@inclusive.....(true|false)*

angegeben und kann sowohl in einem *<low>* als auch in einem *<high>* Element benutzt werden. Wenn dies nicht spezifiziert ist, wird "true" als Default angenommen.

Zu beachten ist, dass die Interpretation u.a. von der Präzision des Zeitpunkts abhängt: "vor 2008" bedeutet "vor dem 1. Januar 2008", "vor 20081201" bedeutet "vor dem 1. Dezember 2008", und "vor 200412011200" bedeutet "vor dem 1. Dezember 2004 12:00".

*<center> ..... In der Mitte des Zeitintervalls (TS)*

Enthält den Zeitpunkt, der den Mittelpunkt des Zeitintervalls kennzeichnet. In Deutschland wird dies ausschließlich benutzt, wenn man (in einem Attribut des Datentyps IVL<TS> einen spezifischen Zeitpunkt angeben will, anstatt eines Zeitintervalls. *Center* wird in Deutschland nie zusammen mit *low*, *high*, oder *width* benutzt.

Zu beachten ist, dass die Interpretation u.a. von der Präzision des Zeitpunkts abhängt: "(In) 2005" bedeutet "im Jahr 2005", "am 20051201" bedeutet "am 1. Dezember 2005", und "200512011200" bedeutet "am 1. Dezember 2005 um 12:00 Uhr".

*<width> ..... Zeitdauer des Intervalls (PQ)*

Enthält die Zeitdauer des Intervalls. In Deutschland wird dies ausschließlich benutzt, wenn *low*, *high* und/oder *center* bei einem Intervall nicht bekannt ist, sondern nur die Dauer. *Width* wird in Deutschland nie zusammen mit *low*, *high*, oder *center* benutzt.

#### XML Beispiele

Am und später als der 7. Mai 2004 bis einschl. 9. September 2004

```
<effectiveTime>  
  <low value="20040507"/>  
  <high value="20040909"/>  
</effectiveTime>
```

Am und später als der 7. Mai 2004

```
<effectiveTime>  
  <low value="20040507"/>  
</effectiveTime>
```

Am (während dem) 3. Januar 1975

```
<value>  
  <center value="19750103">  
</value>
```

In (während) 2003

```
<value>  
  <center value="2003">  
</value>
```

Am und nach dem 3. Januar 1975, aber vor dem 7. Januar 1975

```
<value>  
  <low value="19750103">  
  <high value="19750107" inclusive="false">  
</value>
```

### 3.34 IVL\_INT (Intervall von Integer – Interval of Integers)

Elemente dieses Datentyp geben Intervalle von Integer (ganze) Zahlen an. Dabei wird in der Regel eine Ober- und eine Untergrenze als Zahl angedeutet. Bei offenen Intervallen (beispielsweise bei der Ausgabe einer Wiederholung "maximal 3 Mal wiederholen") wird nur eines der beiden Elemente angegeben.

*<low> ..... Untergrenze (INT)*

*<high> ..... Obergrenze (INT)*

#### **XML Beispiel**

von 3 bis einschl. 5

```
<repeatNumber>  
  <low value="3"/>  
  <high value="5"/>  
</repeatNumber>
```

maximal 5

```
<repeatNumber>  
  <high value="5"/>  
</repeatNumber>
```

### 3.35 IVL\_PQ (Intervall bei physikalischen Mengen – Interval of Physical Quantities)

Elemente dieses Datentyps geben Intervalle bei physikalischen Mengen an. Dabei wird in der Regel eine Ober- und eine Untergrenze als *value unit* Paar angedeutet.

*<low>* ..... Untergrenze (PQ)

*<high>* ..... Obergrenze (PQ)

Es ist anzumerken, dass die Differenzgrößen (width) nicht notwendigerweise die gleiche Einheit tragen müssen, siehe z.B. Temperaturdifferenz in Kelvin, pH-Werte-Differenz.

#### **XML Beispiel**

Zwischen 7,5 und 10 mmol/l. Bemerkung: 10.1 gehört nicht mehr zum Intervall, die Grenze ist exakt 10.

```
<referenceRange>
  <low value="7.5" unit="mmol/l"/>
  <high value="10" unit="mmol/l"/>
</referenceRange>
```

### 3.36 RTO (Verhältnisangaben zweier Daten – ratio)

Elemente dieses Datentyps haben zwei Sub-Elemente, den Zähler (numerator) und den Nenner (denominator) eines Vergleichs (Ratio). Der QTY (Quantity) Datentyp ist ein abstrakter Datentyp, der in Nachrichten durch einen spezifischen Datentyp ersetzt wird, häufig PQ oder TS. Die relevanten Strukturelemente sind:

**<numerator>** .....Zähler (QTY)

Die Menge, die in der Ratio geteilt wird. Der Defaultwert ist der Integer 1.

**<denominator>** ..... Nenner (QTY)

Die Menge, durch die der *numerator* in der Ratio geteilt wird. Der Defaultwert ist der Integer 1. Der *denominator* darf nicht den Wert 0 haben. Der *denominator* ist häufig eine Zeiteinheit.

#### XML Beispiel

6 (mal/Stück) pro (ein) Tag

```
<maxDoseQuantity>
  <numerator xsi:type="PQ" value="6"/>
  <denominator xsi:type="PQ" value="1" unit="d"/>
</maxDoseQuantity>
```

400 Milligramm pro (ein) Tag

```
<maxDoseQuantity>
  <numerator xsi:type="PQ" value='400' unit='mg' />
  <denominator xsi:type="PQ" value='1' unit='d' />
</maxDoseQuantity>
```

5 mal pro (eine) Stunde

```
<maxDoseQuantity>
  <numerator xsi:type="PQ" value="5"/>
  <denominator xsi:type="PQ" value="1" unit="h"/>
</maxDoseQuantity>
```

1:10000 beispielsweise bei epidemiologischen Daten oder bei Laborbefunden

```
<value xsi:type="RTO_QTY_QTY">
  <numerator xsi:type="INT" value="1"/>
  <denominator xsi:type="INT" value="10000"/>
</value>
```

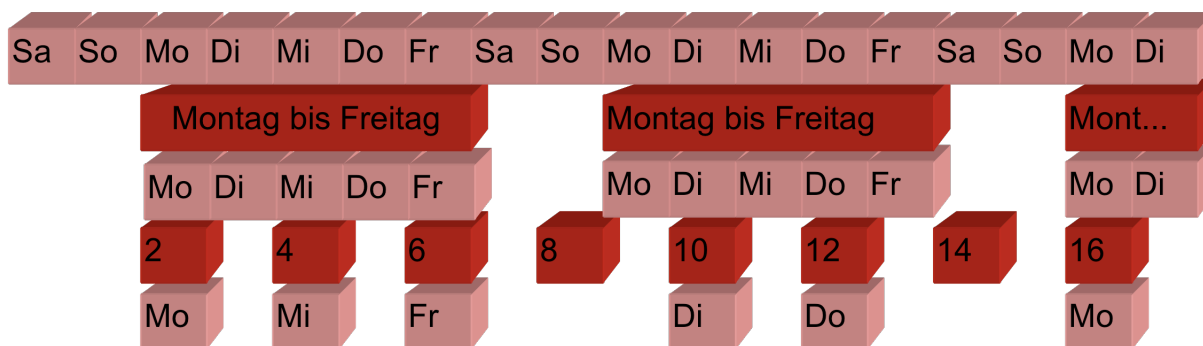


### 3.37 GTS – Der generische Zeit Datentyp

Nachstehend wird der GTS Datentyp (General Timing Specification oder generische Zeitangabe) beschrieben, wobei ausschließlich die Aspekte beschrieben werden, die auf die deutsche Situation zutreffen. Falls es spezifische Anweisungen gibt, wie etwas in Deutschland implementiert werden muss, wird dies im Text angegeben.

Der Datentyp GTS bietet eine sehr umfangreiche (aber auch komplexe) Struktur für die Angabe von Zeitpunkten, Zeitintervallen und Zeitschemas mit einem praktisch unbegrenzt komplexen Aufbau an. Die Analyse von GTS ist ein Thema für sich, aber zum Glück lassen sich die meisten Anwendungen mit einem limitierten Set an GTS Funktionen, wie TS und IVL<TS> beschreiben.

Einfach gesagt ist ein GTS ein SET<TS>. Diese können zu periodischen Sets zusammengefügt werden und durch Set-Operationen ist es möglich, auch komplexe Zeitangaben zu spezifizieren. Das folgende Schema zeigt zur Illustration eine Zeitangabe am Beispiel einer Instruktion zur Arzneiverabreichung "an geraden Tagen (Beginn am Tag 0, dem ersten Tag, an dem die Arznei verabreicht wird), wochentags von Mo. bis Fr." als ein Set von Tagen. Das Set der Wochentage (erste Reihe) wird mit einem zweiten Set, das als Zeitintervall der Tage von Mo. bis Fr. definiert ist, geschnitten. Daraus ergibt sich die dritte Reihe. Diese wird mit einem Set von geraden Tagen geschnitten, wodurch schließlich Reihe fünf mit dem gewünschten Ergebnis entsteht.



Es sind verschiedene XML Repräsentationen eines Elements mit dem Typen GTS denkbar, aber in Deutschland hat ein vollständiges Element des Typen GTS immer folgendes Format:

#### effectiveTime

Set component 1

Set component 2

Set component ...

Das XML Element effectiveTime wird als SXPR\_TS (*parenthetic set expression* des Typen TS) definiert und die Set-Komponenten <comp> werden als Kindelemente zugefügt.

Das bedeutet, dass im XML Element die Set-Komponente Elemente benutzt wird. Bei XML hat ein GTS Datentyp also immer das folgende Muster.

```
<effectiveTime xsi:type="SXPR_TS">
  <comp xsi:type="...">
    ...
  </comp>
  <comp xsi:type="..." operator="A">
    ...
  </comp>
</effectiveTime/>
```

Als Set-Komponenten sind die Datentypen PIVL (*periodic interval of time* oder periodisches Zeitintervall) und EIVL (*event related interval of time* oder ereignisbezogenes Intervall) zugelassen. In Deutschland wird zurzeit nur PIVL benutzt. Eine Set-Komponente wird über die Angabe *xsi:type* als Datentyp PIVL\_TS, IVL\_TS oder TS definiert. Für die Set-Komponenten sind Operatoren verfügbar (siehe auch vorstehendes Beispiel für die Set-Operationen), welche die verschiedenen Komponenten zusammensetzen können: die Vereinigung mit dem vorigen Zeitschema, die Differenz mit dem vorigen Zeitschema oder den Durchschnitt mit dem vorigen Zeitschema (siehe nachstehend unter PIVL für eine nähere Erläuterung).

```
<effectiveTime xsi:type="SXPR_TS">
  <comp xsi:type="IVL_TS">
    <!-- eine normale Intervallangabe -->
  </comp>
  <comp xsi:type="PIVL_TS" operator="A">
    <!-- periodische Intervallangaben (hier mit einem Operator zwischen zwei Sets) -->
  </comp>
</effectiveTime/>
```

Weil GTS eine Superklasse für alle Zeitangaben in HL7 ist, kann von einem Element des Datentyps GTS in einer XML Instanz beispielsweise auch ein Intervall erstellt werden, indem man das *xsi:type* XML Attribut anwendet. Gesetzt den Fall, dass *effectiveTime* als GTS in einem Modell definiert ist, dann enthalten die folgenden XML Instanzen ausnahmslos gültige Zeitangaben.

```
<effectiveTime xsi:type="IVL_TS">
  <low value="20050924"/>
  <high value="20050925"/>
</effectiveTime >
```

```
<effectiveTime xsi:type="SXPR_TS">
  <comp xsi:type="IVL_TS">
    <low value="20050901"/>
    <width value="90" unit="d"/>
  </comp>
  <comp xsi:type="PIVL_TS" operator="A">
    <period value="2" unit="d"/>
  </comp>
</effectiveTime/>
```

### 3.38 PIVL (Periodisches Zeitintervall – Periodic Interval of Time)

Definition: Ein Zeitintervall (oder Zeitpunkt) das/der sich periodisch wiederholt.

Attribute:

*@operator.....Mengenoperator (CS)*

*@alignment..... ausrichten auf ... (CS)*

*@institutionSpecified.....Pünktlichkeitsindikator (BL), nicht verwenden in DE*

Elemente:

*<phase>.....Basisintervall (IVL<TS>)*

*<period>..... Wiederholungsperiode PQ [time]*

Struktur: Der Datentyp PIVL ist eine Erweiterung des Datentyps SXCM (*Set Component*) und gilt daher als ein Teil einer Menge. In diesem speziellen Fall setzt sich eine solche Menge aus **Zeitspezifikationen** zusammen, die gemeinsam die Beschreibung eines bestimmten Zeitschemas darstellen. Die Zeitspezifikationen werden dabei zusammengesetzt mit Operatoren (Set Operators), wie 'Vereinigung', 'Differenz' und 'Durchschnitt'. Ohne Beispiele ist das schwierig zu erklären. Deshalb wird dies nachstehend anhand ausführlicher Beispiele demonstriert.

Anwendung: Nachrichtenelemente mit dem Datentyp PIVL kommen ausschließlich als Teil des (abstrakten) Datentyps GTS vor. *General Timing Specification* nutzt eine sehr allgemeine Methode zur Darstellung von Zeitschemas und besteht grundsätzlich aus einer Menge zusammengesetzter Komponenten. Eine der wichtigsten Komponentenarten ist der Datentyp PIVL. Hiermit werden Zeitphasen angegeben, die sich wiederholen, wie z.B. '2x täglich', oder 'alle 2 Wochen Montags von 10:00 bis 10:30'.

#### XML Beispiele

2x täglich

```
<comp xsi:type="PIVL_TS">
  <period value="0.5" unit="d"/>
</comp>
```

alle 2 Wochen Montags von 10:00 bis 10:30

```
<comp xsi:type="PIVL_TS" alignment="DW">
  <phase>
    <low value="200508291000"/>
    <width value="30" unit="min"/>
  </phase>
  <period value="2" unit="wk"/>
</comp>
```

Anmerkung: In den beiden vorstehenden Beispielen hat das Nachrichtenelement den Namen <comp>. Damit wird angegeben, dass es sich um Elemente binnen des Datentyps SXPR (*Paranthetic Set Expression*) handelt. Das ist die Form, in der PIVL meistens vorkommt. Nähere Erläuterungen zu SXPR finden Sie unter Datentyp GTS (*General Timing Specification*) an anderer Stelle in diesem Dokument.

**@operator.....Mengenoperator (CS)**

Wie gesagt fungiert PIVL als Teil einer Menge Komponenten. Wenn diese Komponenten zusammengesetzt werden, ergibt sich ein Zeitschema. Dies lässt sich am besten anhand praktischer Beispiele erklären.

Beispiel: Ein Rezept hat eine Gültigkeitsdauer von 90 Tagen ab 1.9.2005 (d.h. die geplante Verabreichungsperiode läuft vom 1.9.2005 bis einschl. 29.11.2005). Das Attribut *effectiveTime* in der Gebrauchsanweisung besitzt den Datentyp GTS und besteht aus einer Menge des Datentyps SXPRTS (*Paranthetic Set Expression*), die mit dem Intervall zwischen diesen beiden Daten beginnt. Es handelt sich also um eine Menge, die ein IVL<TS> enthält:

**XML voorbeelden**

Geplante Verabreichungsperiode

```
<effectiveTime xsi:type="SXPRTS">
  <comp xsi:type="IVL_TS">
    <low value="20050901"/>
    <width value="90" unit="d"/>
  </comp>
</effectiveTime/>
```

*Bemerkung: In vorstehendem Beispiel kommt der Datentyp PIVL selbst noch gar nicht vor, aber es soll deutlich machen, wie der Operator im PIVL benutzt wird, um die Menge zu definieren. Siehe auch die Beschreibung des allgemeinen Datentyps GTS.*

Angenommen, dass in der betreffenden Periode alle 2 Tage eine Arznei verabreicht werden muss. De facto muss dann der Durchschnitt des angegebenen Intervalls genommen werden, sowie ein sich periodisch wiederholendes 'Etwas', von nur bekannt ist, dass es alle 2 Tage stattfindet.

Alle 2 Tage innerhalb des angegebenen Intervalls

```
<effectiveTime xsi:type="SXPRTS">
  <comp xsi:type="IVL_TS">
    <low value="20050901"/>
    <width value="90" unit="d"/>
  </comp>
  <comp xsi:type="PIVL_TS" operator="A">
    <period value="2" unit="d"/>
  </comp>
</effectiveTime/>
```

Der Operator "A" in der vorstehend zugefügten PIVL Komponente hat die Definition: 'Nehmen sie den **Durchschnitt** der zuvor angegebenen Zeiten (Intervall von 90 Tagen ab 1.9.2005) mit der Wiederholungsperiode (*period*) die danach angegeben wird. Das Ergebnis ist also 'alle 2 Tage vom 1.9.2005 bis einschl. 29.11.2005'. Dies ergibt also 45

Verabreichungszeitpunkte, ohne dass genau definiert wird, an welchen Tageszeitpunkten diese Ereignisse stattfinden sollen.

Die gültigen Operatoren bei PIVL\_TS (und aller anderen Komponenten eines SXPRTS) sind:

| Tabelle: Domäne SetOperator |   |
|-----------------------------|---|
| Code                        | Definition  |
| I                           | Nehmen Sie die Vereinigung mit dem vorigen Zeitschema.  |
| E                           | Nehmen Sie die Differenz mit dem vorigen Zeitschema.    |
| A                           | Nehmen Sie den Durchschnitt mit dem vorigen Zeitschema. |



Das Attribut *operator* ist in einem PIVL Nachrichtenelement **verpflichtend** in allen Fällen, in denen es als Teil einer **SXPRT Menge** fungiert *und nicht das erste Element* ist. Bei zweiten und weiteren Elementen einer SXPRT Menge muss schließlich die Beziehung zu den vorigen Elementen angegeben werden (oder besser gesagt zu der bis dahin aufgebauten Menge).

Siehe auch die allgemeine Beschreibung bei GTS über die Anwendung von *set operator*.

*<phase>*..... *Basisinterval*

Das Element *phase* ist eigentlich ein **Beispiel (Prototyp)** eines Basisintervalls (oder Basiszeitpunkt), das im PIVL Datentyp periodisch wiederholt wird. Dabei kann in PIVL z. B. 'täglich um 14:00' angegeben werden, indem man den Zeitpunkt '14:00' an einem willkürlichen Tag als *phase* andeutet und bei *period* angibt, dass dies sich täglich wiederholt.

Das Element *phase* ist **nicht verpflichtend**, weil es auch vorkommt, dass nur eine Wiederholungsperiode angegeben werden muss und ein Basiszeitpunkt oder Basisintervall nicht nötig ist. So ist *phase* bei 'alle zwei Tage' im Beispiel weiter oben nicht nötig, weil es nicht relevant ist, an welchem Zeitpunkt innerhalb der zwei Tage das Ereignis (z.B. die Verabreichung der Arznei) stattfindet. Nachstehend finden Sie einige Beispiele von Situationen, in denen dies wohl der Fall ist.

Täglich um 8:00

```
<comp xsi:type="PIVL_TS">
  <phase>
    <center value="200509010800"/>
  </phase>
  <period value="1" unit="d"/>
</comp>
```

Zu beachten ist, dass die Tatsache, dass hier der 1. September 2005 eingetragen ist, de facto irrelevant ist! *Phase* ist nur angegeben, weil der Zeitpunkt 08:00 übermittelt werden muss und dient insofern als Prototyp eines solchen Zeitpunkts. Dabei muss aber ein Datum angegeben werden, weil die Übermittlung eines separaten Zeitpunkts in der HL7 Version 3 (Datentyp TS) nicht möglich ist! Es hätte dort also auch `<center value="198812040800">` stehen können, woraus die Verarbeitungssoftware exakt die gleiche Schlussfolgerung ziehen müsste (täglich um 08:00).

Zu beachten ist auch, dass *phase* immer die Form eines Intervalls hat (Datentyp IVL\_TS). Wenn also *ein* separater Zeitpunkt (oder *ein* separater Tag) angegeben werden muss, geschieht dies am besten, indem man das Element <center> von IVL\_TS anwendet. Damit wird das Intervall praktisch auf einen Zeitpunkt reduziert, behält aber die Struktur von IVL.

Täglich um 8:00, während 10 Minuten

```
<comp xsi:type="PIVL_TS">
  <phase>
    <low value="200509010800"/>
    <width value="10" unit="min"/>
  </phase>
  <period value="1" unit="d"/>
</comp>
```

Der einzige Unterschied zu dem vorigen Beispiel ist, dass *phase* jetzt ein 'echtes' Intervall ist, anstatt ein einfacher Zeitpunkt. Dies ist z.B. der Fall, wenn eine Arzneiverabreichung, eine physiotherapeutische Behandlung oder eine andere Aktivität in einer bestimmten Zeitspanne ausgeführt werden muss. Dies kann auch vorkommen, wenn der Zeitpunkt im Grunde nicht relevant ist:

Täglich, während 10 Minuten

```
<comp xsi:type="PIVL_TS">
  <phase>
    <width value="10" unit="min"/>
  </phase>
  <period value="1" unit="d"/>
</comp>
```

In diesem Beispiel ist von *phase* nur noch die Dauer (*width*) relevant.

**@alignment..... ausrichten auf ... (CS)**

Das Attribut *alignment* wird benutzt, um anzugeben, welcher Aspekt von *phase* (das Basisintervall) relevant ist zum Festlegen des Wiederholungsmusters. Das klingt in erster Instanz sehr verwirrend, bietet aber die Möglichkeit, Wiederholungsmuster, wie 'jeden Montag' oder 'an jedem 14e eines Monats' oder 'jeden 2. Mai' relativ einfach anzugeben.

### Jeden Montag

```
<effectiveTime xsi:type="PIVL_TS" alignment="DW">
  <phase>
    <center value="20050829"/>
  </phase>
```

```
<period value="1" unit="wk"/>
</effectiveTime>
```

Das *alignment* 'DW' gibt hier an, dass der relevante Aspekt der angegebenen *phase* 'day of the week' is. Angesichts der Tatsache, dass 29.8.2005 ein Montag war, steht hier also nichts anderes wie 'jeden Montag'. Wie bereits zuvor erwähnt, ist der exakte Wert von *phase* nicht relevant.

Die gültigen *alignment* Varianten bei PIVL\_TS beschränken sich in den Niederlanden auf:

| Tabelle: Domäne CalendarCycle |      |  |
|-------------------------------|------|--|
| Naam                          | Code | Beschrijving                             |
| day of the week               | DW   | Jeden Montag, Dienstag, Mittwoch, ...    |
| day of the month              | DM   | Jeden 1e, 2e, 3e, 4e, ... des Monats     |
| day of the year               | DY   | Jeden 1/1, 2/1, 3/1, 4/1, ... des Jahres |

Zu beachten ist also, dass *alignment* nur dann sinnvoll ist, wenn *phase* die Form eines einfachen Zeitpunkts oder eines Intervalls hat, die innerhalb *eines* Tages liegt.

Jeden 15. des Monats

```
<effectiveTime xsi:type="PIVL_TS" alignment="DM">
  <phase>
    <center value="20050915"/>
  </phase>
  <period value="1" unit="mo"/>
</effectiveTime>
```

Jeden 1/3 und 1/8, von 14:00 bis 16:00

```
<effectiveTime xsi:type="SXPR_TS">
  <comp xsi:type="PIVL_TS" alignment="DY">
    <phase>
      <low value="200503011400"/>
      <width value="2" unit="h"/>
    </phase>
    <period value="1" unit="a"/>
  </comp>
  <comp xsi:type="PIVL_TS" operator="I" alignment="DY">
    <phase>
      <low value="200508011400"/>
      <width value="2" unit="h"/>
    </phase>
    <period value="1" unit="a"/>
  </comp>
</effectiveTime>
```

**@institutionSpecified ..... Pünktlichkeitsindikator (BL)**

Das Attribut *institutionSpecified* soll angeben, ob es sich bei der Wiederholungsperiode, die der PIVL Datentyp angibt, um eine 'harte' oder 'weiche' Zeitspezifikation handelt. Das bedeutet: Darf der Empfänger bei der Angabe '3x täglich' selbst entscheiden, an welchen Zeitpunkten das Ereignis stattfindet oder muss dies exakt alle 8 Stunden (gleichmäßig verteilt) geschehen.

Es gibt drei Gründe, weshalb dieses Attribut in den Niederlanden vorläufig **nicht benutzt** wird:

- Über die korrekte Interpretation dieses Attributs finden noch die nötigen Diskussionen statt.
- Der Name wurde schlecht gewählt, weil das gleiche Phänomen z.B. auch bei ambulanten Arzneimittelverordnungen auftreten kann. In diesem Fall bedeutet *institutionSpecified* „vom Patienten mit gesundem Menschenverstand zu entscheiden“.
- Das Attribut *institutionSpecified* wurde vor allem hinzugefügt, weil *period* im PIVL eine Angabe wie '3x täglich' nur mit 'alle 8 Stunden' wiedergeben kann, was viel 'härter' (exakter) klingt wie beabsichtigt. Da inzwischen das Element *frequency* an PIVL zugefügt wurde (mit dem '3x täglich' wohl angegeben werden kann), verfällt das Argument für die Anwendung von *institutionSpecified*.

**<period> ..... Wiederholungsperiode (PQ [time])**

Das Element *period* ist in jedem Nachrichtenelement des Datentyps PIVL **verpflichtend**. Es gibt darin die Wiederholungsperiode des Ereignisses an, auf das es sich bezieht.

Die *period* selbst hat den Datentyp PQ (Physical Quantity), mit der Einschränkung, dass ausschließlich Daten einer **Zeitdauer** benutzt werden dürfen. Das Format ist also immer:

<period value=[Anzahl] unit=[Zeiteinheit] />

Die folgenden Zeiteinheiten (konform mit UCUM) müssen dabei immer unterstützt werden:

| Name   | Einheit | Definiert als  |
|--------|---------|--|
| second | s       | SI-Einheit "the duration of 9,192,631,770 periods of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the caesium-133 atom at zero kelvins" |
| minute | min     | 60 s   |
| hour   | h       | 60 min   |
| day    | d       | 24 h   |
| week   | wk      | 7 d  |
| year   | a       | 365.25 d (siehe Anmerkung)   |
| month  | mo      | 1 a/12   |

*Bemerkung: Zu beachten ist, dass die Definition eines Jahres (Unit 'a', abgeleitet von 'annum') auf dem so genannten Julianischen Kalender (mit einem Jahr von exakt 365.25 Tagen) beruht. In Fällen, in denen es wichtig ist, dass ein Ereignis immer auf den gleichen Tag eines Monats oder Jahres fällt, wird das alignment Attribut benutzt, damit es keine Missverständnisse geben kann.*

Im einfachsten Fall kann auf diese Art und Weise z.B. angegeben werden, dass ein Ereignis (z.B. das Verabreichen einer Arznei) alle 2 Stunden auszuführen ist. Das geschieht wie folgt:



Alle 2 Stunden

```
<comp xsi:type="PIVL_TS">
  <period value="2" unit="h"/>
</comp>
```

Bei Arzneimittelverordnungen ist es üblich, um z.B. '3x täglich' als Frequenz für die Verabreichung anzugeben. Dies kann im PIVL auf zweierlei Art und Weise angegeben werden:

3x täglich, in Tagen

```
<comp xsi:type="PIVL_TS">
  <period value="0.3333" unit="d"/>
</comp>
```

3x täglich, in Stunden

```
<comp xsi:type="PIVL_TS">
  <period value="8" unit="h"/>
</comp>
```

Im zweiten Fall findet die Umrechnung also über  $1\text{ d} = 24\text{ h}$ . statt. Weil im PIVL Datentyp keine Frequenz, sondern eine Wiederholungsperiode angewendet wird, kann '3x täglich' nur in ganzen Zahlen ausgedrückt werden, also 'alle 8 Stunden'. Gefühlsmäßig klingt letzteres strenger, aber mathematisch sind beide Angaben natürlich vollkommen äquivalent.



Gerade weil beide Angaben (mit Hilfe von Tagen, bzw. Stunden) äquivalent sind, muss jedes empfangende Anwenderprogramm in der Lage sein, beide zu verarbeiten. Im verarbeitenden Programm müssen sie zur exakt gleichen Interpretation führen.

Ein sendendes System darf selbst entscheiden, welche Methode bevorzugt wird. Als Richtlinie gilt, dass bei Nicht-Ganzen *values* (wie vorstehend die 0.3333) maximal 4 relevante Dezimale nach dem Punkt übermittelt werden.

Ein vergleichbares Phänomen ergibt sich bei Wiederholungsperioden von mehr als einem Tag:

1x in 2 Tagen

```
<comp xsi:type="PIVL_TS">
  <period value="2" unit="d"/>
</comp>
```

Diese Situation ist einfach, aber schwieriger wird es bei '3x wöchentlich' (zwei Möglichkeiten):

3x wöchentlich, in Wochen

```
<comp xsi:type="PIVL_TS">
  <period value="0.3333" unit="wk"/>
```

```
</comp>
```

3x wöchentlich, in Tagen

```
<comp xsi:type="PIVL_TS">  
  <period value="2.3333" unit="d"/>  
</comp>
```

Hier ist deutlich, dass es nicht hilft, die Einheit anzupassen an "d", da in allen Fällen auch ein Dezimalpunkt nötig ist. Die Wahl liegt auch hier bei dem sendenden System, aber jedes empfangende System muss in der Lage sein, beide Varianten zu verarbeiten.

Zudem ist es möglich, eine Frequenz mit '2x monatlich' anzugeben. Dazu muss angemerkt werden, dass dies keine exakte Angabe ist (aufgrund der wechselnden Anzahl Tage pro Kalendermonat), aber im PIVL kann dies folgendermaßen ausgedrückt werden:

2x monatlich

```
<comp xsi:type="PIVL_TS">  
  <period value="0.5" unit="mo"/>  
</comp>
```

oder indem die Frequenz angegeben wird mit '3x pro Jahr':

3x pro Jahr

```
<comp xsi:type="PIVL_TS">  
  <period value="0.3333" unit="a"/>  
</comp>
```



Zu beachten ist, dass eine Umrechnung von Monaten oder Jahren in Tage oder Wochen nicht wünschenswert ist, weil dabei unpraktische Werte entstehen (wenn die exakten Umrechnungsfaktoren angewendet werden). Sogar eine andere Interpretation ist dabei möglich (wenn z.B. 1 mo = 30 d oder 1 mo = 4 wk als Faktor hantiert wird).

Es ist allerdings wohl zugelassen, Jahre und Monate untereinander umzurechnen. Vorstehendes Beispiel könnte dann auch mit dem folgenden PIVL ausgedrückt werden:

3x jährlich, in Monaten

```
<comp xsi:type="PIVL_TS">  
  <period value="4" unit="mo"/>  
</comp>
```

Hier wird also der Umrechnungsfaktor 1 mo = 1 a/12 eingesetzt. Fakt ist, dass dieser Umrechnungsfaktor nicht so 'hart' ist wie der Umrechnungsfaktor zwischen Sekunden, Minuten, Stunden und Wochen.

Daraus ergibt sich die folgende Tabelle mit Umrechnungsfaktoren, die unterstützt werden müssen:

| unit |  | s | min | h | d | wk | mo | a |
|------|--|---|-----|---|---|----|----|---|
| s    |  | X | X   | + | + | +  |    |   |
| min  |  | X | X   | X | + | +  |    |   |
| h    |  | + | X   | X | X | +  |    |   |
| d    |  | + | +   | X | X | X  |    |   |
| wk   |  | + | +   | + | X | X  |    |   |
| mo   |  |   |     |   |   |    | X  | + |
| a    |  |   |     |   |   |    | +  | X |

Die "X" in dieser Tabelle geben Umrechnungen an, die von allen empfangenden Systemen unterstützt werden müssen. Die "+" geben Umrechnungen an, die vorzugsweise unterstützt werden müssen. Die nicht ausgefüllten Zellen geben Umrechnungen an (von gesendeten Informationen an gespeicherte Interpretation oder umgekehrt) die nicht zugelassen sind.

## Allgemeine Beispiele

Um die kombinierte Anwendung von *phase* und *period* zu erläutern, werden nachstehend noch einige Beispiele von periodischen Ereignissen aufgeführt (z.B. bestimmte Arzneiverabreichung), die einen festen Eichpunkt und/oder eine feste Zeitdauer haben.

Dies geschieht meistens in Kombination mit einer SXPB Komponente, die das Intervall angibt, in der das Ereignis stattfindet. Die vollständige *effectiveTime* wird dann:

3x täglich, jeweils um 06:00, 14:00 und 22:00, während 30 Min., ab 2. Sept. 2005 um 14:00

```
<effectiveTime xsi:type="SXPB_TS">
  <comp xsi:type="IVL_TS">
    <low value="200509021400"/>
  </comp>
  <comp xsi:type="PIVL_TS" operator="A">
    <phase>
      <low value="200509022200"/>
      <width value="30" unit="min"/>
    </phase>
    <period value="0.3333" unit="d"/>
  </comp>
</effectiveTime>
```

Die PIVL Komponente erhält hier den Operator "A", weil der Durchschnitt mit dem Intervall genommen werden muss, das zuvor bereits im IVL\_TS angegeben ist ('ab 2.9.2005').

Zu beachten ist, dass der faktische Zeitpunkt in der *phase* Komponente des PIVL nicht relevant ist, d.h. er ist nur der Eichpunkt für *period*, ist aber ansonsten willkürlich. Auf diese Art und Weise steht dort '3x täglich, während 30 Min.', wobei aber wohl angegeben wird, dass dies stets um 14:00, 22:00 und 06:00 wiederholt werden muss (da 22:00 als Eichpunkt für die Wiederholung angegeben wird). In Kombination mit dem zuvor angegebenen Intervall ergeben sich die folgenden Zeitpunkte, an denen das Ereignis stattzufinden hat:

- 2. Sept. 2005 um 14:00 (auch wenn dies vor dem Eichpunkt liegt)
- 2. Sept. 2005 um 22:00 (dies ist 'zufällig' der Eichpunkt)
- 3. Sept. 2005 um 06:00
- .... (und so weiter alle 8 Stunden)

Zu beachten ist, dass wenn die Dauer der Aktivität irrelevant (oder unbekannt) gewesen wäre, die Komponente *width* von *phase* einfach hätte fehlen können. Wenn sogar die Anfangszeit irrelevant (oder unbekannt) gewesen wäre, wäre *phase* überhaupt nicht nötig gewesen. Das würde die nachstehenden PIVL Komponenten in dem vorgenannten Beispiel ergeben:

3x täglich, jeweils um 06:00, 14:00 und 22:00

```
<comp xsi:type="PIVL_TS" operator="A">
  <phase>
    <low value="200509022200"/>
  </phase>
  <period value="0.3333" unit="d"/>
</comp>
```

3x täglich

```
<comp xsi:type="PIVL_TS" operator="A">
  <period value="0.3333" unit="d"/>
</comp>
```

Schließlich ein vergleichbares Beispiel mit einer Wiederholung an zwei festen Wochentagen:

jeden Mo. und Fr. um 13:00, während 4 Stunden, während Sept. 2005

```
<effectiveTime xsi:type="SXPR_TS">
  <comp xsi:type="IVL_TS">
    <low value="20050901"/>
    <high value="20050930"/>
  </comp>
  <comp xsi:type="SXPR_TS" operator="A">
    <comp xsi:type="PIVL_TS" alignment="DW">
      <phase>
        <low value="200508291300"/>
        <width value="4" unit="h"/>
      </phase>
      <period value="1" unit="wk"/>
    </comp>
    <comp xsi:type="PIVL_TS" operator="I" alignment="DW">
      <phase>
        <low value="200509021300"/>
        <width value="4" unit="h"/>
      </phase>
      <period value="1" unit="wk"/>
    </comp>
  </comp>
</effectiveTime>
```

Zu beachten ist, dass hier zwei PIVL Komponenten definiert sind, die miteinander vereinigt werden (Operator "I"). Gemeinsam wird dieses Sub-Set wieder geschnitten (Operator "A") mit dem zuvor festgelegten Intervall ('der Monat September'). Dies ist ein Beispiel einer geschachtelten Anwendung des Datentyps SXP, das unter GTS näher erläutert wird.

Zu beachten ist auch, dass das Basisintervall des ersten PIVL (für die Wiederholung am Montag) nicht einmal innerhalb des Intervalls liegt, mit dem geschnitten wird. (29/8 wird nämlich als Eichpunkt benutzt). Die relevanten Aspekte des Eichpunkts sind nur der Wochentag (durch die Kombination mit dem *alignment* auf "DW") und der Anfangszeitpunkt (ab 13:00). Mit anderen Worten: PIVL beschreibt de facto **jeden Montag zwischen 13:00 und 17:00**, von der unendlichen Vergangenheit bis in die unendliche Zukunft. Der Durchschnitt sorgt für die Begrenzung.

Alles in allem ergibt sich daraus die folgende Liste mit den durch diese Struktur definierten Zeiten:

- 2. Sept. 2005 um 13:00 (der erste Freitag im September 2005)
- 5. Sept. 2005 um 13:00 (der erste Montag im September 2005)
- 9. Sept. 2005 um 13:00 (der zweite Freitag im September 2005)
- ....
- 26. Sept. 2005 um 13:00 (der letzte Montag im September 2005)
- 30. Sept. 2005 um 13:00 (der letzte Freitag im September 2005)

Zum Schluss noch ein Beispiel, in dem ein bestimmtes periodisches Intervall ausgeschlossen wird:

Tablettenschema: 21 Tage einnehmen, 7 Tage nicht, 1x täglich um 09:00

```
<effectiveTime xsi:type="SXP_TS">
  <comp xsi:type="IVL_TS">
    <low value="20050901"/>
    <high value="20051130"/>
  </comp>
  <comp xsi:type="SXP_TS" operator="A">
    <comp xsi:type="PIVL_TS">
      <phase>
        <low value="200509010900"/>
      </phase>
      <period value="1" unit="d"/>
    </comp>
    <comp xsi:type="PIVL_TS" operator="E">
      <phase>
        <low value="20050922"/>
        <width value="7" unit="d"/>
      </phase>
      <period value="28" unit="d"/>
    </comp>
  </comp>
</effectiveTime>
```

```
</comp>
</effectiveTime>
```

Auch hier werden zwei PIVL Komponenten definiert, wobei die Differenz zwischen der ersten und der zweiten Komponente festgelegt wird (die zweite wird dabei von der ersten subtrahiert). Gemeinsam wird dieses Sub-Set wieder geschnitten (Operator "A") mit dem zuvor festgelegten Intervall ('die Monate September bis einschl. November').

Der Effekt ist, dass ein sich wiederholendes Muster mit der folgenden Struktur entsteht:

|  |                                    |
|--|------------------------------------|
| 3 Wochen mit täglicher Einnahme um 09:00 | 01/09/2005 bis einschl. 21/09/2005 |
| 1 Woche ohne Einnahme                    | 22/09/2005 bis einschl. 28/09/2005 |
| 3 Wochen mit täglicher Einnahme um 09:00 | 29/09/2005 bis einschl. 20/10/2005 |
| 1 Woche ohne Einnahme                    | 21/10/2005 bis einschl. 27/10/2005 |
| 3 Wochen mit täglicher Einnahme um 09:00 | 28/10/2005 bis einschl. 17/11/2005 |
| 1 Woche ohne Einnahme                    | 18/11/2005 bis einschl. 24/11/2005 |
| 6 Tage mit täglicher Einnahme um 09:00   | 25/11/2005 bis einschl. 30/11/2005 |

Die tägliche Einnahme wird durch *period* im ersten PIVL festgelegt. Die Woche ohne Einnahme wird durch die zweite PIVL mit einer *phase* von 7 Tagen festgelegt, die sich pro *period* von 28 Tagen wiederholt und im Muster des ersten PIVL ausgeschlossen wird.

Zu beachten ist, dass es durchaus relevant ist, welches Datum als Anfangsdatum bei *phase* im zweiten PIVL gewählt wird. Dies muss nämlich so mit dem **Anfangsdatum des Intervalls** übereinstimmen (nicht unbedingt mit dem von *phase* im ersten PIVL!), dass die Periode ohne Einnahme stets in die 4<sup>e</sup> Woche fällt (also im Anschluss an eine Periode von 3 Wochen mit täglicher Einnahme).

# 4 CMETS

---



Dieses Kapitel folgt separat und ist zunächst nicht Bestandteil dieses Dokuments.

# 5

## Referenzen

---

- [HL7V3] HL7 Version 3  
<http://www.hl7.org>  
Ballot 11, September 2005  
Normative Edition 2005
- [XMLSC] World Wide Web Consortium. XML Schema.  
<http://www.w3.org/TR/xmlschema-0/>  
<http://www.w3.org/TR/xmlschema-1/>  
<http://www.w3.org/TR/xmlschema-2/>
- + World Wide Web Consortium. XML Schemas Part 1: Structures.  
<http://www.w3.org/TR/xmlschema-1/>
- + World Wide Web Consortium. XML Schemas Part 2: Datatypes.  
<http://www.w3.org/TR/xmlschema-2/>
- [XML] World Wide Web Consortium. Extensible Markup Language, 1.0, 2nd Edition.  
<http://www.w3.org/TR/REC-xml>
- [OIDK] Object Identifier (OID) Konzept für das Deutsche Gesundheitswesen, Gemeinschaftskonzept der HL7-Benutzergruppe in Deutschland e. V., Köln, der Arbeitsgemeinschaft Sciphox GbR mbH, Köln, der Kassenärztlichen Bundesvereinigung - Körperschaft des öffentlichen Rechts, Berlin, und des Deutschen Instituts für Medizinische Dokumentation und Information (DIMDI), Köln, Entwurf, Version 1.02, Stand: 18.03.2005 ([www.hl7.de](http://www.hl7.de))

# 6

## Anhang

---

Dieses Kapitel ist nicht normativ.

## 6.1 HL7

HL7 (Health Level Seven) ist der weltweit eingesetzte und anerkannte Kommunikationsstandard im Gesundheitswesen. Im Vordergrund stand dabei bisher der Austausch von Nachrichten, sowohl für administrative als auch klinische Belange.

HL7 Version 3 [HL7V3] definiert eine neue Generation von Kommunikationsstandards für die Spezifikation, Entwicklung und Pflege von Nachrichten im gesamten Gesundheitswesen. Dies wird mit einer ausgereiften Methodik zur modell-basierten und Werkzeug-gestützten Entwicklung zugeschnittener Nachrichten erreicht.

Zahlreiche Projekte wurden bereits mit HL7 Version 3 Spezifikationen erfolgreich durchgeführt. Viele europäische Länder, darunter die Niederlande, Großbritannien und Dänemark, haben HL7 Version 3 als strategisches Konzept für eine landesweite Kommunikation im Gesundheitssektor gewählt. In England wurde vom National Health Service (NHS) bereits vor zweieinhalb Jahren das GP2GP-Projekt initiiert, das sich gerade HL7 Version 3 zur Unterstützung für die Kommunikation Niedergelassener zu Nutze machte. Daraus ist mittlerweile ein nationales, staatlich stark gefördertes und deutlich ausgedehntes Projekt geworden. Auch in Deutschland ist in den zurzeit vom Gesundheitsministerium initiierten Bestrebungen rund um die elektronische Gesundheitskarte (bIT4health) aktiv HL7 Version 3 als Zieltechnologie und Kernelement sektorenübergreifender IT-Anwendungen vereinbart worden.

Die für dieses Projekt zur Anwendung gelangenden Basis-Modelle kommen aus dem Bereich „Health&Clinical Management“, der Domäne „Patient Care Provision“ und „Clinical Documents“.



Hinweis: in dieser Spezifikation kann naturgemäß nur sehr eingeschränkt auf die HL7 Version 3 Nachrichtenkonzepte und Methodologie eingegangen werden. Es wird empfohlen, entsprechende weiterführende Informationen zu Rate zu ziehen (z. B. [HL7V3]) oder entsprechende Fortbildungsangebote zu nutzen.

## 6.2 Hinweise zur Vergabe und Verwendung von Object Identifiern (OIDs)

### 6.2.1 Identifikationen von Objekten

In HL7 muss für jeden der beteiligten Kommunikationsteilnehmer (Systeme, die an der Kommunikation teilnehmen, sog. Devices) ein eindeutiger Object Identifier (OID) existieren [OIDK]. Das bedeutet für die Software, dass bei den Stammdaten einer jeden Senderinstanz für alle Kommunikationspartner eine eindeutige OID bekannt sein muss.

Auch ein Heilberufler ist eindeutig mit einer OID zu versehen. Dabei ist zu beachten, dass ein Arzt selber in diesem Sinne nicht als Sender- oder Empfänger-Gerät auftritt, sondern als Person z. B. in der Rolle Autor. Es handelt sich also um verschiedene OIDs, sendende und empfangende Anwendungen müssen darüber hinaus auch identifizierbar sein.

Eine mögliche und häufig anzutreffende Lösung ist, dass die Hersteller, die sendende Systeme installieren, die OID selbst vergeben. Dabei wird davon ausgegangen, dass jeder Hersteller über eine eigene OID verfügt. Diese wird rechts ergänzt um eine innerhalb des Unternehmens eindeutige Kennung für die jeweilige konkrete Installation (Instanz der Anwendung, bzw. Mandant). Daran angehängt werden die jeweils benötigten Kennungen für die verschiedenen mit einem Identifikator zu versehenen Objekte, wie Dokumenten-Id, Diagnosen-Id, etc.

Verwenden mehrere Software-Prozesse und/oder mehrere Arbeitsplätze einer Praxis/Abteilung/Klinik denselben OID-Zähler, so muss der Zugriff auf diesen Zähler serialisiert sein. Ist dies zu aufwändig, müssen separate Zähler mit jeweils unterschiedlichem Präfix für jede parallel arbeitende Instanz eingerichtet werden.

*Beispiel: Es sei 1.2.3.4.5 die OID eines Systemherstellers. Er ergänzt diese nach rechts um .67, also 1.2.3.4.5.67. Dies soll die Wurzel OID für alle Installationen/Systeminstanzen dieses Herstellers andeuten.*

*Daran anzuhängen wäre dann durchnummeriert eine Zahl für jede Installation, die der Hersteller irgendwo installiert hat. Die Installation des Systems in Arztpraxis A bekäme demnach eine .1 angehängt, was einer Gesamt-OID von 1.2.3.4.5.67.1 entspräche, die Installation im Krankenhaus K bekäme die .2 angehängt und so weiter.*

*Eine Dokumenten-Id ist durch anhängen einer .7 an die Installationskennung (OID) gekennzeichnet. Ein entsprechendes Dokument hätte danach im id Element der ClinicalDocument Klasse als @root die OID 1.2.3.4.5.67.2.7 und eine entsprechende eindeutige Dokumentenkennnummer im @extension Attribut, die innerhalb des sendenden Systems eindeutig sein muss. Gleiches Vorgehen wird auch für die eindeutige Kennung von z. B. Diagnosen, Prozeduren usw. verwendet, wo statt der .7 für Dokumente eine .18 für Diagnosen, eine .19 für Prozeduren angehängt um die eindeutige interne Nummer ergänzt wird.*

Hierbei handelt es sich – wie gesagt – um ein Beispiel, es sind auch andere Vorgehensweisen denkbar oder in realen Implementationen zu finden. Der Hersteller hat dabei dafür Sorge zu tragen, dass jede OID nur einmal vergeben wird und bei Änderungen der Zuordnung von OIDs eine neue OID zur Anwendung kommt.

Einzige Verpflichtung des Herstellers ist also, dass Objekte mit dem OID Konzept weltweit eindeutig und dauerhaft identifiziert werden können.

## 6.2.2 Identifikationen von Codesystemen

Auch die Nutzung der bei HL7 verpflichtend anzugebenden Codesysteme bei der Übermittlung von Codes und anderen Klassifikationen, die ebenso mittels einer OID verwaltet und in Dokumenten spezifiziert werden, stellt ggf. neue Anforderungen an die Hersteller von Software. Im Prinzip muss zu jedem verwendeten Codewert das zugehörige Codesystem im Sinne einer OID bekannt sein.