gematik
Gesellschaft für Telematikanwendungen der Gesundheitskarte mbH

Einführung der Gesundheitskarte

Spezifikation von Versionsnummern in Schnittstellendefinitionen und Software-Komponenten

Version:	1.1.0
Stand:	04.05.2007
Status:	freigegeben

Dokumentinformationen

Änderungen zur Vorversion

Das Dokument wurde im Allgemeinen überarbeitet, Formulierungen wurden präzisiert und ergänzt, kleinere Unstimmigkeiten wurden behoben. Im gesamten Dokument wurden die Hinweise entfernt, wie mit Versionsnummern umzugehen ist, die noch nicht den Vorgaben dieses Dokuments entsprechenden

Speziell wurden Änderungen an folgenden Themen durchgeführt:

- Allgemeine Anforderungen wurden präzisiert und ergänzt (Kapitel 3) und im Anhang in einer Tabelle zusammengefasst (Anhang C).
- Änderungen der Erklärungen zum Aufbau und zur Bedeutung der Versionsnummer (Kapitel 4) wurden vorgenommen.
- die Beschreibung der Versionierung von WSDL- und Schemadateien wurde zum besseren Verständnis in der Reihenfolge geändert (Kapitel 5 und 6), die Beschreibung der Versionsnummern in Schemadateien wurde strukturell umgestellt (Kapitel 6). Die Unterscheidung der verschiedenen Gruppen von Schemadateien wurde erweitert.
- Die Beschreibung der Versionsinfomationen der eGK (Kapitel 9) wurde um die Versionierung der Anwendungsdateien erweitert.
- Der Begriff der „Fachanwendungsversion“ wird als Ersatz der in der Vorversion eingeführten „TUC-Version“ in Kapitel 10 präzisiert.
- Anhang A5 (Versionen einer Fachanwendung) wurde aktualisiert.

Folgendes Kapitel wurde hinzugefügt:

- Kapitel 7 geht auf Versionskennungen für die Versionierung der Konnektor-, Broker- und Fachdienstschnittstellen ein. Die Betrachtung schließt neben Schnittstellenänderungen auch Änderungen im Verhalten der Dienste ein.

Inhaltliche Änderungen gegenüber der letzten freigegebenen Version sind farblich markiert. Sofern ganze Kapitel eingefügt wurden, wurde zur besseren Lesbarkeit lediglich die Überschrift durch gelbe Markierung hervorgehoben.

Referenzierung

Die Referenzierung in weiteren Dokumenten der gematik erfolgt unter:

[gemSpec_VersNr]	gematik (04.05.2007): Einführung der Gesundheitskarte - Spezifikation von Versionsnummern in Schnittstellendefinitionen und Software-Komponenten Version 1.1.0

Dokumentenhistorie

Version	Stand	Kap./ Seite	Grund der Änderung, besondere Hinweise	Bearbeitung
0.0.1	26.10.06		Initiale Version	gematik, AG1
0.0.4	07.12.06		Version für AG1-internes Review	gematik, AG1
0.0.10-0.0.14	08.01.07		Aufteilung in „Spezifikation Versionsnummern in Schnittstellendefinitionen und Software-Komponenten“ (diese Datei) und separates Dokument zur Beschreibung der Prozesse zur Erweiterbarkeit und Versionierung	gematik, AG1
0.0.15	11.01.07		Version für die interne QS	gematik, AG1
0.0.17	22.02.07		Einarbeitung der QS-Kommentare und Erweiterungen Kap. 8, 9 und Anhang A	gematik, AG1
0.9.0	02.03.07		zur Freigabe vorgelegt	gematik
1.0.0	02.03.07		freigegeben	gematik
1.0.1	18.04.07		Überarbeitung und Präzisierung, Neuaufnahme v. Kap. 7, Vertauschung der Reihenfolge von Kapitel 5 und 6	gematik, AG1
1.0.2	19.04.07		mit Ergänzungen und Änderungen für T1 – Version für die interne QS	gematik, AG1
1.0.3	02.05.07		Einarbeitung der Review-Kommentare, Aktualisierung von Anhang A5	gematik, AG1
1.1.0	04.05.07		freigegeben	gematik

Inhaltsverzeichnis

1 Zusammenfassung

Im fortlaufenden Betrieb der Telematikinfrastruktur (TI) und deren über WebServices ansprechbaren Fachanwendungen können aus einer Vielzahl von Gründen Änderungen erforderlich werden, die eine Erweiterung oder Veränderung bestehender Komponenten und Services zu einem definierten Zeitpunkt zur Folge haben werden. Diese Änderungen müssen durch ein Release- und Changemanagement abgebildet werden.

Dieses Dokument definiert die Verwendung von Versionskennungen in den einzelnen Bausteinen der Service-Architektur der TI und bildet damit eine der Grundlagen für das Versionsmanagement für sämtliche von der gematik spezifizierten Software-Komponenten der TI. Die Versionskennungen ermöglichen die eindeutige Identifizierung und Referenzierung des Änderungsstandes einzelner Komponenten und Services im laufenden Betrieb.

Für die wesentlichen zu versionierenden Bestandteile der TI, die als serviceorientierte Architektur (SOA) mittels WebServices realisiert wird, legt dieses Dokument normativ die zu verwendende Versionsnummerierung der Schnittstellendefinitionen fest, die durch XML-Schema- und WSDL-Dateien[1] beschrieben werden. Auch für spezielle XML-Datenstrukturen und Softwarekomponenten wird in diesem Dokument die Verwendung von Versionsinformationen festgelegt. Schließlich geht das Dokument noch kurz auf die Versionsnummer der eGK ein.

[1] Die WebService Description Language (WSDL) [WSDL1.1] ist eine Sprache für die Beschreibung von WebService-Schnittstellen. Dabei können XML-Schemadefinitionsdateien (XSD) verwendet werden, die den Aufbau von Nachrichten beschreiben.

2 Einführung

2.1 Zielsetzung und Einordnung des Dokumentes

Das Dokument definiert die Vergabe und Nutzung von Versionsnummern, die für Schnittstellendefinitionen, Softwarekomponenten und die eGK in den von der gematik spezifizierten Dokumenten anzuwenden sind. Damit wird eine einheitliche Verwendung von Versionsinformationen ermöglicht. Das Dokument ist somit Grundlage für die Erweiterbarkeit und Änderbarkeit aller für den Betrieb der Telematikinfrastruktur benötigten Komponenten.

2.2 Zielgruppe

Das Dokument richtet sich an Entwickler von Komponenten und Diensten sowohl innerhalb als auch außerhalb der gematik.

2.2.1 Interne Zielgruppen

Innerhalb der gematik sind folgende Aufgabenbereiche identifiziert, für welche Festlegungen getroffen werden:

- **Gesamtarchitektur und Facharchitekturen**: Das Dokument definiert die Verwendung von Versionsnummern bei Änderungen an Schnittstellendefinitionen und ist durch die Facharchitekturen und in der Gesamtarchitektur zu verwenden.
- **Arbeitsgruppe Karten und Kartensysteme**: Das Dokument definiert das Format der eGK-Version.
- **Arbeitsgruppe Dezentrale Komponenten**: Das Dokument definiert die Grundlage des Versionierungskonzepts bei Änderungen der Schnittstellen und der Software-Komponenten, die diese Schnittstellen bereitstellen, und ist dadurch auch für die Spezifikation des Konnektors relevant.
- **Arbeitsgruppe Infrastruktur:** Das Dokument definiert die Grundlage des Versionierungskonzepts bei Änderungen und ist für Spezifikationen der Infrastrukturdienste zu verwenden.
- **Anforderungsmanagement**: Das Dokument definiert Anforderungen, die vom Anforderungsmanagement übernommen werden müssen.
- **Arbeitsgruppe Betrieb**: Das Dokument definiert Vorgaben zur Versionierung und Ausgabe von Versionsinformationen der Software-Komponenten, die für die Betriebsprozesse relevant sind.

- **Arbeitsgruppe Test**: Die Umsetzung der Vorgaben zur Versionierung von Telematikinfrastruktur-Komponenten muss durch die AG Test überprüft werden.

2.2.2 Externe Zielgruppen

Das Dokument richtet sich auch an externe Zielgruppen, die Software-Komponenten entwickeln und diese aufgrund neuer Spezifikationen der gematik erweitern und dabei mit Versionsinformationen versehen müssen. Beispielsweise gehören dazu die Hersteller von Software für Komponenten der Telematikinfrastruktur.

2.3 Geltungsbereich

Das vorliegende Dokument entsteht im Rahmen der Arbeiten der gematik für das Projekt „Durchführung der Testmaßnamen zur Einführung der elektronischen Gesundheitskarte“ gemäß Rechtsverordnung in der Fassung vom 05.10.2006 [RVO2006]. Es kann weiterentwickelt und durch eine Nachfolgeversion abgelöst werden. Die Inhalte der Spezifikation haben Gültigkeit für die darin benannten Funktionsabschnitte 1 bis 3.

Diese Funktionsabschnitte werden in der Umsetzung auf die Releases 1 und 2 verteilt. Entsprechend ist der Geltungsbereich dieses Dokumentes auf die Releases 1 und 2 beschränkt.

2.4 Arbeitsgrundlagen

Ausgangspunkt für die Ausführungen dieses Dokuments ist die Gesamtarchitektur [gemGesArch].

2.5 Abgrenzung des Dokumentes

Die Aufgabe dieses Dokuments ist die Definition eines übergreifenden Konzeptes für die Versionsbezeichnung wichtiger Artefakte der Telematikinfrastruktur. Zu den betrachteten Artefakten gehören fachspezifische Datenstrukturen (XML-Schemadateien), WebService-Schnittstellen (XML-Schemadateien und WSDL-Dateien), Software-Komponenten, die eGK sowie Fachanwendungen. Es werden dazu grundlegende Anforderungen an das Format und die Bereitstellung dieser Versionsbezeichnungen gestellt.

Die Versionierung der Dokumentation zu diesen Artefakten (Fachkonzepte, Facharchitekturen und deren UML-Modelle, Spezifikationen, Implementierungsdokumentationen etc.) ist nicht Gegenstand dieses Dokuments.

Der Prozess der Migration von Anwendungen aufgrund von Änderungsanforderungen und Erweiterungen ist nicht Gegenstand dieses Dokuments. Er wird in [gemGA_Wartg] behandelt.

Die spezifizierten Versionsnummern sollen den Umgang mit Änderungen erleichtern und Hinweise zu Umfang und Kompatibilität der Änderungen zu anderen Versionen des gleichen Artefakts liefern. Die einzelnen Prozesse der Versionierung, die zur Erstellung

und Freigabe einer neuen Version einer Schnittstellendefinition oder einer Software-Komponente führen, werden nicht in diesem Dokument spezifiziert.

Das Dokument beschränkt sich auf die Software-Artefakte. Dazu gehören auch die Algorithmen und Dateien der eGK. Hardware-basierte Artefakte der TI sind nicht Gegenstand dieses Dokuments.

2.5.1 Verwendung von Schüsselworten

Für die genauere Unterscheidung zwischen normativen und informativen Inhalten werden die dem RFC 2119 [RFC2119] entsprechenden in Großbuchstaben geschriebenen, deutschen Schlüsselworte verwendet:

- **MUSS** bedeutet, dass es sich um eine absolutgültige und normative Festlegung bzw. Anforderung handelt.
- **DARF NICHT** bezeichnet den absolutgültigen und normativen Ausschluss einer Eigenschaft.
- **SOLL** beschreibt eine Empfehlung. Abweichungen zu diesen Festlegungen sind in begründeten Fällen möglich.
- **SOLL NICHT** kennzeichnet die Empfehlung, eine Eigenschaft auszuschließen. Abweichungen sind in begründeten Fällen möglich.
- **KANN** bedeutet, dass die Eigenschaften fakultativ oder optional sind. Diese Festlegungen haben keinen Normierungs- und keinen allgemeingültigen Empfehlungscharakter.

3 Anforderungen

3.1 Einleitung

Die Anwendungen der Gesundheitskarte und die diese realisierenden Softwarekomponenten sind ständigen Änderungen unterworfen. Daraus ergibt sich der Bedarf nach einer Konvention für die eindeutige Bezeichnung der entstehenden Vielfalt an Versionen. Nur über eindeutige Versionskennungen wird es möglich, Artefakte (vgl. [gemGA_Wartg]) innerhalb der Änderungsprozesse eindeutig identifizieren, sowie auf diese Weise referenzieren zu können. Eine nachvollziehbare, vereinheitlichende Regelung für die Verwendung von Versionsnummern bei der Auszeichnung einzelner Artefakte bildet die Grundlage für die Zusammenstellung in sich konsistenter Gesamtkonfigurationen.

Daher sind für alle Bestandteile der TI, die prinzipiell Änderungen unterliegen können, eindeutige Versionskennungen anzugeben. Im Folgenden werden dazu allgemeine Anforderungen formuliert.

3.2 Fachliche Anforderungen

- Eine Versionsinformation MUSS in jeder veröffentlichten Datei, die ganz oder teilweise eine WebService-Schnittstelle beschreibt, enthalten sein.
- Alle Änderungen an Schemadateien und WSDL-Dateien MÜSSEN zu neuen Versionen führen, auch wenn die Änderungen aufwärts- bzw. abwärtskompatibel sind. Dadurch wird eine korrekte Dokumentation und Referenzierung sowie eine Validierung der erzeugten XML-Datenstrukturen ermöglicht.
- Die in der TI transportierten XML-Daten der Fachanwendungen werden durch Schemadateien unabhängig vom Transport strukturell und inhaltlich beschrieben. Unabhängig von der Struktur kann sich die inhaltliche Bedeutung eines Datenfeldes (z. B. die Kodierung) ändern, ohne dass es zu einer strukturellen Änderung kommt. Anwendungsspezifische XML-Instanzdokumente MÜSSEN Auskunft über ihre inhaltliche Version geben können. Diese Version unterscheidet sich von der Schemaversion der zugehörigen XSD-Datei. Dazu MUSS in jeder anwendungsspezifischen XSD-Datei ein Attribut im Wurzel-Element definiert werden, das diese inhaltliche Versionsinfomation enthält.
- Software-Komponenten, die die von der gematik spezifizierten Funktionalitäten realisieren, MÜSSEN Versionsinformationen besitzen und diese geeignet zurückgeben können.
- Software-Komponenten, die versionierte Schnittstellen anbieten, MÜSSEN in der Lage sein, Auskunft zu den aktuell unterstützten Schnittstellenversionen geben zu können.

- Die eGK MUSS eine eigene Versionskennung besitzen (unabhängig von weiteren auf der eGK abgelegten Versionskennungen), mit der ermittelbar ist, welche eGK-Spezifikation die jeweilige Karte umsetzt.

4 Übergreifende Festlegungen

Dieses Kapitel definiert allgemeine Festlegungen zur Bildung und Bedeutung der meisten in den nachfolgenden Kapiteln aufgeführten Versionsnummern. Ausnahmen sind möglich, werden aber in den entsprechenden Kapiteln näher erläutert.

4.1 Spezifikation des Formats von Versionsnummern

Versionsnummern, die von der gematik vergeben werden, SOLLEN folgenden grundlegenden Aufbau haben:

Hauptversionsnummer.Nebenversionsnummer.Revisionsnummer

Die kleinste Versionsnummer ist 0.0.1. Die einzelnen Bestandteile sind maximal 3-stellig und numerisch. Führende Nullen sind nicht erlaubt.

Neue Versionsnummern SOLLEN nur erzeugt werden, wenn sie intern oder öffentlich zur Nutzung freigegeben werden, unabhängig vom Grund ihrer Erstellung (z. B. nur für interne Testzwecke). Die Nummerierung der öffentlich freigegebenen Versionen ist damit nicht zwangsweise lückenlos. Sichergestellt ist nur, dass eine neue Version eine höhere Nummer als eine Vorgängerversion hat und somit eine Datei mit höherer Versionsnummer zeitlich später freigegeben wurde.

4.2 Bedeutung der Bestandteile der Versionsnummer

Die Bestandteile geben im Allgemeinen absteigend den Grad der Änderungen zur Vorgänger-Version wieder:

- Die **Hauptversionsnummer** eines Artefakts MUSS sich erhöhen, falls daran signifikante Änderungen durchgeführt werden.
- Die **Nebenversionsnummer** MUSS sich erhöhen, falls moderate Änderungen durchgeführt werden.
- Die **Revisionsnummer** MUSS sich erhöhen, falls Änderungen notwendig werden, die die Funktionalität des Artefakts nicht beeinflussen.

Die Bedeutungen der einzelnen Teile der Versionsnummer für die einzelnen Artefakte sind im Rahmen dieses Dokuments in den folgenden Kapiteln näher erläutert und dienen als Hilfestellung zur Interpretation der vorgenommenen Änderungen. Die konkreten Versionsnummern der einzelnen Artefakte werden von der gematik in einem zentralen Prozess (dieser ist nicht Gegenstand dieses Dokuments) vergeben und veröffentlicht.

5 WSDL-Version

5.1 Einleitung

WSDL-Dateien werden für die Definition von WebService-Schnittstellen verwendet. Diese existieren nicht nur für die Konnektor-, Broker- und Fachdienstschnittstellen, sondern auch z.B. für die Management-Schnittstelle des System Monitoring [gemSysSLM]. Eine Versionierung der Dateien ist erforderlich, da sich Schnittstellen und Nachrichtenformate ändern können.

Generell kann man bei der Versionierung von Schnittstellen bei Web Services folgende Fälle unterscheiden:

- Nachrichtenversionierung: Versionierung der Schemata, die für die Definition von Nachrichten verwendet werden. Dazu gehören die Änderung und Erweiterung existierender sowie die Definition weiterer Nachrichtentypen.
- Kontraktversionierung: Versionierung der WSDL- und Kontraktinformationen, die einen Dienst beschreiben.

Laut einer gematik-internen Konvention importiert jede WSDL-Datei genau eine XSD-Datei, in der die Nachrichtendefinitionen stehen. Beide Dateinamen sind bis auf die Endungen gleich.

5.2 WSDL-Version

Jede von der gematik spezifizierte WSDL-Datei hat eine eindeutige Versionsnummer, die **WSDL-Version**. Sie wird von der gematik definiert. Die Versionsnummer MUSS das Format gemäß Abschnitt 4.1 haben. Die Bestandteile MÜSSEN folgende Bedeutung haben:

- Die **Hauptversionsnummer** erhöht sich, falls aus Sicht des Verwenders der Schnittstelle Änderungen derart an der WSDL-Datei oder der importierten XSD-Datei vorgenommen werden, dass Anpassungen in der Nachrichtenverarbeitung erforderlich werden können, die über eine Anpassung des Namensraumes hinausgehen. Dies können je nach Schnittstelle z. B. Änderungen an Nachrichtendefinitionen oder Parametern sein.
- Die **Nebenversionsnummer** erhöht sich, falls aus Sicht des Verwenders der Schnittstelle nur die Anpassung des Namensraumes der Nachricht erforderlich ist (vgl. 6.3).
- Die **Revisionsnummer** erhöht sich, falls Änderungen vorgenommen werden, die keine Schnittstellenänderungen zur Folge haben, z. B.
    - wenn Kommentare geändert, eingefügt oder gelöscht wurden,

- wenn die Struktur der WSDL-Datei geändert wurde, ohne dass sich die bisher beschriebenen WSDL-Definitionen ändern,
- wenn andere neutrale Änderungen an der WSDL-Datei vorgenommen werden.

Jede WSDL-Datei wird einzeln für sich versioniert.

Es MUSS die Festlegung eingehalten werden, dass die Haupt- und Nebenversionsnummern einer WSDL-Datei immer identisch mit den Haupt- und Nebenversionsnummern der importierten schnittstellenspezifischen XSD-Datei ist, da nur beide Dateien zusammen als Spezifikation der Schnittstelle angesehen werden.

Die Version MUSS in der WSDL-Datei unterhalb des „wsdl:definitions“-Knotens im Element „wsdl:documentation“ in der Form „version=<Versionsnummer>“ abgelegt werden. Der Eintrag MUSS als Substring des Textknotens genau einmal vorhanden sein. Im Folgenden ist ein beispielhafter Auszug aus einer WSDL-Datei dargestellt:

```
<wsdl:definitions ...>
      <wsdl:documentation>
            ...
            version=1.1.3
      </wsdl:documentation>
      ...
</wsdl:definitions>
```

5.3 Namespace-Version

Alle von der gematik definierten WSDL-Dateien MÜSSEN im *definitions*-Knoten einen Zielnamensraum (TargetNamespace) definieren, in dem eine zweistellige Versionskennung steht. Der Zielnamensraum der WSDL-Datei MUSS identisch mit dem Zielnamensraum der importierten XSD-Datei sein, damit sind auch die jeweiligen Versionsnummern identisch.

Die Angabe der Versionsnummer erfolgt abweichend zur Festlegung in Abschnitt 4.1 nur mit zwei Stellen:

- Die **Hauptversionsnummer** ist identisch mit der Hauptversionsnummer der WSDL-Datei.
- Die **Nebenversionsnummer** ist identisch mit der Nebenversionsnummer der WSDL-Datei.

Die Festlegung des Aufbaus der URI des Zielnamensraums ist in Anhang A4 ausführlich beschrieben.

6 Versionierung von Schemadefinitionsdateien (XSD-Dateien)

6.1 Einleitung

Dieses Kapitel behandelt die Versionierung von XML-Schemadefinitionsdateien (XSD-Dateien), die von der gematik spezifiziert werden. Diese werden sowohl zur Nachrichtendefinition in WSDL-Dateien verwendet (vgl. Kapitel 5) als auch zur Spezifikation weiterer schnittstellenunabhängiger Datenstrukturen. Erstere werden im Folgenden als **schnittstellenspezifische XSD-Dateien**, letztere als **anwendungsspezifische XSD-Dateien** bezeichnet. Darunter fallen derzeit die Datenstrukturen der Fachanwendungen (fachspezifische XSD-Dateien) und des Ticketservices (XSD-Datei für Tickets). Eine dritte Gruppe von XSD-Dateien definiert Datenstrukturen, die für administrative Zwecke verwendet werden. Dazu gehört die Selbstauskunft von Schnittstellen (vgl. Kapitel 7) und von Softwarekomponenten (vgl. Kapitel 8).

In späteren Ausbaustufen der Telematikinfrastruktur können weitere Datenstrukturen hinzukommen.

Es werden im Weiteren folgende Versionsnummern unterschieden:

- Die **Schemaversion** dient der Kennzeichnung aller Änderungen in XSD-Dateien.
- Zur Kennzeichnung von Änderungen des Namensraums wird die **Namespace-Version** verwendet.
- Mit der **logischen Version** werden Änderungen an der Bedeutung von Struktur und Inhalten in anwendungsspezifischen Datenstrukturen nachvollziehbar.

Die Festlegungen der zu verwendenden Versionsnummern in XSD-Dateien sind für diese betrachteten drei Gruppen unterschiedlich, daher werden diese in den Abschnitten 6.3 und 6.4 getrennt betrachtet.

Tabelle 1: XSD-Dateien und ihre Versionskennungen gibt einen Überblick darüber, welche Versionskennungen in welchen Kategorien von XSD-Dateien definiert werden MÜSSEN. Der Eintrag ‚X' bedeutet, dass die in der aktuellen Zeile vorhandene Versionskennung in der Datei der aktuellen Spalte definiert werden muss.

Tabelle 1: XSD-Dateien und ihre Versionskennungen

Versions-kennung/XSD-Datei	Schnittstellen-spezifische XSD-Datei	Fachspezifische XSD-Datei	XSD-Datei für Tickets	XSD-Datei für Selbstauskunft
Schemaversion	X	X	X	X

Versions-kennung/XSD-Datei	Schnittstellen-spezifische XSD-Datei	Fachspezifische XSD-Datei	XSD-Datei für Tickets	XSD-Datei für Selbstauskunft
Namespace-Version	X	X	X	X
Logische Version	-	X	X	-

6.2 Allgemeine Festlegungen

Dieser Abschnitt enthält allgemeine Festlegungen, die für alle XSD-Dateien Gültigkeit haben. Weitere Festlegungen, die von der Kategorie der XSD-Datei abhängen, befinden sich in den nachfolgenden Abschnitten.

6.2.1 Schemaversion der XSD-Datei

Jede von der gematik spezifizierte XSD-Datei hat eine von der gematik festgelegte eindeutige Versionsnummer, die Schemaversion. Die Schemaversion MUSS das Format gemäß Abschnitt 4.1 haben.

Jede XSD-Datei MUSS einzeln versioniert werden. Die Version MUSS im Attribut „version“ des „xs:schema“-Knotens in der XSD-Datei abgelegt werden. Im Folgenden ist ein beispielhafter Auszug aus einer XSD-Datei dargestellt:

```
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
elementFormDefault="qualified" attributeFormDefault="unqualified"
version="1.3.2">
...
</xs:schema>
```

6.2.2 Namespace-Version

Alle XSD-Dateien MÜSSEN einen Zielnamensraum (TargetNamespace) definieren, der die **Namespace-Version** enthält. Diese entspricht der Hauptversions- und Nebenversionsnummer der zugehörigen Schemaversion. Sie ist also zweistellig. Dadurch wird erreicht, dass dieser Namensraum auch in den zum Schema passenden XML-Instanzdokumenten enthalten ist und bei der Validierung zur Auswahl der zugehörigen Schema-Datei herangezogen werden kann.

Die Festlegungen zum Aufbau des Zielnamensraums zum Format der Versionsnummer sind in Anhang A4 beschrieben.

6.3 Versionierung schnittstellenspezifischer XSD-Dateien

Bei schnittstellenspezifischen XSD-Dateien ist die Haupt- und Nebenversionsnummer der Schemaversion immer identisch mit der Haupt- und Nebenversionsnummer der WSDL-Datei, welche die XSD-Datei importiert (WSDL-Version, vgl. Abschnitt 5.2). Die Bedeutung ist daher identisch mit der der WSDL-Version.

Zur Kennzeichnung von Schema- bzw. Schnittstellenänderungen wird der Namensraum der XSD-Datei versioniert. Die im Zielnamensraum einer schnittstellenspezifischen Schemadefinitionsdatei vorhandene Haupt- und Nebenversionsnummer ist dabei identisch mit der Haupt- und Nebenversionsnummer des Zielnamensraums der WSDL (siehe Abschnitt 5.2), die diese Schemadefinitionsdatei importiert.

6.4 Versionierung anwendungsspezifischer XSD-Dateien

Für alle anwendungsspezifischen XSD-Dateien MÜSSEN die Bestandteile der **Schemaversion** folgende Bedeutung haben:

- Die **Hauptversionsnummer** erhöht sich, falls, abgesehen von einer Zielnamensraumänderung, inkompatible Änderungen an der XSD-Datei vorgenommen werden.
- Die **Nebenversionsnummer** erhöht sich, falls, abgesehen von einer Zielnamensraumänderung, Erweiterungen an der XSD-Datei vorgenommen werden, die als kompatibel zu XSD-Dateien mit der gleichen Hauptversionsnummer eingestuft werden.
- Die **Revisionsnummer** erhöht sich, wenn die vorgenommenen Änderungen für die Schemavalidierung und die Verarbeitung des Inhalts irrelevant sind, z. B.
    - wenn Kommentare geändert, eingefügt oder gelöscht wurden,
    - wenn die Struktur der XSD-Datei (z. B. aus Gründen der Übersichtlichkeit) geändert wurde, ohne dass sich die beschriebenen Elemente oder Typen der XSD ändern (z. B.: geeignete Auflösungen von „import“- Anweisungen),
    - wenn andere neutrale Änderungen an der XSD-Datei vorgenommen werden.

Zur Kennzeichnung von Änderungen in der Bedeutung fachlicher Inhalte in anwendungsspezifischen Datenstrukturen ist eine entsprechende inhaltliche Versionierung erforderlich, die im Folgenden als logische Version bezeichnet wird. (vgl. Abschnitt 3.2). Diese Änderungen umfassen sowohl strukturelle Änderungen als auch Änderungen in der Interpretation der Inhalte, die nicht notwendigerweise zu Änderungen von XSD-Dateien führen müssen.

Die Bestandteile der **logischen Version** MÜSSEN folgende Bedeutung haben:

- Die **Hauptversionsnummer** erhöht sich, falls sich die Hauptversionsnummer der Schemaversion geändert hat oder falls Änderungen an der Bedeutung einzelner oder mehrerer Datenfelder vorgenommen werden, die bei der Interpretation der Inhalte eine geänderte Verarbeitung erfordern. Dies ist insbesondere bei Veränderungen an Werten von kodierten Einträgen zu berücksichtigen, wenn sich die zu Grunde liegende Tabelle und damit die Inhalte der Tabelle ändern.
- Die **Nebenversionsnummer** erhöht sich, falls sich die Nebenversionsnummer der Schemaversion geändert hat oder falls Änderungen an der Bedeutung

derart vorgenommen werden, dass keine veränderte Verarbeitung erforderlich ist, z. B. bei Einschränkungen eines Wertebereichs eines Datenfeldes, der nicht mehr benutzt wird.

- Die **Revisionsnummer** erhöht sich, wenn die vorgenommenen Änderungen für die Verarbeitung des Inhalts irrelevant sind, z. B. wenn die Bedeutung eines Feldes näher präzisiert wird durch Kommentare in der Schemadatei, ohne das sich die Bedeutung grundlegend ändert.

Zu jeder logischen Version gibt es eine eindeutige korrespondierende Schemaversion. Eine Abbildung von logischen Versionen zu Schemaversionen wird von der gematik bereitgestellt. Daher kann für jedes anwendungsspezifische XML-Instanzdokument die Version des zugehörigen Schemas, etwa zu Validierungszwecken, ermittelt werden. In Anhang A2 ist für fachspezifische Datenstrukturen ein Beispiel angegeben, in dem die Gründe für die Unterscheidung der beiden Versionen noch vertieft werden.

Die logische Version für fachliche Datenstrukturen (VSDM, VODM, NFDM) wird mit dem Namen „CDM_VERSION“ („Corresponding Data Modell“, Versionskennung mit Bezug zum jeweiligen Architektur-Modell) bezeichnet.

Die logische Version für Tickets (Objekt- und ServiceTickets) wird mit dem Namen „Version“ bezeichnet.

Die logische Versionsnummer DARF in der XSD-Datei in der Attributdefinition NICHT als fixer Wert aufgenommen werden.

Wie schon in Tabelle 1: XSD-Dateien und ihre Versionskennungen definiert wurde, haben Dateien für die Selbstauskunft keine logische Version.

7 Versionierung von Schnittstellen und Diensten

Dieses Kapitel geht auf die Versionskennungen für die Versionierung der Konnektor-, Broker- und Fachdienstschnittstellen ein. Die Betrachtung schließt neben Schnittstellenänderungen auch Änderungen im Verhalten der Dienste ein. Weitere Informationen zu dem Thema „Versionierung von Schnittstellen“, vor allem über mögliche Änderungsprozesse, befinden sich in [gemGA_Wartg].

7.1 Versionierung der Konnektordienste und -schnittstellen

Die Versionierung der Konnektordienste erfolgt über den Dienstverzeichnisdienst des Konnektors (siehe [gemSpec_Kon]). Er ermöglicht die Versionierung der für die Primärsysteme zur Verfügung gestellten Dienste, d. h. der Basis- und Fachanwendungen. Jeder Dienst kann in unterschiedlichen Versionen und mit unterschiedlichen Endpunkten bereitgestellt werden. Die Versionskennung eines Dienstes wird als Dienstversion bezeichnet.

7.1.1 Dienstversion

Bei der Verwendung von Fachmodulschnittstellen bezieht sich ein Primärsystem immer auf eine bestimmte Facharchitektur oder Spezifikation, in der die Schnittstelle und das Verhalten der Anwendung festgelegt werden. Deshalb MUSS die Dienstversion bei Fachanwendungen immer identisch mit einer Fachanwendungsversion (FA-Version) sein. Der Aufbau und die Bedeutung dieser Versionskennung ist in Kapitel 10 beschrieben.

Für die Basisanwendungen des Konnektors in [gemSpec_Kon] MUSS die Dienstversion das Format gemäß Abschnitt 4.1 haben. Weiterhin MUSS folgendes zur Bedeutung der Versionsnummer gelten:

- Die **Hauptversionsnummer** erhöht sich, falls sich die Hauptversionsnummer des Zielnamensraums der zur Schnittstelle der Basisanwendung gehörenden WSDL ändert.
- Die **Nebenversionsnummer** erhöht sich, falls sich die Nebenversionsnummer des Zielnamensraums der zur Schnittstelle der Basisanwendung gehörenden WSDL ändert.
- Die **Revisionsnummer** erhöht sich, falls an der Basisanwendung nur Änderungen vorgenommen werden, die keine Schnittstellenänderung zur Folge haben. Das wird im Allgemeinen dann der Fall sein, wenn das Verhalten des Dienstes versioniert werden soll.

7.1.2 Selbstauskunft

Wie oben und in [gemSpec_Kon] beschrieben, erfolgt die Selbstauskunft der von einem Konnektor unterstützten Dienstversionen über den Dienstverzeichnisdienst. Für jede Dienstversion (sowohl für Basis- als auch für Fachanwendungen) MUSS es eine eindeutige Abbildung auf den Namensraum der Schnittstelle (WSDL) geben. Dabei

können, z. B. bei Verhaltensänderungen des Konnektormoduls, unterschiedliche Dienstversionen auf den gleichen Namensraum abbildbar sein.

7.2 Versionierung der Transportschnittstelle bei Broker und Fachdienst (TelematikTransport-Nachricht)

Strukturelle oder inhaltliche Änderungen der TelematikTransport-Nachricht werden durch eine Versionskennung, der InterfaceVersion, identifiziert (siehe [gemGA_Wartg]).

7.2.1 InterfaceVersion

Der Ort der InterfaceVersion innerhalb der TelematikTransport-Nachricht wird in [gemSpec_TTD] festgelegt.

Das Format der InterfaceVersion MUSS den in Abschnitt 4.1 gemachten Festlegungen entsprechen. Nicht nur strukturelle sondern auch inhaltliche Änderungen der TelematikTransport-Nachricht können zur Änderung der InterfaceVersion führen. Deshalb haben die Bestandteile der InterfaceVersion die gleiche Bedeutung wie bei der logischen Version in Abschnitt 6.4.

7.2.2 Selbstauskunft

Um während des Betriebs alle InterfaceVersionen ermitteln zu können, welche eine bestimmte Brokerinstanz aktuell unterstützt, muss es für den Betreiber bzw. den Administrator eine entsprechende Abfragemöglichkeit des Brokers geben. Dazu MUSS jede in Betrieb befindliche Broker-Instanz Auskunft zu allen InterfaceVersionen der TelematikTransport-Nachrichten geben können, die sie aktuell unterstützt und verarbeiten kann. Die Schnittstelle SOLL nur innerhalb der Betreiberumgebung verwendbar sein. Die Rückgabe des Brokernamens, des Abfragedatums und der Versionsnummern MUSS in dem in Anhang A3 beschriebenen XML-Format erfolgen.

Weitere Festlegungen zur Schnittstelle und zu den Zugriffswerkzeugen sind nicht Gegenstand dieses Dokuments.

7.3 Versionierung der Fachdienste

Die Requests an den Fachdienst beziehen sich entweder auf Fachanwendungen oder auf den Ticketservice. Die Entscheidung wird durch den Inhalt des Component-Elements der TelematikTransport-Nachricht getroffen (vgl. [gemSpec_TTD]).

Gemäß [gemGA_Wartg] lassen sich Fachdienste anhand von Änderungen ihrer logischen Schnittstellen, aber auch aufgrund von Implementierungsänderungen versionieren. Weiterhin kann sich auch die Schnittstelle des Ticketservice auf Fachdienstseite ändern. In all diesen Fällen wird die Fachdienstversion für die Auswahl und Prüfung der Schnittstellen- und Implementierungsversion herangezogen.

7.3.1 Fachdienstversion

Die Fachdienstversion wird in der TelematikTransport-Nachricht im Element Component-Version abgelegt. Sie wird vom Konnektor gesetzt, der die Nachricht erzeugt und initialisiert. Der genaue Ort innerhalb der Nachricht wird in [gemSpec_TTD] festgelegt.

Die Fachdienstversion MUSS das in Abschnitt 4.1 festgelegte Format haben. Für die Bedeutung der Fachdienstversion MÜSSEN folgende Festlegungen gelten:

- Die **Hauptversionsnummer** erhöht sich, falls aus Sicht des Konnektors inkompatible Änderungen der logischen Fachdienstschnittstelle vorgenommen werden
- Die **Nebenversionsnummer** erhöht sich, falls aus Sicht des Konnektors kompatible Änderungen der logischen Fachdienstschnittstelle vorgenommen werden
- Die **Revisionsnummer** erhöht sich, falls am Fachdienst oder Ticketservice nur Änderungen vorgenommen werden, die keine logischen Schnittstellenänderungen zur Folge haben. Das wird im Allgemeinen dann der Fall sein, wenn sich das Verhalten des Dienstes geändert hat und wenn die Änderung versioniert werden soll.

7.3.2 Selbstauskunft

Auch bei Fachdiensten ist es notwendig, dass Administratoren oder das Betriebspersonal alle unterstützten Fachdienst-Versionen ermitteln können, welche eine bestimmte Fachdienstinstanz unterstützt. Dazu MUSS jede in Betrieb befindliche Fachdienstinstanz Auskunft zu allen Fachdienstversionen geben können, die sie aktuell unterstützt und verarbeiten kann. Die Schnittstelle SOLL nur innerhalb der Betreibergrenzen verwendbar sein. Die Rückgabe des Fachdienstnamens, des Abfragedatums sowie der Namen und Versionsnummern aller aktuell unterstützten Fachdienstkomponenten MUSS in dem in Anhang A3 beschriebenen XML-Format erfolgen

Weitere Festlegungen zur Schnittstelle und zu den Zugriffswerkzeugen bleiben den Herstellern überlassen.

8 Versionierung von Software-Komponenten

8.1 Einleitung

Dieses Kapitel beschreibt den Aufbau der Versionsnummern sowie weitere versionierungsbezogene Anforderungen an Software-Komponenten der TI. Die Anforderungen beziehen sich auf alle Software-Komponenten, die die von der gematik spezifizierten Funktionalitäten implementieren und dem Release-, Change- und Configuration-Management der gematik unterliegen. Dazu gehören die Komponenten der zentralen Telematik Tier (Broker Service, Infrastrukturdienste) sowie die Provider Adapter, die in den einzelnen Betreiberumgebungen eingesetzt werden. Weiterhin fallen darunter die Komponenten des Konnektors, da sie von der gematik getestet und die dafür benötigten Versionskennungen von der gematik verwaltet werden. Im Folgenden bezieht sich der Begriff „Software-Komponente“ auf die hier beschriebene Menge von Komponenten.

8.2 Versionsnummern von Software-Komponenten

Jede Software-Komponente hat einen Namen und eine eindeutige Versionsnummer. Der Name ist eindeutig und wird von der gematik vorgegeben. Anhang A6 enthält eine Auflistung der Komponenten und ihrer Namen. Die Versionsnummer MUSS abweichend zu Abschnitt 4.1 folgenden Aufbau haben:

Hauptversionsnummer.Nebenversionsnummer.Revisionsnummer.HerstellerID.Herstellerversion

Die Bestandteile MÜSSEN folgende Bedeutung haben:

- Die **Hauptversionsnummer** erhöht sich, falls aufgrund von Spezifikationsänderungen signifikante Änderungen in der Komponente durchgeführt werden, die zu einer aus Sicht des Verwenders nicht mehr abwärtskompatiblen Funktionalität führen.
- Die **Nebenversionsnummer** erhöht sich, falls funktionelle Erweiterungen an der Software-Komponente durchgeführt werden, die aus Sicht des Verwenders abwärtskompatibel sind (z. B. weil eine alte Schnittstelle weiterhin unterstützt wird).
- Die **Revisionsnummer** erhöht sich, falls herstellerübergreifende Änderungen notwendig werden, die die Funktionalität der Software-Komponente nicht beeinflussen. Beispiele sind kleinere Änderungen in der Spezifikation sowie andere Vorgaben, die von allen Herstellern der Software-Komponente umgesetzt werden müssen.
- Die **Hersteller-ID** identifiziert den Hersteller der Software-Komponente eindeutig. Das Feld ist maximal 3-stellig und alphanumerisch. Die Hersteller-IDs werden von der gematik auf Anforderung vergeben.

- Die **Herstellerversion** ist eine herstellerspezifische Versionskennung im beliebigen Format, mit der der Hersteller einen spezifischen Build (d. h. die Erstellung einer neuen Version) der Komponente identifizieren kann. Diese Versionskennung wird nicht von der gematik vorgegeben. Die einzige Bedingung ist, dass bei jedem Build eine neue und für die Software-Komponente eindeutige Herstellerversion erzeugt wird. Das Feld kann eine maximale Länge von 30 Zeichen haben. Neben den alphanumerischen Zeichen können noch folgende Satzzeichen aufgenommen werden: - (Bindestrich) . (Punkt) _ (Unterstrich) : (Doppelpunkt).

Logisch gesehen besteht die Versionsnummer aus einer gematik-spezifischen Versionsnummer (Hauptversionsnummer, Nebenversionsnummer und Revisionsnummer) und einer Hersteller-spezifischen Versionsnummer (Hersteller-ID und Herstellerversion). Erstere wird von der gematik definiert, letztere vom Hersteller. Die Beziehung zwischen der Versionsnummer der zugehörigen gematik-Spezifikation und der zugehörigen gematik-spezifischen Versionsnummer wird von der gematik verwaltet.

8.3 Selbstauskunft

Um während des Betriebs feststellen zu können, welche Versionen der einzelnen Software-Komponente aktuell eingesetzt werden, muss es für den Betreiber bzw. den Administrator möglich sein, den Ist-Stand der installierten Software zu ermitteln und diese Informationen ggf. an die gematik weiterzureichen. Dazu MUSS jede installierte und lauffähige Software-Komponente Auskunft zu sich selbst und zu allen Komponenten, von denen sie abhängig[2] ist oder aus denen sie sich zusammensetzt[3] (Teilkomponenten), geben können. Für die Selbstauskunft MUSS jede Software-Komponente eine geeignete Schnittstelle bereitstellen.

Die Selbstauskunft einer Komponente MUSS folgende beiden Varianten vorsehen:

(1) Rückgabe des Namens, des Abfragedatums (einschl. Uhrzeit) und der Versionsnummer der angesprochenen Software-Komponente

(2) Rückgabe des Namens, des Abfragedatums und der Versionsnummer der angesprochenen Software-Komponente sowie die Namen und Versionsnummern aller Software-Komponenten, von denen sie direkt oder indirekt abhängig ist bzw. aus denen sie sich zusammensetzt (Teilkomponenten)

Weiterhin MÜSSEN folgende Bedingungen erfüllt sein:

- Die Rückgabe des Abfragedatums, der Namen und der Versionsnummern einer Komponente MUSS in dem in Anhang A3 beschriebenen XML-Format erfolgen. Der Name, das Abfragedatum und die Version der Komponente stehen im Wurzelknoten. Bei der Rückgabe gemäß Variante 2 SOLL die Anordnung der Knoten gemäß ihrer Abhängigkeits- bzw. Teilerelation (d. h. in Baumdarstellung) erfolgen. Es KÖNNEN aber auch alle Knoten der

[2] Eine Komponente A ist von der Komponente B abhängig, wenn A Operationen von B aufruft oder auf öffentliche Daten von B zugreift.

[3] Eine Komponente muss nicht notwendigerweise von ihren Teilkomponenten abhängen. Das betrifft besonders dann zu, wenn Teilkomponenten optional sind.

Komponenten, von der die Software-Komponente abhängig ist, direkt unter dem Wurzelknoten stehen (Listendarstellung).

- Der Name und die Versionsnummer sind Bestandteil der Software-Komponente und DÜRFEN sich NICHT in von der Software-Komponente getrennten Speicherorten (z. B. Konfigurationsdatei oder Datenbank) befinden. Dadurch sollen Fehler aufgrund von inkonsistenten Updates (z. B. die Komponente wurde aktualisiert, aber nicht die von der Komponente getrennt gespeicherte Versionsnummer) vermieden werden.
- Die Namen und die Versionsnummern sowie das Rückgabeformat MÜSSEN für jeden Build konstant sein und DÜRFEN sich bei einem wiederholten Aufruf NICHT verändern. Es MUSS durch einen einfachen Zeichenkettenvergleich feststellbar sein, ob sich eine Version geändert hat.

Weitere Festlegungen zur Schnittstelle und zu den Zugriffswerkzeugen bleiben den Herstellern überlassen.

Ist eine Komponente nicht lauffähig und nicht in der Lage, ihren Namen und ihre Versionsnummer zurückzugeben, so MUSS der Name und die Versionsnummer über eine geeignete zum Installationszeitpunkt erstellte Datei ermittelbar sein.

Weiterhin MÜSSEN auch die Versionsstände der Basissysteme (Betriebssystem, Datenbanksystem etc.) einschließlich eingespielter Updates (Patches, Service Packs usw.) ermittelbar sein. Das kann durch die Verwendung von Betriebssystem-Funktionen erfolgen.

9 Versionierung der eGK

9.1 eGK-Version

Aufgrund von Änderungen und Erweiterungen des Kartenbetriebssystems, der Dateien der eGK sowie weiteren in der eGK-Spezifikation aufgeführten Artefakten wird die Versionierung der eGK notwendig. Zugreifende Systeme wie der Konnektor müssen erkennen können, welchen Kommando- und Dateiumfang die aktuelle Karte besitzt. Deshalb ist geplant, dass die Chipkarte zukünftig Versionskennungen gemäß Kapitel 8.2 erhalten wird. Die Kennungen für Haupt-, Neben- und Revisionsnummer, welche gemäß Kapitel 4.1 zu wählen sind, sind mit den entsprechenden Nummern der zugrunde liegenden eGK-Spezifikation identisch.

Die genaue Bedeutung der einzelnen Stellen der Versionskennung ergibt sich aus der Bedeutung der Versionskennung der eGK-Spezifikation. Der Ort auf der Karte, in dem die Versionskennungen gespeichert werden, wird Inhalt einer zukünftigen eGK-Spezifikation sein[4]. Da die aktuelle Spezifikation der eGK ([gemSpec_eGK_P1] und [gemSpec_eGK_P2]) diese Versionskennungen noch nicht definieren, kann man beim Fehlen der Versionskennungen auf einer eGK implizit auf diese Versionen schließen.

Da sich die Gültigkeitszeiträume der eGK-Versionen überschneiden können, MÜSSEN zugreifende Systeme (Konnektoren, Primärsysteme etc.) in der Lage sein, verschiedene Versionsstände zu verarbeiten. Die jeweils gültigen Versionsstände werden von der gematik festgelegt.

Es ist geplant, die für die eGK-Version getroffenen Festlegungen analog für die HBA- und SMC-B-Karten zu übernehmen. Die genauen Festlegungen werden in zukünftige Spezifikationen dieser Karten aufgenommen.

9.2 Versionen der Anwendungsdateien

Auf der eGK sind in der Datei DF.HCA (Health Care Applications) verschiedene Elementary Files (EF) definiert (vgl. [gemSpec_eGK_P2]), die die Daten zu den verschiedenen Anwendungen enthalten. Die Definition der Inhalte dieser Dateien erfolgt in [gemeGK_Fach]. In DF.HCA ist für die verschiedenen Anwendungen auch jeweils eine Datei EF.Status definiert. Diese Datei enthält Informationen zur Version der zugehörenden Anwendungsdatei, zur Gültigkeit und zum Datum der letzten Aktualisierung. Die Versionsinformation ist notwendig, da sich die Struktur der Anwendungsdatei im Verlauf der Einführung und Nutzung der eGK ändern kann. Die Änderungen können sich aus neuen gesetzlichen Vorgaben, aber auch aus den Erfahrungen im Test- und Wirkbetrieb ergeben.

Folgende unterschiedliche Versionskennungen existieren in der Datei DF.HCA

[4] Der aktuelle Ansatz sieht dafür vor, in jeder Applikation ein EF.Version vorzusehen, das stets lesbar ist und die Versionsnummer enthält.

- Version in EF.StatusVD: Diese Version ist identisch mit der den CDM_Version der VSD-Datensätze PD, VD und GVD. Sie verändert sich, falls mindestens einer der drei Datensätze ergänzt bzw. geändert wird.
- Ticket Container-Version in EF.StatusRezept: Diese Version bezieht sich auf die Datei EF.eRezept_Tickets, welche mehrere eRezept-Tickets enthalten kann. Mit dieser Versionskennung lassen sich strukturelle Änderungen des Ticket-Containers unterscheiden
- Rezept Container-Version in EF.StatusRezept: Diese Version bezieht sich auf die Datei EF.eRezept_Container, welche mehrere Rezepte (Verordnungen) enthalten kann. Mit dieser Versionskennung können strukturelle Änderungen des Rezept-Containers unterschieden werden
- Version in eRezept-Ticket: Diese Version ist identisch mit der CDM_Version aus dem Verordnungsdatensatz in der Datei EF.eRezept_Container, auf den sich das Ticket bezieht.
- EF.StatusNotfalldaten: Diese Version ist identisch mit der CDM_Version der Notfalldatenstruktur in EF.eNotfalldaten

Weitere Informationen zur Länge sowie zum Typ, Format und Initialwert der Versionskennungen befindet sich im Dokument [gemeGK_Fach].

Da sich die Gültigkeitszeiträume der Anwendungsdatei-Versionen überschneiden können, MÜSSEN zugreifende Systeme (Konnektoren, Primärsysteme etc.) in der Lage sein, verschiedene Versionsstände zu verarbeiten. Die jeweils gültigen Versionsstände werden von der gematik festgelegt.

10 Versionierung von Fachanwendungen

Eine geeignete Klammer für die Zusammenfassung logisch zusammengehörender Änderungen bilden die Facharchitekturen, welche jeweils ein oder mehrere zusammengehörende technische Anwendungsfälle spezifizieren. Technische Anwendungsfälle (Use Cases) beschreiben die Schnittstellen, Abläufe und Datenstrukturen der verschiedenen Komponenten (Konnektor, Broker und Fachdienst), die für sie relevant sind. Fachanwendungen werden versioniert. Eine Version beschreibt die vollständigen, in sich konsistenten Anwendungsfälle einer Fachanwendung. Jede Version wird durch eine eindeutige Versionskennung, der Fachanwendungsversion (FA-Version), identifiziert. Das Format der Fachanwendungsversion MUSS mit den Festlegungen von Abschnitt 4.1 übereinstimmen. Dabei MÜSSEN die Bestandteile folgende Bedeutung haben:

- Die **Hauptversionsnummer** erhöht sich, falls aufgrund von Use Case-Änderungen signifikante Änderungen an Schnittstellen, Komponenten oder Datenstrukturen durchgeführt werden müssen.
- Die **Nebenversionsnummer** erhöht sich, falls Use Case-Änderungen moderate Änderungen zur Folge haben.
- Die **Revisionsnummer** erhöht sich, falls die Änderungen von geringer Bedeutung sind, z. B. bei Fehlerbereinigungen.

Über die FA-Version einer Fachanwendung wird eine Abbildung auf fachlich relevante und in diesem Dokument beschriebene Versionen festgelegt. Diese Abbildung ist Teil des fachlichen Architekturdokuments. Anhang A5 enthält als Vorgabe eine Tabelle mit den Namen und Versionskennungen, die von den fachlichen Architekturen spezifiziert werden MÜSSEN.

Anhang A – Erläuterungen

A1 – Zusammenfassung der Versionen

Der Abschnitt fasst die in den vorigen Kapiteln beschriebenen Versionskennungen zusammen und beschreibt ihre wichtigsten Merkmale.

Tabelle 2: Zusammenfassung der Versionen

Versionskennung	Ort	Bedeutung	Aufbau	Herleitung
Fachspezifische Schemaversion (FSV)	Version-Attribut im Schema-Element der fachspezifischen XSD-Datei	Versionierung von Änderungen auf Dateiebene	H.N.R	Hochzählung bei Änderung
Ticket-Schemaversion (TSV)	Version-Attribut im Schema-Element der XSD-Datei des Tickets	Versionierung von Änderungen auf Dateiebene	H.N.R	Hochzählung bei Änderung
Schnittstellen-spezifische Schemaversion (SSV)	Version-Attribut im Schema-Element der schnittstellen-spezifischen XSD-Datei	Versionierung von Änderungen auf Dateiebene	H.N.R	H = WV.H N = WV.N Hochzählung von R bei SS-neutralen Änderungen
Fachspezifische Namespace-Version (FNV)	TargetNamespace im Schema-Element der fachspezifischen XSD-Datei	Versionierung von Schema-änderungen; Ermittlung des Schemas zur Laufzeit	H.N	H = FSV.H N = FSV.N
Namespace-Version des Tickets (TNV)	TargetNamespace im Schema-Element der XSD-Datei des Tickets	Versionierung von Schema-änderungen; Ermittlung des Schemas zur Laufzeit	H.N	H = TSV.H N = TSV.N
Schnittstellen-spezifische Namespace-Version (SNV)	TargetNamespace im Schema-Element der schnittstellen-spezifischen XSD-Datei	Versionierung von Schnittstellen-änderungen; Ermittlung des Schemas zur Laufzeit	H.N	H = WNV.H N = WNV.N
WSDL-Version	documentation-	Versionierung von	H.N.R	Hochzählung bei

Versionskennung	Ort	Bedeutung	Aufbau	Herleitung
(WV)	Element der WSDL-Datei	Änderungen auf Dateiebene		Änderung der WSDL oder SS-relevanten Änderungen der zugehörigen XSD
WSDL Namespace-Version (WNV)	TargetNamespace im Definitions-Element der WSDL-Datei	Versionierung von Schnittstellen-änderungen des Konnektors	H.N	H = WV.H N = WV.N
Fachspezifische Logische Version (FLV)	CDM_VERSION-Attribut im Root-Knoten der fachspezifischen XSD-Datei	Versionierung fachlicher Datenstrukturen; Ermittlung des Schemas zur Laufzeit	H.N.R	Hochzählung bei Schema- und inhaltlichen Änderung
Logische Version des Tickets (TLV)	VERSION-Attribut im Root-Knoten der XSD-Datei des Tickets	Versionierung von Ticketänderungen; Ermittlung des Schemas zur Laufzeit	H.N.R	Hochzählung bei Schema- und inhaltlichen Änderung
Dienstversion (DV)	Fachliche Architektur (FAV); Konnektor-spezifikation; Service-Directory des Konnektors	Bestimmung des Konnektor-Endpunktes	H.N.R	Basisanwen-dung:Hochzäh-lung bei Änderungen; Fachanwendung: Identisch mit FAV
Interface-Version (IV)	Fachliche Architektur; Release-Plan	Versionierung der Telematik-Transport-Nachricht	H.N.R	Hochzählung bei Änderungen
Fachdienst-Version (FDV)	Fachliche Architektur; Release-Plan	Versionierung der Fachdienste und Ticketservices	H.N.R	Hochzählung bei Änderungen
SW-Komponenten-Version (SWV)	Intern in SW-Komponente	Versionierung von Code-Änderungen auf Komponenten-ebene	H.N.R. HID.HV	Hochzählung bei Spezifikations- und Code-Änderungen
eGK-Version (EGKV)	EF in zukünftiger eGK	Versionierung der auf der Karte umgesetzten eGK-Spezifikationen Teil1 und Teil2	H.N.R Teil1 H.N.R Teil2	Hochzählung bei Spezifikations-änderungen
Versionen der eGK-Anwendungs-dateien	EF.StatusVD, EF.StatusRezept, eRezept-Ticket, EF.Status-	Versionierung der auf der Karte befindlichen Anwendungs-	siehe [gem-eGK_-Fach]	siehe [gemeGK_Fach]

Versionskennung	Ort	Bedeutung	Aufbau	Herleitung
	Notfalldaten	dateien		
FA-Version (FAV)	Fachliche Architektur; Release-Plan	Versionierung von Fach-anwendungen	H.N.R	Hochzählung bei Änderungen

Legende:

SS: Schnittstelle; H: Hauptversionsnummer; N: Nebenversionsnummer; R: Revisionnummer; HID: Hersteller-ID; HV: Herstellerversion

Weiterhin gilt:

SSV.H = WV.H = WNV.H = SNV.H

SSV.N = WV.N = WNV.N = SNV.N

A2 – Erläuterungen zur Verwendung logischer Versionen bei fachspezifischen Datenstrukturen

Betrachtet man Änderungen an fachspezifischen Datenstrukturen genauer, ergeben sich Fälle, wo eine fachspezifische Datenstruktur und damit die zugehörige Schema-Datei unverändert bleiben, obwohl sich eine inhaltliche Änderung ergeben hat. Selbst wenn man zwangsweise eine neue Version der zugehörigen Schemadatei erzeugt, hat diese keinen anderen Inhalt, vorausgesetzt, dass keine weiteren gleichzeitigen Änderungen an den in der Schema-Datei definierten Strukturen vorgenommen wurden. Betrachtet wird daher ein informatives, fiktives Beispiel, das anhand eines Ausschnitts einer VOD-Datei erläutert wird:

In diesem Beispiel wird die Annahme gemacht, dass sich die zulässigen Werte des Feldes „Rechtskreis“ ändern. Dieses Datenfeld hat einen kodierten Eintrag - der Inhalt darf nur aus definierten Werten bestehen. Die zulässigen Werte werden im XML-Schema in diesem Fall nicht als Aufzählung (enumeration) definiert. Die Inhalte sind in einer externen Referenz hinterlegt und können sich ändern, ohne dass technische Komponenten dies prüfen könnten. Die Bedeutung dieser kodierten Inhalte wird nur in solchen Komponenten ausgewertet, die eine Verarbeitung dieser Daten aufgrund dieser Information vornehmen. Eine fachbezogene Verarbeitung ist somit in der Regel den Primärsystemen und den Fachdiensten vorbehalten.

Die für das Feld „Rechtskreis“ derzeit definierten Werte sind „1“ und „9“ und bezeichnen kodiert die Bedeutung „Ost“ bzw. „West“. Eine beispielhafte Änderung im Sinne einer Erweiterung könnte auf folgende Arten geschehen (weitere Möglichkeiten sind denkbar):

- Durch Hinzufügen eines Feldes, ohne die Bedeutung der bisherigen Werte zu verändern (z. B. durch „5“ bzw. „Helgoland“),
- durch Veränderung der den Werten zugehörigen Bedeutung (z. B.: „1“: „Ost“, „2“: „West“, „3“: „Nord“).

Die zugehörige Schemadefinition und das XML-Instanz-Dokument sind hier zur Verdeutlichung ausschnittsweise dargestellt. In beiden oben beschriebenen Fällen bleibt die Schemadatei unverändert.

XML-Instanz-Dokument	XML-Schemadatei
… <Versicherter> … <Zusatzinfos_GKV> … <Rechtskreis>1</Rechtskreis> … </Zusatzinfos_GKV> … <Versicherter> …	… <element name="ZusatzinfosGKV"> <complexType> <sequence> … <element name="Rechtskreis" type="vod:D0302_TS_Textstring_1"/> … </sequence> </complexType> </element> …

Würde man für diese inhaltsgleichen Schema-Dateien eine Änderung der Versionsnummer vornehmen, hätte dies weitreichende und unter Umständen unnötige Folgen:

- Die technischen Komponenten, also solche, die keine Interpretation sondern lediglich eine Schemavalidierung durchführen, müssten diese „neue“ Version verarbeiten können und eine Software-Aktualisierung vornehmen, um dieses Schema zu unterstützen, obwohl effektiv keine Änderung vorliegt.
- Für jede Änderung würde eine neue Schemaversion erzeugt, auch wenn keine Änderung am Schema selbst erkennbar ist. Versionsänderungen sind dann nicht immer echte Änderungen, was verwirrend und fehleranfällig ist.
- Systeme, die eine fachliche Interpretation der Daten vornehmen, müssten anhand der im Dokument enthaltenen Schemaversion auf die geänderte Fachlichkeit schließen.

Zur Unterscheidung der erweiterten bzw. geänderten Bedeutung des Datenfelds kann also die Version der Schemadatei nicht sinnvoll herangezogen werden. Daher wird aufgrund solcher fachlogischen Änderungen die so genannte „logische Versionsnummer“ verwendet. Immer dann, wenn sich Änderungen am Dateninhalt oder am Schema ergeben, die den Kommunikationspartnern beim Austausch von Daten bekannt sein müssen, wird die logische Version hochgezählt. Für jede logische Version einer fachspezifischen Datenstruktur wird zudem mit Hilfe einer Liste zugeordnet, mit welchem Schema sie definiert wird und validiert werden kann. Diese Liste der logischen Versionen und zugeordneten Schemaversionen ist Bestandteil der Facharchitekturen. Die Änderung der fachlichen Bedeutung ist dem zugehörigen Fachkonzept zu entnehmen.

Es ergeben sich zusammengefasst folgende Vorteile:

- Fachliche Änderungen wirken sich nicht immer auf technische Komponenten aus, die keine inhaltliche Verarbeitung der enthaltenen Daten vornehmen, aber technische Prüfungen (z. B. Schemavalidierung) implementieren. Technische Komponenten brauchen nicht aktualisiert zu werden, wenn keine Schemaänderungen vorgenommen wurden (z. B. der Konnektor für die Facharchitektur VOD).
- Die logische Version kann um mehrere zusammengehörende XML-Dateien bzw. deren zugehörige Schemadateien eine „logische“ Klammer bilden, die verschiedene Schema-Versionen der jeweiligen XSD-Dateien miteinander

verknüpft: Beim VSD kann dann beispielsweise die logische Version 1.2.3 die drei XSD-Dateien mit jeweils unterschiedlichen Versionen („allg.xsd“: 1.1.2, „per.xsd“: 1.1.0, „ges.xsd“: 1.0.1) bezeichnen. Das Primärsystem kann anhand der logischen Version, die im Status-Container EF.StatusVD abgelegt ist und mit übergeben wird, die Auswertung aller drei Dateien vornehmen.

A3 – Schemadefinitionen für die Selbstauskunft

Interface-Versionen

Das in Abbildung 1 dargestellte Schema wird verwendet, um die unterstützten Interface-Versionen einer Brokerinstanz zusammenzufassen. Es wird der Name der Broker-Instanz, das Abfragedatum sowie alle InterfaceVersionen erfasst, die der Broker zum Abfragezeitpunkt unterstützt.

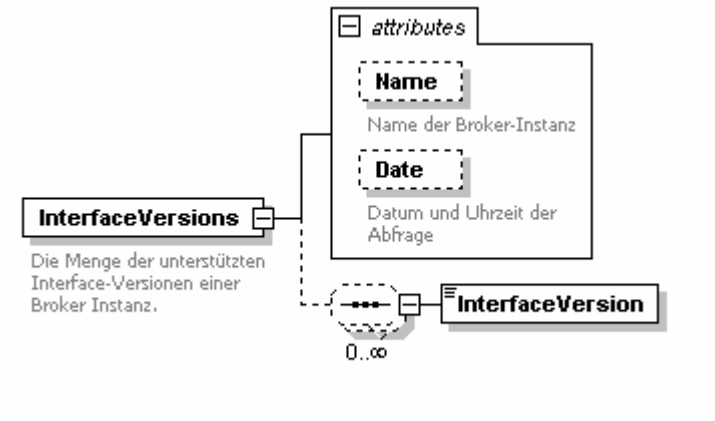

Abbildung 1: XML-Schema von InterfaceVersions

Fachdienstversionen

Abbildung 2 gibt das Schema für die Fachdienstversionen wieder, die ein Fachdienst zum Abfragezeitpunkt unterstützt. Neben dem Namen der Fachdienstinstanz und dem Abfragezeitpunkt werden die Namen und Versionsnummern aller unterstützten Fachdienst-Interfaces (Element Component) aufgeführt. Interfaces gibt es bisher eines für die fachliche Verarbeitung einer Nachricht und ggf. auch eines für die Abfrage von Tickets und für die Rechteverwaltung.

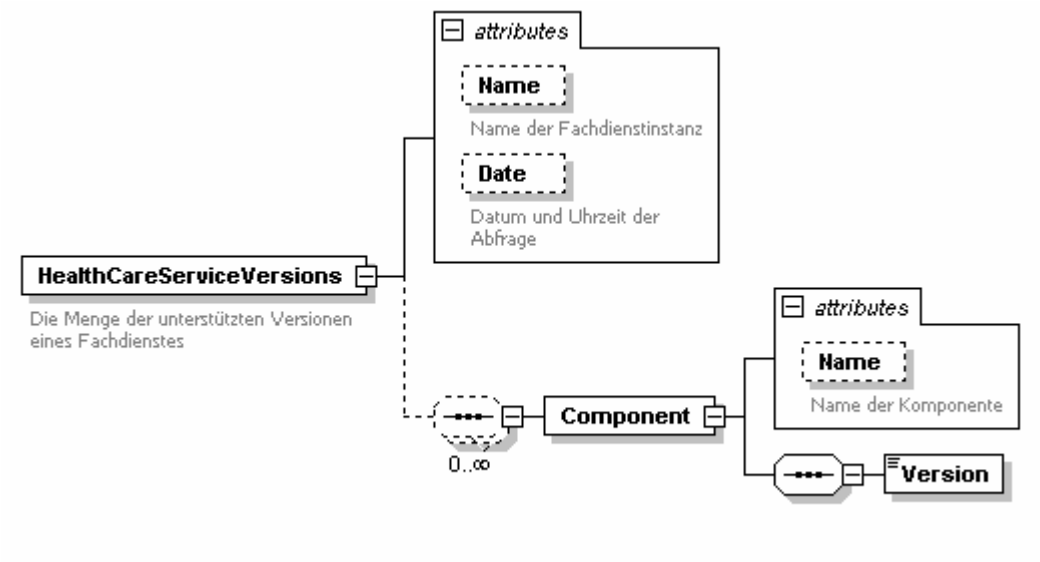

Abbildung 2: XML-Schema von HealthCareServiceVersions

Versionsnummer einer Software-Komponenete

Die folgende Abbildung veranschaulicht das XSD-Schema für die Rückgabe der Komponentenversion. Es wird im Wurzelknoten der Name, das Abfragedatum und die Version der abgefragten Komponente dokumentiert. Weiterhin erlaubt es die Dokumentation der Versionen aller Komponenten, von denen die Komponente abhängt. Die Versionskennung dieser Komponenten kann sowohl in Baum- als auch in Listendarstellung angeordnet werden.

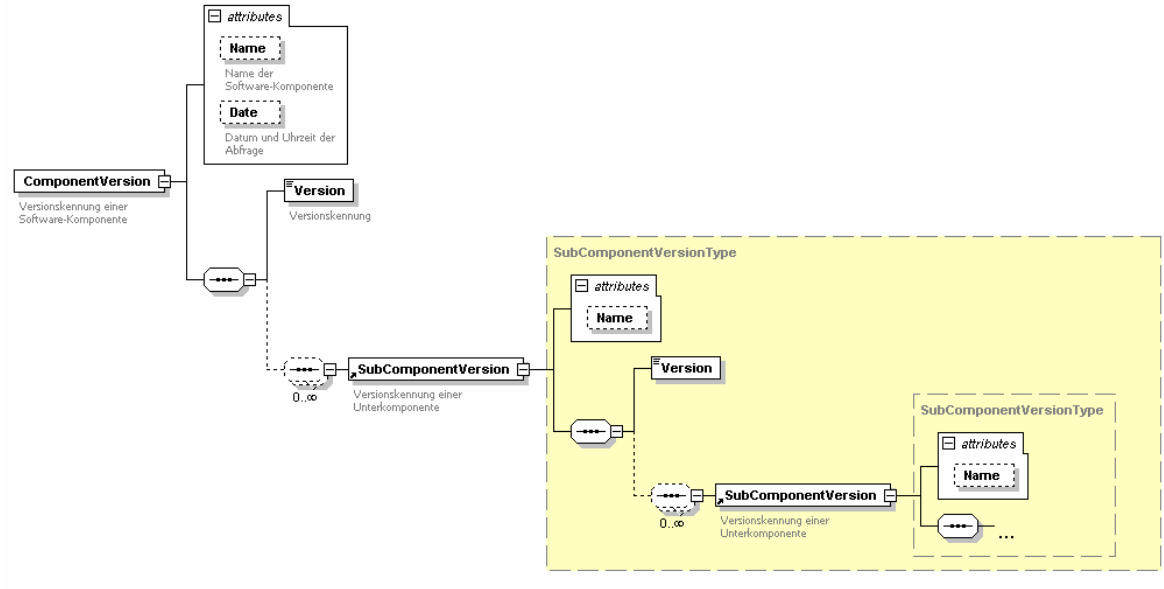

Abbildung 3: XML-Schema von ComponentVersion

A4 – Struktur von Namensräumen

Die Zielnamensräume der XSD- und WSDL-Dateien MUSS folgenden Aufbau haben:

http://<*Root*>/<*Physische Komponente*>/<*Dienstname*>/<*Dateiname*>/<*Version*>

Die Hierarchie der TargetNamespaces MUSS folgendermaßen festgelegt werden:

(1) Root: „ws.gematik.de“

(2) Physische Komponente: Die Komponente, die Dienste anbietet. Die zulässigen Bezeichnungen werden von der gematik festgelegt.

(3) Dienstname: Der Name des Dienstes, den die Komponente zur Verfügung stellt. Die zulässigen Bezeichnungen werden von der gematik festgelegt.

(4) Dateiname: Name der XSD- oder WSDL-Datei ohne Endung

(5) Version: Die Versionskennung ist **zweistellig** und hat den Aufbau v<*Hauptversionsnummer*>.<*Nebenversionsnummer*>

Nichtnormative Beispiele für Namespaces sind:

- http://ws.gematik.de/tel/transport/TelematikTransport/v1.0
- http://ws.gematik.de/conn/vsds/VSDService/v2.1
- http://ws.gematik.de/cm/cam/CmCamServiceRequest/v1.1
- http://ws.gematik.de/fa/vsds/UC_geschuetzteVersichertendatenXML/v2.0

A5 – Versionen einer Fachanwendung

Die Facharchitekturen definieren die Abbildung der Version einer Fachanwendung (FA-Version) auf die zu verwendenden Versionen der Artefakte, die an den Schnittstellen zum Konnektor, Broker und Fachdienst zu verwenden sind.

Bei den Versionsnummern werden folgende drei Kategorien unterschieden:

- Versionsnummern, die von den Facharchitekturen festgelegt werden und sich auf eine Fachanwendung insgesamt beziehen;
- Versionsnummern, die von den Facharchitekturen festgelegt werden und sich auf einzelne Anwendungsfälle einer Facharchitektur beziehen;
- Versionsnummern, die außerhalb der Facharchitekturen festgelegt werden, auf die sich aber die Facharchitekturen beziehen.

Für die einzelnen Kategorien der Versionsnummern gibt es unterschiedliche Tabellen.

Die folgende Tabelle beschreibt die Namen und Versionen der Artekfakte einer Fachanwendung, die von den Facharchitekturen festgelegt werden MÜSSEN:

Tabelle 3: Fachanwendungsspezifische Versionskennungen

NR	PARAMETER	WERT
Fachanwendung		
	FA-Name	
	FA-Version	
Fachliche Payloads		
	Schemaname	
	Schemaversion	
	TargetNamespace des Schemas	
	Logische Version (CDM_VERSION)	
	hier ggf. weitere fachliche Payloads	
Primärsystem/Konnektor-Schnittstelle		
	WSDL-Name	
	WSDL-Version	
	WSDL-TargetNamespace	
	Schemaname	
	Schemaversion	
	TargetNamespace des Schemas	
Konnektor/Broker-Schnittstelle (FA-abhängig)		
	Schemaname	
	Schemaversion	
	TargetNamespace des Schemas	
	Schemaname	
	Schemaversion	
	TargetNamespace des Schemas	
eGK		

NR	PARAMETER	WERT
	<Version eGK-Anwendungsdatei>	
Fachdienst-Schnittstelle (einschl. Ticket-Service)		
	Fachdienstversion	
	Fachdienstname	
Software-Komponenten		
	Fachmodul des Konnektors	
	Fachdienst	
	Ticket-Service	

Pro Anwendungsfall (Use Case) einer Fachanwendung ist eine Tabelle mit folgendem Inhalt zu erstellen:

Tabelle 4: Anwendungsfallspezifische Versionskennungen

NR	PARAMETER	WERT
Technischer Use Case		
	Techn. Use Case Name	
Primärsystem/Konnektor-Schnittstelle		
	Name der Operation	
Broker/TelematikTransport-Schnittstelle		
	Brokersequenz Name	
Fachdienst-Schnittstelle (einschl. Ticket-Service)		
	Fachdienstoperation	

Die nächste Tabelle beschreibt die Namen und Versionen der Artekfakte, auf die sich eine Facharchitektur bezieht, die aber an anderer Stelle festgelegt werden:

Tabelle 5: Allgemeine Versionskennungen

NR	PARAMETER	WERT
Fachanwendung		
	FA-Name	
	FA-Version	
Ticket		
	Schemaname	
	Schemaversion	
	TargetNamespace des Schemas	
	Logische Version (VERSION)	
Konnektor/Broker-Schnittstelle (FA-übergreifend)		
	Schemaname	z.B. CmCcServiceRequest.xsd
	Schemaversion	
	TargetNamespace des Schemas	
	Schemaname	z.B. CmCcServiceResponse.xsd
	Schemaversion	
	TargetNamespace des Schemas	
Broker/TelematikTransport-Schnittstelle		
	Interface Version der TelematikTransport-Nachricht	
eGK		
	eGK-Version	

Die Dienstversion des Endpunktes ist bei Fachanwendungen immer mit der FA-Version identisch.

A6 – Software-Komponentennamen

Wie in Abschnitt 8.2 beschrieben wurde, werden die der Selbstauskunft unterliegenden Software-Komponenten einschließlich ihrer Namen von der gematik vorgegeben. Dieser Abschnitt gibt einen Überblick der Komponenten der TI und ihrer Namen, die bei der Selbstauskunft verwendet werden MÜSSEN.

Komponenten des Konnektors

Der Konnektor besteht aus den Hauptkomponenten Netzkonnektor, Trusted Viewer und Anwendungskonnektor. Im Anwendungskonnektor enthalten sind die Teilkomponenten der Fachmodule des Konnektors (VSDM, VODM, NFDM)[5]. Die Selbstauskunft des Anwendungskonnektors gibt neben der eigenen Versionsnummer (ComponentVersion, siehe Anhang A3) auch die Versionsnummern aller Teilkomponenten (SubComponentVersion, siehe Anhang A3) zurück. Die folgende Tabelle listet die Namen der einzelnen Komponenten auf.

Tabelle 6: Komponenten des Konnektors

Komponente	Name	Bedeutung
Netzkonnektor	netconn	Der Netzkonnektor bildet u.a. die Funktionalität eines VPN Clients ab.
Trusted Viewer	tv	
Anwendungs-konnektor	appconn	Der Anwendungskonnektor bildet u.a. die Anwendungsfälle der Fachanwendungen ab.
Fachmodul VSDM	vsdm	Die Spezifikation erfolgt in der Facharchitektur
Fachmodul VODM	vodm	Die Spezifikation erfolgt in der Facharchitektur
Fachmodul NFDM	nfdm	Die Spezifikation erfolgt in der Facharchitektur

Die Komponentennamen der restlichen Bestandteile der TI werden in einer späteren Version des Dokuments ergänzt.

[5] Die Basisanwendungen des Konnektors sind Teil des Anwendungskonnektors und werden deshalb nicht einzeln versioniert.

Anhang B

B1 – Abkürzungen

Kürzel	Erläuterung
CDM_Version	„Corresponding Data Modell“, zugehöriges Datenmodell
eGK	elektronische Gesundheitskarte
FAV	Fachanwendungsversion
GVD	Geschützte Versichertendaten
HBA	Heilberufsausweis
NFDM	Notfalldatenmanagement
PD	Persönliche Daten
SMC	Security Module Card
SOA	Serviceorientierte Architektur
TI	Telematikinfrastruktur
TUC	Technischer Use Case
VD	Versichertendaten
VODM	Verordnungsdatenmanagement
VSDM	Versichertenstammdatenmanagement
WSDL	WebService Description Language
XML	eXtensible Markup Language
XSD	XML Schema Definition

B2 – Glossar

Das Projektglossar wird als eigenständiges Dokument zur Verfügung gestellt.

B3 – Abbildungsverzeichnis

B4 – Tabellenverzeichnis

B5 – Referenzierte Dokumente

[Quelle]	Herausgeber (Erscheinungsdatum): Titel
[gemGA_Wartg]	gematik (04.05.2007): Einführung der Gesundheitskarte - Anforderungen an die Komponenten zur Unterstützung der Wartungs- und Änderungsprozesse Version 0.9.0, www.gematik.de
[gemGesArch]	gematik (15.05.2007): Einführung der Gesundheitskarte - Gesamtarchitektur, Version 1.0.0, www.gematik.de
[gemeGK_Fach]	gematik (04.05.2007): Einführung der Gesundheitskarte - Speicherstrukturen der eGK für Gesundheitsanwendungen Version 1.1.0
[gemSpec_eGK_P1]	gematik (07.02.2006): Einführung der Gesundheitskarte – Die Spezifikation elektronische Gesundheitskarte; Teil 1 – Kommandos, Algorithmen und Funktionen des Kartenbetriebssystems Version 1.1.0, www.gematik.de
[gemSpec_eGK_P2]	gematik (15.05.2007): Einführung der Gesundheitskarte – Die Spezifikation elektronische Gesundheitskarte ; Teil 2 – Anwendungen und anwendungsspezifische Strukturen Version 1.3.0, www.gematik.de
[gemSpec_Kon]	gematik (04.05.2007): Einführung der Gesundheitskarte - Konnektorspezifikation Version 2.0.0, www.gematik.de
[gemSpec_TTD]	gematik (04.05.2007): Einführung der Gesundheitskarte - Spezifikation TelematikTransport-Details Version 1.1.0, www.gematik.de
[gemSysSLM]	gematik (04.05.2007): Einführung der Gesundheitskarte – Spezifikation System und Service Level Monitoring, Version 1.1.0

[Quelle]	Herausgeber (Erscheinungsdatum): Titel
[RFC2119]	RFC 2119 (März 1997): Key words for use in RFCs to Indicate Requirement Levels S. Bradner, http://www.ietf.org/rfc/rfc2119.txt (zuletzt geprüft am 14.12.2006)
[RVO2006]	Bundesgesetzblatt I (2006) vom 05.10.2006, Seite 2199 ff.: Neufassung der Verordnung über Testmaßnahmen für die Einführung der elektronischen Gesundheitskarte
[WSDL1.1]	Eric Christensen et. al. (2001): Webservices Description Language (WSDL) 1.1

B6 – Klärungsbedarf

Das Format und der Inhalt der Konfigurationstabelle aus Anhang A5 sind derzeit noch in Abstimmung. Änderungen können sich daher noch ergeben.

Anhang C

Dieses Kapitel fasst die allgemeinen Anforderungen aus Kapitel 3 zusammen. Die Anforderungen der übrigen Kapitel beziehen sich hauptsächlich auf das Format und die Bedeutung der Versionskennungen sowie auf die Ausgestaltung der Selbstauskunft. Sie werden an dieser Stelle nicht nochmals wiederholt.

Tabelle 7: Allgemeine Anforderungen

AnforderungsID	Bezeichnung	Anford.-Level	Beschreibung
AF-VN-00-0001	Versionsinformation im WebServices	MUSS	Eine Versionsinformation MUSS in jeder veröffentlichten Datei, die ganz oder teilweise eine WebService-Schnittstelle beschreibt, enthalten sein.
AF-VN-00-0002	Jede Änderung führt zu neuer Version	MUSS	Alle Änderungen an Schemadateien und WSDL-Dateien MÜSSEN zu neuen Versionen führen, auch wenn die Änderungen aufwärts- bzw. abwärtskompatibel sind. Dadurch wird eine korrekte Dokumentation und Referenzierung sowie eine Validierung der erzeugten XML-Datenstrukturen ermöglicht
AF-VN-00-0003	Inhaltliche Versionsangabe	MUSS	Die in der TI transportierten XML-Daten der Fachanwendungen werden durch Schemadateien unabhängig vom Transport strukturell und inhaltlich beschrieben. Unabhängig von der Struktur kann sich die inhaltliche Bedeutung eines Datenfeldes (z. B. die Kodierung) ändern, ohne dass es zu einer strukturellen Änderung kommt. Anwendungsspezifische XML-Instanzdokumente MÜSSEN Auskunft über ihre inhaltliche Version geben können. Diese Version ist unterschiedlich zur Schema¬version der zugehörigen XSD-Datei. Dazu MUSS in jeder anwendungsspezifischen XSD-Datei ein Attribut im Wurzel-Element definiert werden, das diese inhaltliche Versionsinformation enthält
AF-VN-00-0004	Versionsinformation in Software-Komponenten	MUSS	Software-Komponenten, die die von der gematik spezifizierten Funktionalitäten realisieren, MÜSSEN Versionsinformationen besitzen und diese geeignet zurückgeben können.
AF-VN-00-0005	Selbstauskunft zu Schnittstellenversionen	MUSS	Software-Komponenten, die versionierte Schnittstellen anbieten, MÜSSEN in der Lage sein, Auskunft zu den aktuell unterstützten Schnittstellenversionen geben zu können.
AF-VN-00-0006	Version der eGK	MUSS	Die eGK muss eine eigene Versionskennung besitzen (unabhängig von weiteren auf der eGK abgelegten Versionskennungen), mit der ermittelbar ist, welche eGK-Spezifikation die jeweilige Karte umsetzt.