IQTIG

Institut für Qualitätssicherung und Transparenz im Gesundheitswesen

Technische Dokumentation zu PB-Spezifikationen für Leistungserbringer

Erfassungsjahr 2021

Erstellt im Auftrag des
Gemeinsamen Bundesausschusses

Stand: 30. Oktober 2020

Impressum

Thema:
Technische Dokumentation zu PB-Spezifikationen. Erfassungsjahr 2021

Auftraggeber:
Gemeinsamer Bundesausschuss

Datum der Veröffentlichung:
30. Oktober 2020

Herausgeber:
IQTIG – Institut für Qualitätssicherung
und Transparenz im Gesundheitswesen

Katharina-Heinroth-Ufer 1
10787 Berlin

Telefon: (030) 58 58 26-0
Telefax: (030) 58 58 26-999

info@iqtig.org
https://www.iqtig.org

Hinweis:
Aus Gründen der leichteren Lesbarkeit wird im Folgenden auf eine geschlechtsspezifische Differenzierung verzichtet. Entsprechende Begriffe gelten im Sinne der Gleichbehandlung für alle Geschlechter.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

Abkürzung	Bedeutung
AES	Advanced Encryption Standard (Verschlüsselungsalgorithmus)
AIS	Arztinformationssystem
AG	Arbeitsgruppe
ASCII	American Standard Code for Information Interchange (Amerikanischer Standard-Code für den Informationsaustausch)
BAS	Auswertungsstelle
BE	Bundesebene
BSI	Bundesamt für Sicherheit in der Informationstechnik
BSNR	Betriebsstättennummer
CSV	Comma-separated values (Dateiformat)
DAS	Datenannahmestelle
DB	Datenbank
dv	direkte Verfahren
DMP	Disease-Management-Programm
DK	Programm zur Früherkennung von Darmkrebs
DPP	Datenprüfprogramm
DRG	Diagnosis Related Groups (diagnosebezogene Fallgruppen)
DÜV	Datenübermittlungsvereinbarung
EBM	Einheitlicher Bewertungsmaßstab
eGK	elektronische Gesundheitskarte
G-BA	Gemeinsamer Bundesausschuss
GKV	Gesetzliche Krankenversicherung
GOP	Gebührenordnungsposition
GUID	Globally Unique Identifier
HTML	Hypertext Markup Language (Hypertext-Auszeichnungssprache)
ID	Identifikationsnummer
IK	Institutionskennzeichen
IKNR	Institutionskennzeichennummer

Abkürzung	Bedeutung
IQTIG	Institut für Qualitätssicherung und Transparenz im Gesundheitswesen
IV	integrierte Versorgung
iv	indirekte Verfahren
JRE	Java Runtime Environment (Java-Laufzeit-Umgebung)
JVM	Java Virtual Machine, ist Teil der Java-Laufzeitumgebung
K	Kann-Feld
KBV	Kassenärztliche Bundesvereinigung
oKFE-RL	Richtlinie für organisierte Krebsfrüherkennungsprogramme
KV	Kassenärztliche Vereinigung
KVDT	Kassenärztliche Vereinigung-Datentransfer (Datenformat)
LANR	Lebenslange Arztnummer
LE	Leistungserbringer
LE-amb	Leistungserbringer ambulant
LID	Leistungserbringeridentifizierenden Daten
LIS	Laborinformationssystem
M	Muss-Feld
MDS	Minimaldatensatz
OR	ODER-Operator
PB	Programmbeurteilung
PBDOK	Access-Datenbank, in der die PB-Dokumentation spezifiziert wird
PBF	Access-Datenbank, in der der PB-Filter spezifiziert wird
PID	Patientenidentifizierende Daten
PlanQI	planungsrelevante Qualitätsindikatoren
PR	Arztpraxis
QS	Qualitätssicherung
RL	Richtlinie
RSA	Verfahren zur Datenverschlüsselung, entwickelt von R. Rivest, A. Shamir und L. Adleman
SGB	Sozialgesetzbuch
SGB V	Sozialgesetzbuch Fünftes Buch

Abkürzung	Bedeutung
SNK	sichere Netze der KVen
SWA	Softwareanbieter
TB	Testbetrieb
TDS	Teildatensatz
TPacker	Programm für die Transportverschlüsselung
URL	Uniform Resource Locator (einheitlicher Ressourcenzeiger)
V	Versionierung
VST	Vertrauensstelle
VST-DAS	Vertrauensstelle des G-BA in der Funktion als Datenannahmestelle
VST-PSN	Vertrauensstelle des G-BA in der Funktion als Pseudonymisierungsstelle
XML	Extensible Markup Language
XPacker	Verschlüsselungsprogramm
XSD	XML-Schema-Datei
XSLT	Extensible Stylesheet Language Transformation (Programmiersprache zur Transformation von XML-Dokumenten)
ZIP	zipper, Abkürzung für ein Format für verlustfrei komprimierte Dateien
ZK	Programm zur Früherkennung von Zervixkarzinomen

Änderungsindex

Änderungen der Datenbanken im Vergleich zur Vorversion lassen sich anhand der Delta-Tabellen nachvollziehen.

Kapitelübergreifende Änderungen:

- Konkretisierungen und Optimierung von Formulierungen
- Anpassung von Jahreszahlen, Beispielen, Abbildungen und Tabellen
- Korrektur von Fehlern und Ergänzung von fehlenden Inhalten
- Anpassung von Abkürzungen

Konkrete Informationen zu den inhaltlichen Änderungen sind der Spezifikationskomponente Uebersicht_Aenderungen bzw. den aktuellen Beschlüssen des G-BA zu entnehmen.

Die spezifischen Änderungen der vorliegenden technischen Dokumentation werden im Folgenden mit Bezug zur jeweiligen Version dargestellt.

Änderung	**Kapitel/Abschnitt**	**Version**
Streichung des Hinweises zum ausstehenden Beschluss zur Bestimmung der Auswertungsstelle	1	2021 V01
Aufnahme eines Hinweises in Bezug auf den dokumentationspflichtigen Leistungserbringer	A 1.1	**2021 V02**
Anpassung des Absatzes zur manuellen Auslösung der PB-Module in Bezug auf die beschlossenen EBM-Ziffern	A 1.1	2021 V01
Anpassung des Absatzes zu den Besonderheiten beim Modul DKK in Bezug auf die teilautomatische Auslösung	A 1.1.1	2021 V01
Aufnahme eines Hinweises zum direkten Export bereits vorliegender Daten	A 1.2	**2021 V02**
Anpassung der Erläuterung zur Verwendung des öffentlichen Schlüssels der BAS	A 1.3.3	**2021 V02**
Ergänzung der Informationen zur Veröffentlichung der öffentlichen Schlüssel in dem separaten Spezifikationspaket der Datenserviceinformationen	A 1.3.3	2021 V01
Streichung des Hinweises zum ausstehenden Beschluss zur Bestimmung der Auswertungsstelle	A 1.3.3	2021 V01
Hinweis, dass ab der Spezifikation 2021 das Pseudonymisierungsprogramm und die Verschlüsselungsprogramme nicht mehr als Spezifikationskomponente sondern losgelöst von der Spezifikation veröffentlicht werden.	B „Komponeten“	2021 V01

Änderung	**Kapitel/Abschnitt**	**Version**
Anpassung des Absatzes zur manuellen Auslösung der PB-Module in Bezug auf die beschlossenen EBM-Ziffern	B 1	2021 V01
Löschen der Attribute direkt, indirekt, pid, oKFE, fkPseudonymVerfahren aus der Tabelle Modul.	B 2.3.1	2021 V01
Ergänzung von Vorgaben zur Vergabe von technischen Feldnamen	B 2.3.3	2021 V01
Streichung des Passus zu den nicht vorhandenen Ausfüllhinweisen	B 2.3.5	2021 V01
Hinzufügen der Attribute direkt, indirekt, pid, oKFE, fkPseudonymVerfahren in die Tabelle Exportmodul.	B 2.5.1	2021 V01
Ergänzung eines Hinweises zur Abfrage `vPruefung`	B 2.7	2021 V01
Dummy-Wert „000“ für //software/vendor/@registration	B 3.4.3	**2021 V02**
Ergänzung eines Hinweises zur Löschung der `qs_data` in der VST bei vorliegendem Widerspruch	B 3.4.4	2021 V01
Ergänzung der NBSNRAMBULANT in Tabelle 78	B 3.4.4	2021 V01
Streichung des Kind-Elementes <perineo_pid>	B 3.4.4	**2021 V02**
Aktualisierung des Kapitels an die aktuellen Regelungen der Basisspezifikation	B 4	2021 V01

Leseanleitung

Diese technische Dokumentation orientiert sich in ihrem Aufbau an den Abläufen der Erfassung und Übermittelung der erforderlichen Daten. Ziel dieser Struktur ist es, eine nachvollziehbare und logische Sicht auf die Umsetzung und Durchführung der beschriebenen Schritte zu gewährleisten. Die Prozesse und Unterprozesse werden im Abschnitt A beschrieben und spiegeln die reale, chronologische Abfolge wider. Jede Prozessbeschreibung berücksichtigt zudem die unterschiedlichen Komponenten, die für die Umsetzung benötigt und in Abschnitt B detailliert beschrieben werden. Im Abschnitt C wird ein Glossar mit den wichtigen Begriffen zum Themenbereich der Spezifikation zur Verfügung gestellt.

Für eine korrekte Umsetzung der Spezifikationen ist es notwendig, die Dokumentation entsprechend ihrer Anordnung von Prozessen zu Komponenten zu befolgen. Einige Bereiche, die sich ausschließlich an bestimmte Zielgruppen richten, sind entsprechend gekennzeichnet.

Legende

Die in dieser Dokumentation verwendeten Symbole heben bestimmte Aspekte bei der Umsetzung der Spezifikationen hervor.

Achtung

Beschreibt Ursache, Folge und Vermeidung einer besonderen Fehlanwendung, die zu Problemen bei der Implementierung oder Ähnlichem führen kann.

Hinweis

Nützliche Informationen, Tipps oder Ratschläge zur Anwendung. Keine wesentlichen oder für das korrekte Funktionieren erforderlichen Informationen.

Beispiel:
Beispiele sind ein Hilfsmittel, um zuvor vermittelte Informationen oder konkrete Abschnitte der Anwendung zu verdeutlichen.

1 Einleitung

Die technische Dokumentation für Datenannahmestellen beschreibt die Spezifikationen zur Programmbeurteilung (PB) für Leistungserbringer gemäß Richtlinie für organisierte Krebsfrüherkennungsprogramme (oKFE-RL) und richtet sich dabei an die Datenannahmestellen.

Die PB-Spezifikationen umfassen alle Komponenten im Zusammenhang mit der Datenerfassung, d. h. von der Bestimmung einer Dokumentationspflicht bis hin zur Rückprotokollierung übermittelter Datensätze. Die vorliegende technische Dokumentation beschreibt die für die Datenannahmestellen relevanten Prozesse und Komponenten.

Regelungsbereich der PB-Spezifikationen sind die verschiedenen Verfahren gemäß der Richtlinie für organisierte Krebsfrüherkennungsprogramme (oKFE-RL), die vom Gemeinsamen Bundesausschuss (G-BA) beschlossen werden.

Für eine Nutzung von Sozialdaten bei den Krankenkassen gemäß § 299 SGB V wird jeweils eine unabhängige Spezifikation veröffentlicht. Für eine spezifikationskonforme Datenannahme, -übermittlung und Rückprotokollierung sind alle Spezifikationskomponenten zu berücksichtigen.

1.1 Spezifikationsbegriff

Die jeweilige Spezifikation ist die Gesamtheit aller Vorgaben, nach denen die Bestimmung der dokumentationspflichtigen Fälle, die Dokumentation zur Programmbeurteilung selbst sowie die Übermittlung der Daten bezogen auf ein Erfassungsjahr erfolgen sollen. Die Zuordnung eines Falles zu einer Spezifikation richtet sich bei ambulanten Fällen nach dem Kalenderjahr des Behandlungsdatums der gemäß Richtlinie durchgeführten Untersuchungen.

Um die komplexen Anforderungen an die PB-Dokumentation sowie die zugehörigen Datenflüsse zu erfüllen, bestehen die PB-Spezifikationen aus verschiedenen Komponenten, die je nach Anwender spezifisch zusammengestellt werden. Als Komponenten werden dabei Access-Datenbanken, technische Dokumentationen, Ausfüllhinweise und anderes bezeichnet. Jeder Anwender bekommt damit das für ihn Relevante in einem eigenen Spezifikationspaket als Download zur Verfügung gestellt. Jedes dieser Pakete kann auf diese Weise auch unabhängig von den anderen aktualisiert werden.

Damit gibt es ein Spezifikationspaket für

- den Regelbetrieb (oKFE)
- ggf. Sonderexporte
- ggf. Probebetriebe
- ggf. Testbetriebe

Sowohl die Spezifikationspakete als auch die einzelnen Komponenten werden nach einem einheitlichen Schema benannt, das bereits im Namen übersichtlich die relevanten Informationen wie Betriebsart, Exportformat und Versionierung enthält. Dieses Schema wird im nächsten Abschnitt erläutert. Durch die Versionierung sowohl auf der Ebene der Pakete als auch auf der

Ebene der Komponenten ist gewährleistet, dass der aktuelle Stand leicht ersichtlich ist. Zudem wird die Kommunikation über die anzuwendenden Bestandteile der Spezifikation erleichtert.

Jedem Paket liegt eine Auflistung der einzelnen Komponenten und ggf. eine Übersicht über die Änderungen zur vorhergehenden Version bei.

1.1.1 Benennungsschema für Spezifikationspakete

Die Benennung der Spezifikationspakete setzt sich wie folgt zusammen:

`<Erfassungsjahr>_<Richtlinie>_<Name>_[<DAS>]<Betriebsart>_<Exportformat>_V<Versionsnummer>`

Das Erfassungsjahr gilt für alle Spezifikationspakete und -komponenten, die Daten dieses Erfassungsjahres betreffen, egal in welchem Jahr das jeweilige Paket veröffentlicht wurde.

Bei der Angabe `<Name>` kann der die Spezifikation kennzeichnende Name angegeben werden. Namen können beispielsweise (verfahrensübergreifend bzw. programmunabhängig) wie folgt definiert werden:

- `PBDOK:` fallbezogene PB-Dokumentation
- `FDOK:` fallbezogene QS-Dokumentation
- `EDOK:` einrichtungsbezogene QS-Dokumentation
- `SDOK:` Spezifikation für Strukturabfragen
- `PBDOK:` Spezifikation für die Programmbeurteilung
- `SozDat:` Nutzung von Sozialdaten bei den Krankenkassen

Bei der optionalen Angabe `[<DAS>]` kann beispielsweise zwischen folgenden Kürzeln unterschieden werden:

- `LKG:` LQS/LKG/LAG[2]
- `KV:` DAS-KV
- `KK:` DAS-KK

Bei der Betriebsart kann zwischen folgenden Kürzeln unterschieden werden:

- `RB`: Regelbetrieb
- `SE`: Sonderexport
- `PB`: Probebetrieb
- `TB`: Testbetrieb

`V<Versionsnummer>`: Die Versionierung erfolgt in ganzen Zahlen, die zweistellig angegeben sind (unter 10 mit einer vorlaufenden 0, z. B. V01). Davon abweichend können Alphaversionen für eine Zurverfügungstellung vorab in Vorbereitung auf gemeinsame Abstimmungsprozesse mit dem Zusatz „_Alpha“ versehen werden. Bei jeder Erhöhung des Erfassungsjahres beginnt die Versionierung wieder bei V01.

[2] Die Angabe LKG wird in der folgenden Dokumentation für die jeweils zuständige Datenannahmestelle des Landes verwendet. Das Kürzel schließt daher neben der LKG auch die LQS und/oder LAG ein. Aus technischen Gründen kann z.B. in Dateinamen auch noch das Kürzel „lqs“ oder „LQS“ Anwendung finden.

Beispiele:
Als erstes planmäßiges Release wird folgendes Paket für die fallbezogenen PB-Dokumentation veröffentlicht:

`2021_oKFE_PBDOK_RB_XML_V01`

Im Regelfall wird im September das Paket mit der Version 02 für die fallbezogenen PB-Dokumentation veröffentlicht.[3]

Ausformuliert bezeichnet dies die Spezifikation zur fallbezogenen PB-Dokumentation für den Regelbetrieb des Erfassungsjahres 2021. Das Format für die zu exportierenden und von den beteiligten Stellen (DAS, VST) zu übermittelnden Datensätze der PB-Dokumentation ist XML.

Beispiele:
Neben der fallbezogenen PB-Dokumentation können beispielsweise folgende Spezifikationspakete veröffentlicht werden.

Die Spezifikation für die Nutzung der Sozialdaten bei den Krankenkassen:

`2021_oKFE_SozDat_KK_RB_XML_V01`

1.1.2 Benennungsschema für Spezifikationskomponenten

Die Benennung der Spezifikationskomponenten lehnt sich an das bei den Spezifikationspaketen verwendete Prinzip an:
`[<Erfassungsjahr>_]<Art der Komponente>_[<Exportformat>_][V<Versionsnummer>].<Dateierweiterung>`

„Art der Komponente“ bezieht sich auf die jeweilige Funktion und wird durch ein Kürzel angegeben.

Die in eckige Klammern gesetzten Benennungsbestandteile sind optional und werden nur dann gesetzt, wenn dieses Merkmal für die Komponenten von Relevanz ist. Das Erfassungsjahr ist für Hilfsprogramme wie z. B. den XPacker nicht relevant; bei diesen erfolgt die Versionierung mit einer Versionsnummer, die über die Jahre fortgeführt wird.

Die Bezeichnungen der einzelnen Komponenten werden in Abschnitt B auf Seite 73 erläutert.

Die folgende Abbildung dient der Veranschaulichung des Benennungsschemas. Die abgebildeten Daten sind beispielhaft und bilden nur einen Teil der Pakete und Komponenten in Bezug auf die Spezifikationen 2020 zur PB-Dokumentation ab. Die gesamte Spezifikation 2020 ist durch einen dunkelgrünen Rahmen begrenzt. Die grünen Kästen innerhalb sind die Spezifikationspakete, die

[3] Für das Erfassungsjahr 2021 wird außerplanmäßig eine Version 02 bereits im Juli veröffentlicht. Im Regelfall erfolgt die Bereitstellung jedoch im September.

je nach Anwender zusammengestellt werden. Innerhalb der Pakete sind die Spezifikationskomponenten aufgelistet. In deren Bezeichnungen ist hier jeweils rot markiert, wenn sich etwas von Paket zu Paket geändert hat.

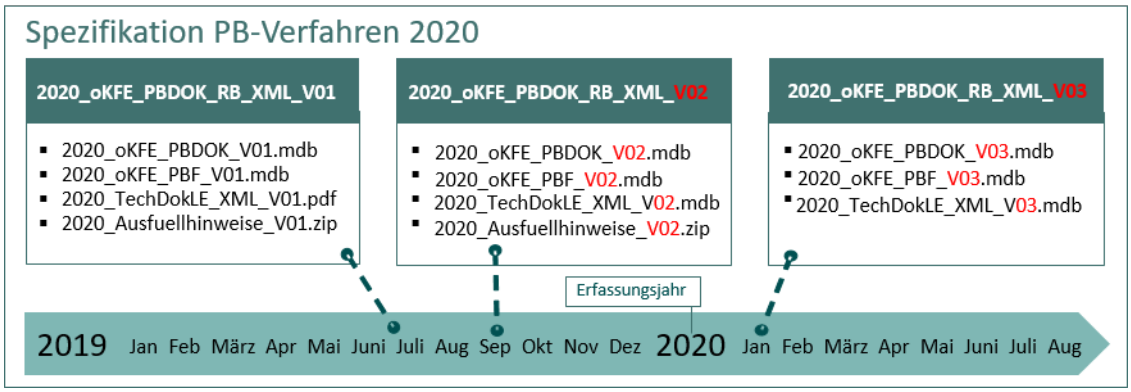


Abbildung 1: Beispiel für Benennung von Paketen und Komponenten der PB-Verfahren für die Spezifikation 2020

1.2 Zielsetzung und Zielgruppe

Die PB-Spezifikation ist ein komplexes Regelwerk, das mithilfe verschiedener Komponenten verbindliche Grundlagen für alle Prozesse im Zusammenhang mit der Erfassung und Übermittlung von PB-Daten bei den unterschiedlichen Verfahrensteilnehmern (Leistungserbringer, Datenannahmestellen, Vertrauensstelle) vorgibt und beschreibt. Die Komponenten der Spezifikation sind daher so ausgestaltet, dass sie von IT-/EDV-Expertinnen und -Experten verstanden werden. Die Spezifikation richtet sich ausschließlich an diesen Teilnehmerkreis. Die Regelung und die Art der Darlegung der Spezifikationskomponenten sind auf eine möglichst automatisierte Nutzung durch diesen Personenkreis ausgerichtet.

Die Spezifikationen zur Programmbeurteilung stellen eine Reihe von Anforderungen an die Datenerhebung, Datenerfassung und Plausibilitätsprüfung, um valide, reliable und vergleichbare Daten gewinnen zu können. Die Erfassung und Plausibilitätsprüfung durch unterschiedliche Softwareumsetzungen beinhaltet grundsätzlich die Gefahr einer Verzerrung der Daten. Die Vorgaben der Spezifikationen, die eine einheitliche Festlegung von Datenfeldbeschreibungen, Plausibilitätsregeln, Grundsätzen der Benutzerschnittstellengestaltung und Datenübermittlungsformaten umfassen, sollen dazu dienen, dieser Gefahr entgegenzuwirken. Dadurch werden die Erhebung valider und vergleichbarer Daten sowie ein unter datenschutzrechtlichen Gesichtspunkten sicherer Datenfluss gewährleistet.

Die vorliegende technische Dokumentation erklärt einleitend, worauf sich die Spezifikation bezieht, wie das Benennungsschema aufgebaut ist und welche Veröffentlichungstermine und Fristen für Rückmeldungen zur Basisspezifikation bestehen.

Anschließend werden alle für die Leistungserbringer bzw. ihre Softwarehersteller relevanten Prozesse beschrieben. Diese Prozessbeschreibung dient der näheren Erläuterung der erforderlichen Schritte beim Leistungserbringer und ist als verbindliche Handlungsanleitung zu betrach-

ten. Damit soll erreicht werden, dass alle Leistungserbringer die Komponenten korrekt anwenden, Dokumentationspflichten erkennen und Klarheit darüber besteht, wie Datenlieferungen zu verschlüsseln und an welche Datenannahmestelle sie zu versenden sind.

Darüber hinaus werden in der Technischen Dokumentation die Spezifikationskomponenten aufgeführt und beschrieben. Wesentliche Komponenten sind die Datenbanken, die die Grundlage zur Erstellung der Softwareprodukte bilden. Auch die weiteren Komponenten enthalten Vorgaben für die Softwareprodukte und zur Umsetzung der Prozesse. In der Komponentenbeschreibung werden unter anderem Struktur, Funktionsweise und Inhalte der Datenbanken, des XML-Schemas und der weiteren Tools (z. B. Datenprüfprogramm und Verschlüsselungsprogramm) erläutert.

Die Komponenten der Spezifikation sind als verbindliche Handlungsanleitung zu betrachten. Damit soll erreicht werden, dass alle Leistungserbringer die Komponenten korrekt anwenden und Dokumentationspflichten erkennen sowie Klarheit darüber besteht, wie Datenlieferungen zu verschlüsseln und an welche Datenannahmestelle sie zu versenden sind. Diese verbindlichen Vorgaben der Spezifikation sind einzuhalten. Die Art der Umsetzung kann jedoch individuell auf die Zielgruppen der Software ausgerichtet werden. Ein Beispiel hierfür wäre die verfahrensspezifische Zurverfügungstellung und Erläuterung der generischen technischen Fehlermeldungen. Da beispielsweise die Fehlermeldungstexte der administrativen Prüfungen allgemein formuliert sind, können Verfahrensteilnehmer (Softwareanbieter, Datenannahmestellen, Vertrauensstelle) die Meldungstexte so konkretisieren, dass sie für den Empfänger (insbesondere für Ärztinnen und Ärzte) für den individuellen Fall verständlich sind.

Auf der Website des IQTIG stehen Informationen für Endanwender zu den einzelnen Verfahren und zur Erleichterung der Dokumentation bereit. Zu Letzterem gehören die Dokumentationsbögen, Ausfüllhinweise und Anwenderinformationen. Diese Dokumente, die sich an Leistungserbringer richten, die Anwender der PB-Software sind (z. B. Ärztinnen oder Ärzte), sind unter Berücksichtigung verschiedener Anforderungen möglichst anwenderorientiert und verständlich formuliert. Neben der Verständlichkeit werden beispielsweise auch Aspekte wie Einheitlichkeit, technische Umsetzbarkeit und Aufwand bei Verfahrensteilnehmern berücksichtigt.

Die vorliegende technische Dokumentation richtet sich an alle beteiligten Leistungserbringer und die von ihnen beauftragten Softwarehersteller, unabhängig vom Versorgungssektor oder vom Abrechnungskontext der Leistungserbringung.

Abbildung 2 stellt das serielle Datenflussmodell gemäß oKFE-RL dar. Unabhängig vom Leistungserbringer bzw. Abrechnungskontext ist nur eine Datenannahmestelle (DAS-KV) vorgesehen.

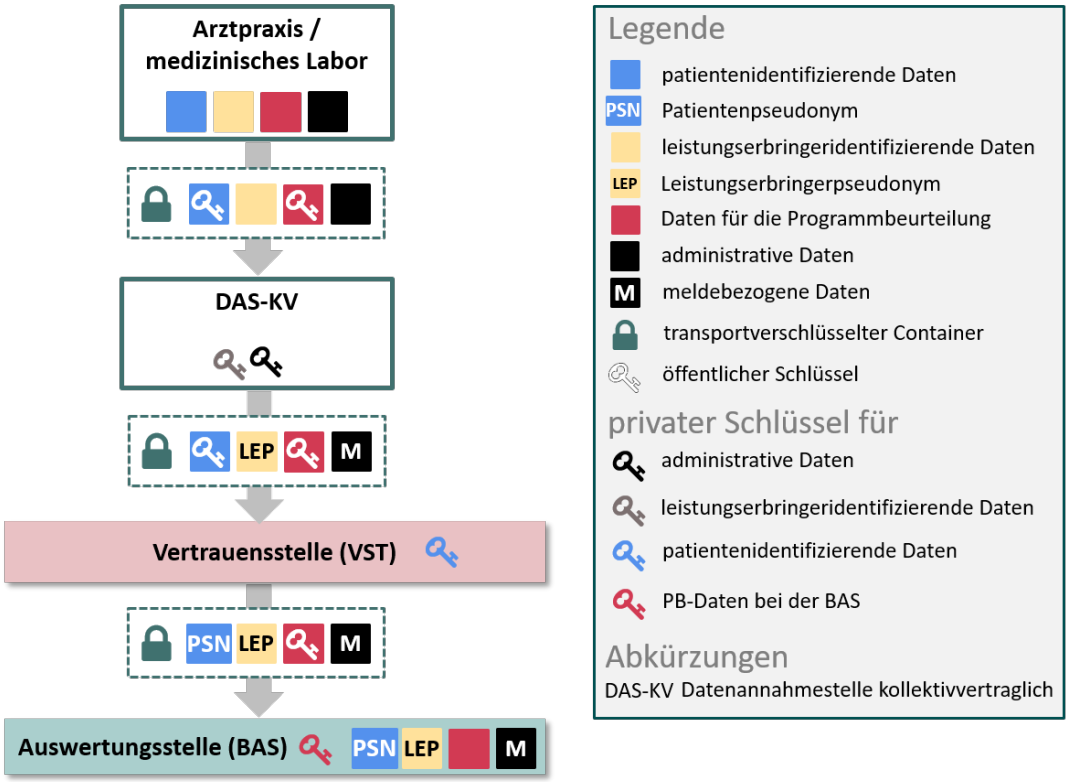


Abbildung 2: Serielles Datenflussmodell für die PB-Verfahren gemäß oKFE-RL

Die Zordnung der einzelnen Fälle zu den Leistungserbringern gemäß Datenfluss der oKFE-RL ist sowohl im Abrechnungskontext als auch auf den Ort der Leistungserbringung im ambulanten Bereich ausgerichtet.

In der technischen Dokumentation werden die Prozesse jeweils mit Bezug auf die Leistungserbringer und den Datenfluss dargestellt.

Die Spezifikation richtet sich insbesondere an die Softwarehersteller, die üblicherweise durch die Leistungserbringer mit der Umsetzung der Spezifikation betraut werden und geeignete Softwareprodukte zur Verfügung stellen. Die Programme müssen die Erfassung aller Daten gemäß der oben genannten Richtlinie und dieser sich daraus abgeleiteten Spezifikation ermöglichen, diese Daten auf Vollständigkeit und Plausibilität prüfen, im vorgegebenen Format an die datenentgegennehmenden Stellen exportieren und fehlerhafte (von den Datenstellen abgelehnte) Datensätze in geeigneter Weise dem Anwender zur Korrektur und zum erneutem Export vorlegen. Darüber hinaus bietet die Spezifikation den Softwareanbietern Hinweise für die Gestaltung der Benutzeroberfläche und die benutzerfreundliche Plausibilitätsprüfung unmittelbar bei der Dateneingabe.

1.3 Releaseplanung

Um Planungssicherheit zu gewährleisten und angemessen auf Fehler reagieren zu können, werden die Termine zur Veröffentlichung von Spezifikationspaketen (Versionen) eines Erfassungsjahres und zu Rückmeldefristen in der Technischen Dokumentation angekündigt. Tabelle 1 stellt eine entsprechende Übersicht für das Paket der PB-Spezifikation 2021 beim Leistungserbringer

für den Regelbetrieb dar. Die genannten Termine sind als Zielwerte zu betrachten und basieren auf Vorgaben des G-BA, Abstimmungen mit Softwareanbietern sowie Anforderungen aus der Umsetzung. Regulär wird die erste Version der Spezifikation eines Erfassungsjahres am 30. Juni des Vorjahres veröffentlicht.

Tabelle 1: Meilensteine der Releaseplanung der Spezifikationen 2021 für den Regelbetrieb

Frist	Meilenstein	Bereitstellung	Bemerkung
30.06.2020	Version 2021 V01	Veröffentlichung auf der IQTIG-Webseite (http://www.iqtig.org)	Finale Version für Verfahren gemäß oKFE-RL
15.09.2020	Frist für Fehlerrückmeldungen	E-Mail an verfahrenssupport@iqtig.org oder Nutzung der Kommunikationsplattform	
30.09.2020	Version 2021 V02[4]	Veröffentlichung auf der IQTIG-Webseite (http://www.iqtig.org)	Fehlerkorrekturen
20.10.2020	Frist für Fehlerrückmeldungen	E-Mail an verfahrenssupport@iqtig.org oder Nutzung der Kommunikationsplattform	
November 2020	Version 2021 V03	Veröffentlichung auf der IQTIG-Webseite (http://www.iqtig.org)	ggf. Aktualisierung von GOP gemäß EBM-Katalog; ggf. Fehlerkorrekturen;

Über die in der Tabelle aufgeführten Meilensteine hinaus erfolgt eine regelmäßige Abstimmung mit Softwareherstellern und weiteren Verfahrensteilnehmern (z. B. Datenannahmestellen, Vertrauensstelle) in Form von Informationstreffen, Workshops und Kommunikationsplattform. Zudem wurden Festlegungen getroffen, die die Qualität der Spezifikation erhöhen und die Richtlinienkonformität sicherstellen (z. B. werden wesentliche Änderungen nur im Rahmen finaler Versionen berücksichtigt).

Change- und Fehlermanagement

Das IQTIG empfiehlt für die Optimierung der Zusammenarbeit mit den beteiligten Stellen die folgenden Aktivitäten:

- Meldung von festgestellten Fehlern (z. B. Spezifikations- und Softwarefehlern)
- Verbreitung von Änderungsvorschlägen
- Abstimmung von Terminen und Umsetzungen im Rahmen der Releaseplanung
- Erfahrungsaustausch, um eine möglichst einheitliche Vorgehensweise zu ermöglichen
- Abstimmung der Spezifikationsänderungen

[4] Für das Erfassungsjahr 2021 wird außerplanmäßig eine Version 02 bereits im Juli veröffentlicht. Im Regelfall erfolgt die Bereitstellung jedoch im September.

Vorschläge, Fehlermeldungen und Diskussionspunkte können per E-Mail an den Verfahrenssupport oder über die Kommunikationsplattform (https://forum.iqtig.org/) mitgeteilt werden.

Sollten Sie über keine Zugangsdaten zur Kommunikationsplattform verfügen, obwohl Sie eine beteiligte Institution (z. B. Softwareanbieter, Datenannahmestelle) sind, kontaktieren Sie uns bitte und lassen sich bei uns registrieren.

Ihr Ansprechpartner:

Institut für Qualitätssicherung und Transparenz im Gesundheitswesen
Katharina-Heinroth-Ufer 1
10787 Berlin

Telefon: (+49) 30 58 58 26 340
Fax: (+49) 30 58 58 26 341

verfahrenssupport@iqtig.org
www.iqtig.org/

A Prozesse

Im Folgenden werden die einzelnen Prozesse und Unterprozesse der Spezifikation beschrieben (siehe Tabelle 2). Die einzelnen Abschnittsnummern verweisen auf die entsprechenden Abschnitte, in denen sie genauer erläutert werden.

Tabelle 2: Übersicht über die Prozesse der PB-Dokumentation und ihre Unterprozesse

Zielgruppe	**KV**
Richtlinie / **Prozesse**	**oKFE**
Auslösung	A 1.1**Fehler! Verweisquelle konnte nicht gefunden werden.**
Erfassung	A 1.2
Datenexport	A 1.3
Datenübermittlung	A 1.4
Rückprotokollierung	A 1.5

1 PB-Dokumentation

In diesem Kapitel werden die Prozessschritte sowie die in jedem Prozessschritt benötigten Werkzeuge der PB-Dokumentation in Bezug auf die Auslösung, Erfassung, Verarbeitung und Datenübermittlung der PB-Dokumentation beschrieben (Abbildung 3).

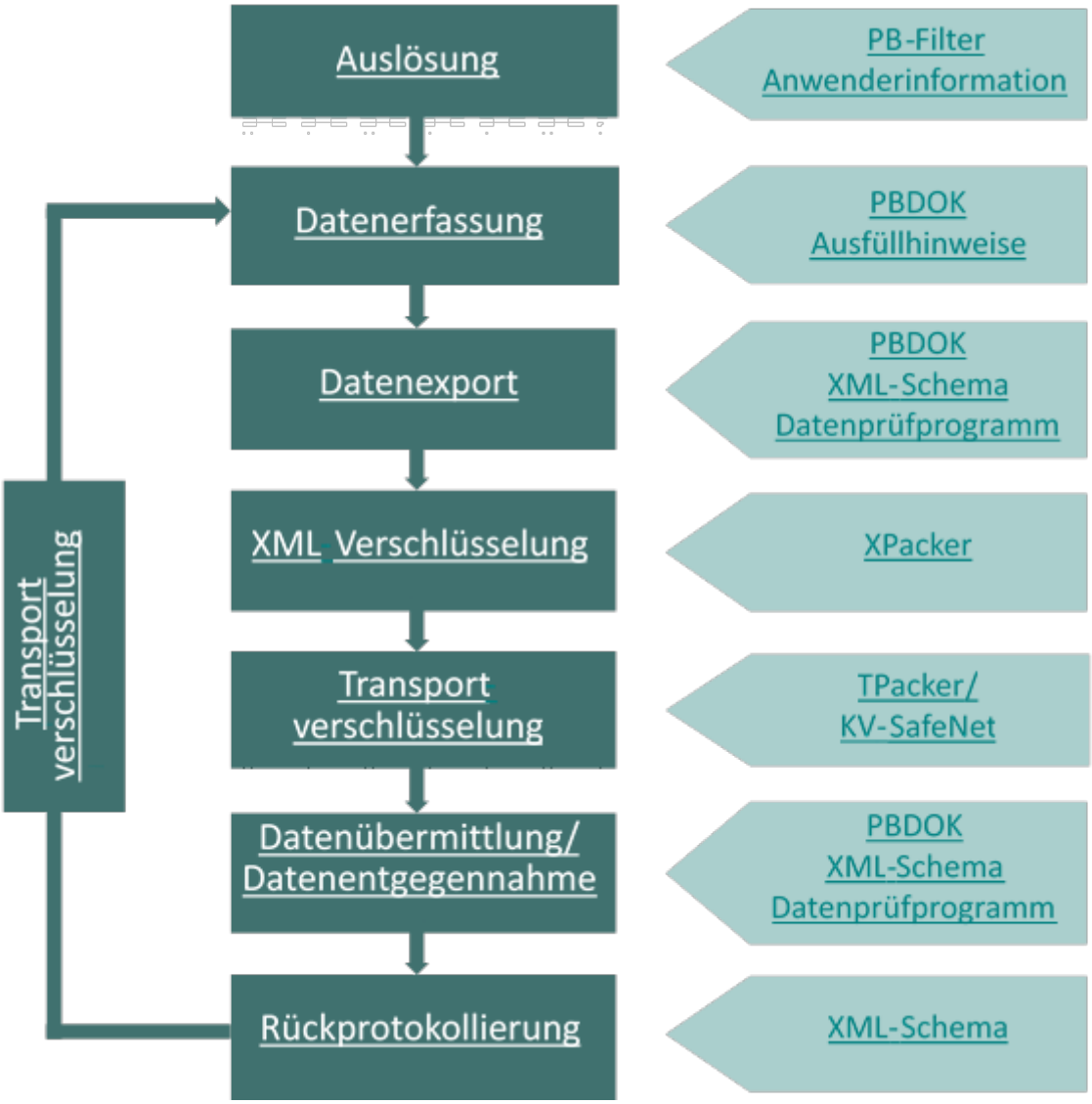


Abbildung 3: Überblick über die Prozesse und Werkzeuge der Datenannahme der PB-Daten

Nachfolgend werden die einzelnen Prozesse und Unterprozesse der PB-Dokumentation beschrieben.

1.1 Auslösung

Die PB-Filter-Software entscheidet für jeden Fall in der Arztpraxis oder medizinischem Labor als Leistungserbringer, welche Datensätze (Module) dokumentationspflichtig sind. Mit der vorliegenden Spezifikation können Systeme entwickelt werden, die eine zeit- und prozessnahe Auslösung von PB-Dokumentationen ermöglichen.

Hinweis

Dokumentationspflicht besteht für den Leistungserbringer, der die zu dokumentierende Leistung mit der KV abrechnet.

Achtung

Der Prozess der Auslösung soll **automatisch** durch die Softwareimplementierung (PB-Filter-Software, integrierte Software) der Auslösekriterien erfolgen.

Teilweise manuelle Auslösung des Moduls DKK

Durch das Fehlen einer spezifischen EBM-Ziffer für die Abklärungskoloskopie nach positivem i-FOB-Test im Rahmen des Programms zur Früherkennung von Darmkrebs kann die Auslösung der **Dokumentation der Abklärungskoloskopie** im Rahmen des Programms zur Früherkennung von Darmkrebs (Modul DKK) nicht automatisiert erfolgen, sondern muss manuell durch die Leistungserbringer ausgelöst werden.

Die Auslösung der **Dokumentation der Früherkennungskoloskopie** im Rahmen des Programms zur Früherkennung von Darmkrebs (Modul DKK) kann jedoch automatisiert erfolgen. Hierfür wurde mit der Spezifikation 2021 ein neuer Modulauslöser in der QS-Filter-Datenbank hinterlegt.

Der Algorithmus (Abschnitt B 1.4) trifft seine Entscheidung auf der Grundlage der medizinischen Routinedokumentationen (Gebührenordnungspositionen gemäß EBM-Katalog). In Arztpraxen bzw. medizinischen Laboren sind die medizinischen Routinedaten im Arztinformationssystem (AIS) und Laborinformationssystem (LIS) verfügbar. Die Informationen werden in einzelnen Fällen auch über spezialisierte Systeme, die auf die individuellen Anforderungen im jeweiligen Bereich zugeschnitten sind, zur Auslösung herangezogen. Im PB-Filter-Eingangsdatensatz (Abschnitt B 1.3) ist definiert, welche Daten verwendet werden. Diese basieren auf der KVDT-Datensatzbeschreibung der Kassenärztlichen Bundesvereinigung für den Einsatz von IT-Systemen in der Arztpraxis zum Zwecke der Abrechnung gemäß § 295 Abs. 4 SGB V.

Abbildung 4 stellt den Prozess der Auslösung der Dokumentationspflicht für die Module zu den Programmbeurteilungen im Rahmen der Richtlinie für organisierte Krebsfrüherkennungsprogramme in der Arztpraxis bzw. im medizinischen Labor dar. Die Pfeile symbolisieren den auslösenden Dokumentationsprozess der PB-Verfahren anhand spezieller EBM-Ziffern, die für die jeweiligen Untersuchungen Verwendung finden.

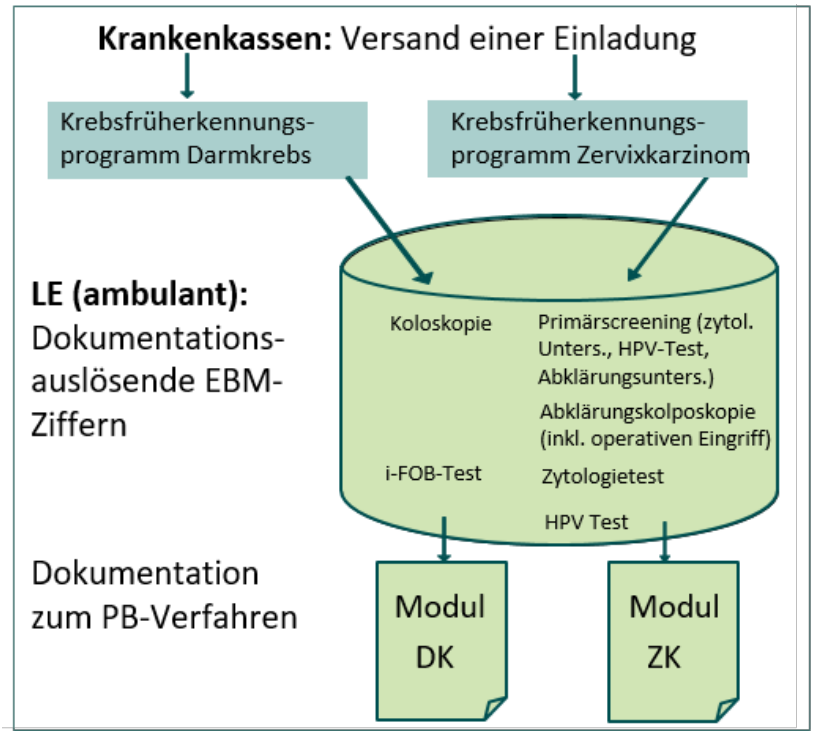


Abbildung 4: Grundfunktionalität der PB-Filter-Software: Berechnung der dokumentationspflichtigen Module auf der Grundlage der Routinedokumentation eines ambulanten Behandlungsfalles

Ein mögliches Szenario für den Einsatz von PB-Filter-Software findet somit an der Schnittstelle zwischen der Anwendungssoftware (AIS/LIS) und PB-Dokumentationssoftware statt. PB-Filter-Software kann als Service für die anderen Systeme realisiert werden: Über eine Anfrage an den PB-Filter-Service wird dem anfragenden System eine Modulliste geliefert. Bei diesem Szenario ist die PB-Filter-Software zustandsfrei: Sie protokolliert nicht die Erfüllung der Dokumentationspflicht (Soll/Ist).

Dies ist nur eine von vielen Einsatzmöglichkeiten. Am Beispiel wird jedoch deutlich, dass festgelegt werden muss, in welchem System die Statusverwaltung der PB-Dokumentation und ein kontinuierliches internes, ggf. standortbezogenes Vollständigkeitsmonitoring implementiert wird. Die Spezifikationskomponenten zum PB-Filter sind in Kapitel B beschrieben.

1.1.1 Der PB-Filter-Eingangsdatensatz

Die PB-Filter-Software bestimmt für jeden Behandlungsfall die dokumentationspflichtigen Datensätze[5] (technisch: Module; inhaltlich: Verfahren). Hierfür soll sie auf bestimmte, bereits dokumentierte Behandlungsdaten des Falles zurückgreifen, die sich im PB-Filter-Eingangsdatensatz wiederfinden. Während einer Behandlung können verschiedene Gebührenordnungspositionen dokumentiert werden, die Teil des PB-Filter-Eingangsdatensatzes sind. Der PB-Filter-Algorithmus basiert zum einen auf diesen medizinischen und zum anderen auf administrativen Datenfeldern.

[5] Definiert in der Spezifikationsdatenbank für PB-Verfahren-Dokumentation.

Der PB-Filter-Eingangsdatensatz enthält somit diejenigen Datenfelder, die in der Datenbank als Kodes hinterlegt sind. Diese benötigt der PB-Filter-Algorithmus (Abschnitt B 1.4) um ein Ergebnis zu ermitteln, d. h. die für den entsprechenden Fall dokumentationspflichtigen PB-Module. Der Aufbau des PB-Filter-Eingangsdatensatzes wird in Abschnitt B 1.3.1 beschrieben.

Eine Besonderheit der PB-Verfahren ist, dass sie sich ausschließlich auf GKV-Versicherte beziehen. Nicht-GKV-Versicherte werden über den PB-Filter identifiziert und bereits bei der Auslösung ausgeschlossen.

Fallkonzept der Programmbeurteilung

Die Auslösung der Dokumentation in den PB-Verfahren erfolgt über EBM-Ziffern (Teilbedingung `EBM EINSIN [MODUL]_EBM`). Gemäß Abrechnungsbestimmungen können die in den Listen definierten EBM-Kodes einmal pro Quartal abgerechnet werden. Somit kann ein Behandlungsfall pro Quartal ausgelöst und dokumentiert werden.

Besonderheit beim Modul DKK

Die Dokumentation Koloskopien im Rahmen des Programms zur Früherkennung von Darmkrebs (Modul DKK) kann aufgrund des Fehlens einer spezifischen EBM-Ziffer für die Abklärungskoloskopien nach positivem i-FOB-Test nicht vollständig automatisch erfolgen, sondern muss teilweise manuell durch den Leistungserbringer ausgelöst werden.

1.2 Erfassung

Nach Auslösung eines Moduls durch den PB-Filter folgt der Prozess der Erfassung. Hierbei werden Daten entweder automatisch aus dem AIS/LIS in die Eingabemaske der Erfassungssoftware übertragen oder durch den Dokumentierenden manuell erfasst.

Hinweis

Für Leistungserbringer, die auf Grundlage ihrer vorliegenden Daten direkt eine spezifikationskonforme Exportdatei erzeugen können, ist die Implementierung einer zusätzlichen Erfassungsmaske nicht notwendig.

In den Modulen der PB-Verfahren gelten besondere Regelungen zur Erhebung von auslösenden EBM-Ziffern (z. B. Abschnitte A 1.2.5 und A 1.2.6). Neben PB-Daten (Abschnitt A 1.2.4) sind auch einrichtungsidentifizierende Daten wie das Institutionskennzeichen und die PB-relevante Betriebsstättennummer des niedergelassenen Leistungserbringers (Abschnitt A 1.2.2) sowie ggf. patientenidentifizierende Daten (Abschnitt A 1.2.3) zu dokumentieren. Die Erfassung ist abgeschlossen, wenn alle zu dokumentierenden Datenfelder unter Berücksichtigung von Abhängigkeiten und Plausibilitätsprüfungen vollständig erfasst wurden (Abschnitt A 1.2.5).

Als Vorlage für die Gestaltung der Eingabemaske (Abschnitt A 1.2.1) durch den Softwareanbieter dient die Datenfeldbeschreibung des jeweiligen Moduls (Abschnitt B 2.3).

1.2.1 Gestaltung von Eingabemasken

Die Benutzeroberfläche einer Erfassungssoftware (Graphical User Interface = GUI) soll ergonomisch und anwenderfreundlich gestaltet sein. Gestaltung und Layout der Eingabemaske sind Aufgabe der Softwarehersteller. Neben Anforderungen der Kunden werden üblicherweise firmeninterne Standards bzw. Vorgaben des Betriebssystems (z. B. Windows) für das „look and feel“ berücksichtigt.

Diese Spezifikation definiert als Minimalstandard die für den Anwender sichtbaren Inhalte der Dokumentationsformulare. Als Referenz für die sichtbaren Inhalte dienen die Dokumentationsbögen, die als Bestandteil der Spezifikation durch das IQTIG veröffentlicht werden. Die Dokumentationsbögen werden als PDF-Dokumente bereitgestellt, die aus der Spezifikationsdatenbank automatisch generiert worden sind. Bei den Dokumentationsbögen handelt es sich um Formulare zur Ansicht (Muster), die nicht zur Dokumentation zu verwenden sind.

Tabelle 3 gibt einen Überblick darüber, welche Informationen der Spezifikationsdatenbank (identifiziert durch Tabelle und Attribut) bei der Erstellung der Dokumentationsbögen berücksichtigt werden und somit auch in den Erfassungssystemen sichtbar sein sollen.

Tabelle 3: Informationen aus der Datenbank, welche im GUI verwendet werden

Tabelle	Attribut	Bemerkung	sichtbar für Anwender
`Modul`	`name`	Kürzel des Datensatzes, erscheint üblicherweise im Titel des Formulars	ja
`Modul`	`bezeichnung`	Bezeichnung des Datensatzes, erscheint üblicherweise im Titel des Formulars	ja
`Bogen`	`bezeichnung`	Bezeichnung des Teildatensatzes	ja
`Bogen-Feld`	`gliederungAufBogen`	Nummer des Datenfelds, dient bei umfangreicheren Bögen zur besseren Orientierung	(ja)
`Bogen-Feld`	`bezeichnung`	Bezeichnung des Datenfelds	ja
`Bogen-Feld`	`ergaenzendeBezeichnung`	Ergänzende Bezeichnung zum Datenfeld, kann z. B. durch Wahl der Schrift von der Bezeichnung abgesetzt werden	ja
`Feld`	`laenge`	Definiert die Länge des Eingabefelds. Für die Gestaltung des Eingabefelds sind weitere Informationen aus der Datenbank wichtig (z. B. `Feld.nachKommaLaenge`).	ja

Tabelle	Attribut	Bemerkung	sichtbar für Anwender
`Feld`	`einheit`	Einheiten (wie z. B. `ml`) müssen angezeigt werden.	ja
`BasisTyp`	`format`	Formatanweisungen (wie z. B. `TT.MM.JJJJ`) sollen – sofern nicht durch übergeeignete Eingabefelder unterstützt – für den Anwender angezeigt werden.	(ja)
`SchluesselWert`	`Code`	Bei Schlüsselfeldern sollen die Kodes möglichst in Auswahllisten angezeigt werden. Bei einigen Realisierungsvarianten (z. B. Checkbox) kann auf die Anzeige der Kodes verzichtet werden.	(ja)
`SchluesselWert`	`bezeichnung`	Bei Schlüsselfeldern müssen die Textdefinitionen der Kodes (z. B. in einer Auswahlliste) angezeigt werden. Für die Sortierung sind die Attribute `sortierNrVerwendet` und `zahl` der Tabelle `Schluessel` relevant (Abschnitt B 2.3.3).	ja
`Abschnitt`	`bezeichnung`	Die Überschriften sind wichtig für die Strukturierung und das Verständnis des Datensatzes und müssen deshalb in der PB-Dokumentationssoftware angezeigt werden.	ja

Abbildung 5 zeigt für ein Datenfeld eines Dokumentationsbogens (PDF) den Zusammenhang zu den Informationen, welche in der Datenbank vorhanden sind.

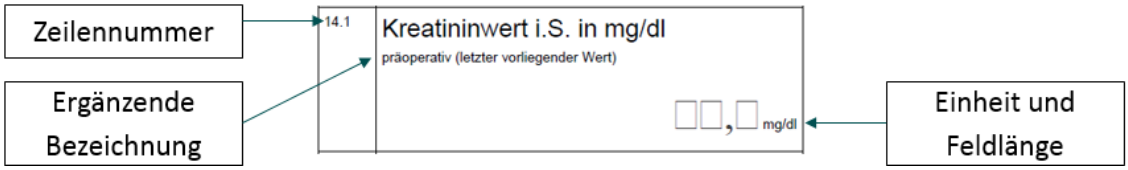


Abbildung 5: Beispiel für Informationen, die in der Oberfläche angezeigt werden sollen (Spezifikation 2018)

Werden Datenfelder (z. B. besondere Personengruppe) eines PB-Datensatzes aus Fremdsystemen über Schnittstellen importiert, so sollen die übernommenen Daten auch in der Erfassungssoftware angezeigt werden. Es ist für den Anwender wichtig, die vollständigen PB-Daten im Kontext eines „PB-Formulars“ zu sehen und auch auf Richtigkeit und Vollständigkeit zu prüfen.

Allgemeine Grundsätze für die Gestaltung der Eingabemaske

Die Grundsätze für die Plausibilitätsprüfungen wirken sich insbesondere auf die Gestaltung der Benutzeroberflächen in Erfassungssoftware aus. Durch die funktionale Gestaltung sollte ein Kompromiss zwischen Dateneingabekomfort einerseits und Zwang zur aktiven Eingabe korrekter Daten andererseits gefunden werden. Im Folgenden werden die Regeln für die Gestaltung von Benutzeroberflächen aufgeführt:[6]

- Keine Suggestion von Feldinhalten durch Vorbelegung (Defaults): Oberstes Prinzip bei der Gestaltung der Benutzeroberflächen ist, dass dem Anwender des Programms keine Angaben suggeriert werden. Insbesondere darf keine Vorbelegung mit Standardwerten erfolgen, die den „Nicht-Problemfall“ dokumentieren.

Hinweis

Eine Vorbelegung mit Standardwerten ist nicht zulässig. Die Übernahme von im AIS/LIS vorhandenen Angaben in die PB-Dokumentation ist hingegen zulässig.

Dies gilt ebenfalls für verschiedene Strukturparameter, die durch den Softwareanbieter in den Bogen übernommen werden können.

Beispielsweise können die Angaben der Datenfelder BSNRAMBULANT oder NBSNRAMBULANT als Strukturparameter übernommen werden, wenn diese Angaben vorliegen.

Eine Vorbelegung des Datenfeldes PROGRAMMNUMMER ist vorgesehen und soll als Default vorbelegt werden. Die Programmnummer zum PB-Verfahren zur Früherkennung von Darmkrebs wird mit „DK“ und für das PB-Verfahren zur Früherkennung von Zervixkarzinomen mit „ZK“ vorbelegt.

- Verwendung der vorgeschriebenen Fehler- und Warnmeldungen bei feldübergreifenden Regeln: Die Fehler- und Warnmeldungen sind so formuliert, dass sie möglichst nicht suggerieren, auf welche Weise widersprechende Angaben korrigiert werden sollen. Insofern sollen sie wörtlich übernommen werden.
- Keine zusätzlichen Ober-/Untergrenzen für Maße, Zeitdauern und Anzahlen: Außer den durch die Datenfeldbeschreibung und die Plausibilitätsregeln vorgegebenen Wertebereichen darf in Erfassungsprogrammen keine Einengung möglicher Merkmalsausprägungen in Wertefeldern erfolgen.
- Zwang zur aktiven Entscheidung zwischen „ja“ und „nein“: An entsprechenden Stellen in den Dokumentationsbögen, bei denen die Auswahl „0“ (nein) und „1“ (ja) (vgl. z. B. Schlüssel JN) zu treffen ist, darf keine Voreinstellung des Wertes im Eingabefeld erfolgen. Es besteht somit der Zwang zur Eingabe eines Wertes. Nur an Stellen, an denen im Erfassungsformular lediglich „1“ als Option angegeben wird, soll die Nicht-Eingabe eines Wertes als Verneinung interpretiert werden. Hintergrund dieser Differenzierung ist, dass einerseits in qualitätskritischen Bereichen eine Unterscheidung zwischen „keine Angabe“ und „nein“ erfolgen muss, es andererseits der Benutzerakzeptanz abträglich ist, wenn diese Systematik auch an allen anderen

[6] Externe Systeme, die Daten an ein Erfassungsprogramm übergeben, sollten diese Grundsätze sinngemäß anwenden.

Stellen durchgängig verfolgt wird.
- Kein automatisches Verändern von Feldinhalten in Abhängigkeit von anderen Feldinhalten.

Beispiel[7]**:**
Wenn die Transfusion von Blut zunächst bejaht und das Feld „Fremdblut“ angekreuzt worden ist, soll das Entfernen des Kreuzes des übergeordneten Feldes „Bluttransfusion“ nicht automatisch zum Entfernen des Kreuzes bei „Fremdblut“ führen. Vielmehr soll eine Fehlermeldung erfolgen und der Anwender gezwungen sein, zunächst das Kreuz bei „Fremdblut“ zu entfernen, bevor „Bluttransfusion“ verneint werden kann.

Umrechnung von Einheiten bei numerischen Feldern
In Einzelfällen ist es aus Anwendersicht hilfreich, wenn die Eingabemaske die Eingabe von Messwerten in Einheiten ermöglicht, die von den spezifizierten Einheiten abweichen. Beispielsweise soll im Modul DKI laut Spezifikation das Datenfeld `IFOBHBKONZENTRATION` in µg/g mit zwei Nachkommastellen dokumentiert werden. In einigen Bundesländern wird dieser Wert aber in µg/ml angegeben. Diese Funktionalität sollte möglichst in die Erfassungssoftware integriert werden, um den Dokumentationsaufwand zu verringern.

Gestaltung von Eingabemasken mit Layout-Feldgruppen
Sogenannte Layout-Feldgruppen sollen in der Erfassungsmaske separat kenntlich gemacht werden. Hierbei handelt es sich um Datenfelder, die zu einer logischen Gruppe zusammengefasst werden können. Feldgruppen können Filterfelder oder abhängige Felder beinhalten. Abhängige Felder von Layout-Feldgruppen werden auf den generierten Dokumentationsbögen eingegraut dargestellt. Hierbei handelt es sich um Feldgruppen, bei denen das Attribut `grauWennNegativ` in der Datenbanktabelle `FeldGruppe` gesetzt ist. Diese sollen für die Gestaltung von Eingabemasken verwendet werden.

Die Eigenschaften von Layout-Feldgruppen sind in Abschnitt B 2.4.7 erläutert.

Empfehlung zur Umsetzung von Layout-Feldgruppen
Ist das Attribut `grauWennNegativ` gesetzt, so darf die Benutzereingabe für die abhängigen Felder durch die Erfassungssoftware deaktiviert werden, falls die negative Filterbedingung zutrifft. Bei der Umsetzung muss Folgendes sichergestellt werden:

- Nach jeder Änderung der Inhalte der Filterfelder im Erfassungsformular muss das Programm die Filterbedingung der Feldgruppe evaluieren und ggf. eine Aktualisierung der Oberfläche durchführen.
- Die Benutzereingabe für die abhängigen Felder darf nur dann deaktiviert werden, wenn keines dieser Felder ausgefüllt ist. Ansonsten ist der Anwender auf eine Plausibilitätsverletzung hinzuweisen.
- Wenn nach einer Benutzereingabe die positive Filterbedingung zutrifft, so sind ggf. vorher deaktivierte Eingabefelder wieder zu aktivieren.

[7] Die im Beispiel genannten Felder sind NICHT Bestandteil der Spezifikationsdatenbank für die PB-Dokumentation. Sie sind frei erfunden und dienen lediglich der Erläuterung.

- Deaktivierte Felder dürfen nicht ausgeblendet werden.

1.2.2 Einrichtungsidentifizierende Daten

In Hinblick auf eine einrichtungs- bzw. standortbezogene Auswertung sind einrichtungsidentifizierende Daten in der PB-Dokumentation wie folgt zu dokumentieren.

Die wesentliche Angabe zur Identifikation eines ambulanten Leistungserbringers ist die von der Kassenärztlichen Vereinigung (KV) vergebene Betriebsstättennummer (BSNR; Feld BSNRAMBULANT). Sie identifiziert die Arztpraxis als abrechnende Einheit. Dabei umfasst der Begriff Arztpraxis auch Medizinische Versorgungszentren (MVZ), Institute, medizinische Labore, Notfallambulanzen sowie Ermächtigungen von am Krankenhaus beschäftigten Ärzten.

Hinweis

Im Rahmen der PB-Verfahren gemäß oKFE-RL werden als ambulante Leistungserbringer sowohl die Leistungserbringer der Arztpraxen als auch die medizinischen Labore unter dem Begriff Leistungserbringer zusammengeführt. Die Identifikation der ambulanten Leistungserbringer erfolgt mittels der von der Kassenärztlichen Vereinigung (KV) vergebenen Betriebsstättennummer (BSNR; Feld BSNRAMBULANT), Nebenbetriebsstättennummer (NBSNR; Feld NBSNRAMBULANT) und lebenslangen Arztnummer (LANR; Feld LANR).

1.2.3 Patientenidentifizierende Daten

Mit dieser Spezifikation werden PB-Daten mithilfe patientenidentifizierender Daten (PID) im Regelbetrieb patientenbezogen zusammengeführt. Hierbei werden auch PB-Daten verschiedener, inhaltlich zusammenhängender Module zusammengeführt.

Tabelle 4 gibt einen Überblick über die betroffenen Module. Die Spalte „Über PID-Verfahren verknüpfte Module“ verdeutlicht, für welche Module von der Vertrauensstelle jeweils dasselbe Patientenpseudonym (PSN) aus den PID-Daten zu generieren ist. Hintergrund ist die Anforderung, dass bei der Auswertungsstelle über das generierte Patientenpseudonym modulübergreifend Datensätze mit inhaltlichem Bezug zusammengeführt werden müssen, während Datensätze, die denselben Patienten, aber einen anderen Leistungsbereich betreffen, nicht zusammengeführt werden dürfen. Die hier aufgeführten Module sind der PB-Spezifikation für Leistungserbringer zugehörig.

Tabelle 4: Module mit patientenidentifizierenden Daten (PID-Module)

Beginn des PID-Verfahrens im Regelbetrieb	Modul	Bezeichnung	Über PID-Verfahren verknüpfte Module[8]
2020	DKK	PB-Verfahren zur Früherkennung von Darmkrebs Koloskopie	DKK, DKI
2020	DKI	PB-Verfahren zur Früherkennung von Darmkrebs i-FOB-Test	

[8] Module in der fallbezogenen PB-Verfahren-Basisspezifikation für Leistungserbringer

Beginn des PID-Verfahrens im Regelbetrieb	Modul	Bezeichnung	Über PID-Verfahren verknüpfte Module[8]
2020	ZKP	PB-Verfahren zur Früherkennung von Zervixkarzinomen Primärscreening/Abklärungsuntersuchung	ZKP, ZKA, ZKZ, ZKH
2020	ZKA	PB-Verfahren zur Früherkennung von Zervixkarzinomen Abklärungskolposkopie	
2020	ZKZ	PB-Verfahren zur Früherkennung von Zervixkarzinomen Zytologie	
2020	ZKH	PB-Verfahren zur Früherkennung von Zervixkarzinomen HPV-Test	

Für die Module in der Tabelle 4 wurden in der PB-Dokumentation die folgenden Bogenfelder integriert:

Tabelle 5: Felder für patientenbezogene Fallzusammenführung

Datenfeld	Beschreibung	Exportweg	PID-Feld
KASSEIKNR	Institutionskennzeichen der Krankenkasse der Versichertenkarte	Export der ersten beiden Ziffern über das Ersatzfeld `kasseiknr2Stellen`	nein
VERSICHERTENID-NEU	eGK-Versichertennummer	Gesonderter Export über das Ersatzfeld `versichertenidgkv`	ja
PERSONENKREIS-KVDT	besondere Personengruppe	Gesonderter Export über das Ersatzfeld `versichertenstatusgkv`	nein

Dokumentation der Felder KASSEIKNR, PERSONENKREISKVDT und VERSICHERTENIDNEU

Die EDV-Systeme der Arztpraxen bzw. medizinischen Labore sollen die oben genannten Felder, die in der Regel über die elektronische Gesundheitskarte (eGK) eingelesen werden, automatisiert in die PB-Dokumentationssoftware übernehmen. Zur Erfassung der Felder ist statt des direkten Auslesens der Karte auch die elektronische Übernahme aus dem AIS/LIS möglich. Arztpraxen und medizinische Labore müssen in Zusammenarbeit mit Softwareanbietern geeignete Wege finden, die Daten für die Felder in den PB-Dokumentationsbögen der PID-Module verfügbar zu machen.

Die fehlerfreie Übertragung der Versichertendaten in die PB-Software ist eine zentrale Voraussetzung für die leistungserbringer- und einrichtungsübergreifende Qualitätssicherung, daher ist eine manuelle Erfassung der PID-Daten unmittelbar im PB-Dokumentationsbogen durch den Anwender nicht zulässig.

Achtung
Dokumentation der Felder KASSEIKNR, PERSONENKREISKVDT und VERSICHERTENIDNEU

Liegen Daten zur elektronischen Gesundheitskarte (eGK) zum Zeitpunkt der Erfassung noch nicht vor, erlischt **nicht** die Dokumentationspflicht der Felder KASSEIKNR, PERSONENKREISKVDT und VERSICHERTENIDNEU für GKV-Versicherte. Die Angabe ist bei Vorliegen der elektronischen Gesundheitskarte zu dokumentieren.

Dokumentation und Export patientenidentifizierender Daten

Die eGK-Versichertennummer (VERSICHERTENIDNEU) stellt ein sogenanntes PID-Feld dar. PID-Felder werden in den vorgesehenen PID-Datencontainer integriert und mit dem öffentlichen Schlüssel der Vertrauensstelle verschlüsselt, sodass nur diese die Daten entschlüsseln kann. Die PID-Felder dienen der Generierung des Patientenpseudonyms.

Ergänzend zum PID-Feld VERSICHERTENIDNEU werden das Institutionskennzeichen der Krankenkasse der Versichertenkarte (KASSEIKNR) und die besondere Personengruppe (PERSONENKREISKVDT) erfasst. Diese Felder stehen in enger Verknüpfung mit dem PID-Feld VERSICHERTENIDNEU. Sie dienen zur Identifikation davon, ob es sich bei dem Behandlungsfall um einen GKV-Versicherten handelt und ob somit PID-Daten zu erfassen sind. Gleichermaßen dient es der Sicherstellung, dass keine PID-Daten von Nicht-GKV-Versicherten exportiert werden.

Das Ersatzfeld versichertenidgkv beinhaltet die eGK-Versichertennummer, wenn diese dem vorgegebenen Format entspricht und das 9-stellige Institutionskennzeichen der Krankenkasse der Versichertenkarte mit der Zeichenkette ‘10’ beginnt und kein besonderer PERSONENKEISKVDT vorliegt. Ist diese Bedingung nicht erfüllt, ist das Ersatzfeld leer. Dies gilt auch für den Fall, dass das Institutionskennzeichen der Krankenkasse nicht 9-stellig vorliegt.

Das Ersatzfeld kasseiknr2Stellen beinhaltet die ersten beiden Zeichen des Institutionskennzeichens der Krankenkasse der Versichertenkarte, wenn dieses Feld 9-stellig und ausgefüllt ist. Ist dies nicht der Fall, ist das Ersatzfeld leer. Dieses Ersatzfeld wird zweifach exportiert: Innerhalb der PB-Daten sowie im vorgesehenen Datencontainer der Patientendaten. Damit ist die Information für alle Datenannahmestellen, die Vertrauensstelle und die Auswertungsstelle einsehbar und eine Prüfung, dass nur für GKV-Versicherte PID-Daten bzw. ein Patientenpseudonym vorliegt, möglich.

Das Ersatzfeld versichertenstatusgkv beinhaltet den Wert 1, wenn die 9-stellige KASSEIKNR mit der Zeichenkette '10' beginnt und kein besonderer PERSONENKEISKVDT vor-

liegt. Ist dies nicht der Fall, beinhaltet das Ersatzfeld den Wert 0. Um die Stabilität und die Sicherheit des Verfahrens bei allen Beteiligten zu gewährleisten, werden die zur Identifikation von GKV-Versicherten notwendigen Datenfelder (z. B. das Feld KASSEIKNR und das zugehörige Ersatzfeld `kasseiknr2Stellen`)mit allen bestehenden Regeln und Prüfalgorithmen auch für diejenigen Leistungsbereiche/Verfahren beibehalten, die ausschließlich GKV-versicherte Patienten umfassen (ein Ausschluss der Nicht-GKV-Versicherten erfolgt dort bereits über den PB-Filter-Algorithmus).

Die Abbildung von Ersatzfeldern in der Datenbank zur PB-Dokumentation wird in Abschnitt B 2.5.2 erläutert.

Bestätigung oder Korrektur fehlender oder fehlerhafter patientenidentifizierender Daten (PID)

Werden fehlerhafte Daten zum „Institutionskennzeichen der Krankenkasse der Versichertenkarte“ und zur „eGK-Versichertennummer“ in die PB-Dokumentationssoftware übertragen, können die Datenfelder nicht korrekt oder nur unvollständig dokumentiert werden. Eine manuelle Korrektur fehlerhafter Angaben der beiden Datenfelder ist unzulässig.

Liegen in der PB-Dokumentationssoftware fehlerhafte Angaben zum „Institutionskennzeichen der Krankenkasse der Versichertenkarte“ und zur „eGK-Versichertennummer“ vor, sollte die interne EDV-Abteilung oder der entsprechende Softwareanbieter frühzeitig kontaktiert werden, um diese Werte zu korrigieren und um den Fall korrekt abzuschließen.

1.2.4 PB-Daten

Es sind alle erforderlichen Daten zu dokumentieren. Hierbei kann eine automatische Übertragung der PB-Daten aus dem AIS/LIS möglich sein.

1.2.5 Plausibilitätsprüfungen

Fehlende und widersprüchliche Angaben in den Datensätzen sollen durch umfangreiche Plausibilitätsprüfungen verhindert werden. In der PB-Dokumentationssoftware muss die vollständige Plausibilitätsprüfung für jeden Datensatz spätestens bei Dokumentationsabschluss erfolgen. Teile der Plausibilitätsprüfungen sollen bereits während der Erfassung erfolgen. Dadurch wird sichergestellt, dass ein aufwendiges Korrekturverfahren – Verschlüsselung und Übermittelung der Datensätze, Prüfung durch DAS, VST oder BAS, Fehlerprotokollierung über die DAS (Datenflussprotokoll), Korrektur der Dokumentation und erneute Übermittlung des Datensatzes – weitgehend entfällt.

Die Datenannahmestellen führen für jeden Datensatz alle harten Plausibilitätsprüfungen der Spezifikation durch. Bei einer Regelverletzung ist der Datensatz zurückzuweisen. Die Datenannahmestellen dürfen keine zusätzlichen (in der Spezifikation nicht definierten) Plausibilitätsprüfungen durchführen.

Es gelten folgende Grundsätze für die Plausibilitätsprüfung:

- Alle Felder müssen ausgefüllt sein, wenn andere logische Sachverhalte dem nicht entgegenstehen.

- Jedes Feld, das auszufüllen ist, muss einen sinnvollen Feldinhalt haben.
- Es wird jede harte Plausibilitätsprüfung vorgenommen, die definiert ist.
- Harte Plausibilitätsprüfungen werden nur vorgenommen, wenn Sachverhalte zwingend miteinander gekoppelt sind.
- Es werden keine Sachverhalte suggeriert (keine Default-Werte, keine Vorbelegungen, keine Profile. Fehlermeldungen werden vorgegeben).
- Keine Angabe (bzw. kein Feldinhalt) wird ergänzt oder gelöscht.

Arten der Plausibilitätsprüfungen

Es wird zwischen drei Arten von Plausibilitätsprüfungen unterschieden:[9]

- harte Prüfungen
- weiche Prüfungen in der PB-Dokumentationssoftware
- warnende Prüfungen bei der Datenentgegennahme

Die drei Arten der Prüfungen werden in unterschiedlichen Kontexten (PB-Software bzw. Datenentgegennahme) durchgeführt und haben unterschiedliche Konsequenzen. Tabelle 6 gibt einen Überblick:

Tabelle 6: Arten der Plausibilitätsprüfungen

Art der Prüfung	Kürzel	Prüfung durch PB-Software	Prüfung durch Datenannahmestelle	Konsequenz: Verhindert Dokumentationsabschluss/Datenentgegennahme
Hart	H	Ja	Ja	Ja
Weich	W	Ja	Optional	Nein
Warnend	D	Optional	Ja	Nein

Harte Prüfungen

Harte Prüfungen sind sowohl in der PB-Dokumentationssoftware als auch bei der Datenentgegennahme anzuwenden. Bei einer harten Regelverletzung ist:

- ein Dokumentationsabschluss der PB-Dokumentation eines Vorgangs unzulässig.
- ein Datensatz von der entgegennehmenden Stelle zurückzuweisen.

Die in der technischen Dokumentation und der Datenbank definierten Plausibilitätsprüfungen sind hart, außer wenn sie explizit als weich oder warnend gekennzeichnet sind.

Weiche Plausibilitätsprüfungen

Die weichen Plausibilitätsprüfungen der Spezifikation sind von der PB-Dokumentationssoftware bis spätestens zum Dokumentationsabschluss durchzuführen. Bei einer Regelverletzung erhält der Benutzer einen Warnhinweis, anhand dessen er entscheidet, ob eine Änderung von Feldinhalten notwendig ist. Ebenso wie harte Plausibilitätsprüfungen müssen weiche Regeln immer vom Softwareanbieter umgesetzt werden.

[9] Die Komponentensicht ist in Abschnitt B „Plausibilitätsprüfungen“ auf S. 153 beschrieben.

Warnende Plausibilitätsprüfungen

Unterjährig (in weiteren Versionen während des Erfassungsjahres) findet keine Verschärfung der Plausibilitätsprüfungen statt, um den Anwendern in den Einrichtungen den Dokumentationsabschluss nach transparenten Regeln zu ermöglichen.

Trotz einer Vielzahl von Prüfungen kann eine hundertprozentige Plausibilität nicht gewährleistet werden, sodass es vorkommen kann, dass Leistungserbringer inhaltlich unplausible Datensätze liefern. Fallen derartige Lücken in der Datenqualität nach Start des Erfassungsjahres auf, so kann die in der Einrichtung installierte PB-Dokumentationssoftware den Anwender nicht auf diesen Mangel hinweisen.

Warnende Plausibilitätsprüfungen können, z. B. auf Anregung einer Datenannahmestelle, vom IQTIG unterjährig in weiteren Versionen publiziert werden, um die Einrichtungen auf gravierende Mängel der Datenqualität hinzuweisen. Warnende Plausibilitätsprüfungen sind somit als ernste Warnungen zu verstehen. Die beteiligten Stellen haben dabei folgende Zuständigkeiten:

- Datenannahmestellen setzen die warnenden Plausibilitätsprüfungen um.
- Einrichtungen erhalten über das Fehlerprotokoll Informationen zu mangelhaften Datensätzen.
- Die PB-Dokumentationssoftware zeigt den Anwendern die Warnungen nach der Rückprotokollierung an.
- Anwender haben dann die Möglichkeit, die mit Warnungen versehenen Datensätze zu korrigieren und erneut einzusenden.

Einzelregeln

Einzelregeln sind in der Datenbank zur PB-Dokumentation in der Regelsyntax in der Tabelle Regeln hinterlegt. Außerdem gibt es Regeln, die nur in Form von Feldeigenschaften – nicht aber in Regelsyntax – in der Datenbank hinterlegt sind. Die standardisierten Fehlertexte dieser Prüfungen sind Abschnitt B 2.4.8 zu entnehmen.

Die PB-Dokumentationssoftware muss sowohl die harten als auch die weichen feldbezogenen Prüfungen ausführen. Die Evaluation soll direkt bei der Dateneingabe geschehen. Fehleingaben sollen dem Benutzer direkt mitgeteilt werden. Einige Prüfungen erübrigen sich durch adäquate Gestaltung von Eingabemasken, z. B. durch Bereitstellung von Auswahlmenüs für Schlüsselkodes. Bei Regelverletzung muss die PB-Dokumentationssoftware dem Benutzer verständliche Fehlertexte anzeigen.

Bei der Datenentgegennahme sind alle harten Prüfungen zu evaluieren und bei Regelverletzung die unten definierten standardisierten Fehlertexte im Rahmen des im Abschnitt B 2.4.8 beschriebenen Korrekturverfahrens an die Einrichtung zu übermitteln.

Feldgruppenregeln

Datenfelder (Bogenfelder) eines Moduls können zu einer Feldgruppe zusammengefasst werden, um logische Abhängigkeiten von Bogenfeldern abzubilden. Das bedeutet in der Praxis, dass der Anwender daran gehindert wird, Felder mit Werten auszufüllen, die der Logik der Feldgruppe widersprechen.

Die explizite Definition von Feldgruppen strukturiert sowohl die Bogenfelder als auch die Plausibilitätsregeln, indem diese die Bogenfelder eines Moduls zu einer logisch zusammenhängenden Gruppe von Feldern zusammenfassen. Die Feldgruppen ergeben sich dabei indirekt aus der Definition von Plausibilitätsregeln.

Die Abbildung von Feldgruppenregeln in der Datenbank für PB-Dokumentation ist in Abschnitt B 2.4.7 erläutert.

1.3 Export der Daten aus der PB-Dokumentation

Die Exportdateien werden beim Leistungserbringer erstellt und dann an die zuständige Datenannahmestelle (DAS-KV) weitergeleitet. Datenannahmestellen nehmen die Datenlieferungen im Rahmen der oKFE-RL entgegen (Abschnitt A 2.1.1). Der Export und die Übermittlung der PB-Daten finden im XML-Format statt.

1.3.1 Erzeugen der Exportdatei

Die zu exportierenden Daten der PB-Dokumentation werden vom Dokumentationssystem in Exportdateien geschrieben und die entsprechenden Vorgänge (identifiziert durch Vorgangsnummern[10]) im absendenden Dokumentationssystem als „exportiert“ markiert.

Inhalte der Exportdatei
Der Inhalt der einzelnen Exportdateien orientiert sich an den in der „Übersicht über die Exportverfahren“ als zusammen exportierbar gekennzeichneten Modulen (Abschnitt A 2.1.1, Tabelle 13). Nur Datensätze aus dort entsprechend gekennzeichneten Modulen dürfen gemeinsam exportiert werden.

Hinweis

Dateien, in denen diese Trennung nicht eingehalten wird, werden von der Datenannahmestelle zurückgewiesen.

Export von Teildatensätzen
Beim Export einer PB-Dokumentation durch ein Dokumentationssystem werden die Inhalte der für den betreffenden Vorgang angelegten Teildatensätze hierarchisch in den XML-Code des passenden Basisbogens geschrieben und können nur gemeinsam mit dem Inhalt des Basisbogens exportiert werden. Die Struktur der Einbettung ist durch den Datentyp des Exportmoduls im Schema definiert.

Die Reihenfolge der Unterbögen (Teildatensätze) ist in der Spezifikationsdatenbank festgelegt. In der Tabelle Bogen ist in der Spalte SortierNr die Reihenfolge der Unterbögen bei der Erfassung und beim Export aufgeführt.

Anonymisierung
Für die Übermittlung der Programmbeurteilungsdaten an die jeweilige Datenannahmestelle müssen personenbezogene Daten durch die PB-Dokumentationssoftware anonymisiert werden.

[10] Datensatznummern.

Die Vorschriften zur Anonymisierung von Bogenfeldern sind in den Tabellen `ErsatzFeld` und `ErsatzFuerFeld` in der Spezifikationsdatenbank für PB-Dokumentation zu finden (Abschnitt B 2.5.2).

Aufbau der Exportdatei

Die innere Struktur der Exportdatei ergibt sich unter anderem aus der Datenfeldbeschreibung der einzelnen Module und den oben beschriebenen Anonymisierungsmaßnahmen zur Erfüllung datenschutzrechtlicher Vorgaben. Aufbauend auf dieser Beschreibung wird ein XML-Schema abgeleitet. Die Struktur der Exportdatei wird durch entsprechende XML-Schemata festgelegt (Abschnitt B 3).

Das XML-Schema beschreibt und definiert die Struktur des XML-Dokuments (Exportdatei) sowie den Inhaltstyp (Datentypen der einzelnen Bögen und Felder).

Die Exportdateien sind wie folgt aufgebaut:

- Header-Bereich enthält die Metadaten (administrative Daten)
- Body-Bereich enthält die tatsächlichen Daten der Datenlieferung

Hinweis

Es ist zu beachten, dass die Zeichenkodierung deklariert wird und diese der UTF-8-Kodierung entsprechen muss.

Sonderzeichen in XML

Das Und-Zeichen (&) und die spitzen Klammern (<, >) müssen geschützt werden, falls sie benötigt werden. Dies kann durch „&“ bzw. „<“ geschehen. Die schließende spitze Klammer (>) kann durch die Zeichenkette „>“ dargestellt werden. Um Attributwerten zu erlauben, sowohl das einfache als auch das doppelte Anführungszeichen zu enthalten, kann der Apostroph (‘) als „'“ und das doppelte Anführungszeichen („) als „"“ dargestellt werden.

Felder der Exportdatei

Einen Überblick über die zu exportierenden Felder eines Moduls liefert die Abfrage `Export-FelderFürEinModul`.[11]

Export von Listenfeldern

Alle Elemente von Listenfeldern werden exportiert, ohne die Nummer des Listenfelds (im Namen des Exportfelds) an den Namen des Listenfelds anzuhängen (zu Listenfeldern siehe Abschnitt B 2.3.3).

[11] Metadaten des Datencontainers sind nicht Teil der Spezifikationsdatenbank und werden daher nicht durch die Abfrage ausgegeben.

Beispiel:
XML-Darstellung eines Listenfeldes

```
<ENTLDIAG V="P52.4"/>
<ENTLDIAG V="P90"/>
<ENTLDIAG V="Z76.3"/>
<ENTLDIAG V="Z38.1"/>
<ENTLDIAG V="I63.9"/>
```

Zusatzfelder des Datenexports
Zusatzfelder und administrative Felder im Header, die nicht in der Datenfeldbeschreibung (Tabelle BogenFeld) eines Moduls enthalten sind, sind von der PB-Dokumentationssoftware zu füllen.[12]

Die Zusatzfelder sind in der Tabelle ZusatzFeld definiert. Das übertragene Speicherdatum DokAbschlDat (Datum des Dokumentationsabschlusses bzw. der Freigabe des Datensatzes für den Export) ist nicht Teil der Datenbank für Auswertungen und wird nur für organisatorische Zwecke verwendet.

Bei der Organisation im XML-Dokument ist weiter die Abfrage vExportZieleXml aus den administrativen Objekten zu berücksichtigen (Abschnitt B 2.7.1). Die Informationen in dieser Abfrage schließen aus, dass bestimmte Felder (wie PID-Daten) dem Element <qs_data> zugeordnet werden, indem anhand eines XPath-Ausdrucks (xmlXpath) auf die richtige Stelle im XML-Dokument (z. B. <patient>) verwiesen wird.

In den in Tabelle 13 in Abschnitt A 2.1 aufgeführten Module sind patientenidentifizierende Daten (PID) zu exportieren.

Organisation der Exportfelder im XML-Dokument
Die Exportfelder sind abhängig von der Modul- und Teildatensatzzugehörigkeit des Datensatzes im Element <qs_data>, einem Unterelement des Elements <case> unterzubringen (Element qs_data, Abschnitt B 3.4.4).

Stornierung
Um den Datensatz zu stornieren, muss <case>/<case_admin><action> auf „delete“ gesetzt werden. Die Datenannahmestelle wird dadurch veranlasst, den betreffenden Datensatz einschließlich aller Vorversionen und Teildatensätze als „storniert“ zu kennzeichnen. Der Stornovorgang wird in der Datenbestätigung protokolliert.

Der zu stornierende Datensatz muss ebenfalls eine hochgezählte/fortgeschriebene Versionsnummer enthalten, um die Stornierung unabhängig von der Reihenfolge der Verarbeitung von Datensätzen sicherzustellen. Ein Storno mit einer bereits verwendeten Versionsnummer wird zurückgewiesen (Bestätigungsstatus ERROR, Fehlerart DOPPELT). Ein Stornoversuch eines noch nicht übermittelten Datensatzes wird ebenfalls zurückgewiesen.

[12] Hier gilt demnach nicht der Grundsatz, dass Felder nicht vorbelegt sein dürfen.

Zur Stornierung eines Datensatzes (Vorgang) genügt der Export der entsprechenden administrativen Daten <case>/<case_admin>. Sowohl die PID (<patient>) als auch die PB-Daten (<qs_data>) des zu stornierenden Datensatzes sind nicht erneut zu übermitteln.

1.3.2 Datenprüfung

Zusätzlich zur bereits im Rahmen der Erfassung durchzuführenden feldbezogenen und -übergreifenden Plausibilitätsprüfungen kann nun nach dem Datenexport die gesamte Struktur der XML-Datei durch aus der Spezifikationsdatenbank abgeleitete Schemata geprüft werden (Abschnitt B 3.2).

Welches Schema für einen Leistungserbringer vorgesehen ist, zeigt Tabelle 7.

Tabelle 7: XML-Schemata zur Prüfung vor der Verschlüsselung

Richtlinie	Bereich	Schnittstelle	Schema
oKFE	kollektivvertraglich	LE interface_LE	2021_kv_pid_1.0_Export

Die einfachste Lösung für die Prüfung der Datenstruktur und der Plausibilität liegt in der Nutzung eines Datenprüfprogramms, das auf der Basis von XSLT die aus der Spezifikationsdatenbank PBDOK ausgeleiteten Plausibilitätsregeln in dem XML-Dokument prüft und Verstöße entsprechend im XML-Code dokumentiert. Ein solches Datenprüfprogramm setzt einen Standard für die Güte der Daten, der unbedingt einzuhalten ist. Die Datenprüfung muss an der Exportdatei vor der nachfolgend beschriebenen XML-Verschlüsselung vorgenommen werden.

Das Datenprüfprogramm ersetzt jedoch nicht die Verpflichtung der Softwareanbieter, schon bei der Eingabe der Daten eines Datensatzes, d. h. dokumentationsbegleitend, für die Einhaltung der Plausibilitätsregeln zu sorgen.

Details zur Verwendung können der Dokumentation des Datenprüfprogramms entnommen werden (Abschnitt B 4.2).

1.3.3 XML-Verschlüsselung

Nach Fertigstellung des Dokuments und seiner Prüfung mit dem Datenprüfprogramm sind – sofern vorhanden – die Datenbereiche (/patient), (/qs_data) mit einer XML-Verschlüsselung zu versehen.

Die Verschlüsselung erfolgt asymmetrisch mit einem öffentlichen Schlüssel. Die PB-Daten sind immer mit dem öffentlichen Schlüssel der Auswertestelle zu verschlüsseln. Die PID-Daten sind immer mit dem öffentlichen Schlüssel der Vertrauensstelle als Pseudonymisierungsstelle VST-PSN zu verschlüsseln.[13]

Tabelle 8 zeigt, welche XML-Elemente mit welchen Schlüsseln zu verschlüsseln sind:

[13] http://www.vertrauensstelle-gba.de/kontakt.htm

Tabelle 8: Asymmetrische Verschlüsselung der XML-Elemente

Richtlinie	Bereich	Datenart	XML-Elemente	Öffentlicher Schlüssel der
oKFE	kollektivvertraglich	PB-Daten	`qs_data`	BAS
		PID-Daten	`patient`	VST

Die öffentlichen Schlüssel der Vertrauensstelle (VST) und der Auswertungsstelle stehen den Anwendern als Spezifikationskomponente im Spezifikationspaket der Datenserviceinformationen zur Verfügung.

Diese Verschlüsselung baut auf dem W3C-XML-Encryption-Standard auf. Das IQTIG stellt ein Verschlüsselungsprogramm bereit, mit dem eine verfahrenskonforme XML-Verschlüsselung nach diesem Standard durchgeführt werden kann. Das Einbinden der Funktionen des Verschlüsselungsprogramms erfolgt u. a. über einen Befehlszeilenaufruf mit Parametern.

Details zur Verwendung können der Dokumentation der Verschlüsselungsprogramme entnommen werden (Abschnitt B 4.3).

1.3.4 Ausgangsvalidierung

Als letzte Maßnahme vor der Weiterleitung muss das Dokument gegen das Übertragungsschema auf Gültigkeit geprüft werden. Durch diese Prüfung wird sichergestellt, dass die richtigen Bereiche des XML-Codes verschlüsselt sind, und sie schließt aus, dass kritische Daten versehentlich unverschlüsselt den Leistungserbringer verlassen.

Das für einen bestimmten Leistungserbringer geltende Schema kann der gesonderten Beschreibung der Schema-Familie entnommen werden (Abschnitt B 3.2).

Tabelle 9: XML-Schemata zur Prüfung nach der Verschlüsselung

Richtlinie	Bereich	Schnittstelle	Schema
oKFE	kollektivvertraglich	`LE/DAS` `interface_LE_DAS`	`interface_LE_KV`

Hinweis

Die für die Ausgangsvalidierung zu verwendenden Schemata sind dieselben, wie die bei der Eingangsvalidierung der DAS verwendeten Schemata. Bei nicht bestandener Schemaprüfung wird die Annahme des Dokuments verweigert.

1.3.5 Beispiele für Exportdateien

Beispieldateien unterschiedlicher Schnittstellen können der Spezifikationskomponente „XML-Schema“ entnommen werden. Tabelle 10 zeigt, welche Beispiele für welche Schnittstelle relevant sind:

Tabelle 10: XML-Beispiele im gesonderten Beispielordner

Richtlinie	Bereich	Schnittstelle	Beispiele
oKFE	kollektivvertraglich	LE vor der Verschlüsselung	`./interface_LE`
		LE/DAS nach der Verschlüsselung	`./interface_LE_DAS`

1.4 Datenübermittlung

Im folgenden Abschnitt werden die abschließend zu unternehmenden Arbeitsschritte Dateibenennung und Datenversand beschrieben.

Vor der Datenübermittlung muss die XML-Verschlüsselung erfolgreich durchgeführt worden sein. Ob sie erfolgreich durchgeführt wurde, kann mit den entsprechenden Schemata für die Ausgangskontrolle geprüft werden.

Für die PB-Verfahren gemäß **oKFE-RL** werden die Datenlieferfristen in den Themenspezifischen Bestimmungen festgelegt. Die Datenübermittlung erfolgt quartalsweise, die Exportfristen pro Exportmodul sind der Tabelle `Exportmodul` der Datenbank zur PB-Dokumentation zu entnehmen.

1.4.1 Dateibenennung

Die Daten werden als XML-Datei an die VST weitergeleitet. Die Exportdatei muss nach einem bestimmten Schema benannt werden (Abschnitt A „Benennung der Exportdateien“ auf S. 54).

Beispiel:
PB-Daten eines LE

47d16341-9e27-4e75-a27e-b791fbbd2dc8_**Q_LE**.xml

1.4.2 Datenversand via gesicherter Schnittstellen – Arztpraxen, MVZ und medizinische Labore – für ambulant kollektivvertraglich erbrachte Leistungen

In der Regel wird derselbe Übertragungskanal wie für die Übertragung der Abrechnungsdaten, beispielweise KV-Connect, genutzt. KV-Connect wird als Anwendung im sicheren Netz der KVen (SNK) betrieben und ermöglicht eine sichere Ende-zu-Ende-Verschlüsselung sensibler und vertraulicher Patientendaten im Gesundheitswesen. Eine umfassende Sicherheitsanalyse[14] des TÜV Rheinland bestätigt, dass KV-Connect alle Kriterien des Anforderungskataloges zur Zertifizierung „Geprüfter Datenschutz“ erfüllt. Weiterhin wurden im Rahmen der Zertifizierung von KV-

[14] https://www.kv-telematik.de/partner-und-softwarehaeuser/weitere-infos/tuev-zertifizierung/

Connect als Anwendung im SNK die Anforderungen der Richtlinie an die KV-Applikationen[15] erfüllt. Diese Richtlinie orientiert sich an den Anforderungen des BSI Grundschutzkataloges.

Hinweis

Anders als bei der Datenübermittlung über nicht gesicherte Übertragungswege (E-Mail), muss die Exportdatei nicht mit dem Verschlüsselungsprogramm TPacker transportverschlüsselt werden.

1.4.3 Zusammenfassung Datenversand

Tabelle 11 fasst die zuvor beschriebenen Datenübertragungswege zusammen:

Tabelle 11: Zuständige Datenannahmestelle

Richtlinie	Bereich	Datenannahmestelle	Datenübertragung	Symmetrische Verschlüsselung
oKFE	kollektivvertraglich	DAS-KV	Abhängig vom Übertragungskanal der KV (z. B. KV-Connect, D2D)[16]	Nein

1.5 Rückprotokollierung

Nach erfolgreicher Eingangsverarbeitung durch die DAS erhält der Leistungserbringer von der DAS auf dem Eingangskanal[17] eine Empfangsbestätigung, die den Erhalt der Exportdatei bestätigt. Nach erfolgreicher Prüfung der XML-Datei erhält der Leistungserbringer zudem ein Datenflussprotokoll mit allen Prüfergebnissen der DAS und ggf. der VST und der BAS.

Hinweis

Da die Protokolle dem LE auf dem Eingangskanal von der DAS zur Verfügung gestellt werden, werden die per E-Mail übermittelten Protokolle von der DAS mit demselben Passwort für Transportverschlüsselung der PB-Daten verschlüsselt und nach demselben Schema benannt.

Das Protokoll ist von der Dokumentationssoftware einzulesen und dem Anwender darzustellen. Gegebenenfalls müssen die fehlerhaften Datensätze korrigiert und erneut übermittelt werden.

Die genaue Struktur ist Abschnitt A 2.3 zu entnehmen.

Regelungen für ein Vorgehen bei Verarbeitungsabbrüchen im besonderen Fehlerfall

Diese Regelungen dienen dazu, ein abgestimmtes Vorgehen durch alle am Datenfluss beteiligten Stellen zu etablieren, um die Datenbestände in allen Instanzen zu konsolidieren, wenn es zu unerwarteten Störungen in der Verarbeitung von Datenlieferungen in der DAS, VST oder BAS kommt. Solche Störungen sind selten und stellen einen Ausnahmefall dar.

[15] http://www.kbv.de/media/sp/KBV_SNK_RLKV_KV_Apps_V3_0.pdf
[16] In der Regel sollen die Daten analog zum Übertragungsweg der Abrechnungsdaten übermittelt werden.
[17] Es können Abweichungen auftreten. Nähere Informationen sind bei der zuständige KV einzuholen.

Solche Störungen können in äußerst seltenen Fällen aufgrund einer fehlerhaften automatisierten Verarbeitung von Datenlieferungen in der DAS, VST oder BAS dazu führen, dass fehlerhafte Datenflussprotokolle versandt werden. Dadurch können beim Leistungserbringer unter Umständen mehrere Datenflussprotokolle zu einer GUID eintreffen, die widersprüchliche Einstufungen der Fälle enthalten. Um diesen Umstand an allen am Datenfluss beteiligten Stellen aufzulösen, müssen die involvierten Vorgänge erneut verarbeitet werden.

Das präferierte Vorgehen in dieser Situation ist ein erneuter Versand aller betroffenen Datensätze, die unter der betroffenen GUID gesandt wurden, durch den Leistungserbringer in einer höheren Version. Dieser Versand muss nach Rücksprache mit der Stelle, bei der die fehlerhafte Verarbeitung aufgetreten ist, und unter Einbindung der anderen Stationen im Datenfluss geschehen und wird deshalb telefonisch von der zuständigen DAS beim betroffenen Leistungserbringer initiiert. Für den erneuten Versand ist beim Leistungserbringer ein Export aller betroffenen Vorgangsnummern einschließlich der Erhöhung der zugehörigen Versionsnummer unter einer neuen GUID durchzuführen. Das im Zuge des Exports erstellte XML-Dokument erhält alle Fälle als Update in der jeweils höchsten Versionsnummer, die das Softwaresystem beim Leistungserbringer erstellt, und erhält eine neue und damit unverbrauchte GUID. Somit können alle Datensätze im Datenfluss über alle folgenden Instanzen (DAS, VST, BAS) einschließlich der zugehörigen Rückprotokollierung mittels des Datenflussprotokolls bis hin zum Leistungserbringer regulär verarbeitet werden.

Sollte der Leistungserbringer nicht in der Lage sein, die Daten erneut zu exportieren, so ist der alternative Weg zu wählen, bei dem ein durch alle Instanzen bereits verarbeitetes Dokument (einschließlich Rückprotokollierung mittels Datenflussprotokoll bis hin zum Leistungserbringer) – und somit eine bereits verarbeitete GUID – erneut verarbeitet wird. Das genaue Vorgehen ist im Abschnitt A 2.3 beschrieben. Es bedarf ebenfalls einer telefonischen Abstimmung aller Stationen im Datenfluss. Auf Leistungserbringerseite muss die Software in der Lage sein, zu einer GUID mehr als ein Datenflussprotokoll aufzunehmen, um die Einstufung der Daten gemäß dem zweiten Datenflussprotokoll analog zu den anderen Verarbeitungsstellen anzupassen.

1.6 Zusammenfassung

Tabelle 12 stellt einen Überblick über die Aufgabenbereiche der Leistungserbringer in Bezug auf die PB-Dokumentation dar.

Weitere Informationen zu den Datenflüssen können Abschnitt A 2.1 entnommen werden.

Tabelle 12: Aufgaben der Leistungserbringer in Bezug auf die PB-Dokumentation

Prozesse	Unterprozesse	Anmerkungen
Auslösung		Automatisch softwaregestützter Prozess durch Kommunikation mit dem Informationssystem. Implementierung der Auslösekriterien
Erfassung		Softwaregestützte Dokumentation.

Prozesse	Unterprozesse	Anmerkungen
		Implementierung des Spezifikationsregelwerks (PBDOK). Übernahme von Daten aus dem Informationssystem und manuelle Ergänzungen
Export	Export in XML-Format	Umsetzung der Vorgaben der PBF und der XML-Schemata oKFE-RL: verfahrensspezifisch
	Konforme Benennung der XML-Datei	
	Datenprüfung vor der Verschlüsselung	Mittels DPP, XML-Schemata oder eigener Tools
	Verschlüsselung der XML-Elemente `qs_data` und `patient`	Mittels XPacker
	Ausgangskontrolle (Schema)	Mittels XML-Schema für die Schnittstelle LE-DAS (mit oder ohne DPP)
Datenübermittlung	KV-Schnittstellen	Mit der zuständigen KV abzustimmen.
Rückprotokollierung	Erhalt einer Empfangsbestätigung	Bestätigung des Erhalts der Exportdatei
	Erhalt der Datenflussprotokolle von der DAS	Ergebnisse über alle Prüfungen durch die DAS und ggf. durch die VST und die BAS
	Einlesen der Protokolle	Ausgabe der Fehlermeldungen und ggf. Vornahme Korrekturen

2 Allgemeine Regelungen zur Datenübermittlung

In diesem Kapitel werden allgemeingültige Regelungen für die Datenflüsse, Datenübermittlung, Datenprüfung und Rückprotokollierung beschrieben.

2.1 Datenfluss

Die Zuordnung von Datenpaketen zu einem Datenfluss ist vom Abrechnungskontext, von den zu erhebenden Modulen sowie davon abhängig, ob patientenidentifizierende Daten (PID) erhoben, exportiert und pseudonymisiert werden müssen. Verfahren gemäß oKFE-RL beinhalten immer PID (Übersicht über die Verfahren, siehe Tabelle Tabelle 13). In den PB-Verfahren ist auch die Durchführung einer Pseudonymisierung von leistungserbringeridentifizierenden Daten bei der Zuordnung zu beachten.

2.1.1 Datenfluss der PB-Daten

Module mit PID:

Datensätze zu Verfahren gemäß oKFE-RL (ambulant erbrachte Leistungen) werden über die zuständigen DAS-KV und die VST an die BAS übermittelt.

Alle Module, die PID enthalten, werden durch die DAS nach der Plausibilitätsprüfung und ggf. der Pseudonymisierung der leistungserbringeridentifizierenden Daten an die Vertrauensstelle als Pseudonymisierungsstelle (VST-PSN) gesendet, die die PID pseudonymisiert.

Die folgende Abbildung gibt einen Überblick über denDatenfluss:

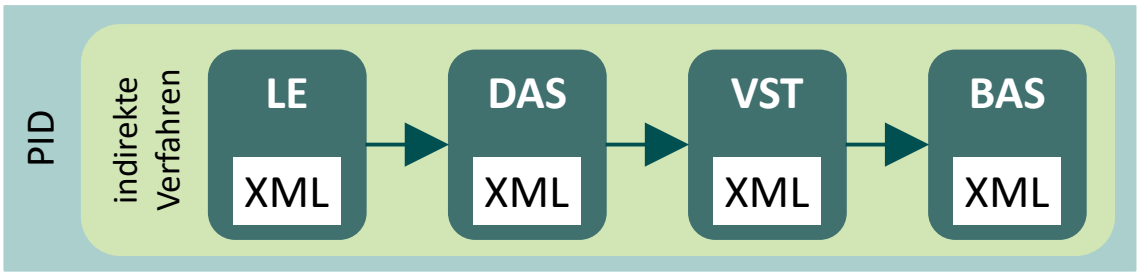


Abbildung 6: Übersicht der Datenflüsse direkte/indirekte PID-/Nicht-PID-Verfahren

Mit den Daten eines oder mehrerer abgeschlossener Module wird vom Dokumentationssystem eine XML-Exportdatei erzeugt, in eine Transaktionsdatei komprimiert und über einen sicheren Übertragungsweg (z. B. KV-Connect) oder als verschlüsselte E-Mail Anhang an die zuständige DAS übermittelt.

Die DAS müssen Dokumente zurückweisen, wenn Datensätze, die nicht in ihren Zuständigkeitsbereich fallen, oder Mischlieferungen von Modulen unterschiedlicher Datenflüsse enthalten sind. Auch sind Mischlieferungen unterschiedlichen oKFE-Verfahren von den Datenannahmestellen abzulehnen.

Folgende Module können hinsichtlich der verfahrensspezifischen Leistungserbingerpseudonymisierunggemeinsam übermittelt werden:

Tabelle 13: Übersicht über die Exportverfahren

PB-Verfahren	Exportmodul
Darmkrebs (DK)	DKK DKI
Zervixkarzinom (ZK)	ZKP ZKA ZKZ ZKH

2.1.2 Datenfluss der Rückprotokolle

Der Datenfluss verläuft mit einer Empfangsbestätigung vom jeweiligen Empfänger (Abschnitt A 2.3.2) zum jeweiligen Absender.

Mit dem Abschluss der Datenverarbeitung in der BAS erfolgt eine weitere Rückprotokollierung – das sog. Datenflussprotokoll – durch die BAS über die DAS zum Leistungserbringer (siehe Abbildung 7). Die Vertrauensstelle (VST-PSN) wird in diesem Datenfluss übersprungen.

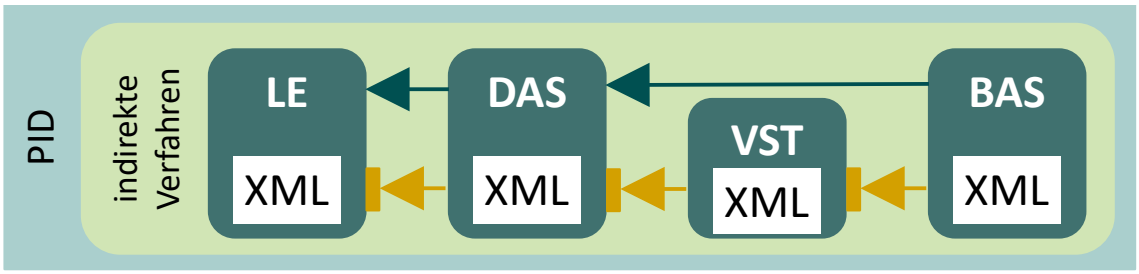


Abbildung 7: Übersicht der Datenflüsse der Rückprotokollierung (PB-Verfahren)

2.2 Datenübermittlung

In diesem Abschnitt werden allgemeine Reglungen in Bezug auf die gesicherte Datenübertragung, Registrierung und Verschlüsselung beschrieben.

2.2.1 Gesicherte Datenübertragung

Die Datenübertragung darf ausschließlich über gesicherte Übertragungskanäle erfolgen. Im Folgenden werden die verschiedenen Übertragungswege beschrieben.

Übertragungswege

Eine an der Datenübertragung beteiligte Einrichtung kann auf zwei verschiedenen Übertragungswegen Daten entgegennehmen:

- Verschlüsselung und Authentifizierung des Übertragungswegs:

Die Daten werden auf Übertragungswegen versandt, die eine Authentifizierung des Absenders und eine Transportverschlüsselung automatisch implizieren, wie beispielsweise KV-Connect und KV-SafeNet. Eine zusätzliche Registrierung des Absenders ist nicht nötig, da dieser so bereits eindeutig identifiziert werden kann.

- Verschlüsselung und Authentifizierung des Datenpakets:
  Die Auswahl des Übertragungswegs ist frei. In der Regel kommen Datenträger und E-Mail-Übertragungen zum Einsatz. Die Transportverschlüsselung und der E-Mail-Versand sind in dieser Spezifikation geregelt. Die Authentifizierung und Verschlüsselung wird über den vorgelagerten Prozess einer Registrierung ermöglicht.

Registrierung von Datenannahmestellen bei der Auswertungsstelle

Die DAS-KV müssen bei der Auswertungsstelle registriert sein.

Registrierung von Datenannahmestellen bei der Vertrauensstelle

Die DAS-KV müssen zudem neben der Registrierung bei der BAS auch bei der Vertrauensstelle registriert sein. Hier ist darauf zu achten, dass die von der VST vergebene Registrierungsnummer der Registriernummer der DAS bei der BAS entspricht. Für die Transportverschlüsselung wird ein anderer geheimer Schlüssel vereinbart.

Registrierung von Softwareanbietern beim IQTIG für Testzwecke

Softwareanbieter, die an Testbetrieben teilnehmen wollen, müssen sich bei den Stellen registrieren, die den Test-Datenservice bereitstellen.

Eindeutige Kennzeichnung der XML-Exportdateien

Jede Exportdatei wird durch eine universell eindeutige ID (GUID) von der PB-Software gekennzeichnet.

Ein Globally Unique Identifier (GUID) ist eine global eindeutige Zahl mit 128 Bit, die eine Implementierung des Universally Unique Identifier Standards (UUID) darstellt.

GUIDs haben das Format XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX, wobei jedes X für ein Zeichen aus dem Hexadezimalsystem steht und damit eine Ziffer 0–9 oder ein Buchstabe A–F sein kann.

Erläuterung zur GUID:

- Die GUID wird im Exportprozess von der PB-Software einer bestimmten Exportdatei zugewiesen.
- Das registrierte Dokumentationssystem ordnet jeder an eine Datenannahmestelle übermittelten Exportdatei eine eindeutige GUID zu.
- Diese GUID wird im XML-Code des Dokuments als ID gesetzt. Sie muss bei dateibasierten Übertragungsverfahren in der Dateibenennung verwendet werden.
- Eine GUID wird durch eine gelungene Transaktion zwischen den jeweils beteiligten Übertragungspartnern verbraucht. Jede Datenannahmestelle muss dafür sorgen, dass ein eindeutig über die GUID identifizierbares Dokument nur einmal denselben Verarbeitungsschritt durchlaufen kann. Anderenfalls ist die Verarbeitung mit einem entsprechenden Fehlerprotokoll

abzulehnen.

Achtung

Aufgrund von fehlerhaften, automatisierten Verarbeitungen in der DAS, VST oder BAS kann der Ausnahmezustand entstehen, dass ein bereits verarbeitetes Dokument nebst Rückprotokollierung bis hin zum Leistungserbringer erneut verarbeitet werden muss.

Für den Fall, dass ein erneuter Datenfluss von PB-Daten ab der zuständigen DAS erfolgt, muss auch die Dokumentationssoftware beim Leistungserbringer über die Möglichkeit verfügen, mehr als ein Datenflussprotokoll zu einer GUID aufnehmen können.

Identifizierung von Datensätzen

Die Vorgangsnummer (auch Datensatz-ID oder ID genannt) kennzeichnet in eindeutiger Weise jeden dokumentierten Vorgang eines registrierten Dokumentationssystems.

Im einfachsten Fall könnten die Vorgangsnummern jeweils um 1 inkrementiert werden, wenn ein neuer Datensatz angelegt wird. Die Vorgangsnummer ist daher allein ein Merkmal des PB-Dokumentationssystems, um einen Datensatz innerhalb des registrierten Dokumentationssystems eindeutig identifizieren zu können. Aus diesem Grund wäre es falsch, die Patientenidentifikationsnummer oder die Fallnummer aus dem AIS/LIS zu verwenden bzw. zu pseudonymisieren.

Eine Vorgangsnummer darf keine Rückschlüsse auf Personen ermöglichen. In der Vorgangsnummer darf z. B. nicht das Geburtsdatum enthalten sein.

Die PB-Dokumentationssoftware verwaltet jahresübergreifend die Vorgangsnummern der PB-Dokumentationen. Sie soll dem Leistungserbringer eine Zuordnung der Vorgangsnummern zu seinen internen Fall- oder Patientennummern (vgl. nicht übermitteltes Datenfeld IDNRPAT) ermöglichen. Zum Zweck der Datenvalidierung und der Qualitätsverbesserung muss es dem Leistungserbringer möglich sein, über die Vorgangsnummer Zugang zur Fall- bzw. Patienten-Akte zu bekommen.

Annahme oder Ablehnung unterschiedlicher Versionen eines Datensatzes

Bei der Datenannahmestelle eingehende Datensätze werden anhand der Kombination aus Registrierungsnummer und Vorgangsnummer als ein Vorgang identifiziert. Der für einen bestimmten Vorgang gespeicherte Datensatz kann nur durch eine neuere Version (mit höherer Versionsnummer) überschrieben werden.[18]

Unterschiedliche Versionen eines Datensatzes müssen demselben Primärmodul[19] zugeordnet sein. Ein Datensatz mit einer Vorgangsnummer aus derselben Registrierung, die bereits unter einem anderen Modul eingeschickt wurde, wird abgelehnt.

[18] Gegebenenfalls ist der geänderte Datensatz mit einer neuen Versionsnummer zu übermitteln.
[19] Jeder Datensatz ist einem Primärmodul zugeordnet.

Benennung der Exportdateien

Die Daten werden als XML-Datei an die VST gesendet. Die Exportdatei muss nach dem folgenden Schema benannt werden:

Syntax: `<GUID>_<Inhaltskennung><Protokolltyp>_<Rolle Absender>.xml`

Tabelle 14: Benennungselemente der Exportdateien

Element	Bedeutung
`GUID`	Die verwendete GUID ist die im Dokument verwendete ID des Datenpakets (Abschnitt A „Eindeutige Kennzeichnung der XML-Exportdateien“ auf S. 52).
`Inhalts-kennung`	Q → PB-Daten R → Routinedaten
`Protokoll-typ`	T → Transaktionsprotokoll bzw. Empfangsbestätigung D → Datenflussprotokoll
`Rolle Ab-sender`	LE → Leistungserbringer DAS → Datenannahmestelle VST → Vertrauensstelle BAS → Auswertungsstelle

Beispiele:

47d16341-9e27-4e75-a27e-b791fbbd2dc8_Q_LE.xml
(PB-Daten eines Leistungserbringers)

47d16341-9e27-4e75-a27e-b791fbbd2dc8_QD_DAS.xml
(PB-Daten-Datenflussprotokoll einer DAS)

47d16341-9e27-4e75-a27e-b791fbbd2dc8_QT_DAS.xml
(PB-Daten- Empfangsbestätigung einer DAS)

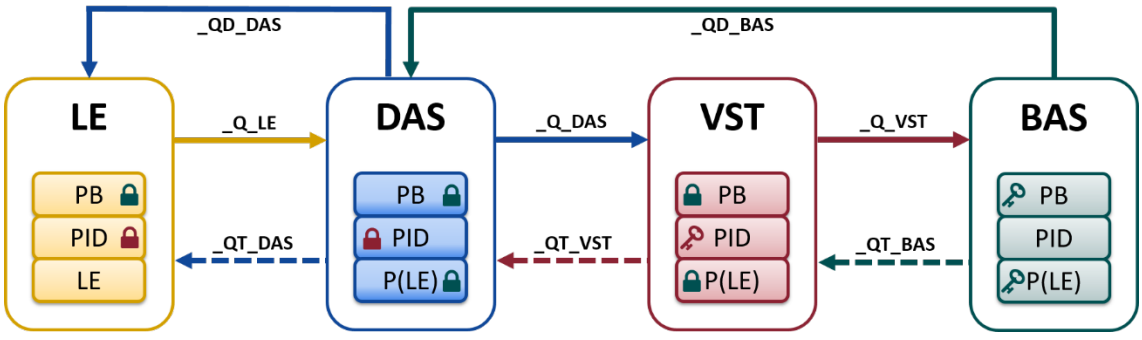


Abbildung 8: Übersicht über die einzusetzenden Suffixe in PB-Verfahren

Die Verschlüsselung

Für die Verschlüsselung der PB-Daten wurde der W3C-Standad „XML-Encryption“ eingesetzt. XML-Encryption bietet eine adäquate Lösung, um komplexe Anforderungen an die Sicherheit des Datenaustausches zu erfüllen. Mit XML-Encryption können unterschiedliche Dokumentenabschnitte (XML-Knoten) für unterschiedliche Datenempfänger mit unterschiedlichen, öffentlichen Schlüsseln verschlüsselt werden, ohne diese Dokumentabschnitte, die logisch miteinander verbunden sind, in unterschiedlichen Dokumenten an die jeweiligen Datenempfänger zu senden. Auf diese Weise bleibt der Zusammenhang der Daten erhalten. Dennoch ist jeder verschlüsselte Dokumentenabschnitt nur für den Besitzer des jeweiligen Schlüssels lesbar.

Des Weiteren werden bestimmte Elemente (Tags) innerhalb der XML-Datei gezippt bzw. komprimiert, damit die Dateigröße in einem überschaubaren Rahmen bleibt.

Abschließend wird für die Datenübermittlung noch eine Transportverschlüsselung angewendet, die die gesamte Datei verschlüsselt und vor unberechtigten Zugriffen schützt.

Verschlüsselungsverfahren

XML-Encryption-Spezifikation
Zusätzlich zur Verschlüsselung von einzelnen XML-Knoten und deren Unterelementen in einem XML-Dokument, definiert die XML-Encryption-Spezifikation weitere Möglichkeiten, wie XML-Dokumente ver- und entschlüsselt werden können:

- Verschlüsselung des gesamten XML-Dokuments
- Verschlüsselung des Inhalts eines XML-Elements
- Verschlüsselung für mehrere Empfänger
- Verschlüsselung eines einzelnen Elements und seiner Unterelemente

Verschlüsselungsalgorithmen

Symmetrische Verschlüsselung
Die symmetrische Verschlüsselung wird mit einem einzigen Schlüssel durchgeführt, d. h. zur Ver- und Entschlüsselung wird derselbe Schlüssel von Sender und Empfänger verwendet.

Der verwendete Algorithmus für die symmetrische Verschlüsselung: AES128
Quelle: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Asymmetrische Verschlüsselung
Verschlüsselung mit asymmetrischen Schlüsseln, d. h. einem Schlüsselpaar (öffentlicher und geheimer Schlüssel). Die Daten werden mit dem frei verfügbaren, öffentlichen Schlüssel („Public Key“) verschlüsselt und können nur mit dem privaten Schlüssel („Private Key“) entschlüsselt werden. Diese Verschlüsselung ist zeitaufwendiger, aber auch sicherer, da kein geheimer Schlüssel übertragen werden muss.

Der verwendete Algorithmus für die asymmetrische Verschlüsselung: RSA mit 2048-Bit.
Quelle: http://www.w3.org/TR/xmlenc-core/

Hybride Verschlüsselung
Das grundlegende Szenario dieser Verschlüsselung besteht aus folgenden Einzelschritten:

- Ein zufälliger symmetrischer Schlüssel wird erzeugt.
- Mit diesem Schlüssel wird ein XML-Element (z. B. PB-Daten) verschlüsselt.
- Der Schlüssel wird nun mit dem „Public Key“ des Empfängers (z. B. BAS) verschlüsselt.
- Der mit dem „Public Key“ verschlüsselte symmetrische Schlüssel wird dem Empfänger zusammen mit den verschlüsselten Daten übergeben.
- Der Empfänger entschlüsselt den chiffrierten Schlüssel mit seinem „Private Key“ und erhält so den symmetrischen Schlüssel, mit dem die Daten verschlüsselt wurden.
- Mit diesem symmetrischen Schlüssel entschlüsselt der Empfänger die verschlüsselten Daten.

Für die Verschlüsselung der XML-Knoten wird die hybride Verschlüsselung nach dem W3C-Standard „XML Encryption Syntax and Processing“ verwendet, da diese Verschlüsselung die Vorteile der symmetrischen und asymmetrischen Verschlüsselungsverfahren – nämlich die Schnelligkeit und die Sicherheit – kombiniert.

Als Verschlüsselungsalgorithmen werden `AES128` für die symmetrische Verschlüsselung der XML-Elemente und `RSA mit 2048-Bit` für die asymmetrische Verschlüsselung des generierten symmetrischen Schlüssels verwendet.

Übermittlung der Daten im Datenfluss

Die folgenden Aspekte der Datenübermittlung werden spezifiziert:
- Datenpaket, innere Struktur
- Datenpaket, äußere Struktur

Während die innere Struktur immer eingehalten werden muss, ist die äußere Struktur nur dann einzuhalten, wenn als Übertragungsweg der Versand per E-Mail oder auf einem Datenträger gewählt wird.

Die äußere Struktur dient allein einem sicheren Übertragungsprozess. Dieser Übertragungsprozess ist für den Austausch von Dateien (z. B. per E-Mail) spezifiziert. Die innere Struktur des Datenpakets muss eingehalten werden und es müssen datenschutzrechtlich unbedenkliche Übertragungsverfahren gewählt werden. Eine Abweichung von der Übertragung mittels E-Mail soll im Konsens zwischen den Übertragungspartnern getroffen werden, wenn die Unbedenklichkeit der Übertragung sichergestellt ist.

Die innere Struktur jedes Datenpakets stellt alle notwendigen Metainformationen bereit, um dieses eindeutig zuzuordnen. Die Unbedenklichkeit der Übertragung muss nachgewiesen werden. Die innere Struktur wird durch ein XML-Schema (Übertragungsschema) definiert.

Zur äußeren Struktur gehören Festlegungen zu Dateibenennung, Transportverschlüsselung, Archivierung und Archivbenennung.

Ausgangsvalidierung gegen das Übertragungsschema

Als letzte Maßnahme vor der Weiterleitung des Dokuments muss die innere Struktur des Dokuments gegen das Übertragungsschema auf Gültigkeit geprüft werden.

Die Vorteile der Ausgangsvalidierung:
- Sicherstellung der Datenintegrität nach Verarbeitung der Daten

- frühe Feststellung von Fehlerquellen in der eigenen Datenverarbeitung
- Entlastung des nachfolgenden Datenservice von nicht validen Daten
- Vermeidung des Versands von Daten, die gegen den Datenschutz verstoßen

Durch diese Prüfung wird sichergestellt, dass die richtigen Bereiche des XML-Kodes verschlüsselt sind und ausgeschlossen ist, dass kritische Daten versehentlich unverschlüsselt die nächste Stelle im Datenfluss erreichen. Sie schließt ebenfalls von vornherein aus, dass Daten an den nachfolgenden Datenservice übermittelt werden, die dieser nicht verarbeiten kann.

Das an einer Übertragungsstelle gültige Schema, kann der Dokumentation über die Schemafamilie entnommen werden.

Beispiel:
Der Leistungserbringer (kollektivvertraglich) verwendet das Schema zur Schnittstelle LE-DAS:
`interface_LE_DAS\interface_LE_KV.xsd`
Die Datenannahmestelle verwendet das Schema zur Schnittstelle DAS-VST:
`interface_DAS_VST\interface_DAS_VST.xsd`

Die Validierung kann über zahlreiche frei verfügbare Tools erfolgen.[20] Für diese Validierung kann auch das Datenprüfprogramm des IQTIG verwendet werden.

2.2.2 Abgrenzung von Test-, Probe- und Regelbetrieb

Im Folgenden wird beschrieben, welche Testmöglichkeiten es im Datenfluss gibt, wie diese genutzt werden können und sollen und wie sich diese voneinander und vom Produktivbetrieb abgrenzen lassen.

Definition Test- und Echtdaten

Es wird zwischen Test- und Echtdaten unterschieden.

Als **Testdaten** werden PB-Daten bezeichnet, die der Datenstruktur der Spezifikation, aber nicht Datensätzen von realen Personen entsprechen.

Als **Echtdaten** werden solche PB-Daten bezeichnet, die der Datenstruktur der Spezifikation und Datensätzen von realen Personen entsprechen, sodass die besonderen Regeln des Datenschutzes auf sie zutreffen.

Datenziele

Alle auswertenden Einrichtungen (DAS, BAS) stellen drei Datenziele zur Verfügung. Diese Datenziele werden als
- Testdatenpool
- Probedatenpool
- Echtdatenpool

[20] http://www.w3.org/XML/Schema.

bezeichnet.

Die Datensicherheit bemisst sich in allen Datenpools an den Maßstäben, die für personenbezogene Daten gelten. Das bedeutet, dass prinzipiell alle drei Datenziele geeignet sind Echtdaten entgegenzunehmen, weil die Sicherheit gewährleistet wird.

Das Datenziel wird im Headerbereich des XML-Dokuments festgelegt (`/document/data_target`).

Testdatenpool
Der Testdatenpool ist sowohl für generierte Testdaten als auch für Echtdaten, die zu Testzwecken versendet werden, bestimmt. Für Echtdaten gilt aber, dass diese auch zu Testzwecken ausschließlich über Produktivinstanzen (siehe unten) geschickt werden dürfen.

Für Daten, die an den Testdatenpool geschickt werden, gibt es keine Erhaltungsregel. Ein Testdatenpool kann ohne vorherige Absprache geleert werden. Die Daten werden spätestens nach einem Jahr gelöscht.

Der Testdatenpool wird nicht inhaltlich ausgewertet. Er soll Testungen von Datenbankoperationen des automatisierten Datenservice und die Simulation echter Rückprotokollierungen ermöglichen.

Probedatenpool
Der Probedatenpool ist für Echtdaten bestimmt, die in Sonderfällen für Probeauswertungen genutzt werden. Diese Daten werden nach Abschluss der Auswertung oder spätestens nach einem Jahr gelöscht.

Dieser Datenpool ist nur für Machbarkeitsprüfungen, Probebetriebe und Sonderexporte relevant.

Echtdatenpool
Der Echtdatenpool ist für Echtdaten bestimmt, die dem Bundesdatenpool zugeführt werden. Diese Daten werden entsprechend der Regeln für den Bundesdatenpool gepflegt. Alle Dokumentationen für den Regelbetrieb werden dem Echtdatenpool zugeführt.

Datenservice

Als ein Datenservice wird eine automatisierte Datenverarbeitung unter einem von außen erreichbaren Endpunkt (z. B. E-Mail-Adresse oder URL) verstanden.

Für jede Instanz eines Datenservice stehen folgende Eigenschaften fest:

- wer der Anbieter des Datenservice ist
- welchen Endpunkt er bedient (E-Mail-Adresse, URL)
- welche Übertragungsarten er bedient (E-Mail, Restservice, KV-Connect etc.)
- welcher öffentliche Schlüssel zu verwenden ist
- welche Bundesländer/Regionen bedient werden
- welche Einsender-Gruppen bedient werden (LE-amb, SWA, DAS, KK)
- welche Spezifikation bedient wird (Spezifikationskennung z. B. 2020_oKFE_RB_XML)
- welche Module bedient werden

- welche Datenziele bedient werden (Testdatenpool, Probedatenpool, Echtdatenpool)
- an welche nachfolgenden Datenservice die verarbeiteten Daten weitergereicht werden
- ob Datensicherheit garantiert wird

Es wird bei den Datenservice grundsätzlich zwischen Testinstanzen und Produktivinstanzen unterschieden. Diese Unterscheidung hat wesentliche Auswirkungen auf bestimmte Eigenschaften eines Datenservice.

Produktivdatenservice

Produktivdatenservice sind Datenservice, mit denen alle Datenpools erreicht werden können und deren Betreiber für die Datensicherheit garantieren muss.

Wegen der Datensicherheit dürfen Daten aus einem Produktivdatenservice auch nur an einen anderen registrierten Produktivdatenservice oder einen registrierten Leistungserbringer weitergeleitet werden.

Eine Testung kann auch über Produktivdatenservice erfolgen. Dabei ist unerheblich ob Echt- oder Testdaten verwendet werden. Ausschlaggebend ist, dass in dem Datenpaket, welches zur Testung verwendet werden soll, als Datenziel „Testdatenpool“ eingetragen ist.

Testdatenservice

Ein Testdatenservice ist ein Datenservice, der von der betreibenden Einrichtung vorgehalten wird, um Daten zu Testzwecken anzunehmen und gleichzeitig um die Verarbeitung der Daten durch den Datenservice zu testen.

Ein Testdatenservice beruht auf Spezifikationen des Probe- oder Regelbetriebs. In Ausnahmefällen kann auch eine Spezifikation für den Testbetrieb die Grundlage bilden, wenn es keine entsprechenden Spezifikationen für die zu testenden Aspekte gibt.

Die angenommenen Daten dürfen unabhängig von der Ausweisung des Datenziels durch den Absender nie in einen Echt oder Probedatenpool, sondern immer nur einem Testdatenpool zugeführt werden.

Anders als die garantierte Datensicherheit des Testdatenpools (siehe oben), wird die Datensicherheit des Übertragungsweges über Testdatenservice nicht garantiert. Deswegen dürfen an Testdatenservice immer nur Testdaten (also generierte Daten ohne Bezug zu realen Personen) geliefert werden.

Es wird empfohlen Testinstanzen von DAS und VST ab zwei Monaten nach Veröffentlichung der ersten Spezifikation eines Erfassungsjahres (Release V01) bereitzustellen. Im Rahmen aller weiteren Releases für ein Erfassungsjahr wird die Bereitstellung der jeweiligen Testinstanz nach einer Woche empfohlen.

Datenservice und Testdatenservice im Datenfluss

Datenpakete dürfen von Testinstanzen nur an nachgeschaltete Testinstanzen anderer Einrichtungen weitergereicht werden. Damit wird verhindert, dass versehentlich Testdaten in Echtdatenpools aufgenommen werden.

Datenpakete dürfen von Produktivinstanzen nur an nachgeschaltete Produktivinstanzen anderer Einrichtungen weitergeleitet werden. Damit wird verhindert, dass mögliche schutzbedürftige personenbezogene Daten in Testinstanzen ohne Datenschutzgarantie verarbeitet werden.

Daraus ergeben sich die folgenden Datenflüsse:

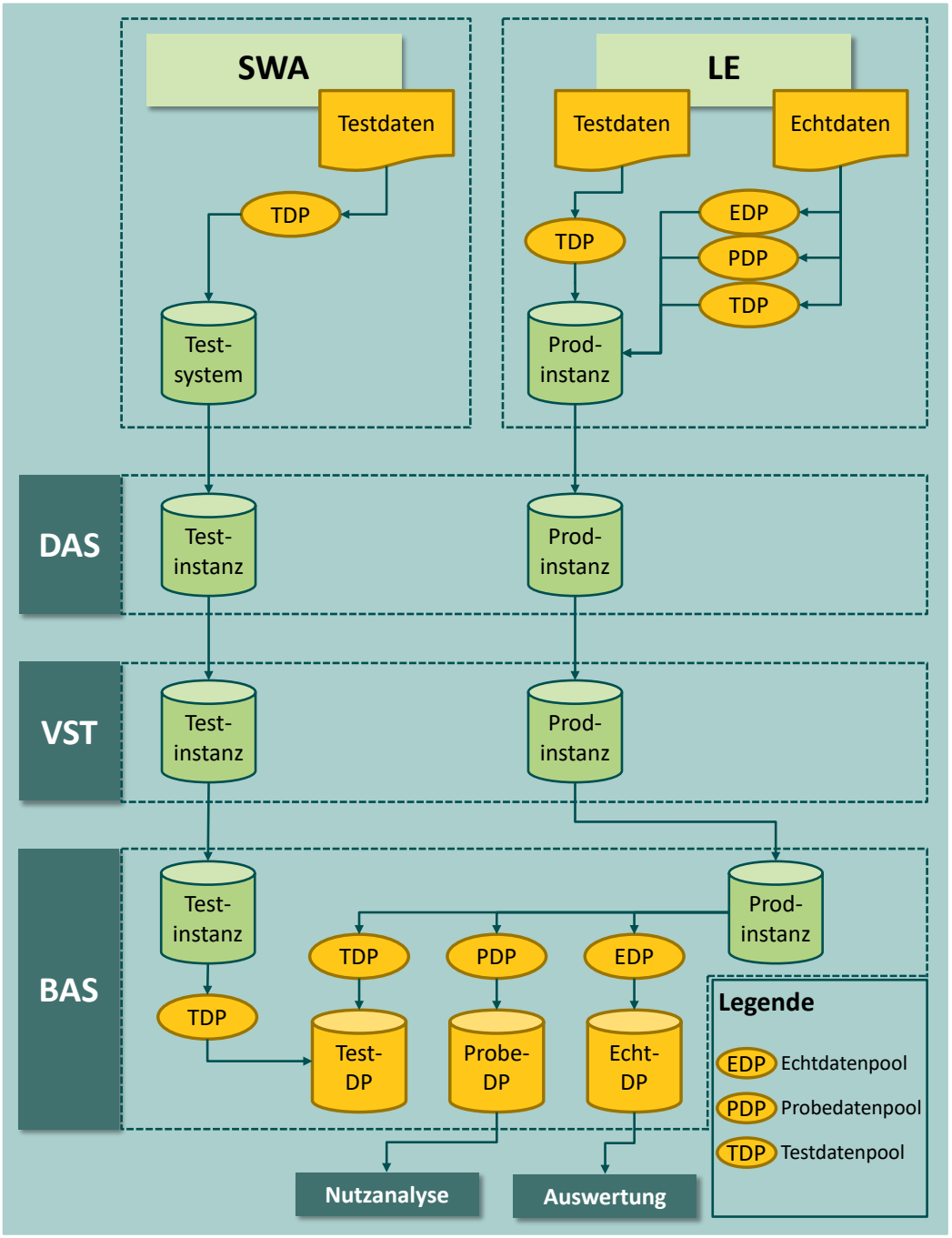


Abbildung 9: Datenflüsse im Test-, Probe- und Regelbetrieb am Bsp. der PID-Verfahren

Testbetrieb mit Testsystemen und Testungen mit Produktivinstanzen

Testbetriebe finden auf der Grundlage einer Spezifikation für den Probebetrieb oder den Regelbetrieb mit Testinstanzen statt.

Achtung

Beim Testbetrieb mit Testsystemen dürfen keine Echtdaten mit Personenbezug verwendet werden, da die strengen Datensicherheitsrichtlinien nur in den Produktivsystemen garantiert werden können.

Für Testinstanzen gelten unter Umständen andere Registrierungen als für Produktivsysteme (Abschnitt A „Abgrenzung von Test-, Probe- und Regelbetrieb“ auf S.58). Diese Frage ist bilateral zwischen den Datenlieferanten und der Datenannahmestelle zu klären.

In Ausnahmen kann als Grundlage für einen Testbetrieb auch eine eigene Spezifikation erstellt werden (Betriebsart TB, siehe Einleitung Abschnitt 1.1), wenn z. B. Änderungen in Exportverfahren unverbindlich getestet werden sollen.

Testungen mit Produktivinstanzen können jederzeit mit Test- und mit Echtdaten durchgeführt werden. Dazu muss zwingend das Datenziel „Testdatenpool“ angegeben werden.

Probebetrieb

Probebetriebe erfordern eine eigene Spezifikation.

Regelbetrieb

Der Regelbetrieb erfordert eine eigene Spezifikation.

2.3 Rückprotokollierung

In diesem Kapitel wird die Rückprotokollierung in Bezug auf die Funktion, den Aufbau und die Erstellung beschrieben.

2.3.1 Funktion von Empfangsbestätigung und Datenflussprotokoll im Datenfluss

Empfangsbestätigung

Die Empfangsbestätigung wird nach Erhalt und abschließender erfolgreicher Eingangsverarbeitung und Weiterleitung eines Dokuments über den Eingangskanal an den Absender übermittelt. Sie bestätigt dem Absender den Übergang der Verantwortung für das Dokument an den Aussteller.

Eine Empfangsbestätigung ist nur für den Absender bestimmt und wird nicht weitergeleitet. Bei Ausbleiben ist von einer fehlgeschlagenen Übermittlung auszugehen. Es gibt zurzeit keine verbindliche Vereinbarung, in welchem zeitlichen Rahmen eine Empfangsbestätigung erwartet werden kann. Angestrebt werden soll allerdings eine Echtzeitverarbeitung, sodass allein die Verarbeitungsdauer eines Dokuments die Verzögerung einer Empfangsbestätigung bedingt.

Datenflussprotokoll

Ein Datenflussprotokoll wird erstellt, wenn das Dokument keine weitere Verarbeitung mehr erlaubt. Das ist dann der Fall, wenn das Dokument durch einen der vorgesehenen Prüfungs- und Verarbeitungsschritte den Status ERROR erhält oder wenn das Dokument in der Auswertungsstelle vollständig und erfolgreich verarbeitet wurde und den Status WARNING oder OK trägt.

Das Datenflussprotokoll dokumentiert alle an dem Dokument durchgeführten Prüfungen und deren Ergebnisse.

Ein Datenflussprotokoll wird in der Regel bis zum Leistungserbringer zurückübermittelt. Aus diesem Grund muss die Datenannahmestelle nach der Prüfung des erhaltenen Datenflussprotokolls den Leistungserbringer depseudonymisieren.

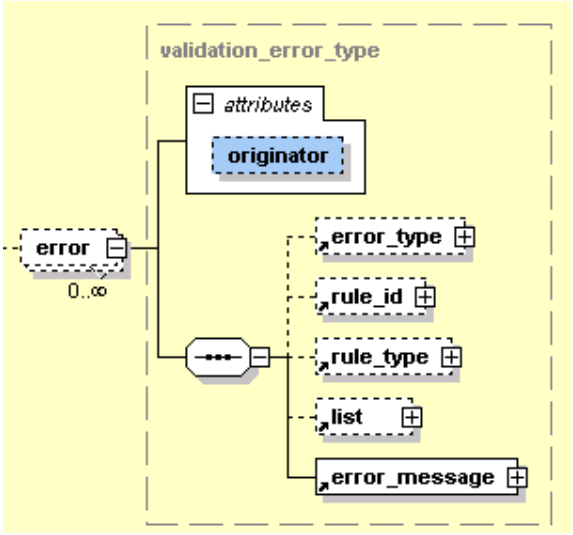


Abbildung 10: Attribut "originator" im Prüfungs- und Fehlerprotokoll

Die im Protokoll im Attribut „originator“ ggf. enthaltenen Hinweise auf den Verursacher eines Fehlers müssen von der Datenannahmestelle so ausgewertet werden, dass Verarbeitungsfehler, die weder auf Fehler des Leistungserbringers noch auf die Software des Leistungserbringers zurückzuführen sind, nicht an den Leistungserbringer weitergeleitet werden. Dieses ist daran erkennbar, dass im Attribut „originator“ ein von Leistungserbringer oder Softwareanbieter abweichender Wert steht. In einem solchen Fall soll verhindert werden, dass das Datenflussprotokoll an den Leistungserbringer verschickt wird.

Statt der Weiterleitung sollte die fehlerhafte Verarbeitung des auslösenden Beteiligten (Datenannahmestelle, Vertrauensstelle, Auswertungsstelle) korrigiert werden und ein korrekt verarbeitetes Dokument, dessen Stand in Abstimmung mit der Auswertungsstelle über die ID zurückgesetzt wurde, erneut in den Datenfluss eingebracht werden. Für das Zurücksetzen eines Dokuments steht kein automatisierter Prozess zur Verfügung. Dieses kann über die E-Mail-Adresse verfahrenssupport@iqtig.org vereinbart werden.

Erzeugung der Protokolle

Die Datenflussprotokolle werden durch eine Reduktion der erhaltenen PB-Export-Daten erstellt. Dabei werden die Elemente `<patient>` und `<qs_data>` aus der Datei entfernt. Es verbleiben Header, Protokoll und Admin-Daten in der Datei.

Für das Datenflussprotokoll wird das Attribut `feedback_range` des Elements `<protocol>` auf `dataflow` gesetzt.

Für diesen Verarbeitungsschritt kann das Datenprüfprogramm eingesetzt werden.

2.3.2 Die Rückprotokollierung

Die Empfangsbestätigung

Die Empfangsbestätigung wird vom Datenempfänger über den Eingangskanal an den Datensender zurückgeschickt, sofern das angenommene Datenpaket in Bezug auf die Prüfungen verarbeitbar ist und weitergeleitet werden kann.

Beispiel:

Wie die folgende Abbildung zeigt, enthält die Empfangsbestätigung lediglich eine Benachrichtigung, dass die vom Leistungserbringer übermittelte XML-Datei verarbeitbar war und an die nachfolgende Stelle weitergeleitet wurde:

```
<?xml version="1.0" encoding="UTF-8"?>
<root xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 content_version="1.0" container_version="2.0"
 xsi:noNamespaceSchemaLocation="../../../interface_LE_DAS/response_receipt.xsd"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <header>
        <document>
            <id V="{55664537-5642-9056-8676-456468327446}"/>
        </document>
        <provider email="datenannahmestelle@test-datenannahmestelle.de"
            name="Test-Datenannahmestelle" registration="bu00000"
            function="Datenannahmestelle" address="Test Adresse"/>
    </header>
    <body>
        <nachricht>
        Ihre Dateneinsendung konnte erfolgreich eingelesen und an die Vertrauensstelle weitergeleitet werden.
        Ein umfassendes Datenflussprotokoll über die Prüfergebnisse der Datenannahmestelle, der Vertrauensstelle
        und der Bundesauswertungsstelle erhalten Sie von uns in den nächsten 24 Stunden.
    </nachricht>
    </body>
</root>
```

Abbildung 11: Beispiel einer Empfangsbestätigung

Die Empfangsbestätigung soll ohne Verschlüsselung an den Sender geschickt werden. Für eine weitere Vereinfachung der Benachrichtigung können die Datenannahmestellen den Inhalt der spezifizierten Benachrichtigung z. B. in die E-Mail übernehmen (betreff/body). Die vollständige Benachrichtigung lautet:

> *Ihre Dateneinsendung konnte erfolgreich eingelesen und an die <nachfolgende Stelle (Vertrauensstelle oder Auswertungsstelle)> weitergeleitet werden.*

> *Ein umfassendes Datenflussprotokoll über die Prüfergebnisse der Datenannahmestelle, der Vertrauensstelle und der Auswertungsstelle erhalten Sie von uns innerhalb der nächsten 24 Stunden.*

Die Benennung der unverschlüsselten Datei leitet sich aus dem Benennungsschema ab (Abschnitt A „Benennung der Exportdateien“ auf S. 54).

Das Datenflussprotokoll

Die vorgenommenen Prüfungen werden in den dafür vorgesehenen Bereichen im XML-Code des übermittelten Dokuments protokolliert. Das Protokoll des Dokuments wächst damit mit jeder Prüfung an.

Nachdem alle Prüfungen der datenentgegennehmenden Stelle abgeschlossen sind, wird für die Rückprotokollierung der Prüfungsergebnisse eine Kopie des Dokuments von allen PB-Daten (Element `<qs_data>`) und patientenidentifizierende Daten (Element `<patient>`) befreit. Das übriggebliebene XML enthält innerhalb der ursprünglichen Struktur des Dokuments die bis dahin protokollierten Prüfungen und die sich daraus ergebenden Statusmeldungen der Datensätze und des Dokuments. Der Aufbau des Protokolls ist weiter unten im Abschnitt A „Prüfungsprozess und Ergebnisprotokollierung“ auf S. 68 detailliert beschrieben.

Das XML-Protokoll kann von der datenentgegennehmenden Stelle oder von der PB-Software mithilfe einer Template-Definition z. B. nach HTML transformiert werden. Den zuvor beschriebenen Plausibilitätsprüfungen schließen sich auf Ebene der Vertrauensstelle und auf Bundesebene weitergehende Prüfungen an. Diese führen zu einem weiteren Anwachsen des Prüfprotokolls und der Datenqualität.

Bezogen auf einen bestimmten Datensatz ist es erst nach der letzten abgeschlossenen Prüfung auf Bundesebene möglich, eine Aussage darüber zu treffen, ob sich dieser Datensatz für die Aufnahme in den Bundesdatenpool eignet oder nicht.

Um dem Leistungserbringer das konkrete Ergebnis seiner Datenlieferung in Bezug auf den Bundesdatenpool mitteilen zu können, wird auch das bis zum Schluss weitergeführte Dokument von PB- und patientenidentifizierenden Daten befreit und als Datenflussprotokoll an die datenentgegennehmende Stelle versandt, die dann die Aufgabe hat, dieses an die – nur am Pseudonym erkennbaren – Leistungserbringer zu übermitteln.

Die Datenflussprotokolle werden jedoch auch von der Datenannahmestelle oder der Vertrauensstelle für den Fall erstellt, dass eine Weiterleitung der PB-Exporte nicht möglich war. In diesem Fall wird keine Eingangsbestätigung an den Absender verschickt. Um das Dokument als Datenflussprotokoll kenntlich zu machen, muss das Attribut `root/header/protocol/@feedback_range` auf den Wert `dataflow` gesetzt werden.

Miniprotokoll

Die Erstellung eines vollständigen Datenflussprotokolls ist nur möglich, wenn die ursprüngliche XML-Datei lesbar ist und nach Entfernung der PID und der PB-Daten schemakonform bleibt. Andernfalls ist ein reduziertes Protokoll („Miniprotokoll“) zu erstellen, das die ID (GUID) des Dokuments (sofern diese zur Verfügung stand und lesbar war, sonst wird diese nicht angegeben) und

die konkrete Fehlermeldung enthält. Ein Miniprotokoll ist z. B. zu erstellen, wenn die Exportdatei nicht wohlgeformt ist.

Die Fehlermeldungen

Für jeden Fehler wird wenigstens das Element `<error_message>` ausgefüllt. Andere Elemente bleiben bei einzelnen Fehlerarten leer. Tabelle 15 gibt einen Überblick darüber, unter welchen Bedingungen in den Feldern der Fehlerdatei Angaben erforderlich sind.

Die Bogenliste `<list>` umfasst einen oder mehrere Namen von Teildatensätzen, welche einen Bezug zu einer Regel haben. Entscheidend für den Bogenbezug sind die in der Tabelle `Regeln` formulierten Regeln, nicht die für den Exportdatensatz umformulierten Regeln.

Die Bogenfeldliste umfasst einen oder mehrere Namen von Bogenfeldern, welche einen Bezug zum Fehler haben. Bei der Fehlerart WERT enthält die Liste nur ein Element. Der Bogenfeldname umfasst auch den Namen des zugehörigen Teildatensatzes[21].

Für jede Regel gibt es eine Liste von Bogenfeldern, identifiziert über die Feldnamen der Regeln. Damit die Liste nicht durch Parsen ermittelt werden muss, wird sie auch über die Tabelle `RegelFelder` zur Verfügung gestellt. Über die Regelnummer können die Teildatensätze, welche Bezug zu einer Regel haben, durch folgende Abfrage identifiziert werden:

```
SELECT DISTINCT Bogen.name FROM (Modul INNER JOIN (Feld INNER JOIN (Bogen INNER JOIN BogenFeld ON Bogen.idBogen = BogenFeld.fkBogen) ON Feld.idFeld = Bogen-Feld.fkFeld) ON Modul.idModul = Bogen.fkModul) INNER JOIN RegelFelder ON BogenFeld.idBogenFeld = RegelFelder.fkBogenFeld WHERE RegelFelder.fkRegeln=<Regelnummer>;
```

Für die `Regelnummer <rule_id>` ist die entsprechende Nummer (Attribut `idRegeln`) der Tabelle `Regeln` anzugeben.

Bei Teildatensätzen, welche mehrfach angelegt werden können, muss die Nummer des betreffenden Teildatensatzes in eckigen Klammern angehängt werden (z. B. `P[1], P[2]` usw.). Mit „Nummer des betreffenden Teildatensatzes“ ist die Position des Teildatensatzes im XML.

Tabelle 15: Ausfüllen der Elemente eines Validation-Items in Abhängigkeit von den Fehlerarten

Feld (csv)	Fehlerart	Regelnr	Regeltyp	Liste	Meldung
Element (xml)	**<error_type>**	**<rule_id>**	**<rule_type>**	**<list>**	**<error_message>**
	STEUER	-	-	-	ja
	EXPORT	Ja	-	<Bogen>	ja
	DOPPELT	Ja	-	-	ja
	TDS	Ja	-	<Bogenliste>	ja

[21] Der Bezug zum Modul kann entfallen, da dieses über die Vorgangsnummer identifiziert werden kann.

Feld (csv)	Fehlerart	Regelnr	Regeltyp	Liste	Meldung
Element (xml)	**<error_type>**	**<rule_id>**	**<rule_type>**	**<list>**	**<error_message>**
	WERT	Ja	-	<Bogenfeldliste>[22]	ja
	REGEL	Ja	ja	<Bogenliste>	ja

Beispiel:

Beispiel eines Protokolls

```
<protocol>
    <status_case V="ERROR"/>
    <validation_item V="Spezifikation" c_date="2014-06-25T08:53:34" id="1">
        <status V="ERROR">
            <error>
                <error_type V="WERT"/>
                <rule_id V="1001022"/>
                <rule_type V="H"/>
                <list V="B/DokAbschlDat"/>
                <error_message V="Das Datenfeld 21/3:B:DokAbschlDat muss einen gültigen Wert enthalten."/>
            </error>
            <error>
                <error_type V="WERT"/>
                <rule_id V="1001022"/>
                <rule_type V="H"/>
                <list V="PROZ/OPDATUM"/>
                <error_message V="Das Datenfeld 21/3:PROZ:OPDATUM &#34;Datum der Prozedur&#34; muss einen gültigen Wert enthalten."/>
            </error>
            <error>
                <error_type V="REGEL"/>
                <rule_id V="7466"/>
                <rule_type V="W"/>
                <list V="PCI/OPSCHLUEPTCA"/>
                <error_message V="Hinweis: Keine QS-Filter-PTCA-Einschlussprozedur für Datensatz 21/3 dokumentiert"/>
            </error>
        </status>
    </validation_item>
</protocol>
```

Um die Fehleranalyse zu vereinfachen, wurden die potenziellen Fehler in Kategorien (XML-Element `<error_type>`) unterteilt, die bestimmte Prüfprozesse (XML-Element `<validation_item>`) durchlaufen. Das Ergebnis ist folgende Tabelle:

Tabelle 16: Mögliche Fehlerarten in Prüfprozessen

Prüfprozess	Fehlerart					
Plausibilitätsprüfung PB-Verfahren gegen Spezifikation: „Spezifikation“		Regel	Doppelt	Export	Wert	TDS
PID_Pseudonym	Kollision				Wert	
Schema-Konformität: „Schema“						
Entschlüsselungsprozess: „Dechiffrierung“	PID	PB-Verfahren	LE			

[22] In der Regel wird hier nur ein Bogenfeld aufgeführt. Ausnahme ist, wenn Kombinationsfelder geprüft werden.

Prüfprozess	**Fehlerart**					
LE_Pseudonym					Wert	
sonstige Prüfung						

Die konkreten Fehlermeldungen sind in der Spezifikationsdatenbank PBDOK hinterlegt:

- Tabelle `Regeln.meldung`: enthält spezifische Fehlermeldungen bei entsprechenden Regelverletzungen
- Tabelle `Fehlermeldung.meldung`: enthält standarisierte und allgemeingültige Fehlermeldungen. Folgende Tabelle zeigt Beispiele bei bestimmten Fehlerfällen.

Tabelle 17: Beispiele von Fehlermeldungen

Fehlerart	**Standardisierte Meldung**	**Beschreibung**
Standardisierung der Meldungen bei Bestätigungsstatus mit Fehlerart DOPPELT.	*Es wurde bereits ein Datensatz mit derselben Registrier- und Vorgangsnummer und derselben oder einer höheren Versionsnummer übermittelt.*	Bei feldbezogenen Fehlern sind die standardisierten Fehlermeldungen der Plausibilitätsregeln zu verwenden.
Standardisierung der Meldungen bei Bestätigungsstatus mit Fehlerart TDS.	Erforderlicher Teildatensatz <Bogen.name> („<Bogen.bezeichnung>“) existiert nicht.	Wenn ein obligatorischer Teildatensatz (Attribut `Bogen.fkBogenZahl + 1` ist, oder ein zu einem Kindteildatensatz zugehöriger Mutterteildatensatz) eines Vorgangs in den Exportdateien einer Transaktion nicht vorkommt
	Die Angaben im Datensatz erfordern einen Teildatensatz <Bogen.name> ("<Bogen.bezeichnung>"). Dieser fehlt.	Wenn die Existenzbedingung eines Kindteildatensatzes im zugehörigen Mutterteildatensatz erfüllt ist, aber kein Kindteildatensatz vorhanden ist (Abschnitt B 2.3.2).
	Die Angaben im Datensatz lassen keinen weiteren Teildatensatz <Bogen.name> ("<Bogen.bezeichnung>") zu, obwohl ein solcher übermittelt wurde.	Wenn die Existenzbedingung eines Kindteildatensatzes im zugehörigen Mutterteildatensatz nicht erfüllt ist, aber trotzdem ein Kindteildatensatz existiert (Abschnitt B 2.3.2).

Die Fehlermeldungen, die nicht von der Tabelle `Regel` in der Access-Datenbank abgedeckt sind, sind in der Tabelle `Fehlermeldung` hinterlegt. Diese sollen eine Standardisierung von Fehlermeldungen und klaren Bedeutungen unterstützen.

Prüfungsprozess und Ergebnisprotokollierung

Ausgangspunkt ist eine prinzipiell offene Anzahl von Prüfungen. Welche Prüfungen konkret durchgeführt werden, ist abhängig vom Datenfluss. Für die Protokollierung der Prüfungen und deren Ergebnisse gibt es auf Dokumentenebene im Header und auf Fallebene im `<case_admin>` das Element `<protocol>`.

Auf Dokumentenebene sind alle Prüfungen zu dokumentieren, einschließlich der Prüfungen, die ausschließlich die Datensätze betreffen. Eine prüfende Einrichtung trägt sich als `<validation_provider>` in die entsprechende Auflistung ein und dokumentiert dann ihre durchgeführten Prüfungen in der Auflistung `<validation_item>` (Ausnahme: Prüft der Leistungserbringer, sind in keinem Fall die Daten des Leistungserbringers einzutragen. In diesem Fall wird der Softwareanbieter als `<validation_provider>` eingetragen.

Prüfungen, die – wie z. B. die Schemakonformität – das Dokument insgesamt betreffen, sind ausschließlich im Headerbereich einzutragen.

Prüfungen, die – wie zum Beispiel die Prüfung auf Plausibilitätsregeln – auf Fallebene erfolgen, müssen folgendermaßen protokolliert werden:

- Das Ergebnis in Bezug auf das gesamte Dokument muss im `<header>` eingetragen werden.
- Das Ergebnis der Fallprüfung muss in `<case_admin>` eingetragen werden, sofern der Status dieser Prüfung nicht OK ist (siehe auch unten).
- Alle Ergebnisse einer Prüfung, die auf Fallebene erfolgt, müssen mit einer gemeinsamen, dokumentweit eindeutigen ID im Attribut ID des Elements `<validation_item>` eingetragen werden. Dadurch ist es möglich, über die ID eines Prüfungsergebnisses, die man auf Fallebene findet, auf Dokumentenebene den `<validation_provider>` eindeutig zu identifizieren.

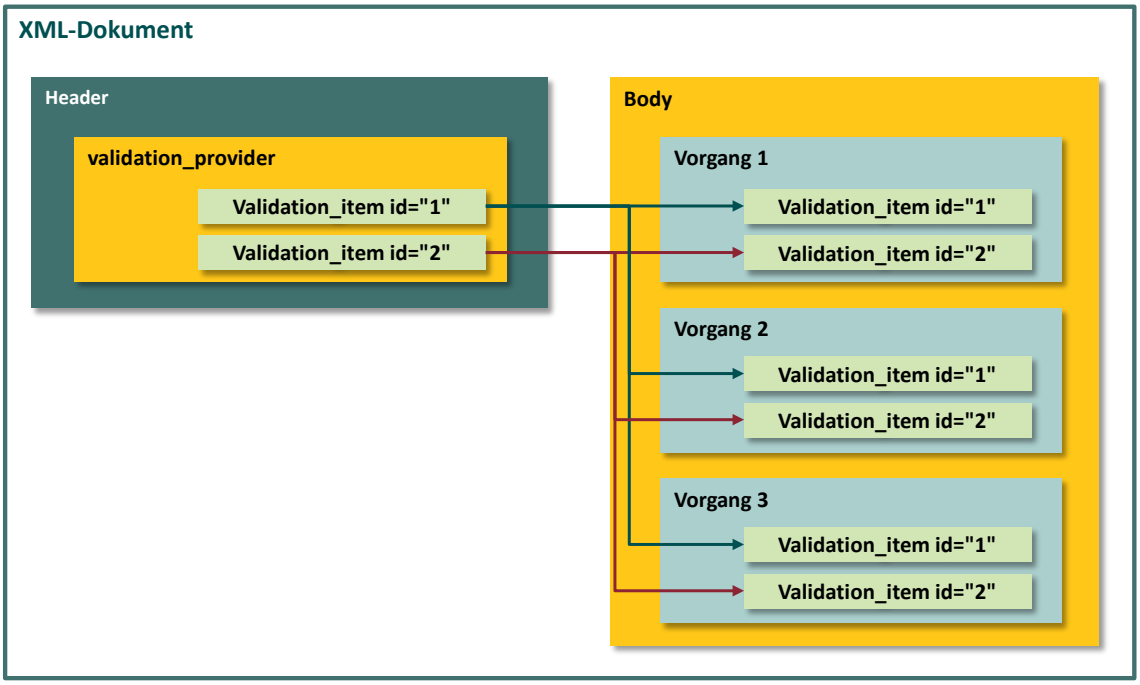


Abbildung 12: Beziehungen zwischen <validation_item> im header und <validation_item> im body über die id

Zur Veranschaulichung dieser Konstruktion soll im Folgenden eine Analogie aus dem relationalen DB-Modell bemüht werden. So kann die Dokumentenebene als Master-Tabelle und die Fallebene als Detail-Tabelle bezeichnet werden. Letztere enthält die zum Master gehörenden Detail-Datensätze, auf die über das Attribut „ID“ referenziert werden kann (Abbildung 12).

Prüfungsergebnisse

Prinzipiell wird als Ergebnis jeder Prüfung eine der folgenden Aussagen über das geprüfte Objekt getroffen:

- Keine Auffälligkeiten
- Auffälligkeiten, die einer Weiterverarbeitung nicht im Weg stehen
- Auffälligkeiten bzw. Fehler, die eine Weiterverarbeitung des Objekts ausschließen.

In der datentechnischen Übersetzung wird dieses durch

- OK
- WARNING
- ERROR

ausgedrückt, die das Ergebnis der Prüfung im Attribut „V“ des Elements `<status>` im Element `<validation_item>` zusammenfassen.

Darüber hinaus gibt es die Möglichkeit, eine beliebige Anzahl von `<error>` Elementen mit einer `<error_message>` im `<status>` Element unterzubringen.

Im Fall einer Auffälligkeit muss wenigstens eine standardisierte Fehlermeldung im `<status>` Element der von der Prüfung betroffenen Ebene (Vorgang oder Dokument) untergebracht werden (Abschnitt B 2.7.3).

Beziehungen Vorgangsebene/Dokumentenebene
Es gibt zwei Kategorien geprüfter Objekte, die zueinander in einer hierarchischen Beziehung stehen:

- Erste Hierarchieebene: das gesamte Dokument
- Zweite Hierarchieebene: der Fall

Jedes dieser Objekte hat einen Status in Bezug auf die Weiterverarbeitung, der sich auf die Ergebnisse der durchgeführten Prüfungen bezieht.

Auf Dokumentenebene ist dieser Status im Unterelement <status_document> von <protocol> im Attribut @V abgelegt.

Auf Fallebene ist dieser Status ebenfalls in einem Attribut @V eines Unterelements von <protocol> abgelegt, welches hier aber <status_case> benannt wird.

In Bezug auf die Weiterverarbeitung gibt es folgende Regeln:
Ein ERROR in einer der Prüfungen verhindert die Weiterverarbeitung des geprüften Objekts. Eine oder mehrere Auffälligkeiten (WARNING, ERROR) auf Fallebene bedeuten ein WARNING in dem korrespondierenden Eintrag auf Dokumentenebene. Wenn bei einer fallbezogenen Validierung in allen Fällen auf Status ERROR erkannt wird, muss auch für das Dokument abweichend von der Regel unter 2. der korrespondierende Eintrag auf Dokumentenebene auf ERROR gesetzt werden. Der Status (<status_case>/<status_document>) eines Objekts kann nicht „besser“ sein als sein schlechtestes Prüfergebnis.

Szenarien
Aus diesen Regeln abgeleitet, soll der Status jedes geprüften Objekts nach jeder Prüfung entsprechend dem Prüfergebnis aktualisiert werden. Daraus ergibt sich folgender Aktualisierungs- und Protokollierungsplan:

Vor der Prüfung und Protokollierung

(1) → Feststellen der höchsten ID in Bezug auf vorhandene <validation_item>-Elemente.

(2) → Festgestellte ID um 1 inkrementieren und als ID der anstehenden Prüfung festlegen.

Protokollierung der fallbezogenen Prüfung
Nachdem die fallbezogene Prüfung erfolgt ist, ist dies auf der Fallebene und der Dokumentenebene folgendermaßen zu protokollieren:

<u>Protokollierung auf Fallebene</u>

Positive Prüfungen werden auf Fallebene nicht protokolliert. Wenn eine Prüfung auf Fallebene keine Auffälligkeit feststellt, wird dieses Ergebnis nicht dokumentiert. Das Ergebnis OK ist implizit anzunehmen, wenn kein Fehler protokolliert wurde.

Auf Fallebene wird nur dann protokolliert, wenn bei der Prüfung eine Auffälligkeit festgestellt wurde. Falls eine Auffälligkeit festgestellt wird, sind die Schritte 3 bis 6 abzuarbeiten.

(3) → <validation_item> der Liste hinzufügen, dabei die unter 2. ermittelte ID verwenden.

(4) → <status_case> des Falls auslesen.

(5) → Ergebnis der Prüfung mit dem Status des Falls vergleichen. In den Fällen, bei denen das Ergebnis der Prüfung schlechter ist als der aktuelle Status des Falls, wird der Status mit dem Ergebnis der Prüfung aktualisiert.

(6) → Falls ein Ergebnis der Prüfung schlechter ist als „OK“, muss dieses als dokumentbezogenes Ergebnis „WARNING“ vermerkt werden.

Protokollierung auf Dokumentenebene

(7) → <validation_item> mit dem unter 4. ermittelten Prüfungsergebnis der fallbasierten Prüfung unterhalb des Elements <validation_provider> eintragen. Falls <validation_provider> für die eigene Einrichtung noch nicht besteht, muss er angelegt werden.

(8) → <status_document> auslesen.

(9) → Das unter 6. ermittelte Gesamtergebnis der Prüfung muss mit dem Status des Dokuments verglichen werden. Falls das Ergebnis der Prüfung schlechter ist als der aktuelle Status des Dokuments, muss dessen Status mit dem Ergebnis der Prüfung aktualisiert werden.

Protokollierung der dokumentenbezogenen Prüfung

Nachdem die dokumentenbezogene Prüfung erfolgt ist, ist dies auf der Dokumentenebene folgendermaßen zu protokollieren:

(10) → <validation_item> mit dem Prüfungsergebnis unterhalb des Elements <validation_provider> eintragen. Falls <validation_provider> für die eigene Einrichtung noch nicht besteht, muss er angelegt werden.

(11) → <status_document> auslesen.

(12) → Das Ergebnis der Prüfung mit dem Status des Dokuments vergleichen und in dem Fall, in dem das Ergebnis der Prüfung schlechter ist als der aktuelle Status des Dokuments, dessen Status mit dem Ergebnis der Prüfung aktualisieren.

Achtung

Rückschlüsse auf den Leistungserbringer

Der Leistungserbringer/die Krankenkasse darf aus datenschutzrechtlichen Gründen nicht als Validation-Provider im Dokument auftauchen. Wenn das Datenprüfprogramm beim Leistungserbringer ausgeführt wird, ist als Validation-Provider der Softwarehersteller des PB- Programms zu verwenden, damit keine Rückschlüsse auf den Leistungserbringer gezogen werden können. Um dieses auch über das Schema abzusichern, ist im Attribut /validation_provider@function der Bezeichner „Leistungserbringer“ nicht zulässig.

Rückprotokoll – Bereitstellung eines XSLT für die Transformation

Für alle Leistungserbringer/Krankenkassen, die keine Möglichkeit haben, das Datenflussprotokoll in die PB-Dokumentationssoftware zu importieren und in geeigneter Form darzustellen, stellt das IQTIG ein XSLT-Skript zur Verfügung, das die Darstellung der XML-Protokolle in Browsern in vereinfachter HTML-Darstellung ermöglicht.

Lokale Transformation (Empfehlung)

Die einfachste und sicherste Variante ist das Transformieren vom Browser selbst. Dafür soll das XML-Protokoll im Browser (z. B. Internet-Explorer, Firefox) geöffnet werden.

Die Voraussetzung für die fehlerfreie Umwandlung ist,

- das lokale Ablegen des dazugehörigen XSLT-Skripts
- der entsprechende Link zum XSLT-Skript muss in das XML-Protokoll unmittelbar nach der ersten Zeile eingetragen werden:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="protocol.xsl"?>
<root xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:gba:sqg ../../../interface_LE/2017_BAS_dv_1.0_Export.xsd"
    xmlns="urn:gba:sqg" container_version="2.0" content_version="1.0">
    <header>
    </header>
    <body>
    </body>
</root>
```

Abbildung 13: Aufnahme des XSLT-Pfads in das XML-Protokoll

Der Eintrag der Referenz (siehe Pfeil) im Rückprotokoll erfolgt durch die DAS.

In der HTML-Darstellung wird der Inhalt des Protokolls besser lesbar und durch die Kategorisierung der Prüfergebnisse unter Verwendung einer Ampelanzeige

- Rot für fehlerhafte Datensätze (ERRORS)
- Gelb für Datensätze mit Hinweisen (WARNINGS)
- Grün für fehlerfreie Datensätze (OK)

übersichtlicher gestaltet.

Dokument-Status: ERROR

» Details ein/aus

Datenannahmestelle **BAQ** (DAS10000) DAS, 80331 München		tel: 089 000000 0 fax: (+49)089/000 00-00 datenannahmestelle@das.de	
Validierung 1	2016-05-22T15:14:48	Spezifikation	OK
Vertrauensstelle **VST-PSN** (bu20000)		tel: - fax: - vst-psn@vertauensstelle-gba.de	
Validierung 2	2016-05-22T16:15:13	PID_Pseudonym	WARNING
Bundesauswertestelle **IQTIG** (BU10000) Katharina-Heinroth-Ufer 1; 10787 Berlin		tel: (+49)030/585826-0 fax: (+49)030/585826-999 verfahrenssupport@iqtig.org	
Validierung 3	2016-05-21T17:41:14	Spezifikation	ERROR

Abbildung 14: HTML-Darstellung nach einer XSLT-Transformation am Beispiel einer QS-Übertragung (entsprechend der PB-Übertragung)

B Komponenten

In diesem Kapitel werden die einzelnen Komponenten der Spezifikation beschrieben. Ein Spezifikationspaket bildet die Gesamtheit seiner Spezifikationskomponenten ab.

Spezifikationskomponenten

Ein Spezifikationspaket kann sich aus folgenden Komponenten zusammensetzen:

- **TechDok** – bezeichnet alle technischen Dokumentationen; diese geben detaillierte Erläuterungen zur Funktionsweise und Verwendung der einzelnen Komponenten. Da es verschiedene spezifisch auf eine jeweilige Zielgruppe hin verfasste TechDoks gibt, wird die Zielgruppe gleich im Kürzel vermerkt.
  - **TechDok_LE** – Technische Dokumentation für Leistungserbringer
  - **TechDok_DAS** – Technische Dokumentation für die Datenannahmestelle
  - **TechDok_SozDat** – Technische Dokumentation für die Nutzung der Sozialdaten bei den Krankenkassen
- **PBDOK** – bezeichnet die Access-Datenbank, in der die PB-Dokumentation spezifiziert wird.
- **PBF** – bezeichnet die Access-Datenbank, in der der PB-Filter spezifiziert wird.
- **Schema** – auf der Komponentenebene ist dies eine ZIP-Datei, die die Versionierung und vollständige Bezeichnung enthält. Sie enthält einzelne XML-Schemata, die festlegen, in welcher Struktur XML-Daten an Schnittstellen im Datenfluss vorliegen müssen.
- **Precheck** – auf der Komponentenebene ist dies eine ZIP-Datei, die Schemata für alle administrativen Daten enthält, die ab dem Export dieser Spezifikation gültig sind. Diese Schemata sind ausschließlich für die Verarbeitung der Datenservice der DAS, VST und BAS relevant.
- **Ausfuellhinweise** – auf der Komponentenebene ist dies eine ZIP-Datei, die die Versionierung und vollständige Bezeichnung enthält. Sie enthält einzelne HTML-Dateien für jedes Modul, die mit den Kürzeln der einzelnen Module benannt sind.
- **Anwenderinformationen** – auf der Komponentenebene ist dies eine ZIP-Datei, die die Versionierung und vollständige Bezeichnung enthält. Die ZIP-Datei enthält einzelne HTML-Dateien für jedes im PB-Filter definiertes Modul, die mit den Kürzeln der einzelnen Modulauslöser benannt sind.
- **Dokuboegen** – auf der Komponentenebene ist dies eine ZIP-Datei, die die Versionierung und vollständige Bezeichnung enthält. Sie beinhaltet die Dokumentationsbögen als einzelne PDF-Dateien für jedes Modul, die mit den Kürzeln der einzelnen Module benannt sind. Die Dokumentationsbögen bilden einige wichtige Datenbankinhalte[23] ab.
- **Protocol** – bezeichnet das Stylesheet zur Darstellung des Rückprotokolls (XSLT).
- **Komponentenliste** – bezeichnet die Auflistung aller im Spezifikationspaket enthaltenen Komponenten mit Bezug zu Versionsnummer und Veröffentlichungsdatum (CSV).

[23] Die Papierform ist hier nur als eine Abbildung des Eingabeformulars oder der Eingabemaske zu verstehen. Verbindlich sind daher nur die Inhalte der Datenbank zur PB-Verfahren-Dokumentation.

Hilfsprogramme werden ggf. ebenfalls als Komponenten in ein Spezifikationspaket aufgenommen. Hier sind zur Zeit der XPacker, der Tpacker, das Datenprüfprogramm und das Pseudonymisierungsprogramm zu nennen. Bei der Angabe der Betriebsart und des Exportformats gelten die gleichen Abkürzungen wie bei den Spezifikationspaketen. Diese Angaben erfolgen aber nur dann, wenn sich die Komponenten durch diese Merkmale unterscheiden.

- **Datenpruefprogramm** – ist auf Komponentenebene eine ZIP-Datei, welche die Bestandteile des Datenprüfprogramms enthält.

V<Versionsnummer>: Die Versionierung erfolgt in ganzen Zahlen, die zweistellig angegeben sind (unter 10 mit einer vorstehenden 0, z. B. V01).

Spezifikationsdatenbanken

Als Access-Datenbanken zur Verfügung gestellte Spezifikationskomponenten dienen der (automatisierten) Erstellung von Software für PB-Filter und PB-Dokumentation. Folgende Spezifikationskomponenten werden als Access-Datenbanken (MS Access 2007-2013) zur Verfügung gestellt:

Datenbank zur PB-Dokumentation (PBDOK)

Die Datenbank zur PB-Dokumentation dient der Spezifikation von Datenerhebung und -erfassung unter Berücksichtigung von Plausibilitätsprüfungen und zu exportierenden Datenfeldern. Einige wichtige Datenbankinhalte werden über die Dokumentationsbögen (Dokuboegen) abgebildet.

Folgende Spezifikationskomponenten ergänzen die Datenbank um Hinweise und Informationen:

- Die Ausfüllhinweise (Ausfuellhinweise) dienen als Hilfestellung bei der Dokumentation durch den Anwender. Die Namen der HTML-Dateien für einzelne Datenfelder sind in der Datenbank (`BogenFeld.ahinweis`) hinterlegt.
- Ergänzende Informationen (erginformationen) erläutern die in der Datenbank definierte Syntaxfunktion.

Die Datenbank zur PB-Dokumentation ist in Kapitel B 2 erläutert.

Datenbank zum PB-Filter (PBF)

Die Datenbank zur PB-Dokumentation dient der Spezifikation zur Auslösung von Dokumentationsmodulen. Die Auslösebedingungen pro Modul werden als Übersicht in Form der Anwenderinformationen (Anwenderinformationen) zur Verfügung gestellt.

Die Datenbank zum PB-Filter ist in Kapitel B 0 erläutert.

Datenbank zu Datenserviceinformationen

Seit der Spezifikation 2019 werden relevante Datenserviceinformationen (Abschnitt 2.7.2) in einer separaten Datenbank gepflegt. Die Datenbank zu Datenserviceinformationen ist keine dem Spezifikationspaket zugehörige Komponente, verfügt jedoch über dieselbe Verbindlichkeit. Da sie eine eigenständige Spezifikationsdatenbank darstellt kann sie außerhalb des Releasezyklus angepasst werden.

Verschluesselungsprogramme

Seit der Spezifikation 2020 werden die Verschlüsselungsprogramme nicht mehr als Spezifikationskomponente veröffentlicht. Sie sind daher keine dem Spezifikationspaket zugehörige Komponente, verfügen jedoch über dieselbe Verbindlichkeit. Da sie ein eigenständiges Paket in Form einer ZIP-Datei darstellen, können sie außerhalb des Releasezyklus angepasst werden.

In der ZIP-Datei enthalten sind der X- und der TPacker sowie die zum Veröffentlichungszeitpunkt aktuellen öffentlichen Schlüssel der Datenservices im Datenfluss.

Pseudonymisierungsprogramm

Seit der Spezifikation 2021 wird das Pseudonymisierungsprogramm nicht mehr als Spezifikationskomponente veröffentlicht. Sie ist daher keine dem Spezifikationspaket zugehörige Komponente, verfügt jedoch über dieselbe Verbindlichkeit. Sie kann außerhalb des Releasezyklus zur Verfügung gestellt werden.

- **PSP** – ist eine ZIP-Datei, die neben dem Pseudonymisierungsprogramm ein Informationsmerkblatt beinhaltet.

Tabellenstruktur der Datenbanken

Die Tabellen und deren Spalten (Attribute) unterliegen einem einheitlichen Namensschema. Erlaubte Zeichen sind die Buchstaben a–z, A–Z und die Ziffern 0–9. Umlaute und Sonderzeichen werden nicht verwendet. Das erste Zeichen eines Namens darf keine Ziffer sein.

Ein Tabellenname beginnt immer mit einem Großbuchstaben und ein Attributname mit einem Kleinbuchstaben. Wenn ein Name aus mehreren Teilen (z. B. Substantiven) besteht, so beginnt jeder nachfolgende Namensteil mit einem Großbuchstaben.

`BasisTyp` (Tabelle)
`idBasisTyp` (Spalte)

Für jede Tabelle ist in der Spezifikation ein Primärschlüssel definiert, der nach folgendem Schema aufgebaut ist:

`id<TabellenName>`

Der Ausdruck in spitzen Klammern ist ein Platzhalter für den Namen der Tabelle. Die meisten Tabellen haben einen einfachen Primärschlüssel vom Typ `AUTOINCREMENT`. Zusätzlich enthalten derartige Tabellen mindestens ein identifizierendes Attribut[24], welches durch Setzen eines weiteren, eindeutigen Indexes (bestehend aus einem oder mehreren Attributen) definiert ist.

Beispiele:

Identifizierendes Attribut: Attribut `name` in Tabelle `BasisTyp`

Identifizierende Attributkombination: Attribute `code` und `fkSchluessel` in Tabelle `SchluesselWert`

Es gibt auch Tabellen, deren einziger eindeutiger Schlüssel der Primärschlüssel ist. Ein Beispiel ist die Tabelle `MussKann` mit dem Primärschlüssel `idMussKann` vom Typ KURZER TEXT(1) (entspricht VARCHAR(1)). Diese Tabellen sind als einfache „Nachschlagtabellen“ zu interpretieren. Im Fall der Tabelle `MussKann` soll im entsprechenden Fremdschlüsselfeld der verknüpften Detailtabelle durch das Datenbankschema gewährleistet werden, dass nur ein M oder K eingegeben werden darf.

Die Namen von Fremdschlüsseln sind analog zum Namen der Primärschlüssel aufgebaut:

`fk<FremdTabellenName>`

Die Namensgebung von Primär- und Fremdschlüsseln vereinfacht den Aufbau von komplexeren Abfragen, welche sich über mehrere Entitäten erstrecken (Inklusionsverknüpfungen, Joins).

Die Fremdschlüsselattribute (Namen beginnen mit `fk`) wurden als Datenbankattribute zum Nachschlagen eingerichtet. Zum Beispiel wird beim Fremdschlüsselattribut `fkModul` in der Tabelle `Tds` nicht mehr der Primärschlüssel des jeweiligen Moduls, sondern der Name des Moduls angezeigt.

Hinweis

Diese Änderung betrifft nur die Anzeige, nicht jedoch die Struktur der Datenbank.

---

[24] Oder sie enthalten eine identifizierende Attributkombination, die einen eindeutigen Schlüssel definiert.

Sind zwei Tabellen mehrfach durch Schlüssel-Fremdschlüssel-Beziehungen miteinander verknüpft, so kann der Name eines Fremdschlüssels auch folgendermaßen aufgebaut sein:

`fk<FremdTabellenName><Rolle>`

`<Rolle>` ist der Platzhalter für eine zusätzliche Qualifizierung der Relation.

N-M-Beziehungen werden wie üblich über Verknüpfungstabellen realisiert. In der Spezifikation haben Verknüpfungstabellen gewöhnlich keinen Primärschlüssel[25], jedoch einen eindeutigen Schlüssel, der über die Fremdschlüsselfelder definiert ist.

Folgende Attribute treten in vielen Tabellen auf:

- `name` ist in der Regel als technischer Name zu verstehen. Zum Beispiel wird `Feld.name` als Variablenname in den Plausibilitätsregeln verwendet.
- `bezeichnung` ist eine kurze Beschreibung. Zum Beispiel ist `TdsFeld.bezeichnung` der Text, welcher ein Feld auf einem Eingabeformular beschreibt.
- bedingung enthält einen logischen Ausdruck. Prominentester Vertreter dieses Attributtyps ist das Attribut `bedingung` in der Tabelle `ModulAusloeser`.

[25] Hier: Primärschlüssel im Sinne der Access-Definition eines Primärschlüssels. Streng genommen wird über die beiden Fremdschlüssel ein neuer Primärschlüssel definiert.

1 PB-Filter

Der PB-Filter definiert, unter welchen Bedingungen ein Modul ausgelöst wird. In der Datenbank zum PB-Filter und den dazugehörigen Anwenderinformationen sind die Informationen hierzu hinterlegt. Die PB-Filter-Software löst für jeden ambulanten Fall die Dokumentation des jeweiligen Moduls der PB-Verfahren gemäß oKFE-RL aus.

Achtung

Manuelle Auslösung des Moduls DKK

Durch das Fehlen einer spezifischen EBM-Ziffer für die Abklärungskoloskopie nach positivem i-FOB-Test im Rahmen des Programms Früherkennung von Darmkrebs kann die Auslösung der Dokumentation der Koloskopien im Rahmen des Programms zur Früherkennung von Darmkrebs (Modul DKK) nicht vollständig automatisch erfolgen, sondern muss teilweise manuell durch den Leistungserbringer erfolgen.

1.1 Anmerkungen zur Struktur der Spezifikationsdatenbank für PB-Filter

Die PB-Filter-Spezifikation ist in einer relationalen Datenbank abgelegt. Zurzeit wird sie ausschließlich als Access-Datenbank (MS Access 2007-2013) zur Verfügung gestellt. Der Name der PB-Filter-Spezifikation richtet sich nach folgendem Schema:

`<Erfassungsjahr>_oKFE_PBF_V<Versionsnummer>.mdb`

`<Versionsnummer>` bezeichnet die 2-stellige Versionsnummer (z. B. 01).

Beispiel:
Im Erfassungsjahr 2021 ist die PB-Filter-Spezifikation 2021_oKFE_PBF_V01.mdb[26] gültig. Die Kennung 2021 gilt für das Erfassungsjahr 2021.

Weiterführende Erläuterungen zum Benennungsschema für Spezifikationskomponenten sind der Einleitung in Abschnitt 1.1.2 zu entnehmen.

Die Tabellenstruktur der Spezifikationsdatenbank wird in Abschnitt B Komponenten beschrieben.

Abfragen der Datenbank

Die Abfragen der Access-Datenbank geben einen vereinfachenden Überblick über die Inhalte der Spezifikation:

- `PB-Filter-Leistungsbereiche`
  Diese Abfrage zeigt Name, Textdefinition, Dokumentationsverpflichtung und PB-Datensatz

[26] Die Versionsnummer der gültigen Spezifikation (z.B. V01, V02, usw.) ist dem zuletzt veröffentlichten Update zu entnehmen.

für alle PB-Verfahren an.

- `Datensatz PB-Filter`
  Hier wird die Beschreibung der Struktur des PB-Filter-Eingangsdatensatzes (Teildatensätze z. B. FALL, PROZ) sowie des Ausgabedatensatzes (Erweiterung um die Teildatensätze QSMODUL und FEHLER) angezeigt.
- `EBM-Listen`
  Die EBM-Listen geben die Auslöser der Dokumentation der jeweiligen Module der PB-Verfahren an.
- `Auslösebedingungen`
  Diese Abfrage liefert einen Überblick über die in der Spezifikation enthaltenen Auslösebedingungen der PB-Filter-Leistungsbereiche. Derzeit werden mit der Spezifikation 2021 die PB-Verfahren zur Früherkennung von Darmkrebs (Module DKK, DKI) und zur Früherkennung von Zervixkarzinomen (Module ZKP, ZKA, ZKZ, ZKH) abgebildet.
- `Schlüsselkodes`
  Diese Abfrage liefert eine Übersicht der Schlüssel und der zugehörigen Kodes.

1.2 Grundlegende Tabellen der Datenbank

Dieser Abschnitt beinhaltet die Darstellung der grundlegenden Tabellen der Spezifikationsdatenbank mit ihren zugehörigen Datensätzen ausgehend von den jeweiligen Modulen.

1.2.1 Module (Datensätze der PB-Dokumentation)

In der Tabelle `Modul` sind Referenzen auf die Module hinterlegt, deren Dokumentationspflicht durch den PB-Filter ausgelöst werden kann.

Hinter jedem Modul verbirgt sich ein Datensatz der korrespondierenden Spezifikation für PB-Dokumentationssoftware.[27] Diese Datensätze bilden in der Spezifikation für PB-Dokumentation den auszulösenden Dokumentationsbogen der Software ab. Auch die Exportmodule eines Moduls lösen keinen entsprechenden PB-Datensatz aus.

Beispiel:
Der Modulauslöser DKK löst den PB-Datensatz DKK aus. Der Datensatz wird als Exportmodul DKK exportiert.

Die Module der Spezifikation für PB-Dokumentationssoftware sind in der gleichnamigen Tabelle definiert. Eine Zuordnung ist über das Attribut `Modul.name` möglich. Die Datensätze der PB-Dokumentation werden in Abschnitt B 2.3.1 beschrieben.

[27] Landesweit verpflichtende Module haben keinen Datensatz in der Spezifikation für PB-Verfahren.

Tabelle 18: Struktur der Tabelle Modul

Feldname	Datentyp	Bemerkung
idModul	INTEGER	Primärschlüssel
name	KURZER TEXT	Technischer Name des Moduls (Identifizierendes Attribut)
bezeichnung	KURZER TEXT	Beschreibender Text für den PB-Datensatz
fkSchluesselWert	INTEGER	Verweis auf denjenigen Kode des Schlüssels Modul (Tabelle SchluesselWert), welcher dem betreffenden Datensatz zugeordnet ist.
fkModulParent	INTEGER	Verweis auf das Eltern-Modul eines Zählleistungsbereichs
zaehlLb	BOOLEAN	Wenn WAHR, existiert kein entsprechender PB-Datensatz, es handelt sich um einen „Zählleistungsbereich“ zur separaten Darstellung in der Sollstatistik. Nicht relevant für die Programmbeurteilung
ausloeseModul	BOOLEAN	Wenn WAHR, existiert ein entsprechender PB-Datensatz, der ausgelöst werden kann. Dieses Modul kann gleich dem Exportmodul sein, z. B. DKK.
exportModul	BOOLEAN	Wenn WAHR, handelt es sich um ein Exportmodul, z. B. DKK_KV. Das Exportmodul kann ungleich dem im Modulauslöser referenzierten Modul sein, z. B. DKK.
direkt	BOOLEAN	Handelt es sich um ein direktes Datenexportverfahren? Nicht relevant für die Programmbeurteilung
indirekt	BOOLEAN	Handelt es sich um ein indirektes Datenexportverfahren? Nicht relevant für die Programmbeurteilung
pid	BOOLEAN	Handelt es sich um ein Modul mit Patientenidentifizierenden Daten?
oKFE	BOOLEAN	Handelt es sich um ein Modul der oKFE-Richtlinie?

1.2.2 Struktur der Datensatzdefinitionen

Die Module werden über den definierten Modulauslöser als dokumentationspflichtig erkannt, indem die hinterlegte Bedingung mit den im AIS/LIS gespeicherten Daten geprüft wird. Ist die Bedingung erfüllt, wird das Modul ausgelöst. Da die in der Bedingung enthaltenen Felder im AIS/LIS vorliegen müssen, ist der Eingangs- und Ausgangsdatensatz gemäß § 295 SGB V in der Spezifikationsdatenbank hinterlegt. Die definierten Felder der Datensätze und der Filterbedingungen sind analog zur Spezifikation für PB-Dokumentationssoftware aufgebaut.

In den nachfolgenden Abschnitten dieses Unterkapitels wird die grundlegende Struktur der Tabellen der Datensatzdefinitionen beschrieben und die Definitionen der Datenfelder erläutert.

Datensätze

Jeder Datensatz besteht aus Teildatensätzen, welche ausgehend von einem Basisteildatensatz hierarchisch angeordnet sind. Der PB-Datensatz umfasst den Eingangsdatensatz. Für diesen Datensatz wird in der Tabelle Ds ein Eintrag angelegt.

Tabelle 19: Struktur der Tabelle Ds

Feldname	Datentyp	Bemerkung
idDs	INTEGER	Primärschlüssel
name	KURZER TEXT	Technischer Name des Datensatzes (Identifizierendes Attribut)
bezeichnung	KURZER TEXT	Beschreibender Text

Teildatensätze

Die Definition von Teildatensätzen befindet sich in der Tabelle Tds der Datenbank (Tabelle 20). Jeder Teildatensatz ist eindeutig durch seinen Namen (z. B. FALL) charakterisiert.

Tabelle 20: Struktur der Tabelle Tds

Feldname	Datentyp	Bemerkung
idTds	INTEGER	Primärschlüssel
name	KURZER TEXT	Technischer Name des Teildatensatzes (Identifizierendes Attribut)
bezeichnung	KURZER TEXT	Beschreibender Text
fkTds	INTEGER	Optionaler Fremdschlüssel zu einem Mutterteildatensatz
fkDs	INTEGER	Bezug des Teildatensatzes zum übergeordneten Datensatz in der Tabelle Ds, z. B. PB-Filter-Datensatz
fkRelation-Typ	KURZER TEXT(1)	Relationstyp, bezieht sich auf die Relation zum Mutterteildatensatz: * Eine beliebige Anzahl von Teildatensätzen darf angelegt werden! ? Höchstens ein Teildatensatz darf angelegt werden! + Mindestens ein Teildatensatz muss angelegt werden! 1 Genau ein Teildatensatz muss angelegt werden!

Feldname	Datentyp	Bemerkung
fkEindeu-tigTdsFeld	INTEGER	Optionaler Fremdschlüssel zu einem eindeutigen Teil-datensatzfeld

In der Tabelle Tds ist eine Hierarchie der Teildatensätze definiert. Der Ausgangspunkt („root“) für die Teildatensatzhierarchie eines Datensatzes (z. B. PB-Filter-Datensatz) ist immer der Basisteildatensatz (daraus folgt: Teildatensatz[28] fkTds = NULL in der Tabelle Tds). Über die in den restlichen Teildatensätzen des Moduls definierten Bezüge zu den Mutterteildatensätzen und Relationstypen lässt sich ein Hierarchiebaum der Teildatensätze aufbauen.

Jeder Datensatz besteht aus

- genau einem Basisteildatensatz
- ggf. einem oder mehreren weiteren Teildatensätzen (= Kindteildatensätze)

Beispiele:
Der PB-Filter-Datensatz DATENSATZ_295 besitzt den Basisteildatensatz EBM und die Kinderteildatensätze FALL, QSMODUL und FEHLER.

Felder der Teildatensätze

Die Felder eines Teildatensatzes sind in der Tabelle TdsFeld definiert. Jedes Feld eines Teildatensatzes (kurz TDS-Feld) ist eindeutig charakterisiert durch die Zugehörigkeit zu einem Teildatensatz und zum referenzierten Feld. Jedes Feld darf also nur einmal in einem Teildatensatz verwendet werden. Listenfelder erfordern einen Wert > 1 beim Attribut elemente.

Felder

Ein Feld wird eindeutig über seinen technischen Namen definiert. Jedes Feld hat einen Basistyp (z. B. SCHLUESSEL, NUMSCHLUESSEL, ZAHL). Schlüsselfelder erfordern zusätzlich einen Schlüssel).

Basistypen

Das Hauptmerkmal eines Basistyps ist der technische Typ eines Eingabefeldes (z. B. Zeichenkette, numerischer Typ, Datum usw.). Weiteres Charakteristikum ist die Beschreibung des Eingabeformats. Die Basistypen sind Voraussetzung für die Beschreibung einer formalen Regelsyntax. Das identifizierende Merkmal eines Basistyps ist sein technischer Name (Attribut name).

[28] Es darf nur eine Definition eines Basisteildatensatzes existieren.

Tabelle 21: Struktur der Tabelle BasisTyp

Feldname	Datentyp	Bemerkung
idBasisTyp	INTEGER	Primärschlüssel
name	KURZER TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	KURZER TEXT	Beschreibender Text
formatAnweisung	KURZER TEXT	Regulärer Ausdruck für die Formatprüfung

Hinweis

- In Zeichenketten (Basistyp TEXT) sind alle Zeichen des ASCII-Formats mit einem Kode >= 32 erlaubt. Ausgenommen sind das Semikolon, die doppelten Anführungsstriche und Hochkommata.
- Es gibt zwei Arten von Schlüsseln: numerische und nichtnumerische.
- Das Komma trennt die Nachkommastellen, Vorzeichen + und – sind erlaubt.
- Das Datumstrennzeichen ist der Punkt.

Schlüssel

Identifizierendes Merkmal eines Schlüssels ist sein technischer Name. Die meisten Schlüsselkodes sind in der Tabelle SchluesselWert definiert. Externe Schlüsselkataloge (z. B. EBM) sind von den entsprechenden Anbietern zu beziehen.

Tabelle 22: Struktur der Tabelle Schluessel

Feldname	Datentyp	Bemerkung
idSchluessel	INTEGER	Primärschlüssel
name	KURZER TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	KURZER TEXT	Beschreibender Text
extern	BOOLEAN	Zeigt an, ob der Schlüssel in der Tabelle Schluessel oder in einer externen Tabelle gespeichert ist.
externVerweis	KURZER TEXT	Verweis auf externe Quelle des Schlüsselkataloges
zahl	BOOLEAN	Wenn WAHR, wird das Attribut code der zugehörigen Schlüsselwerte als ganze Zahl interpretiert, ansonsten als Zeichenkette.

Schlüsselkodes können auf zwei Arten kodiert werden. Wenn das Attribut `zahl = WAHR`, so werden die Kodes als ganze Zahl interpretiert. Ansonsten werden sie als Zeichenketten angesehen. In der Syntax der Auslösebedingungen werden die letztgenannten Kodes in einfache Hochkommata gesetzt.

Beispiel:
Attribut **`zahl`** bei Schlüsselfeldern

Felder des Basistyps NUMSCHLUESSEL haben das Attribut `zahl = TRUE`.
Felder des Basistyps SCHLUESSEL haben das Attribut `zahl = FALSE`. Es handelt sich um alphanumerische Schlüssel, die Buchstaben, Ziffern oder Sonderzeichen verwenden (z. B. `ypN0`). Hierbei kann es sich auch um Werte handeln, die lediglich Ziffern verwenden, aber mit führender Null beginnen (z. B. `01`).

Externe Schlüsselkataloge

Externe Schlüsselkataloge sind über das Attribut `extern` deklariert. Externe Schlüsselkataloge werden nicht vom IQTIG bereitgestellt und daher auch nicht verantwortet.

Hinweise zu den Bezugsquellen sind in der Spalte `externVerweis` zu finden (z. B. http://www.dimdi.de). Ein Verweis auf eine Bezugsquelle kann unabhängig vom Attribut `extern` angegeben werden.

Achtung

Der Softwareanbieter hat dafür Sorge zu tragen, dass die aktuellen externen Schlüsselkataloge in der Software verwendet werden.

Schlüsselwerte

Identifizierendes Merkmal ist hier eine Kombination der Spalten `fkSchluessel` und `code`. Das bedeutet, dass jeder Schlüsselkode innerhalb eines Schlüssels nur einmal vorkommen darf.

Tabelle 23: Struktur der Tabelle SchluesselWert

Feldname	Datentyp	Bemerkung
`idSchluesselWert`	`INTEGER`	Primärschlüssel
`fkSchluessel`	`INTEGER`	Fremdschlüssel zur Tabelle Schlüssel
`code`	`INTEGER`	0, 1, 2 …
`bezeichnung`	KURZER TEXT	Textliche Definition des Schlüsselwertes

Hinweis

Die Schlüsselwerte lassen sich am einfachsten über die Abfrage `Schlüsselkodes` ermitteln.

1.2.3 EBM-Listen

Jede EBM-Liste ist charakterisiert durch ihren Namen (Attribut `name` in Tabelle `EBMListe`), welcher nach folgendem Schema gebildet wird:

{<TEXT>_}EBM{_<TEXT>}

Hinter <TEXT> verbirgt sich ein frei wählbarer Name (Erlaubte Zeichen: A–Z, a–z, 0–9, Umlaute sind nicht erlaubt). Die {}-Ausdrücke sind optional.

Beispiel:

DKK_EBM	Einschlussgebührenordnungspositionen PB-Verfahren zur Früherkennung von Darmkrebs - Koloskopie - gemäß EBM-Katalog

String-Vergleich bei EBM-Kodes

EBM-Ziffern können auf Landesebene um weitere Zeichen ergänzt werden. Das Feld `Gebührenordnungsziffer gemäß EBM-Katalog [EBM]` hat daher die Feldlänge 7.

In der Spezifikationsdatenbank für PB-Filtersoftware sind die EBM-Ziffern in Tabelle `EBMWert` fünfstellig – ohne ergänzenden Zeichen – definiert. Für die Prüfung, ob zwei Kodes identisch sind, genügt kein einfacher Stringvergleich. Stattdessen wird ein Stringvergleich der Normkodes[29] durchgeführt, um die Übereinstimmung zwischen dem dokumentierten Kode und dem einer EBM-Liste zu ermitteln.

1.2.4 Versionsverwaltung

Jede Spezifikationsdatenbank hat eine Version. Die Versionsinformation ist in der Tabelle `Version` der Eintrag, welcher den Attributwert `gueltig = WAHR` besitzt.

Die wichtigsten Eigenschaften einer Version sind der Versionsname (Attribut `name`) und der Gültigkeitszeitraum (Attribute `ab` und `bis`). Der Gültigkeitszeitraum einer Version ist in der Regel ein Erfassungsjahr (z. B. Behandlung zwischen dem 01.01.2021 und dem 31.12.2021).

Versionen können den Status `in Entwicklung` oder `final` haben. Diese Zustände werden in der Nachschlagetabelle `VersStatus` verwaltet. Das Attribut `gueltig` zeigt die gültige Version der Datenbank an. Nur eine einzige Version darf als gültig markiert sein. Darüber hinaus

[29] Jeder EBM-Kode lässt sich entweder als Kode mit ergänzenden Zeichen (Normcode + ergänzende Zeichen) oder als Kode ohne ergänzende Zeichen (Normkode) darstellen.

verwaltet die Tabelle `Version` die Historie der Versionen[30]: Welche Vorgängerversion vorher gültig war, kann über das Attribut `fkVersion` ermittelt werden.

Tabelle 24: Struktur der Tabelle Version

Feldname	Datentyp	Bemerkung
`idVersion`	`INTEGER`	Primärschlüssel
`name`	`KURZER TEXT`	Technischer Name der Version (Identifizierendes Attribut)
`bezeichnung`	`KURZER TEXT`	Beschreibender Text
`ab`	`DATUM`	Anfang des Gültigkeitszeitraumes
`bis`	`DATUM`	Ende des Gültigkeitszeitraumes
`pub`	`DATUM`	Datum der Publikation
`gueltig`	`BOOLEAN`	gültige Version (nur ein Eintrag darf als gültig markiert sein)
`fkVersion`	`INTEGER`	Bezug zur Vorgängerversion
`fkVersStatus`	`KURZER TEXT`	Bezug zum Status einer Version (Tabelle VersStatus): E = in Entwicklung F = finale Version S = Service Release zur finalen Spezifikation U = Update der finalen Spezifikation

Das Attribut `Modul.fkVersion` wird verwendet, um die aktuelle Version eines Moduls kenntlich zu machen. Freiwillige Module werden im Rahmen der inhaltlichen Systempflege nicht berücksichtigt. Jahreszahlen in Regeln werden für diese Module weiterhin angepasst. Änderungen aufgrund von modulübergreifenden Anpassungen, z. B. Umbenennung technischer Feldnamen oder Ergänzung von Schlüsselwerten sind nicht auszuschließen.

Zuordnung der PB-Filter-Version zu Behandlungsfällen

Die PB-Filter-Software eines Erfassungsjahres wird für Behandlungsfälle verwendet, deren Behandlungsdatum bei ambulanten Eingriffen in den definierten Gültigkeitszeitraum fällt.

[30] Die Inhalte der Vorversionen sind nicht Teil der aktuellen Spezifikationsdatenbank.

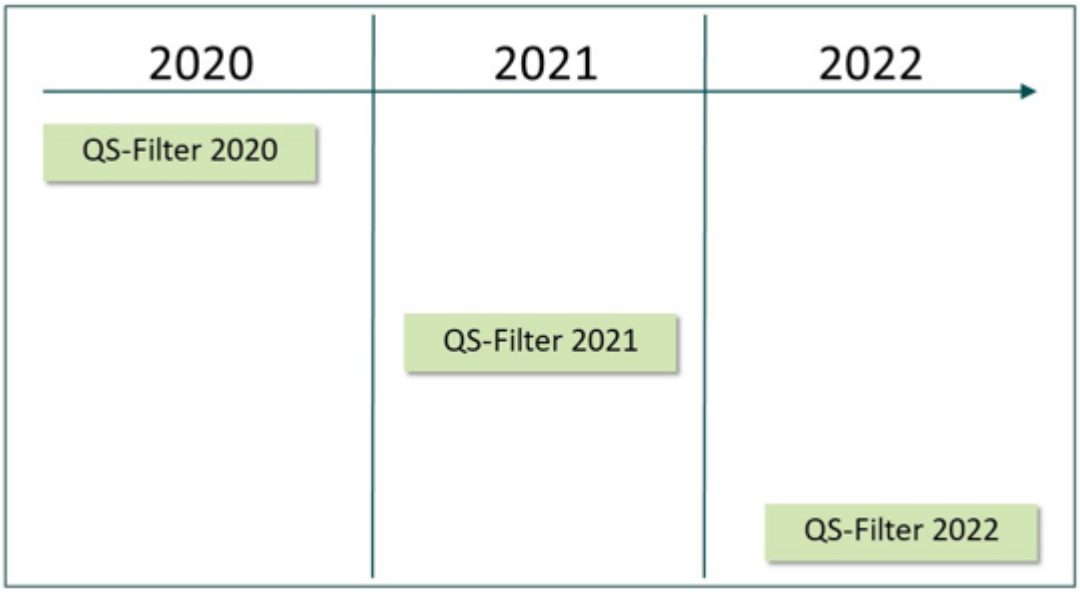


Abbildung 15: Zuordnung der Version des PB-Filters zu den Behandlungsfällen: Kriterium ist das Behandlungsjahr (EBM-Datum)

Abbildung 15 stellt dar, für welche ambulanten Behandlungsfälle welche Version der PB-Filter-Software verwendet wird. Exemplarische Behandlungsfälle sind durch Querbalken visualisiert. Für die ambulanten Fälle gilt lediglich das Datum der Behandlung für die Zuordnung zur korrekten PB-Filter-Software-Version.

1.2.5 Meta-Tabellen

In den Tabellen `TabellenStruktur` und `TabellenFeldStruktur` werden die Tabellen und ihre Attribute aufgelistet. Der Inhalt wird automatisch generiert (Abschnitt B 2.6.2).

1.3 Der PB-Filter-Datensatz

Der PB-Filter-Datensatz umfasst den PB-Filter-Eingangsdatensatz. Verbindlich für ein Erfassungsjahr ist die Datenfeldbeschreibung, welche in der gültigen Spezifikationsdatenbank über die Abfrage `Datensatz PB-Filter` definiert ist. Hier wird mit der Spezifikation 2020 der Datensatz DATENSATZ_295 dargestellt. Dieser bildet den entsprechenden PB-Filter-Eingangs- und PB-Filter-Ausgangsdatensatz ab.

Tabelle 25: Ausschnitt der Tabelle Ds

idDs	name	bezeichnung
1	DATEN-SATZ_295	Datensatzbeschreibung KVDT [KBV_ITA_VGEX_Datensatzbeschreibung_KVDT], Kassenärztliche Bundesvereinigung

1.3.1 Der PB-Filter-Eingangsdatensatz

Der PB-Filter-Eingangsdatensatz ist je nach folgendem Abrechnungskontext zu wählen:

DATENSATZ_295 enthält den PB-Filter-Eingangsdatensatz nach § 295 SGB V

Der PB-Filter-Eingangsdatensatz nach § 295 SGB V bezieht sich auf Fälle, die kollektivvertraglich durch einen niedergelassenen Leistungserbringer (Arztpraxis, MVZ, medizinische Labore)[31] erbracht werden und ist so beschaffen, dass fast alle seine Datenfelder aus der Struktur der KVDT-Datensatzbeschreibung für den Einsatz von IT-Systemen in der Arztpraxis der Kassenärztlichen Bundesvereinigung gemäß § 295 SGB V abgeleitet werden können[32].

Der PB-Filter-Eingangsdatensatz nach § 295 SGB V besteht aus folgenden Teilen:

- Behandlungsfall (Teildatensatz FALL)

Tabelle 26: Felder des PB-Filter-Eingangsdatensatzes nach § 295

Feld	Beschreibung	M/K	Zeichenlänge	BasisTyp
Behandlungsfall (FALL)				
BSNRAMBULANT	Betriebs- (BSNR) (ambulant)	M	9	SCHLUESSEL (BSNRambulant)
LANR	Lebenslange Arztnummer (LANR)	M	9	SCHLUESSEL (LANRambulant)
KASSEIKNR	Kostenträgernummer	M	9	TEXT
VERSICHERTENID-NEU	eGK-Versichertennummer	K	10	TEXT
PERSONENKREIS	besonderer Personenkreis	K	2	SCHLUESSEL (Personenkreis)

[31] Ambulante Leistungen nach §295 SGB V, §116 SGB V

[32] Zeichenart und Feldlänge der im Folgenden beschriebenen Datenfelder können vom KVDT-Datensatz abweichen, da hier die in der Spezifikation definierten Datentypen verwendet werden. Das Format DATUM wird beispielsweise über 10 Zeichen (TT.MM.JJJJ) abgebildet statt über 8 Zeichen (JJJJMMTT).

Achtung
PB-Filter des Eingangsdatensatzes der PB-Verfahren nach § 295

In den PB-Verfahren gemäß oKFE-RL dienen die EBM-Ziffern lediglich als auslösendes Kriterium der Dokumentationspflicht und nicht dem Abrechnungskontext. Aus dem Datensatz nach § 295 SGB V werden für die Filter als Adminkriterium das Datum der Behandlung (`EBMDATUM`) und zur Modulauslösung die entsprechenden Gebührennummern (EBMs), `VERSICHERTENIDNEU`, `KASSEIKNR` und `PERSONENKREIS` verwendet. Alle Felder sind als Muss-Felder entsprechend zu dokumentieren.

Der PB-Filter-Eingangsdatensatz im Rahmen der PB-Verfahren gilt bei kollektivvertraglichen Fällen eines niedergelassenen Leistungserbringers (ambulante Behandlung/medizinische Labore).

Die Gebührenordnungspositionen bzw. Gebührennummern sind nach dem gültigen einheitlichen Bewertungsmaßstab in der jeweils aktuellen Fassung zu kodieren.

1.4 Der Algorithmus zur Ermittlung der Dokumentationspflicht

Dieser Abschnitt beschreibt den Algorithmus zur Ermittlung der Dokumentationsverpflichtung von Modulen und die erforderlichen Fehlerprüfungen.

1.4.1 Einleitung und Überblick

Für jedes Modul ist ein formaler, logischer Ausdruck definiert, welcher sich aus einer verfahrensbezogen Teilbedingung (Abschnitt B 1.4.2) und einer administrativen Teilbedingung (Abschnitt B 1.4.3) zusammensetzt:

Auslösebedingung für ein Modul = `ModulAusloeser.bedingung` `UND` `AdminKriterium.bedingung`

Wenn eine der beiden Teilbedingungen nicht erfüllt ist, so löst der PB-Filter für das jeweilige Modul keine Dokumentationspflicht aus.

Für jeden Fall evaluiert der PB-Filter-Algorithmus sämtliche der in der Spezifikationsdatenbank hinterlegten Auslösebedingungen. Wird die Auslösebedingung erfüllt, so ist der Fall für das verknüpfte Modul dokumentationspflichtig.

Achtung

Generell gilt die Einschränkung, dass ein gleiches Modul pro Fall nur einmal ausgelöst wird. Es können aber mehrere verschiedenartige Module ausgelöst werden.

Dokumentationspflicht eines Falles
Ein Fall ist dokumentationspflichtig, wenn für ihn mindestens ein PB-Modul dokumentationspflichtig ist.

Dokumentationspflicht eines Moduls
Ein Modul (Datensatz) ist dann dokumentationspflichtig, wenn folgende Bedingungen zutreffen:

- Bei der Fehlerprüfung tritt kein Fehler auf (Abschnitt B 1.4.6).
- Die jeweilige modulspezifische Auslösebedingung ist erfüllt (Abschnitt B 1.4.2).

- Die administrativen Bedingungen sind erfüllt (Abschnitt B 1.4.3).

Zu beachten sind die in Abschnitt B 1.4.5 beschriebenen Stufen der Dokumentationsverpflichtung.

1.4.2 Verfahrensbezogen Einschlusskriterien

Überblick:
Für jede PB-Datensatz-Definition (= Modul) sind in der Tabelle `ModulAusloeser` (Tabelle 27) eine oder mehrere Auslösebedingungen (Attribut `bedingung`) hinterlegt.

Definition:
Jede Auslösebedingung der Tabelle `ModulAusloeser` definiert einen PB-Filter-Leistungsbereich.

Ein PB-Filter-Leistungsbereich ist somit ein Komplex von Leistungen, welche über den PB-Filter-Algorithmus zu einer Dokumentationsverpflichtung führen.

Beispiel:
Die Bedingung (Modul DKI)

```
EBM EINSIN DKI_EBM
UND
VERSICHERTENIDNEU <> LEER
UND
format(VERSICHERTENIDNEU;'[A-Z][0-9]{9}') = WAHR
UND
LENGTH(KASSEIKNR) = 9 UND LEFT(KASSEIKNR;2) = '10'
UND
PERSONENKREIS = '00'
```

DKK_EBM stellt die Einschlussschlussliste dar, die in der Tabelle `EBMListe` spezifiziert wurde und deren Schlüsselwerte in der Tabelle `EBMWert` zu finden sind.

Achtung

Einschlusskriterien mit `KASSEIKNR`, `VERSICHERTENIDNEU` und `PERSONENKREIS`

Verfahrensbezogen Einschlusskriterien können die Felder `KASSEIKNR`, `VERSICHERTENIDNEU` und `PERSONENKREIS` enthalten, mithilfe dessen geprüft wird, ob es sich um einen GKV-Versicherten handelt. Liegt die elektronische Gesundheitskarte (eGK) bei der Prüfung der definierten Kriterien noch nicht vor, erlischt **nicht** die Dokumentationspflicht. Die Auslösebedingung ist bei Vorliegen der Angaben `KASSEIKNR`/`VERSICHERTENIDNEU`/`PERSONENKREIS`/eGK erneut zu prüfen.

Üblicherweise entspricht ein PB-Datensatz genau einem PB-Filter-Leistungsbereich.

Definition in der Datenbank

Tabelle 27 gibt einen Überblick über die Tabelle `ModulAusloeser` der Spezifikationsdatenbank.

Tabelle 27: Struktur der Tabelle ModulAusloeser

Feldname	Datentyp	Bemerkung
`idModulAusloeser`	`INTEGER`	Primärschlüssel
`name`	`KURZER TEXT`	Technischer Name des PB-Filter-Leistungsbereichs
`bezeichnung`	`KURZER TEXT`	Bezeichnungstext der Auslösebedingung
`bedingung`	`LANGER TEXT`	Auslösebedingung für den PB-Filter-Leistungsbereich
`textDefinition`	`LANGER TEXT`	Medizinisch-inhaltliche Definition bzw. Erläuterung der Auslösebedingung
`fkModul`	`INTEGER`	Bezug zum Modul
`verpflichtend`	`BOOLEAN`	Wenn WAHR, ist das betreffende Modul bundesweit verpflichtend
`fkAdminKriterium`	`INTEGER`	Bezug zu den administrativen Abgrenzungskriterien
`fkDs`	`INTEGER`	Bezug zum entsprechenden Sollstatistikdatensatz Nicht relevant für die Programmbeurteilung
`internBild`	`KURZER TEXT`	Bild zum Modul
`ambulant`	`BOOLEAN`	Ist die Fallart ambulant?
`kollektiv`	`BOOLEAN`	Wird der Fall ambulant kollektivvertraglich abgerechnet?

Identifizierung der Fallart und der Art der Leistungserbringung

Zur Identifizierung der Fallart (`ambulant`) und Art der Leistungserbringung (`kollektiv`) wurden diese booleschen Attribute in die Tabelle `ModulAusloeser` integriert. Sie können wie folgt identifiziert werden:

`ambulant`

Dieses Attribut ist WAHR, wenn Feld `FALLART = 1` (Modulauslöser DKK und Exportmodul DKK).

Hinweis

Die Exportmodule der PB-Verfahren gemäß oKFE-RL bilden ausschließlich ambulante (kollektivvertragliche) Fälle ab.

kollektiv

Dieses Attribut kann über das Feld EBM identifiziert werden. Das Attribut `kollektiv` ist WAHR, wenn `EBM <> LEER.` In diesem Fall ist das Attribut `krankenhausabrechnung = FALSCH.`

1.4.3 Administrative Einschlusskriterien

Überblick

Die administrativen Regelungen beschreiben diejenigen Auslösebedingungen, welche über die modulspezifischen Kriterien (Abschnitt B 1.4.2) hinausgehen. Moduldokumentationen werden nur dann durch die PB-Filter-Software ausgelöst, wenn

1. der Patient ambulant behandelt wird,
2. der Behandlungsfall einen bestimmten zeitlichen Rahmen aufweist.

Bei ambulant behandelten Patienten existiert kein Aufnahmegrund. Die Bedingung `AUFNGRUND <> LEER` ist daher im administrativen Einschlusskriterium nicht enthalten. Der zeitliche Rahmen wird bei ambulanten Fällen über das Behandlungsdatum definiert:`jahreswertListe(EBMDATUM) EINSIN (2021)`

Definition in der Datenbank

Tabelle 28 gibt einen Überblick über die Tabelle `AdminKriterium`, welche die administrativen Abgrenzungskriterien definiert.

Tabelle 28: Struktur der Tabelle AdminKriterium

Feldname	Datentyp	Bemerkung
`idAdminKriterium`	`INTEGER`	Primärschlüssel
`name`	KURZER TEXT	Technischer Name des administrativen Kriteriums
`bezeichnung`	KURZER TEXT	Inhaltliche Definition des administrativen Kriteriums
`bedingung`	LANGER TEXT	Auslösebedingung für den PB-Filter-Leistungsbereich
`info`	KURZER TEXT	Bezeichnungstext des administrativen Kriteriums

1.4.4 Struktur und Syntax der Auslösebedingungen

Die Variablen der Auslösebedingungen

Die in den Auslösebedingungen erlaubten Variablen sind in der Tabelle SyntaxVariable definiert.

Die Variablennamen (Attribut SyntaxVariable.name) bestehen aus maximal 32 Zeichen. Sie dürfen nur die Buchstaben A bis Z (Großbuchstaben) und die Ziffern 0 bis 9 enthalten. Ein Feldname muss immer mit einem Buchstaben beginnen. Umlaute und Sonderzeichen sind in Feldnamen nicht erlaubt. Ein Feldname darf auch nicht ein reserviertes Wort sein (z. B. Namen von Operatoren wie EINSIN, siehe Tabelle 30).

Typen

Jede Variable hat einen Basistyp. Tabelle 29 gibt einen Überblick über die möglichen Basistypen.

Jeder der in der Tabelle SyntaxVariable definierten Variablen ist über den Wert des Attributs SyntaxVariable.fkTdsFeld ein Feld des PB-Filter-Eingangsdatensatzes zugeordnet. Jedes dieser Felder besitzt einen Basistyp.

Tabelle 29: Basistypen der Variablen

Basistyp	Bezeichnung	Beispiele (Literale)
BOOL	Boolesche Variable	WAHR, FALSCH
TEXT	Zeichenkette (String)	„Spezifikation“
GANZEZAHL	… -2, -1, 0, 1, 2, 3, …	1
ZAHL	Zahl (mit oder ohne Nachkommastellen)	Erfassung: 25,4 oder -100,8 Export: 25.4 oder 100.8
DATUM	Zehnstelliges Datum	'01.01.2013'
NUMSCHLUESSEL	Numerisch kodierter Schlüssel (wie GANZEZAHL)	0
SCHLUESSEL	Alphanumerischer Schlüssel	'19.1', '07'
UHRZEIT	Uhrzeit	'10:15'
JAHRDATUM	Jahresdatum	2018

Die meisten Schlüsselwerte werden als GANZEZAHL kodiert, d. h., dass die Kodes nicht in Hochkommata gesetzt werden dürfen.

Listen

Eine Variable wird als Liste interpretiert, wenn der Wert des Attributs SyntaxVariable.istListe = WAHR ist.

Literale

Alphanumerische Literale (z. B. SCHLUESSEL) werden von einfachen Hochkommata eingeschlossen, während Zeichenketten vom Datentyp TEXT in doppelte Anführungsstriche gesetzt werden müssen.

Dies gilt nicht für numerischen Literale (GANZEZAHL, ZAHL, NUMSCHLUESSEL) und Literale des Datentyps BOOL (Wahrheitswerte).

Die Operatoren der Auslösebedingungen

Tabelle 30 gibt einen Überblick über die in der Syntax zulässigen Operatoren. Der aktuelle Überblick über alle zulässigen Operationen (inkl. Operanden) ist in Tabelle SyntaxOperator der PB-Filter-Datenbank zu finden.

Tabelle 30: Präzedenz und Assoziativität der Operatoren[33]

Präzedenz	Assoziativität	Operator	Erläuterung
0	links	EINSIN	Mengenoperator „EINSIN“
	links	KEINSIN	Mengenoperator „KEINSIN“
3	links	<	Vergleichsoperator „kleiner“
	links	>	Vergleichsoperator „größer“
	links	<=	Vergleichsoperator „kleiner gleich“
	links	>=	Vergleichsoperator „größer gleich“
4	links	=	Vergleichsoperator „gleich“
	links	<>	Vergleichsoperator „ungleich“
5	rechts	NICHT	Logischer Operator „NICHT“
6	links	UND	Logischer Operator „UND“
7	links	ODER	Logischer Operator „ODER“

Anmerkungen:

Operatoren mit beidseitigen Listenfeldern als Operanden:

EINSIN: Wenn mindestens ein Element aus der linken Liste in der rechten Liste enthalten ist, so ist der Ausdruck wahr (nichtleere Schnittmenge).

KEINSIN: Wenn kein Element der linken Liste in der rechten Liste enthalten ist, so ist der Ausdruck wahr (leere Schnittmenge)[34].

[33] In dieser Übersichtstafel hat jeder einzelne Operator eine Präzedenzstufe (höchste Präzedenzstufe ist 0). Operatoren, welche die gleiche Stufe haben, werden nach den Regeln der Assoziativität aufgelöst.

[34] Dieser Operator ist redundant, da er auch durch Negation des EINSIN–Operators abgedeckt ist.

Beispiel:
Folgende Regel prüft, ob ein Element der Listenvariable PROZ einen bestimmten Kode besitzt:

```
PROZ EINSIN TON_OPS
```

String-Vergleich bei Abrechnungsziffern
Die Operation EINSIN basiert für Abrechnungsziffern auf einem Vergleich von Zeichenketten (String-Vergleich): Für jedes Element der linken Liste wird über einen String-Vergleich geprüft, ob es in der rechten Liste enthalten ist.

Eine Modifikation des String-Vergleichs ist beim String-Vergleich von Diagnosen nötig: Hier werden die Suffixe +, * und ! sowie die Zusatzkennzeichen A, G, V und Z ignoriert. Nicht ignoriert wird das Sonderzeichen ..

Achtung

In der PB-Filter-Datenbank sind nur terminale Kodes enthalten. Entsprechend muss sichergestellt sein, dass auch die Kodes aus dem Eingangsdatensatz terminal sind. Darauf ist besonders zu achten, wenn Daten ungeprüft aus Subsystemen übernommen wurden.

1.4.5 Stufen der Dokumentationsverpflichtung

Die Spezifikation für PB-Filter enthält bundesweit **verpflichtende** (dokumentationspflichtige) Verfahren. Bundesweit dokumentationspflichtige Verfahren sind erkennbar am Attribut `verpflichtend` in der Tabelle `ModulAusloeser`.

1.4.6 Fehlerprüfung

Vor Evaluation der in den vorhergehenden Abschnitten beschriebenen Auslösebedingungen ist eine Fehlerprüfung durchzuführen. Die Fehlerprüfung bezieht sich auf die Daten des PB-Filter-Eingangsdatensatzes. Das Ergebnis der Fehlerprüfung wird im Teildatensatz FEHLER gespeichert.

Tabelle 31: Fehlerkodes des PB-Filters

Beschreibung des Fehlers	Fehlerkode	Formale Fehlerbedingung	Fehlermeldung
Überprüfung des Formats	1	Abschnitt B 2.4.8	Der Wert '<WERT>' des Datenfeldes <Feld.name> '<TdsFeld.bezeichnung>' ist kein gültiger <BasisTyp.name>-Wert (<BasisTyp.bezeichnung> <BasisTyp.format>).
Überprüfung der Feldlänge	2	Abschnitt B 2.4.8	Der Wert '<WERT>' des Datenfeldes <Feld.name>

Beschreibung des Fehlers	Fehler-kode	Formale Fehlerbedingung	Fehlermeldung
			'<TdsFeld.bezeichnung>' überschreitet die zulässige Feldlänge <Feld.laenge>.
Sind in den Datenfeldern mit internen und externen Schlüsseln (Basistyp SCHLUESSEL oder NUM-SCHLUESSEL) gültige Schlüsselkodes eingetragen?[35]	3	Abschnitt B 2.4.8	Ungültiger Schlüsselkode <Wert> des Schlüssels <Schluessel.name> im Datenfeld <Feld.name> '<TdsFeld.bezeichnung>'!
Überprüfung numerischer Wertebereiche	4	Abschnitt B 2.4.8	Der Wert '<WERT>' des Datenfeldes <Feld.name> '<TdsFeld.bezeichnung>' ist kleiner als '<Feld.min>' oder Der Wert '<WERT>' des Datenfeldes <Feld.name> '<TdsFeld.bezeichnung>' ist größer als '<Feld.max>'
Ist ein Muss-Feld ausgefüllt?	5	Abschnitt B 2.4.8: Attribut <TdsFeld.fkMussKann>	Das Datenfeld <Feld.name> '<TdsFeld.bezeichnung>' muss einen gültigen Wert enthalten.
Ist der Fall dem Erfassungsjahr der Spezifikation zugeordnet? (Spezifikation <ERFASSUNGSJAHR>)	6	EBMDATUM < '01.01.<ERFASSUNGSJAHR>' ODER EBMDATUM > '31.12.<ERFASSUNGSJAHR>'	Der Fall ist im Jahr <ERFASSUNGSJAHR> nicht dokumentationspflichtig: Aufnahmedatum/OP-Datum = <Wert>

In <Wert> ist der Wert des Datenfeldes der Falldaten einzusetzen, auf den sich die Fehlermeldung bezieht. Ansonsten sind in die <…> -Ausdrücke die entsprechenden Einträge aus der Datenbank einzusetzen.

Im Fehlerfall sind entsprechende Einträge im Teildatensatz FEHLER zu generieren:

- Fehlerkode in Attribut FKODE,

[35] Die Korrektheit (z.B. ausschließlich gültige terminale Kodes oder Verwendung gültiger Katalogversionen) der extern definierten Kodes (EBM) muss vor der Prüfung des Datensatzes durch die PB-Verfahren-Filter-Software sichergestellt sein.

- Fehlermeldung in Attribut FMELDUNG.

Es wird beim Auftreten eines Fehlers nicht weiter geprüft, ob für den Behandlungsfall Moduldokumentationen ausgelöst werden.

2 PB-Dokumentation

Die vorliegenden Spezifikationskomponenten für die PB-Dokumentation dienen der Erstellung von Software zur Datenerfassung, Plausibilitätsprüfung und Datenübermittlung für die Programmbeurteilungen im Rahmen der oKFE-RL. Diese sollen die Bereitstellung valider und vergleichbarer Daten gewährleisten. Neben der Datenbank zur PB-Dokumentation zählen zu den Spezifikationskomponenten der PB-Dokumentation die Ausfüllhinweise und die Dokumentationsbögen. Weiterhin werden ergänzende Informationen zur Verfügung gestellt, sowie der Java-Code für ein Testprogramm und eine Testklasse mit Testfällen.

2.1 Anmerkungen zur Struktur der Spezifikation zur PB-Dokumentation

Die Spezifikation zur PB-Dokumentation ist in einer relationalen Datenbank abgelegt. Zurzeit wird sie ausschließlich als Access-Datenbank (MS Access 2007-2013) zur Verfügung gestellt. Der Name der Spezifikation richtet sich nach folgendem Schema:

`<Erfassungsjahr>_oKFE_PBDOK_V<Versionsnummer>.mdb`

`<Erfassungsjahr>` bezeichnet das Jahr, in dem die PB-Dokumentation stattfindet. `<Versionsnummer>` bezeichnet die 2-stellige Versionsnummer (z. B. 01).

Beispiel:
Im Erfassungsjahr 2021 ist die Spezifikation `2021_oKFE_PBDOK_V01.mdb`[36] gültig.

Weiterführende Erläuterungen zum Benennungsschema für Spezifikationskomponenten sind der Einleitung, Abschnitt A 1.1.2, zu entnehmen. Die Tabellenstruktur der Spezifikationsdatenbank wird in Abschnitt B beschrieben.

Folgende Attribute treten in vielen Tabellen auf:

- `name` ist in der Regel als technischer Name zu verstehen. Zum Beispiel wird `Feld.name` als Variablenname in den Plausibilitätsregeln verwendet.
- `bezeichnung` ist eine kurze Beschreibung. Zum Beispiel ist `BogenFeld.bezeichnung` der Text, welcher ein Feld auf einem Eingabeformular beschreibt.
- `bedingung` enthält einen logischen Ausdruck. Prominentester Vertreter dieses Attributtyps ist das Attribut `bedingung` in der Tabelle `Regeln`.

Abfragen der Datenbank

Die Abfragen der Access-Datenbank geben einen vereinfachenden Überblick über die Inhalte der Spezifikation.

[36] Die Versionsnummer der gültigen Spezifikation (z.B. V01, V02 usw.) ist dem zuletzt veröffentlichten Update zu entnehmen.

- `Datensätze`
  Diese Abfrage liefert einen Überblick über die in der Spezifikation enthaltenen Module (verpflichtende und freiwillige Module), die der aktuell gültigen Version entsprechen.
- `Datenfeldbeschreibung`
  Hier sind alle Bogenfelder der spezifizierten Module, sortiert nach Modulname, Bogenname und Zeilennummer der Bogenfelder dargestellt (Abschnitt B 2.3).
- `DatenfeldbeschreibungFürEinModul`
  Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „DKK“) angegeben werden und man erhält eine entsprechende modulbezogene Auswahl der Datenfeldbeschreibung.
- `Plausibilitätsregeln`
  Diese Abfrage enthält alle Plausibilitätsregeln der spezifizierten Module, sortiert nach Modulname und Nummer der Regel (Abschnitt B 2.4).
- `PlausibilitätsregelnFürEinModul`
  Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „DKK“) angegeben werden und man erhält eine entsprechende modulbezogene Auswahl der Plausibilitätsregeln.
- `Teildatensätze`
  Diese Abfrage liefert einen Überblick über die Teildatensätze und die Regeln für das Anlegen von Teildatensätzen (Abschnitt B 2.3).
- `Ersatzfelder`
  Dies ist eine Auflistung der Ersatzfelder für die Bogenfelder, die modulspezifisch anonymisiert werden müssen (Export von Teildatensätzen).
- `Exportfelder`
  Wenn man diese Abfrage aufruft, erhält man eine Übersicht über alle Exportfelder. Exportfelder für Listenfelder werden nicht pro Listenelement, sondern pro Listenfeld dargestellt. Die Anzahl der Elemente ist der Abfrage zu entnehmen (`Exportfelder.elemente`).
- `ExportfelderFürEinModul`
  Diese Abfrage zeigt eine Auswahl der Exportfelder eines Moduls (Modulname ist explizit anzugeben). Man erhält eine Übersicht über die zu exportierenden Felder inkl. Zuordnung zum Teildatensatz. Exportfelder für Listenfelder werden nicht pro Listenelement, sondern pro Listenfeld dargestellt. Die Anzahl der Elemente ist der Abfrage zu entnehmen (`ExportfelderFürEinModul.elemente`) (Abschnitt B 2.5.2).
- `Feldgruppen`
  Diese Abfrage liefert eine Übersicht über alle Feldgruppen (Abschnitt B 2.4.7).
- `FeldgruppenFürEinModu`
  Wenn man diese Abfrage aufruft, so muss der Modulname (z. B. „DKK“) angegeben werden und man erhält eine entsprechende modulbezogene Auswahl der Feldgruppen eines Moduls.
- `WertebereicheNumerischerFelder`
  Diese Abfrage liefert eine modulübergreifende Anzeige der numerischen Datenfelder (Typ ZAHL und GANZEZAHL) und ihrer Wertebereiche.
- `WertebereicheNumerischerFelderFuerEinModul`
  Hier werden die numerischen Datenfelder (Typ ZAHL und GANZEZAHL) und ihrer Wertebereiche für ein Modul angezeigt. Das Modul muss bei Aufrufen der Abfrage angegeben werden.
- `ÜberschriftenFürEinModul`

Diese Abfrage liefert eine Anzeige der Überschriften für das angegebene Modul. Angegeben werden Start-und Ende-Felder der Überschriften, sowie die Ebene der Überschriften.

- `Schlüsselkodes`
  Diese Abfrage zeigt alle Schlüssel und die zugehörigen Schlüsselwerte an.
- `Ausfüllhinweise`
  Hier wird die Zuordnung von Ausfüllhinweisen (htm.Dateien) zu den Feldern in den einzelnen Modulen angezeigt.
- `AusfüllhinweiseFürEinModul`
  Hier wird die Zuordnung von Ausfüllhinweisen (htm.Dateien) zu den Feldern eines Moduls angezeigt. Das Modul muss bei Aufrufen der Abfrage angegeben werden.

2.2 Patientenidentifizierende Daten

In der PB-Spezifikation werden PB-Daten mithilfe sogenannter patientenidentifizierender Daten (PID) im Regelbetrieb patientenbezogen zusammengeführt.

Mit den PB-Daten werden für jeden Vorgang die folgenden berechneten Ersatzfelder

- `versichertenidgkv`
- `versichertenstatusgkv` und
- `kasseiknr2Stellen` exportiert.

In der Spezifikationsdatenbank werden die Module mit PID-Verfahren über das Attribut `pid` der Tabelle `Modul` gekennzeichnet. Für welche Module dasselbe Patientenpseudonym zu generieren ist, wird über das Attribut `fkPseudonymVerfahren` beschrieben. Ob auch von der DAS eine verfahrensbezogene Pseudonymisierung der LE-Daten durchzuführen ist, wird durch das Feld `das` (WAHR/FALSCH) festgelegt.

Tabelle 32: Struktur der Tabelle `PseudonymVerfahren`

Feldname	Datentyp	Bemerkung
`idPseudonymVerfahren`	`INTEGER`	Primärschlüssel
`bezeichnung`	`KURZER TEXT(255)`	Bezeichnung des Pseudonymverfahrens, z. B. PB-Verfahren zur Früherkennung von Darmkrebs (DKK, DKI)
`name`	`KURZER TEXT(255)`	Name des Pseudonymverfahrens, z. B. DK
`vst`	`BOOLEAN`	PID-Pseudonymisierung durch die Vertrauensstelle
`das`	`BOOLEAN`	LE-Pseudonymisierung (oKFE-Verfahren) durch die Datenannahmestelle

2.3 Datenfeldbeschreibung

Für jedes Modul existiert eine eigene Datenfeldbeschreibung. Sie spezifiziert alle auszufüllenden Datenfelder (Bogenfelder, auch Items genannt) und besteht aus mehreren Tabellen (Abbildung 16), die in den nachfolgenden Abschnitten erläutert werden.

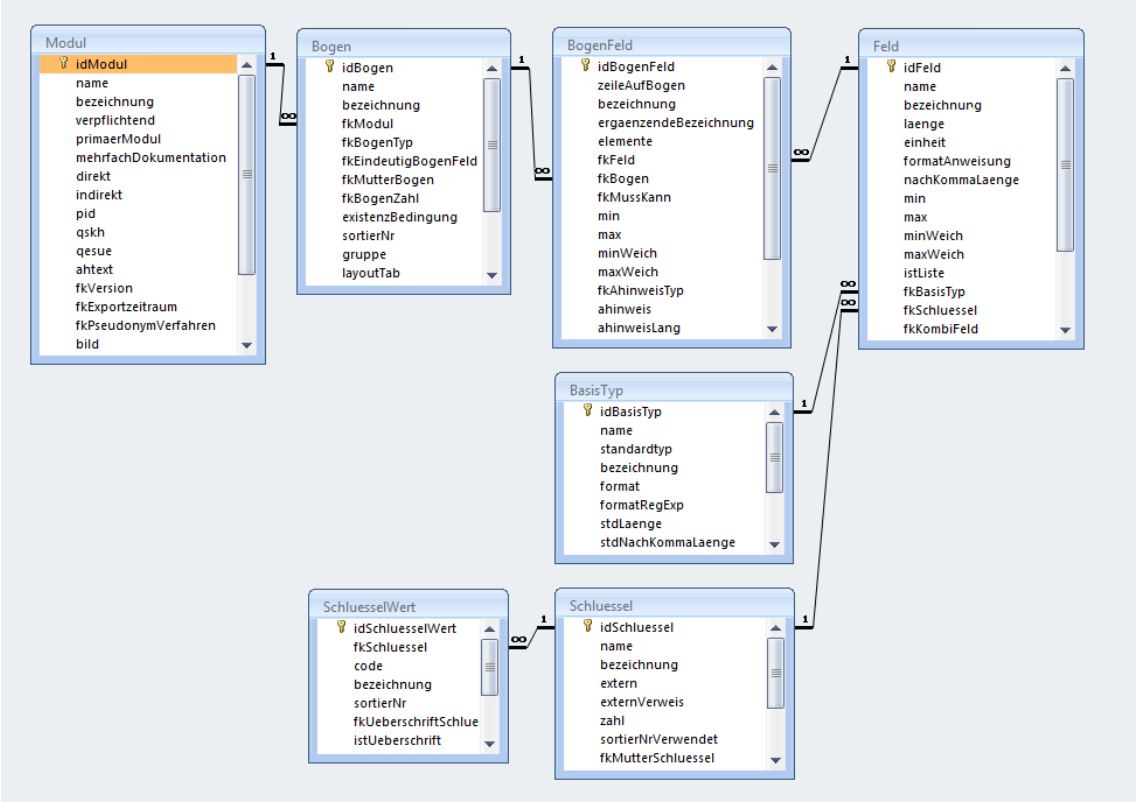


Abbildung 16: Tabellen und Relationen der Datenfeldbeschreibung

Die Abfragen `Datenfeldbeschreibung` und `DatenfeldbeschreibungFürEinModul` der Access-Datenbank ermöglichen den Überblick über diese Struktur.

Die Beschreibung der Datenfelder hat folgende Ziele:

- Bereitstellung der Informationen, welche für die Programmierung des Eingabeformulars und für die Sicherung der eingegebenen Daten nötig sind
- Vermeidung von Redundanzen
- Typisierung der Felder nach fachlichen und datentechnischen Kriterien

Das für den Anwender wichtigste Merkmal ist die Bezeichnung des Datenfelds (Attribut `BogenFeld.bezeichnung`).

Die Datenfeldbeschreibung ist teilweise auf dem jeweiligen Dokumentationsbogen eines Moduls („Bogensicht“) abgebildet. Zu beachten ist dabei, dass die „Bogensicht“ lediglich die Sicht der medizinischen Fachgruppen, die die Module entwickeln, darstellt. Bei verteilten Softwarelösungen für einen Leistungserbringer hingegen ist die Bogensicht dann nicht mehr adäquat, wenn die Bestandteile eines Bogens auf verschiedene Teilsysteme verteilt sind. Die Daten eines Bogens werden in diesen Fällen für den Export aus den einzelnen Teilsystemen zusammengestellt.

Hinweis

Die Papierbögen werden lediglich zu Illustrationszwecken zur Verfügung gestellt. Sie sind zur Dokumentation nicht zugelassen.

Im Kontext einer integrierten, prozessorientierten Dokumentationssoftware müssen die Teildatensätze nicht direkt in Eingabeformulare umgesetzt werden. Es ist sinnvoller, die Teile eines Dokumentationsbogens zu dem Zeitpunkt und in dem Dokumentationskontext zu erfragen, der sich in den Prozessablauf eines Leistungserbringers einordnet.

2.3.1 Dokumentationsmodule (Datensätze)

Ein Modul der Spezifikation enthält die Datensatzdefinition von einem oder mehreren PB-Verfahren (oKFE-RL). Abhängig von (inhaltlich oder organisatorisch) abzugrenzenden Bereichen kann ein PB-Verfahren mehrere Module umfassen (Beispiel: Module Koloskopie und iFOB-Test im PB-Verfahren zur Früherkennung von Darmkrebs). Die PB-Dokumentationssoftware kann für einen Behandlungsfall eine oder mehrere Moduldokumentationen anlegen, die nach Dokumentationsabschluss an die Datenannahmestelle übermittelt werden. Fehlerfreie Moduldokumentationen (verkürzt „Module“), die die Basis der Datenauswertungen bilden, werden dem Leistungserbringer von der Datenannahmestelle bestätigt. Aus technischer Sicht ist ein Modul durch einen eindeutigen Namen gekennzeichnet. Es umfasst mindestens einen Teildatensatz. In der Tabelle `Modul` der PB-Spezifikation finden sich die zentralen Definitionen eines Moduls.

Tabelle 33: Struktur der Tabelle `Modul`

Feldname	Datentyp	Bemerkung
`idModul`	`INTEGER`	Primärschlüssel
`Name`	`KURZER TEXT(32)`	Eindeutiger technischer Name
`bezeichnung`	`KURZER TEXT(255)`	Erläuternde Bezeichnung
`verpflichtend`	`BOOLEAN`	Besteht für das Modul eine Dokumentationsverpflichtung?
`primaerModul`	`BOOLEAN`	Ist das Modul ein Primärmodul?
`mehrfachDokumentation`	`BOOLEAN`	Ist ein mehrfaches Anlegen eines gleichartigen Datensatzes pro Fall zulässig (`WAHR/FALSCH`)?
`ahtext`	`LANGER TEXT`	Einleitender Text für den Ausfüllhinweis eines Moduls
`fkVersion`	`INTEGER`	Gültige Version des jeweiligen Moduls
`bild`	`KURZER TEXT (20)`	Modulspezifisches Bild

Auslösung der Moduldokumentation

Der auslösende Sachverhalt für die Dokumentationspflicht ist in der Spezifikation für PB-Filter-Software definiert. Die PB-Filter-Software greift zu diesem Zweck unter anderem auf Gebührenordnungspositionen (EBM-Kodes) zurück, die im Arztinformationssystem (AIS) bzw. Laborinformationssystem (LIS) verfügbar sind

Primärmodule

Für Primärmodule sind in der Spezifikation für PB-Filter-Software Auslösebedingungen definiert

Mehrfachdokumentation

Pro Fall darf höchstens ein Datensatz eines Moduls angelegt und exportiert werden, wenn `mehrfachDokumentation = FALSCH` ist (Abschnitt B 1.4).

Die Dokumentation aller notwendigen Teildatensätze soll durch die Software sichergestellt werden.

Achtung

Die PB-Dokumentationssoftware muss sicherstellen, dass die Mehrfachdokumentation gleichartiger Datensätze für einen Fall unterbunden wird, sofern diese nicht zulässig ist. Stattdessen sind separate Teildatensätze zu dokumentieren

2.3.2 Teildatensätze

Die Begriffe „Teildatensatz“ und „Bogen“ werden synonym gebraucht. In den der Illustration dienenden Dokumentationsbögen werden alle Teildatensätze aufgeführt. Dabei erfolgt eine chronologische Anordnung, was dazu führen kann, dass ein Teildatensatz durch einen anderen, hierarchisch untergeordneten Teildatensatz unterbrochen wird.

Ein Teildatensatz
ist jeweils einem Modul zugeordnet,
besitzt einen Namen, der innerhalb eines Moduls eindeutig ist,
kann unter definierten Bedingungen mehrfach pro Fall erzeugt werden.

Die Teildatensätze der PB-Spezifikation sind in der Tabelle `Bogen` definiert (Tabelle 34).

Tabelle 34: Struktur der Tabelle `Bogen`

Feldname	Datentyp	Bemerkung
`idBogen`	`INTEGER`	Primärschlüssel
`name`	KURZER TEXT	Technischer Name des Teildatensatzes
`bezeichnung`	KURZER TEXT	Beschreibender Text
`extistenzBedingung`	LANGER TEXT	Logische Bedingung (Regeln für das Anlegen von Teildatensätzen)

Feldname	Datentyp	Bemerkung
sortierNr	INTEGER	Reihenfolge der Unterbögen bei der Erfassung und beim Export
fkModul	INTEGER	Obligatorischer Fremdschlüssel zu einem Modul
fkBogenZahl	KURZER TEXT(1)	Anzahl der auszufüllenden Teildatensätze pro Patient (bezogen auf den Basisbogen oder ggf. auf den Mutterteildatensatz)
fkMutterBogen	INTEGER	Optionaler Fremdschlüssel, welcher den Mutterteildatensatz eines Teildatensatzes definiert
fkBogenTyp	KURZER TEXT(1)	Spezifiziert den für den Export relevanten Bogentyp: Mögliche Werte B, K oder O. Die Angabe ist obligatorisch.
fkEindeutigBogen-Feld	INTEGER	Fremdschlüssel auf ein Bogenfeld, das mehrfach vorhandene Teildatensätze eines Datensatzes identifiziert

Benennung von Teildatensätzen

Ein Teildatensatz wird durch die folgende Kombination von Modulnamen und Bogennamen identifiziert und angesprochen:

<Modul.name>:<Bogen.name>

Beispiele:

DKK: B ist der Basisbogen des Moduls PB-Verfahren zur Früherkennung von Darmkrebs Koloskopie

ZKP: B ist der Basisbogen des Moduls PB-Verfahren zur Früherkennung von Zervixkarzinomen Primärscreening/Abklärungsuntersuchung

Bogentyp

Der Kerndatensatz besteht aus mindestens einem Basisteildatensatz und kann durch einen oder mehrere Teildatensätze ergänzt werden. Das Attribut Bogen.fkBogenTyp definiert für jeden Teildatensatz seine Rolle im und seine Zugehörigkeit zum Kerndatensatz. In Tabelle 35 sind die Bezeichnungen der einzelnen Bogentypen definiert.

Tabelle 35: Inhalte der Tabelle BogenTyp

idBogenTyp	Bezeichnung
B	Basisteildatensatz (Teil des Kerndatensatzes)
K	Teildatensatz ist Teil des Kerndatensatzes und kein Basisteildatensatzes
O	Teildatensatz ist Teil des optionalen Datensatzes

Hierarchie von Teildatensätzen

Der Ausgangspunkt („root“) für die Teildatensatzhierarchie eines Moduls ist immer der Basisteildatensatz (Wert B des Attributs `Bogen.fkBogenTyp`). Ein abhängiger Teildatensatz besitzt einen Mutterteildatensatz, der über das Attribut `fkMutterBogen` der Tabelle `Bogen` definiert ist.[37]

Auf diese Weise lässt sich für jedes Modul ein „Hierarchiebaum“ der Teildatensätze aufbauen.

Regeln für das Anlegen von Teildatensätzen

Jedes Modul muss die Definition genau eines Basisteildatensatzes enthalten (Wert B des Attributs `fkBogenTyp` der Tabelle `Bogen`). Wenn die Dokumentation eines Moduls durchgeführt wird, muss der Basisteildatensatz genau einmal angelegt werden (z. B. in der Exportdatei). Das Attribut `fkBogenZahl` gibt Auskunft darüber, wie oft ein Teildatensatz pro Vorgang angelegt werden darf. Folgende Werte des Attributs sind möglich:

- 1 = Genau ein Teildatensatz muss ausgefüllt werden
- + = Mindestens ein Teildatensatz muss ausgefüllt werden
- ? = Höchstens ein Teildatensatz darf ausgefüllt werden
- * = Eine beliebige Anzahl von Teildatensätzen kann ausgefüllt werden

Die Kardinalität eines abhängigen Teildatensatzes bezieht sich auf den Mutterteildatensatz. Der Basisteildatensatz hat immer die Kardinalität 1.

Die Ausprägung `fkBogenZahl = *` definiert eine 1-n-Beziehung. Man beachte, dass das Attribut `fkBogenZahl` wichtig für den XML-Aufbau des PB-Datensatzes ist und im Schema Berücksichtigung findet.

Beispiele:

Der Teildatensatz `DKK:B` muss als Basisteildatensatz genau einmal ausgefüllt werden (`fkBogenZahl = 1`).

Man beachte weiterhin, dass die im Attribut `fkBogenZahl` der Tabelle `Bogen` definierten Kardinalitäten durch Definitionen in den nachfolgend beschriebenen Attributen `existenzBedingung` bzw. `fkEindeutigBogenFeld` eingeschränkt werden können.

Inhaltliche Voraussetzung für das Anlegen von Teildatensätzen

Das Attribut `existenzBedingung` ist eine logische Bedingung (Syntax gemäß Abschnitt B 2.4.2) für das Anlegen eines Teildatensatzes. Die referenzierten Bogenfelder der Existenzbedingung beziehen sich auf den Mutterteildatensatz.

Die Dokumentationssoftware muss die Existenzbedingung als Trigger für das Anlegen eines abhängigen Teildatensatzes nutzen. Wenn die Existenzbedingung eines potenziellen Kindteildatensatzes erfüllt ist, so muss der Kindteildatensatz auch angelegt und übermittelt werden.

[37] Falls der Mutterteildatensatz nicht über das Attribut `fkMutterBogen` explizit definiert ist, so gilt implizit der Basisteildatensatz des Moduls als Mutterteildatensatz.

Andererseits gilt: Wenn die entgegennehmende Stelle einen Kindteildatensatz erhält, für den die zugehörige Existenzbedingung im Mutterteildatensatz nicht erfüllt ist, so ist das eine relationale Plausibilitätsverletzung.

Identifizierende Attribute mehrfach vorhandener Teildatensätze
Teildatensätze, die mehr als einmal ausgefüllt werden dürfen (Werte + und * des Attributs `fkBogenZahl`), sind nicht mehr durch die Vorgangsnummer voneinander unterscheidbar. Diese Teildatensätze benötigen ein zusätzliches identifizierendes Bogenfeld, das im Attribut `fkEindeutigBogenFeld` festgelegt wird.

Beim Anlegen einer Tabelle für die Speicherung eines mehrfach vorhandenen Teildatensatzes muss der Primärschlüssel mindestens die Attribute `Vorgangsnr`[38], `VersionsNr`[39] und das in `fkEindeutigBogenFeld` definierte Feld umfassen.

Wenn es bei den Teildatensätzen mehr als eine Ebene gibt, muss der Wert des Attributs `fkEindeutigBogenFeld` eines Kindbogens eindeutig in Bezug auf den übergeordneten Bogen sein. Hierbei kann sich die Eindeutigkeit des Wertes auf den Elternbogen (z. B. den Prozedurbogen) beschränken, sodass die Kombination beider Werte in Bezug auf den gesamten Vorgang eindeutig ist. Diese Bedingung wird auch erfüllt, wenn das Attribut `fkEindeutigBogenFeld` in Bezug auf den übergeordneten Basisbogen und damit auf den gesamten Vorgang eindeutig ist.

2.3.3 Datenfelder (Bogenfelder)

Jedes auf einem Teildatensatz vorhandene und auszufüllende Feld wird als Datenfeld (Item, Bogenfeld) bezeichnet. Datenfelder sind charakterisiert durch ihren Namen (Bezeichnung) und die Spezifikation des einzutragenden Inhalts.

Die Bezeichnung[40] wird so gewählt, dass sie einem medizinischen Experten unmittelbar verständlich ist. Die Spezifikation des Inhalts umfasst hingegen sowohl eine fachliche (medizinische) als auch datentechnische Typisierung. Dagegen repräsentieren die in der Tabelle `Feld` aufgelisteten Felder inhaltlich gleiche Dokumentationsfelder mehrerer Module (Abschnitt B 2.3.1), der datentechnische Typ (`BasisTyp`) charakterisiert das Format des Feldes (Abschnitt B 2.3.1).

Jedes Datenfeld hat zwingend einen Bezug zu einem Teildatensatz und zu einem technischen Feld. Weitere Eigenschaften sind die Bogenfeldbezeichnung und die fortlaufende Nummer im Teildatensatz. Die Datenfelder sind in der Tabelle `BogenFeld` gespeichert.

Identifizierendes Merkmal eines Datenfelds ist eine Kombination aus `fkBogen` und `fkFeld`. Das bedeutet, dass das Datenbankschema gewährleistet, dass der technische Feldname (`Feld.name`) in einem Teildatensatz maximal einmal vorkommt. Per definitionem muss ein

[38] Bei den Zusatzfeldern ist zu beachten, dass die Feldnamen beim Export durch die entsprechenden XML-Elemente zu ersetzen sind.
[39] Bei der entgegennehmenden Stelle kommt noch das Feld `RegistrierNr` hinzu, da dort Datensätze verschiedener Krankenhäuser gesammelt werden.
[40] Gegebenenfalls im Kontext der Überschriften (Abschnitt B 2.3.4).

Datenfeldname sogar innerhalb eines Moduls eindeutig sein, d. h. dass eine Abfrage mit dem Primärschlüsselpaar (`modulNr`, `feldNr`) genau einen Primärschlüssel `idBogenFeld` liefert.

Tabelle 36: Struktur der Tabelle `BogenFeld`

Feldname	Datentyp	Bemerkung
idBogenFeld	INTEGER	Primärschlüssel
zeileAufBogen	DOUBLE	bestimmt die Reihenfolge von Datenfeldern im Dokumentationsbogen
gliederungAufBogen	KURZER TEXT	Gliederungsnummer, die im Dokumentationsbogen angezeigt wird
bezeichnung	KURZER TEXT	Beschreibender Text zum Feld auf dem Dokumentationsbogen
ergaenzendeBezeichnung	KURZER TEXT	Optionale ergänzende Bezeichnung zu einem Bogenfeld.
elemente	INTEGER	Anzahl der Elemente bei Listenfeldern
fkFeld	INTEGER	Fremdschlüssel zu dem Teildatensatz und zu dem Feld, bilden zusammen die identifizierenden Merkmale
fkBogen	INTEGER	
fkMussKann	KURZER TEXT(1)	M oder K, Unterscheidung zwischen Muss- und Kann-Feldern
Min	DOUBLE	Harte Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulspezifisch). Die Definition ist optional.
Max	DOUBLE	Harte Obergrenze des Wertebereichs eines numerischen Datenfelds (modulspezifisch). Die Definition ist optional.
minWeich	DOUBLE	Weiche Untergrenze des Wertebereichs eines numerischen Datenfelds (modulspezifisch). Die Definition ist optional.
maxWeich	DOUBLE	Weiche Obergrenze des Wertebereichs eines numerischen Datenfelds (modulspezifisch). Die Definition ist optional.
fkAhinweisTyp	INTEGER	Typisierung des Ausfüllhinweises
ahinweis	KURZER TEXT(32)	Name des HTML-Ausfüllhinweises ohne Endung .htm (Abschnitt B 2.3.5)
ahinweisLang	BOOLEAN	Handelt es sich um einen langen Ausfüllhinweis?

Muss- und Kann-Felder

Jedes Bogenfeld ist als Muss- oder Kann-Feld zu deklarieren:

- Ein Muss-Feld (`M`) muss innerhalb eines angelegten Teildatensatzes immer ausgefüllt sein (Abschnitt B 2.3.2).[41]
- Kann-Felder (`K`) sind optionale Felder.
- Abhängige Muss-Felder (`K`) müssen nur unter bestimmten Bedingungen ausgefüllt werden. Wenn also logische Sachverhalte dem Ausfüllen von Kann-Feldern entgegenstehen, so dürfen sie nicht ausgefüllt werden. Diese Felder unterliegen Feldgruppenregeln und verfügen wie optionale Felder über den Attributwert K.

Anzahl der Elemente von Listenfeldern

Das Attribut `elemente` ist nur relevant bei von Listenfeldern (vgl. Attribut `istListe` der Tabelle `Feld`) abgeleiteten Bogenfeldern (Bogenfeldlisten). Es gibt die Größe der Bogenfeldliste an. Wenn für eine Bogenfeldliste das Attribut `elemente` leer ist, so ist die Größe per Definition 1.

Wenn ein Listenfeld als Muss-Feld deklariert ist, so ist nur das erste Exportfeld der Liste ein Muss-Feld, die restlichen Elemente sind Kann-Felder. Wenn ein Listenfeld als Kann-Feld deklariert ist, so sind alle weiteren exportierten Elemente ebenfalls Kann-Felder.

Felder – ein erster Schritt zur Prozess- und Datenintegration

Die Tabelle `Feld` (Tabelle 37) erleichtert dem Softwarehersteller den Abgleich seines Datenmodells mit dem Datenmodell des IQTIG. Gleiche Informationen in der Menge aller Dokumentationsbögen müssen dadurch nicht redundant abgebildet werden.

Beispielsweise taucht das Feld DATUMUNT (Datum der Untersuchung) in den meisten Modulen auf. Um die mehrfache Pflege dieser Felder zu vermeiden, wird ein Feld mit dem Namen DATUMUNT definiert und jeweils nur noch in der Tabelle `BogenFeld` referenziert.

Jedem Feld ist zwingend ein Basistyp zugeordnet (Abschnitt B 2.3.1). Bei Schlüsselfeldern muss auch ein Schlüssel assoziiert sein. Im Gegensatz zu den (technischen) Basistypen enthalten die Felder die medizinisch-fachliche Information der Datenfelder. Der fachliche Inhalt wird durch den Text im Attribut `bezeichnung`[42] beschrieben.

Identifizierendes Attribut eines Felds ist allein sein technischer Name (Attribut `name`). Dies ist wichtig für die Eindeutigkeit von Feldnamen innerhalb eines Moduls: Felder mit unterschiedlichen Typen oder unterschiedlichen Schlüsseln müssen unterschiedliche Namen haben. Bei der Wahl des technischen Feldnamens sind beispielsweise folgende Punkte zu berücksichtigen:

- Der technische Name eines Feldes muss eindeutig sein
- Bei der Erstellung des technischen Feldnamens ist im Idealfall eine Zeichenlänge von circa 1 bis 15 Zeichen zu nutzen. Eine Zeichenlänge von 25 sollte im Idealfall nicht überschritten werden.

[41] In jedem Muss-Feld muss für jeden angelegten Teildatensatz einmal eine Angabe erfolgen.
[42] Das Attribut `bezeichnung` ist ein Standardtext für das gleichnamige Attribut der Tabelle Bogen-`Feld`. Im Eingabeformular wird die Bezeichnung aus der Tabelle `BogenFeld` angezeigt.

- Sonderzeichen wie z.B. Unterstriche sollten nicht verwendet werden.
- Der Feldname wird in Großbuchstaben geschrieben.
- Der technische Name ist so zu wählen, dass dieser den abzufragenden Sachverhalt zwar grob umfasst, jedoch nicht so konkret, dass eine modulübergreifende Verwendung des Feldes nicht mehr möglich ist.
- Nachträgliche Anpassungen der Bezeichnung des technischen Feldes bei Änderungen der Bogenfeldbezeichnung werden i.d.R. nicht vorgenommen.

Ein Feld kann als Skalar oder als Liste definiert sein. Diese Eigenschaft wird über das Attribut `istListe` gesteuert. Jedes von einem Listenfeld abgeleitete Bogenfeld ist automatisch eine Liste.[43] Die Anzahl der Elemente des von einem Feld abgeleiteten Bogenfelds wird über das Attribut `elemente` der Tabelle `BogenFeld` gesteuert.

Insbesondere für die Verwendung der richtigen Operatoren in den Plausibilitätsregeln und Feldgruppen ist die Listendefinition eines Felds wichtig.

Grundsätzlich gilt: Die Festlegung, ob ein Bogenfeld ein Skalar oder Listenfeld ist, wird durch die Tabelle `Feld` vorgegeben. Alle von einem Listenfeld abgeleiteten Bogenfelder sind automatisch auch Listenfelder. Die Größe der Liste wird individuell in der Tabelle `BogenFeld` konfiguriert.

Die Tabelle `Feld` bietet über die „Bogensicht“ hinausgehende Informationen.

Tabelle 37: Struktur der Tabelle `Feld`

Feldname	Datentyp	Bemerkung
`idFeld`	`INTEGER`	Primärschlüssel
`Name`	KURZER TEXT	Technischer Name
`bezeichnung`	KURZER TEXT	(Erlaubte Zeichen: `A-Z`, `0-9`, Ziffer nicht am Anfang) Beschreibender Text auf dem Dokumentationsbogen (Standardwert für gleichnamiges Feld in Tabelle `BogenFeld`)
`Laenge`	`INTEGER`	Anzahl der Zeichen in der Feldeingabemaske, enthält beim Typ ZAHL auch das Komma, bei SCHLUESSEL die Trennzeichen
`einheit`	KURZER TEXT(50)	Einheit des Felds (z. B. `mm, Stunden`)
`formatAnweisung`	KURZER TEXT	Regulärer Ausdruck für die Formatprüfung (z. B. `[0-9]{9}`)
`funktion`	KURZER TEXT	Formel zur Generierung des Feldinhaltes, z. B. durch Aufruf einer Syntaxfunktion

[43] Man beachte die Besonderheiten der Listenfelder beim Datenexport und in der Syntax der Plausibilitätsregeln.

Feldname	Datentyp	Bemerkung
nachKommaLaenge	INTEGER	Anzahl der Nachkommastellen in der Feldeingabemaske (muss kleiner als laenge sein)
Min	DOUBLE	Harte Untergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
Max	DOUBLE	Harte Obergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
minWeich	DOUBLE	Weiche Untergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
maxWeich	DOUBLE	Weiche Obergrenze des Wertebereichs eines numerischen Datenfelds (modulübergreifend). Die Definition ist optional.
istListe	BOOLEAN	Wenn istListe = WAHR, so sind die vom betreffenden Feld abgeleiteten Bogenfelder Listenfelder.
fkBasisTyp	INTEGER	Fremdschlüssel zur Tabelle Basistypen
fkSchluessel	INTEGER	Fremdschlüssel zur Tabelle Schlüsseltypen
fkKombiFeld	INTEGER	Optionaler Fremdschlüssel auf ein anderes Feld, welches Kombinationsfelder kennzeichnet

Kombinationsfelder

Für manche Bogenfelder ist zwingend vorgeschrieben, dass sie innerhalb eines Moduls in Kombination mit einem anderen Bogenfeld existieren. Die Definition von Kombinationsfeldern geschieht mithilfe des optionalen Fremdschlüssels fkKombiFeld in der Tabelle Feld.

Basistypen

Das Hauptmerkmal eines Basistyps ist der technische Typ eines Eingabefelds (z. B. Zeichenkette, numerischer Typ, Datum usw.). Wichtiges Charakteristikum ist die Beschreibung des Eingabeformats. Die Basistypen sind Voraussetzung für die Beschreibung einer formalen Regelsyntax (Abschnitt B 2.4.2).

Das identifizierende Merkmal eines Basistyps ist sein technischer Name (Attribut name).

Tabelle 38: Struktur der Tabelle BasisTyp

Feldname	Datentyp	Bemerkung
idBasisTyp	INTEGER	Primärschlüssel
Name	KURZER TEXT	Technischer Name (muss eindeutig sein)

standardtyp	KURZER TEXT	Entsprechender Standarddatentyp
bezeichnung	KURZER TEXT	Beschreibender Text
Format	KURZER TEXT	Formatdefinition, z. B. TT.MM.JJJJ beim Basistyp Datum
formatRegExp	KURZER TEXT	Regulärer Ausdruck für die Formatprüfung
stdLaenge	INTEGER	Vorschlagsfeld für das gleichnamige Feld in der Tabelle Feld (einschließlich Vorzeichen und Komma)
stdNachKommaLaenge	INTEGER	Vorschlagsfeld für das gleichnamige Feld in der Tabelle Feld

Hinweis

- In Zeichenketten (Basistyp TEXT) sind alle Zeichen des ASCII-Formats mit einem Kode >= 32 erlaubt. Ausgenommen sind das Semikolon, die doppelten Anführungsstriche und Hochkommata.
- Es gibt zwei Arten von Schlüsseln: numerische und nichtnumerische.
- Das Komma trennt die Nachkommastellen, Vorzeichen + und – sind erlaubt.
- Das Datumstrennzeichen ist der Punkt.

Schlüssel

Identifizierendes Merkmal eines Schlüssels (Kodesystem) ist sein technischer Name. Die meisten Schlüsselkodes sind in der Tabelle SchluesselWert (Tabelle 40) definiert.

Tabelle 39: Struktur der Tabelle Schluessel

Feldname	Datentyp	Bemerkung
idSchluessel	INTEGER	Primärschlüssel
Name	KURZER TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	KURZER TEXT	Beschreibender Text
Extern	BOOLEAN	Zeigt an, ob der Schlüssel in der Tabelle Schluessel (= FALSCH) oder in einer externen Tabelle gespeichert (= WAHR) ist.
externVerweis	KURZER TEXT	Verweis auf die Quelle des externen Schlüssels

Feldname	Datentyp	Bemerkung
Zahl	BOOLEAN	Wenn WAHR, sind die Werte im Attribut code der zugehörigen Schlüsselwerte als ganze Zahl kodiert, ansonsten als Zeichenkette.
sortierNrVerwendet	BOOLEAN	Flag, das anzeigt, ob für die Reihenfolge das Attribut sortierNr der Tabelle SchluesselWert herangezogen wird.
fkMutterSchluessel	INTEGER	Referenz auf einen übergeordneten Schlüssel. Abgeleitete Schlüssel enthalten in der Regel keine Bezeichnungen (Datenbanktabelle SchluesselWert), da diese bereits im „Mutterschlüssel“ definiert sind.

Schlüsselkodes können auf zwei Arten interpretiert werden: Wenn das Attribut zahl gesetzt ist, so werden die Kodes als ganze Zahl gedeutet, ansonsten werden sie als Zeichenketten interpretiert. In der Syntax der Plausibilitätsregeln werden die letztgenannten Kodes in einfache Hochkommata gesetzt (Abschnitt B 2.4.2).

Beispiel:
Attribut **zahl** bei Schlüsselfeldern

- Felder des Basistyps NUMSCHLUESSEL haben das Attribut zahl = WAHR.
- Felder des Basistyps SCHLUESSEL haben das Attribut zahl = FALSCH. Es handelt sich um alphanumerische Schlüssel, die Buchstaben, Ziffern oder Sonderzeichen verwenden (z. B. ypN0). Hierbei kann es sich auch um Werte handeln, die lediglich Ziffern verwenden, aber mit führender Null beginnen (z. B. 01).

Externe Schlüsselkataloge

Externe Schlüsselkataloge sind über das Attribut extern deklariert. Hinweise zu den Bezugsquellen sind in der Spalte externVerweis zu finden. Diese externen Schlüsselkataloge werden nicht vom IQTIG bereitgestellt und somit auch nicht verantwortet.

Achtung

Der Softwareanbieter hat dafür Sorge zu tragen, dass die jeweils aktuellen externen Schlüsselkataloge in der Software verwendet werden.

Die datenentgegennehmenden Stellen müssen ebenfalls die aktuellen Schlüsselkataloge verwenden und fehlerhafte Datensätze abweisen.

Hinweise zu den Bezugsquellen sind in der Spalte externVerweis zu finden. Ein Verweis auf eine Bezugsquelle kann unabhängig vom Attribut extern angegeben werden.

Die Schlüsselkodes sind in der Tabelle SchluesselWert enthalten. Spätere Schlüsseländerungen bzw. -fortschreibungen werden vom IQTIG zeitnah übernommen.

Schlüsselwerte

Tabelle 40 gibt einen Überblick über die Datenbanktabelle SchluesselWert, in der die Kodes und Bezeichnungen der Schlüssel hinterlegt sind. Identifizierendes Merkmal ist hier eine Kombination der Spalten fkSchluessel und code. Das bedeutet, dass jeder Schlüsselkode innerhalb eines Schlüssels nur einmal vorkommen darf.

Tabelle 40: Struktur der Tabelle SchluesselWert

Feldname	Datentyp	Bemerkung
idSchluesselWert	INTEGER	Primärschlüssel
fkSchluessel	INTEGER	Fremdschlüssel zur Tabelle Schlüssel
code	KURZER TEXT(50)	Schlüsselkode (entweder numerisch oder alphanumerisch kodiert)
bezeichnung	KURZER TEXT	Textliche Definition des Schlüsselwertes
sortierNr	INTEGER	Optionale Angabe zur Reihenfolge der Schlüsselwerte: Wenn belegt, so ist diese Reihenfolge bei der Anzeige in der Erfassungssoftware einzuhalten.

Das Attribut code der Tabelle SchluesselWert ist ein Textfeld, das in Abhängigkeit vom Wert des Attributes zahl im zugeordneten Schlüssel entweder numerisch oder nichtnumerisch interpretiert wird. Wenn in einer Plausibilitätsregel (Abschnitt B 2.4.2 und B 2.4.7) Felder mit numerischen Schlüsseln (Basistyp NUMSCHLUESSEL) vorkommen, so werden bei der Evaluierung der Regel die Schlüsselkodes wie ganze Zahlen behandelt.

Sortierung der Kodes

- Für die Kodes (Attribut SchluesselWert.code) eines Schlüssels ist eine Sortierung definiert. Die Art der Sortierung wird über die Attribute zahl und sortierNrVerwendet der Tabelle Schluessel festgelegt.
- Numerische Sortierung: Wenn sortierNrVerwendet = FALSCH und zahl = WAHR, so sind die Kodes nach der Spalte code der Tabelle Schluessel numerisch zu sortieren.
- Alphanumerische Sortierung: Wenn sortierNrVerwendet = FALSCH und zahl = FALSCH, so sind die Kodes nach der Spalte code der Tabelle Schluessel alphanumerisch zu sortieren.
- Spezielle Sortierung: Wenn sortierNrVerwendet = WAHR, so sind die Kodes nach den Werten in der Spalte sortierNr der Tabelle Schluessel numerisch zu sortieren.

2.3.4 Überschriften

Die Überschriften der Dokumentationsbögen in der Spezifikation sind in der Tabelle Abschnitt zu finden.

Tabelle 41: Struktur der Tabelle `Abschnitt`

Feldname	Datentyp	Bemerkung
`idAbschnitt`	`INTEGER`	Primärschlüssel
`bezeichnung`	KURZER TEXT	Text der Überschrift
`ebene`	`INTEGER`	Zeigt die Hierarchie der Überschriften an
`fkStartBogenFeld`	`INTEGER`	Fremdschlüssel auf das erste zur Überschrift gehörende Bogenfeld
`fkEndeBogenFeld`	`INTEGER`	Fremdschlüssel auf das letzte zur Überschrift gehörende Bogenfeld

Zu jeder Überschrift ist angegeben, bei welchem Bogenfeld sie beginnt und bei welchem Bogenfeld sie endet. Über das Attribut `ebene` lassen sich auch Teilüberschriften realisieren. Ein Bogenfeld kann somit mehreren Überschriften zugeordnet sein.

Achtung

Die in der Spezifikationsdatenbank hinterlegten Überschriften sind in die Eingabemasken der PB-Dokumentationssoftware zu integrieren. Viele Datenfelder sind für den Anwender erst im Kontext der Überschriften verständlich.

2.3.5 Ausfüllhinweise

Die Ausfüllhinweise zu den Datenfeldern sind in einem separaten ZIP-Archiv enthalten, das nach dem Benennungsschema für Spezifikationskomponenten bezeichnet wird (Einleitung, Abschnitt A 1.1). Jeder Ausfüllhinweis ist ein HTML-Dokument.

Beispiel:

Ausfüllhinweis IDNRPAT.htm

```
<!DOCTYPE HTML PUBLIC „-//W3C//DTD HTML 4.0 Transitional//EN“>
<html>
  <head>
    <title>IDNRPAT</title>
    </head>
  <body>
  <!--BLOCKANFANG-->
<div class=„AH“><p>
```

Die (einrichtungsinterne) Identifikationsnummer wird dem Patienten von der Einrichtung zugewiesen. Sie verbleibt in der Einrichtung und wird nicht an die Datenannahmestelle übermittelt.

```
  </p></div>
  <!--BLOCKENDE-->
  </body>
</html>
```

In der Spalte ahinweis der Tabelle BogenFeld ist festgelegt, welcher HTML-Ausfüllhinweis mit einem Datenfeld verknüpft ist:

<aHinweis>.htm = Name der HTML-Datei

Wenn der Eintrag in ahinweis leer ist, so existiert für das betreffende Bogenfeld kein Ausfüllhinweis. Das Attribut fkAhinweisTyp lässt die Differenzierung drei verschiedener Arten von Ausfüllhinweisen zu:

Tabelle 42: Arten von Hinweistypen

fkAhinweisTyp	Beschreibung	Beispiel
Feldbezogen	Der Ausfüllhinweis bezieht sich auf den entsprechenden Datensatz in der Tabelle Feld. Der Ausfüllhinweis ist modulunabhängig.	IDNRPAT.htm Der Ausfüllhinweis bezieht sich auf verschiedene Module, beispielsweise auf DKI und ZKH.
Modulspezifisch	Soll sich ein Ausfüllhinweis nur auf ein bestimmtes Modul beziehen, kann der Attributwert modulspezifisch ausgewählt werden.	HPVVIRUS-TYP$ZKH.htm Der Ausfüllhinweis bezieht sich nur auf das Modul ZKH.
Speziell	Soll es für verschiedene Datenfelder der Tabelle Feld einen gemeinsamen Ausfüllhinweis geben, kann dieser als speziell deklariert werden. Der Attributwert ahinweis definiert den Namen des Ausfüllhinweises.	BEFUNDBIOPSKUE-RET.htm Die Felder HISTOLOG-VORBEFUNDVORUNT, BEFUNDBIOPSKUERET und HISTOBEF haben denselben Ausfüllhinweis.

Die Zuordnung von Bogenfeldern und Ausfüllhinweisen ist auch in der Abfrage Ausfüllhinweise dargestellt. Sie zeigt Modul/Teildatensatz, Zeile, Feldname, Bezeichnung und den HTML-Dateinamen des Ausfüllhinweises zu dem Bogenfeld. Im Gegensatz zur Tabelle Bogenfeld ist hier die Endung .htm mit angegeben.

2.4 Plausibilitätsprüfungen

Es wird zwischen drei Arten von Plausibilitätsprüfungen unterschieden, die in Tabelle Regel-Typ definiert sind:

- harte Prüfungen
- weiche Prüfungen in der PB-Dokumentationssoftware
- warnende Prüfungen bei der Datenentgegennahme

Tabelle 43: Tabelle RegelTyp

idRegelTyp	bezeichnung
D	Warnung Datenentgegennahme
H	hart
W	weich

Weiterhin wird zwischen sogenannten **Einzelregeln** (Abschnitt B 2.4.4) und **Feldgruppenregeln** (Abschnitt B 2.4.7) unterschieden.

2.4.1 Die Regeltabelle

Die Bedingungen für unplausible Angaben[44] sind in der Tabelle Regeln abgelegt. Die hier beschriebenen Prüfungen sind in der Spezifikationsdatenbank für PB-Dokumentation hinterlegt. Die Syntax ist in Abschnitt B 2.4.4 beschrieben. Die Bedingungen sind möglichst kurzgefasst (Vermeidung von durch ODER verknüpften Teilbedingungen). Jede Bedingung kommt nur einmal innerhalb eines Moduls vor.

Tabelle 44: Struktur der Tabelle Regeln

Feldname	Datentyp	Bemerkung
idRegeln	INTEGER	Primärschlüssel
fkModul	INTEGER	Fremdschlüssel zur Tabelle Modul
bedingung	LANGER TEXT	Entsprechend der Syntax definierte Regeln
meldung	LANGER TEXT	Fehlermeldung: Diese Texte sind bei Regeln mit Bezug zu Feldgruppen automatisch generiert.
alternativMeldung	LANGER TEXT	Alternative Fehlermeldung: Wenn hier ein Text vorhanden ist, so ist dieser anstelle des Textes in der Spalte meldung zu verwenden.
fkMehrfachRegel	INTEGER	Fremdschlüssel zur Tabelle mit mehrfach vorkommenden Regeln, die mithilfe von Ersatzbedingungen nach dem Export gültig sind.
fkFeldGruppe	INTEGER	Optionaler Fremdschlüssel zur Tabelle FeldGruppe: Indikator dafür, ob eine Regel aus einer Feldgruppe generiert wurde.
fkRegelTyp	KURZER TEXT(1)	Fremdschlüssel zur Tabelle RegelTyp: Die Regeltypen sind die in Abschnitt A 2.4 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D

[44] Eine Plausibilitätsregel müsste eigentlich „Unplausibilitätsregel“ heißen, weil sie unplausible Zustände beschreibt, die zu Fehlermeldungen führen.

Feldname	Datentyp	Bemerkung
gueltigNachExport	BOOLEAN	Regeln, die den Wert FALSCH haben, können von Datenannahmestellen nicht evaluiert werden. Stattdessen werden die referenzierten Ersatzbedingungen der Tabelle MehrfachRegel evaluiert (falls definiert).

Regelfelder (Bogenfelder einer Regel)

Die Tabelle RegelFelder (Tabelle 45) ist eine Verknüpfungstabelle zwischen den Tabellen Regeln und BogenFeld. Durch gezielte Abfragen erhält man unter Verwendung dieser Tabelle einen Überblick über Folgendes:

- Bogenfelder, die in einer Regel verwendet werden.
- Regeln, die sich auf ein Bogenfeld beziehen.

Tabelle 45: Struktur der Tabelle RegelFelder

Feldname	Datentyp	Bemerkung
fkBogenFeld	INTEGER	Fremdschlüssel zu den Tabellen Feld und Regeln, bilden zusammen den Primärschlüssel
fkRegeln	INTEGER	

Mehrfachregeln (Ersatzregeln zur Prüfung nach dem Export)

Wenn in einer Regel von der Pseudonymisierung betroffene Datenfelder benutzt werden, so kann diese von Datenannahmestellen nicht evaluiert werden. Stattdessen wird für solche Regeln in der Tabelle MehrfachRegel eine Ersatzbedingung definiert, deren Referenzierung in der Tabelle Regeln definiert ist. Die Ersatzbedingung ist von den Datenannahmestellen zu evaluieren.

Tabelle 46: Struktur der Tabelle MehrfachRegel

Feldname	Datentyp	Bemerkung
idMehrfachRegel	INTEGER	Primärschlüssel
bedingung	LANGER TEXT	Entsprechend der Syntax definierte Regeln
meldung	LANGER TEXT	Kontextunabhängige Fehlermeldung
ersatzBedingung	LANGER TEXT	Ersatzregel für den pseudonymisierten Datensatz
fkRegelTyp	KURZER TEXT(1)	Fremdschlüssel zur Tabelle RegelTyp: Die Regeltypen sind die in Abschnitt A 2.4 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D

Weitere Regeln

Weitere feldübergreifende Regeln sind die in Abschnitt B 2.3.2 beschriebenen Existenzbedingungen für das Anlegen von abhängigen Teildatensätzen (Attribut `existenzBedingung` in Tabelle `Bogen`).

2.4.2 Regelsyntax

Bedingungen sind in den Tabellen `Regeln`, `MehrfachRegel` und `Bogen` definiert. Die den Bedingungen zugrundeliegende Regelsyntax wird in diesem Abschnitt beschrieben. Jede Regel ist ein logischer Ausdruck, dessen Ergebnis WAHR oder FALSCH lautet. Jede Regel bezieht sich auf einen eingegebenen Datensatz eines Moduls, dessen Daten in Variablen gespeichert sind.

Die Regelsyntax lehnt sich an die logischen Ausdrücke in bekannten Programmiersprachen an. Jedoch haben die Operatoren deutsche Namen, z. B. UND statt AND oder ODER statt OR. Die Regelsyntax ist als Pseudocode zu verstehen.

Typen

Die möglichen Typen der Datenfelder sind in Tabelle 47 aufgelistet.

Tabelle 47: Basistypen der Datenfelder in den Plausibilitätsregeln

Basistyp	Bezeichnung	Beispiele (Literale)
BOOL	Boolesche Variable	WAHR, FALSCH
TEXT	Zeichenkette (String)	"Spezifikation"
GANZEZAHL[45]	… -2, -1, 0, 1, 2, 3, …	1
ZAHL	Zahl (mit oder ohne Nachkommastellen)	25,4 oder -100,8
DATUM	Zehnstelliges Datum	'01.01.2012'
MONDATUM	Monatsdatum	'04.2012'
QUARTDATUM	Quartalsdatum	'3/2012'
JAHRDATUM	Jahresdatum	2012
NUMSCHLUESSEL	Numerisch kodierter Schlüssel (wie GANZEZAHL)	1
SCHLUESSEL	Alphanumerischer Schlüssel	'19.1', '07'
UHRZEIT	Uhrzeit	'10:15'

[45] Beim Typ GANZEZAHL sind auch negative ganze Zahlen erlaubt.

In der Spezifikation für die PB-Dokumentation wird zwischen NUMSCHLUESSEL und SCHLUESSEL unterschieden:

- Schlüsselwerte verfügen über den Datentyp NUMSCHLUESSEL, wenn es sich bei den Kodes um ganze Zahlen handelt. Da dies ein numerischer Schlüssel ist, darf er nicht in Hochkommata gesetzt werden.
- Schlüsselwerte, die alphanumerische Kodes beinhalten, haben den Basistyp SCHLUESSEL.

Achtung

Datumsangaben (Datum, Monats-, Quartalsdatum) müssen in Hochkommata gesetzt werden. Eine Ausnahme ist das Jahresdatum (JAHRDATUM), da es sich hierbei um eine ganze Zahl handelt.

Felder

Feldnamen bestehen aus maximal 32 Zeichen und dürfen nur die Buchstaben A bis Z (Großbuchstaben) und die Ziffern 0 bis 9 enthalten. Ein Feldname muss immer mit einem Buchstaben beginnen. Umlaute und Sonderzeichen sind in Feldnamen nicht erlaubt. Ein Feldname darf kein reserviertes Wort sein (z. B. LEER).

Achtung

In einer Regel dürfen nur die Feldnamen der im betreffenden Modul definierten Bogenfelder[46] enthalten sein. Bei der Evaluierung von Regeln werden die aktuellen Werte der referenzierten Bogenfelder eingesetzt. Kann-Bogenfelder können auch unausgefüllt sein, also den Wert LEER haben.

Listenfelder

Ein Bogenfeld wird dann als Liste interpretiert, wenn im referenzierten Feld (Tabelle Feld) der Wert des Attributs Feld.istListe = WAHR ist. Andernfalls ist das Bogenfeld ein Skalar. Bei der Formulierung von Regeln ist darauf zu achten, dass Listenfelder nicht bei jedem Operator als Operand fungieren können. Listenfelder dürfen z. B. nicht voneinander subtrahiert werden.

Literale

Alphanumerische Literale (z. B. SCHLUESSEL) werden von einfachen Hochkommata eingeschlossen, während Zeichenketten vom Datentyp TEXT in Anführungszeichen gesetzt werden müssen.[47]

Dies gilt nicht für numerische Literale (GANZEZAHL, ZAHL, NUMSCHLUESSEL und JAHRDATUM) und Literale des Datentyps BOOL (Wahrheitswerte).

[46] Bei den Ersatzregeln in Tabelle MehrfachRegel sind stattdessen die Exportfelder des Moduls erlaubt.
[47] Beim Export entfallen die begrenzenden Zeichen.

Beispiel für Regeln mit Literalen:

```
POKOMPLIKAT <> 1 UND PNEUMONIE <> LEER

AUFNVONSTATPFLEGE = 1 UND ENTLGRUND NICHTIN ('07'; '10')

aktuellesJahr() - jahreswert(GEBDATUM) > 100
```

Listen von Literalen

Literale können sowohl als Skalare als auch als Listen angesprochen werden. Der Separator einer Liste von Literalen ist das Semikolon. Um zu prüfen, ob alle Listenfelder ausgefüllt sind, wird die Liste über den Wert LEER angesprochen.

Außerdem gibt es Teildatensatz-Listenfelder, die im Abschnitt B 2.4.6 beschrieben werden.

Operatoren

Tabelle 48 gibt einen Überblick über die in der Syntax zulässigen Operatoren. Der aktuelle Überblick über alle zulässigen Operationen (inkl. Operanden) ist in Tabelle SyntaxOperator in der PBDOK-Datenbank zu finden.

In Tabelle 48 hat jeder einzelne Operator eine Präzedenzstufe (höchste Präzedenzstufe ist 0). Operatoren, die die gleiche Stufe haben, werden nach den Regeln der Assoziativität aufgelöst.

Tabelle 48: Präzedenz und Assoziativität der Operatoren

Präzedenz	Assoziativität	Operator	Erläuterung
0	links	IN	Operator zum Vergleich einer Variablen mit einer Liste (z. B. ein Datenfeld mit Schlüsselwerten). Die Variable und die Feldelemente müssen gleichen Typs sein.
	links	NICHTIN	
	links	EINSIN	Operator zum Vergleich einer Liste mit einer anderen Liste oder einem Listenelement (z. B. ein Listenfeld mit einem Schlüsselwert). Die Listenelemente müssen gleichen Typs sein.
	links	JEDESIN	
	links	EINSNICHTIN	
	links	KEINSIN	
1	links	*	Operator für die Multiplikation „mal”
	links	/	Operator für die Division „geteilt”
2	links	+	Operator für die Addition „plus”
	links	-	Operator für die Subtraktion „minus“
3	links	<	Vergleichsoperator „kleiner“
	links	>	Vergleichsoperator „größer“
	links	<=	Vergleichsoperator „kleiner gleich“

Präzedenz	Assoziativität	Operator	Erläuterung
	links	>=	Vergleichsoperator „größer gleich“
4	links	=	Vergleichsoperator „gleich“
	links	<>	Vergleichsoperator „ungleich“
5	rechts	NICHT	Logischer Operator: „NICHT“
6	links	UND	Logischer Operator: „UND“
7	links	ODER	Logischer Operator: „ODER“

Prüfung auf LEER mit Vergleichsoperatoren

Die Prüfung auf LEER von in Regeln verwendeten Kann-Feldern, welche an anderer Stelle in der Regel mit einem anderen Operator als <> oder = geprüft werden sollen, findet auf der linken Seite einer ODER-Verknüpfung statt. Hintergrund dieser Syntaxregel ist, dass die Vermeidung von Laufzeitfehlern bei der Evaluation ermöglicht werden soll.

Beispiel:

```
FELD = LEER ODER FELD OPERATOR OPERAND
```

Beispielsweise kann bei leeren Feldwerten und der vorgegebenen Linksassoziativität des ODER-Operators die Evaluation bei leerem Feldwert vor der Evaluation des rechtsstehenden Ausdrucks mit der Rückgabe von WAHR abgebrochen werden. Ein Laufzeitfehler, der sich z. B. bei einem Vergleich von LEER < WERT ergeben würde, kann so nicht entstehen.

Operatoren zum Vergleich einer Variablen mit einer Liste

Folgende Operatoren erfordern entweder nur rechts oder links und rechts Listenfelder:

- nur rechts: IN, NICHTIN
- links und rechts: EINSIN, KEINSIN, JEDESIN, EINSNICHTIN

Operatoren mit beidseitigen Listenfeldern als Operanden:

- EINSIN: Wenn mindestens ein Element aus der linken Liste in der rechten Liste enthalten ist, so ist der Ausdruck wahr (nichtleere Schnittmenge).
- KEINSIN: Wenn kein Element der linken Liste in der rechten Liste enthalten ist, so ist der Ausdruck wahr (leere Schnittmenge). Dieser Operator ist redundant, da er auch durch Negation des EINSIN-Operators abgedeckt ist.
- JEDESIN: Der Ausdruck ist dann wahr, wenn jedes Element der linken Liste in der rechten Liste enthalten ist (Teilmenge).
- EINSNICHTIN: Der Ausdruck ist dann wahr, wenn mindestens ein Element der linken Liste nicht in der rechten Liste enthalten ist (nichtleere Differenz).

Beispiel:

- Die Operation GANZEZAHL:= DATUM1 – DATUM2 liefert als Ergebnis die Differenz zwischen zwei Kalenderdaten in Tagen.
- Die Operation ZAHL:= UHRZEIT1 – UHRZEIT2 liefert als Ergebnis die Differenz zwischen zwei Uhrzeiten in Minuten.

Folgende Operatoren sind komplementär:

IN und NICHTIN

EINSIN und KEINSIN

JEDESIN und EINSNICHTIN

Folgende Ausdrücke sind gleich:

```
A EINSNICHTIN B
NICHT A JEDESIN B
```

Hinweis
Wird das Zusatzkennzeichen direkt in der Plausibilitätsregel abgefragt, wird dieses bei der Evaluation der entsprechenden Regel nicht ignoriert. Hierbei kann es sich beispielsweise um die Prüfung der Diagnosesicherheit mithilfe der Funktion format handeln.

Beispiel

```
STATUSLE = 1 UND ARTLE IN (2;3) UND format(ENTLDIAG;'[a-zA-Z][0-9]{2}(\.[0-9]{1,2})?([#†\*\+!])?([RLBrlb])?$') <> WAHR
```

2.4.3 Funktionen

Eine Funktion ist gekennzeichnet durch ihren Namen, an den sich unmittelbar (ohne Leerzeichen) ein Listenausdruck anschließt. Funktionen ohne Übergabeparameter werden ähnlich wie in C oder Java durch ein Klammerpaar abgeschlossen. Funktionen können nicht nur in Regeln, sondern auch zur Berechnung von Exportfeldern genutzt werden (Abschnitt B 2.5.2).

Beispiel:
Das Ersatzfeld quartal wird mithilfe der Syntaxfunktion quartal berechnet:

```
quartal = quartal(DATUMUNT)
```

Der aktuelle Stand der in der Syntax verwendeten Funktionen ist in der Tabelle SyntaxFunktion der Spezifikation zu finden.

In den nachfolgenden Beispielen gilt folgende Notation für Funktionen:

```
<BASISTYP> <FUNKTIONSNAME>([<BASISTYP> <VARNAME>{;
<BASISTYP> <VARNAME>}])
```

mit

- { } Wiederholung
- [ ] Option
- <BASISTYP> Basistyp der Variablen
- <VARNAME> Name der Variablen

Beispiele:

DATUM aktuellesDatum()
Funktion ohne Übergabeparameter und mit Ergebnistyp DATUM

DATUM Minimum(DATUM DATUMLISTE)

Funktion mit Ergebnis vom Typ DATUM, die das Minimum einer Liste von Datumsangaben (DATUMLISTE) liefert.

JAHRDATUM jahreswert(DATUM EINDATUM)
Funktion mit Ergebnis vom Typ JAHRDATUM

Es kommen auch verschachtelte Funktionsaufrufe (z. B. funktionA(funktionB())) oder arithmetische Ausdrücke als Funktionsargumente (z. B. funktion(x+y)) vor (Beispiel: gewichtsperzentile). Häufig wird nur die Signatur von Funktionen bereitgestellt.

Hinweise für die Implementierung von Funktionen
Als Hilfestellung für die Ausprogrammierung wird bei manchen Funktionen ein Pseudokode bereitgestellt. Der Pseudokode ergänzt die Syntax der Plausibilitätsregeln um folgende Sprachelemente:[48]

- Befehlszeilen werden mit Semikolon abgeschlossen ;
- Wertzuweisungen mit dem Operator :=

```
A := B + C;
```

- Auswahlanweisungen

```
if (<Bedingung>){
     ...
}
else {
     ...
}
```

Hinter <Bedingung> verbirgt sich ein logischer Ausdruck, der der Syntax der Plausibilitätsregeln gehorcht.

- Blöcke werden durch geschweifte Klammern definiert.

[48] Der Pseudocode erhebt nicht den Anspruch auf formale Korrektheit.

```
{
...
}
```

- Innerhalb einer Funktion sind die Argumentvariablen verfügbar.
- Eine Variable, die den gleichen Namen wie die Funktion hat, muss am Ende mit `return` zurückgegeben werden.

Hinweise zur Funktion `format (Feld, pattern)`

Die Funktion prüft, ob der erste Parameter (`Feld`) dem regulären Ausdruck (`pattern`) entspricht. Gibt es eine Übereinstimmung, gibt die Funktion ein WAHR zurück.

Beispiel:

`format(STANDORT, [0-9]{2})`

`[0-9]`: der Wert darf nur die Zahlen 0–9 enthalten

`{2}`: der Wert muss genau 2-stellig sein

Die konkrete Implementierung dieser Funktion ist von der eingesetzten Programmiersprache abhängig.

JAVA	C#	C++
`feld.matches (pattern)`	`Regex.IsMatch (feld,pattern)`	`Regex::IsMatch (feld,pattern)`

2.4.4 Syntaxvariablen

Der Eingangsdatensatz bildet die medizinische Routinedokumentation ab, die in jedem Arztinformationssystem (AIS) und Laborinformationssystem (LIS) enthalten ist. Der Eingangsdatensatz wird in der PBDOK nicht explizit aufgeführt, da er der PB-Filterdatenbank entnommen werden kann.

Syntaxvariablen dienen in der PB-Dokumentation der technischen Darstellung der automatischen Generierung von Angaben aus dem Eingangsdatensatz.

Diese sind in Tabelle `SyntaxVariable` hinterlegt.

Jeder der in der Tabelle `SyntaxVariable` definierten Variablen ist über den Wert des Attributes `SyntaxVariable.fkTdsFeld` ein Feld des PB-Filter-Eingangsdatensatzes zugeordnet. Jedes dieser Felder besitzt somit einen Basistyp. Die in den Bedingungen erlaubten Variablen sind in der Tabelle `SyntaxVariable` definiert. Die Variablennamen (Attribut `SyntaxVariable.name`) bestehen aus maximal 32 Zeichen. Sie dürfen nur die Buchstaben A bis Z (Großbuchstaben) und die Ziffern 0 bis 9 enthalten. Ein Feldname muss immer mit einem Buchstaben beginnen. Umlaute und Sonderzeichen sind in Feldnamen nicht erlaubt. Ein Feldname darf auch kein reserviertes Wort sein (z. B. Namen von Operatoren wie `EINSIN`).

2.4.5 Einzelregeln

Sogenannte Einzelregeln können sich als feldbezogene Prüfungen auf ein einziges Datenfeld oder als feldübergreifende Prüfungen auf mehrere Datenfelder beziehen. Einzelregeln sind von den in Abschnitt B 2.4.7 beschriebenen Feldgruppen zu unterscheiden.

Feldbezogene Prüfungen – beispielsweise Wertebereichsüberprüfungen – sind in der formalen Regelsyntax in Tabelle **Regeln** formuliert.

Unter feldbezogenen Prüfungen sind aber auch die in Abschnitt B 2.4.8 beschriebenen Prüfungen des Formates, der Feldlänge, der Wertebereiche, Prüfungen von Schlüsselkodes und von Muss-Feldern zu verstehen. Für diese Prüfungen gibt es keine formale Regelsyntax in Tabelle Regeln.

Feldübergreifende Regeln

- haben eine eigene Syntax,
- haben geringe Komplexität,
- haben einfache, dem Anwender verständliche Fehlertexte,
- enthalten alle Teilregeln der Feldgruppen,
- haben gewöhnlich den Bezug zu zwei oder mehreren Feldern,
- können zum Teil direkt nach der Benutzereingabe in ein Feld geprüft werden,
- enthalten Bedingungen für unplausible Angaben[49].

Feldübergreifende Regeln können auch teildatensatzübergreifende Regeln sein, wenn die Datenfelder der Regel aus mehreren Teildatensätzen eines Moduls stammen (Abschnitt B 2.4.6).

2.4.6 Teildatensatzübergreifende Regeln

Eine Regel ist teildatensatzübergreifend, wenn die Datenfelder der Regel aus mehreren Teildatensätzen eines Moduls stammen.

Es gibt zwei Arten von teildatensatzübergreifenden Regeln:
- Die Felder sind in verschiedenen Teildatensätzen eines Moduls definiert.
- Ein Feld der Regel ist in einem wiederholbaren Teildatensatz definiert und die Regel bezieht sich auf alle Werte des Datenfeldes innerhalb eines Datensatzes (= Summe aller Teildatensätze eines Vorgangs).

2.4.7 Feldgruppenregeln

Logische Abhängigkeiten von Bogenfeldern werden über Feldgruppen dargestellt. Die Plausibilitätsregeln, die einen Bezug zu einer Feldgruppe aufweisen (Tabelle Regeln), werden anhand der Feldgruppendefinition (Tabelle FeldgruppeFelder) automatisch generiert. Die Menge der abgeleiteten Einzelregeln wird in diesem Abschnitt erläutert.

[49] Eine Plausibilitätsregel müsste eigentlich „Unplausibilitätsregel“ heißen, weil sie unplausible Zustände beschreibt, die zu Fehlermeldungen führen.

Die möglichen Antworten[50] eines jeden Datenfeldes werden in zwei Gruppen aufgeteilt. Die erste Gruppe ist die Menge der positiven, die zweite Gruppe die Menge der negativen Antworten.[51]

Typische positive `Antworten` sind beispielsweise:

`Feld <> LEER` oder `Feld IN (2;3)`

Die komplementären negativen Antworten würden entsprechend wie folgt lauten:

`Feld = LEER` oder `Feld NICHTIN (2;3)`

Eine Feldgruppe kann ein Filterfeld haben. Wenn die Antwort dieses Filterfeldes negativ ausfällt (Bspw. Bedingung: `Feld = 3`; Antwort: `Feld <> 3`), so darf keines der abhängigen Felder positiv beantwortet werden. Die folgende Tabelle gibt einen Überblick über die Typen von Feldgruppen. Der aktuelle Stand findet sich in der Tabelle `FeldGruppenTyp` der Spezifikation.

Tabelle 49: Typen von `Feldgruppen`

Name	Bemerkung
mit Filterfeld	
EF_FILTER	Einfachauswahl, genau ein abhängiges Feld muss positiv beantwortet sein
EF_OPTIONAL_FILTER	Einfachauswahl, genau ein abhängiges Feld kann positiv beantwortet sein
MF_OPTIONAL_FILTER	Mehrfachauswahl, alle abhängigen Felder können positiv beantwortet sein
MF_MINDESTENS1_FILTER	Mehrfachauswahl, mindestens ein abhängiges Feld muss positiv beantwortet sein
MF_ALLES_FILTER	Mehrfachauswahl, alle abhängigen Felder müssen positiv beantwortet sein
ohne Filterfeld	
EF	Einfachauswahl, genau ein Feld muss positiv beantwortet sein
MF_OPTIONAL	Mehrfachauswahl, alle Felder können positiv beantwortet sein
MF_MINDESTENS1	Mehrfachauswahl, mindestens ein Feld muss positiv beantwortet sein
UND	Einfache Regel mit Und-Verknüpfungen

[50] Die Antworten eines Datenfeldes umfassen hier neben möglichen Werten (z.B. Schlüsselwerten) oder Wertemengen auch die Kategorie „nicht ausgefüllt“ (LEER).
[51] Die negativen Antworten sind abhängig von der definierten Bedingung eines Feldes in der entsprechenden Feldgruppe.

In der Tabelle `BogenFeld` sind abhängige Datenfelder einer Feldgruppe immer als Kann-Felder definiert. Nach Abhängigkeit der Feldgruppenlogik können/müssen diese Felder leer bleiben oder zwingend ausgefüllt werden. Im letztgenannten Fall können die Datenfelder auch als bedingte Muss-Felder bezeichnet werden.

Die Muss- oder Kann-Definition der Datenfelder (Bogen- und Ersatzfelder) im Exportformat unterliegt ebenfalls der Feldgruppenlogik. Ist die Berechnung eines Ersatzfeldes von bedingten Datenfeldern abhängig, so gilt die Feldgruppenlogik auch für diese Ersatzfelder. Wenn die bedingten Datenfelder zwingend ausgefüllt werden müssen, so muss auch das Ersatzfeld zwingend berechnet bzw. exportiert werden.[52]

Struktur der Tabellen `FeldGruppe` und `FeldgruppeFelder`

Die Feldgruppen sind in den Tabellen `Feldgruppe` und `FeldgruppeFelder` definiert. In der Tabelle `FeldGruppe` (Tabelle 50) sind Name, Typ und die Zuordnung zu einem Modul definiert. Die Verknüpfungstabelle `FeldgruppeFelder` (Tabelle 51) definiert die abhängigen Bogenfelder. Zusätzlich wird hier festgelegt, welche Bogenfelder der Feldgruppe als Filterfeld dienen.

Tabelle 50: Struktur der Tabelle `FeldGruppe`

Feldname	Datentyp	Bemerkung
`idFeldGruppe`	`INTEGER`	Primärschlüssel
`name`	`KURZER TEXT(64)`	Technischer Name der Feldgruppe
`fkModul`	`INTEGER`	Obligatorischer Fremdschlüssel zu einem Modul
`fkFeldgruppenTyp`	`INTEGER`	Obligatorischer Fremdschlüssel zu einem Feldgruppentyp
`hinweis`	`KURZER TEXT`	Bei Filter-Feldgruppen relevant für die Gestaltung der Eingabemaske. Der Hinweistext informiert den Anwender über die Bedingungen, welche das Ausfüllen von ein oder mehreren abhängigen Feldern erforderlich machen. Der Hinweistext kann bei der Erstellung der Eingabemasken verwendet werden. Beispiel: Der Hinweistext „Bei postoperativen Komplikationen“ wird oberhalb eines Blocks von zusammengehörigen Feldgruppenfeldern angezeigt.
`fkFilterFeldTyp`	`KURZER TEXT (1)`	Attribut wird bei Feldgruppen mit mehreren Filterfeldern gesetzt:

[52] Die Funktion `verkettenmt` verkettet (zwei oder mehrere) Zeichenfolgen zu einer Zeichenfolge. Hierbei müssen nicht alle im Attribut `Ersatzfeld.formel` aufgeführten Datenfelder ausgefüllt sein. Ist lediglich eine übergebene Zeichenfolge nicht leer, wird diese ohne Verkettung zurückgeliefert.

Feldname	Datentyp	Bemerkung
		O = Oder-Verknüpfung der positiven Filterbedingungen U = Und-Verknüpfung der positiven Filterbedingungen
fkRegelTyp	KURZER TEXT (1)	Fremdschlüssel zur Tabelle RegelTyp: Die Regeltypen sind die in Abschnitt A 2.4 beschriebenen Arten der Plausibilitätsprüfungen: H, W oder D Die generierten Einzelregeln der Feldgruppe haben den gleichen Regeltyp.
nurPositiv	BOOLEAN	Wenn WAHR, dann umfasst die Feldgruppe nur diejenigen Regeln, welche sich auf die positive (Filter)bedingung beziehen.
grauWennNegativ	BOOLEAN	Definiert eine Layout-Feldgruppe, wenn WAHR (siehe unten S. 131, Layout-Feldgruppen)

Tabelle 51: Struktur der Tabelle FeldgruppeFelder

Feldname	Datentyp	Bemerkung
idFeldgruppeFelder	INTEGER	Primärschlüssel
fkFeldGruppe	INTEGER	Obligatorischer Fremdschlüssel zur Feldgruppe
fkBogenFeld	INTEGER	Obligatorischer Fremdschlüssel zum Bogenfeld
bedingung	KURZER TEXT	Positive Bedingung für das jeweilige Bogenfeld
istFilter	BOOL	Legt fest, ob das jeweilige Bogenfeld ein Filterfeld ist
bezeichnungSchluesselListe	KURZER TEXT	Abkürzende Bezeichnung für eine Schlüsselliste in der Bedingung, wird beim Generieren von Fehlermeldungen verwendet.
tdsFilter	BOOLEAN	Das Bogenfeld wird in Regeln als TDS-Listenfeld (Abschnitt 2.4.6) verwendet (Voranstellen des @-Zeichens vor Feldnamen).

Syntax der Feldgruppenregeln

In den Tabellen `FeldGruppe` bzw. `FeldgruppeFelder` sind die positiven Bedingungen für das Filterbogenfeld bzw. die abhängigen Bogenfelder einer Feldgruppe definiert. Jede Bedingung hat folgenden Aufbau:

`<Operator> <Operand>`

Der linke Operand wird hier weggelassen, weil er immer der Name des jeweiligen Bogenfeldes ist. Die komplette Bedingung für das Bogenfeld einer Feldgruppe lautet also:

`<Bogenfeld> <Operator> <Operand>`

Als Operator kann jeder dyadische Operator der Tabelle 48 verwendet werden. Die auf der rechten Seite erlaubten Operanden sind nachfolgend aufgelistet:

- Literale (Tabelle 47)
- LEER
  Kodelisten, in denen auch die Kodes eines Schlüssels referenziert werden können; Beispiel: `(1;2;3)` oder `(MaDCIS)`

Hinweis

Der rechte Operand darf kein Bogenfeld sein, da sich eine Feldbedingung immer genau auf ein Bogenfeld bezieht.

Formale Definition von Feldgruppen

A sei ein Bogenfeld in einer Feldgruppe. Dann seien `p(A)` die positiven und `n(A)` die negativen Bedingungen, welche jeweils das Ergebnis wahr oder falsch haben können.

Eine Feldgruppe kann ggf. ein Filterfeld haben, das mit `F` bezeichnet wird. Eine Feldgruppe lässt sich dann in folgender Tabelle darstellen:

Tabelle 52: Formale Definition einer Feldgruppe

Feld	Positive Bedingung	Negative Bedingung	Bemerkung
F	p(F)	n(F)	falls Feldgruppentyp mit Filter
A1	p(A1)	n(A1)	
A2	p(A2)	n(A2)	
A3	p(A3)	n(A3)	
…			
An	p(An)	n(An)	

Eine Feldgruppe besteht insgesamt aus `n` abhängigen Bogenfeldern:

`A1, A2, ... An`

In Abhängigkeit von den Feldgruppentypen werden unterschiedliche Einzelregeln generiert.

Feldgruppen mit Filter

- Regeln der Feldgruppe „Optionale Mehrfachauswahl mit Filterfeld“ (MF_OPTIONAL_FILTER)

  n(F) UND p(Ai) i = 1,.., n

Insgesamt sind n Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Obligatorische Mehrfachauswahl mit Filterfeld“ (MF_MINDESTENS1_FILTER)

  n(F) UND p(Ai) i = 1,.., n

  p(F) UND n(A1) UND n(A2)UND ... UND n(An)

Insgesamt sind n+1 Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Mehrfachauswahl mit Filterfeld, alle abhängigen Felder müssen positiv beantwortet sein“ (MF_ALLES_FILTER)

  n(F) UND p(Ai) i = 1,.., n

  p(F) UND n(Ai) i = 1,.., n

Insgesamt sind 2n Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Einfachauswahl mit Filter“ (EF_FILTER)

  n(F) UND p(Ai) i = 1,.., n

  p(F) UND n(A1) UND n(A2)UND ... UND n(An)

  p(F) UND p(Aj) UND p(Ai) für alle unterschiedlichen i,j = 1,.., n

Insgesamt sind n(n+1)/2+1 Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Optionale Einfachauswahl mit Filter“ (EF_OPTIONAL_FILTER)

  n(F) UND p(Ai) i = 1,.., n

  p(F) UND p(Aj) UND p(Ai) für alle unterschiedlichen i,j = 1,..,n

Insgesamt sind n(n+1)/2 Einzelregeln mit der Feldgruppe verknüpft.

Feldgruppen mit Filter: Attribut nurPositiv

Wenn in einer Feldgruppe mit Filter das Attribut nurPositiv gesetzt ist, so sind nur die Einzelregeln mit positiver Filterbedingung Bestandteil der Feldgruppe.

Beispiel:

Die Feldgruppe EF_FILTER mit nurPositiv= ja hat folgende Einzelregeln:

p(F) UND n(A1) UND n(A2)UND ... UND n(An)

p(F) UND p(Aj) UND p(Ai) für alle unterschiedlichen i,j = 1,.., n

---

Feldgruppen ohne Filter

- Regeln der Feldgruppe „Einfachauswahl“ (EF)

  n(A1) UND n(A2) UND ... UND n(An)

  p(Aj) UND p(Ai) für alle unterschiedlichen i,j = 1,.., n

Insgesamt sind n(n-1)/2+1 Einzelregeln mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Obligatorische Mehrfachauswahl“ (MF_MINDESTENS1)

  n(A1) UND n(A2) UND ... UND n(An)

Insgesamt ist eine Einzelregel mit der Feldgruppe verknüpft.

- Regeln der Feldgruppe „Und-Regel“ (UND)

  p(A1) UND p(A2) UND ... UND p(An)

Insgesamt ist eine Einzelregel mit der Feldgruppe verknüpft.

Feldgruppen mit mehreren Filterfeldern

Es besteht die Möglichkeit, Feldgruppen mit mehr als einem Filterfeld zu definieren:
Formal gibt es dann die Filterfelder F1, F2, ..., Fn mit den positiven bzw. negativen Bedingungen p(Fj) bzw. n(Fj). Für alle Filterfelder wird eine positive Bedingung p(F1, ..., Fn) und eine negative Bedingung n(F1, ..., Fn) gebildet. Diese modifizierten Filterbedingungen ersetzen die im Abschnitt B 2.4.7 definierten Filterbedingungen p(F) und n(F) bei den Einzelregeln.

Die Filterfelder können entweder über eine ODER-Verknüpfung oder eine UND-Verknüpfung miteinander verbunden sein:

p(F1, ..., Fn)= p(F1) ODER p(F2) ODER ... ODER p(Fn)

(ODER-Verknüpfung)

p(F1, ..., Fn)= p(F1) UND p(F2) UND ... UND p(Fn)

(UND-Verknüpfung)

Der Verknüpfungstyp ist im Attribut fkFilterFeldTyp der Tabelle FeldGruppe hinterlegt.

Layout-Feldgruppen

Feldgruppen, bei denen das Attribut grauWennNegativ in der Datenbanktabelle FeldGruppe WAHR ist, werden nachfolgend als Layout-Feldgruppen bezeichnet. Der Attributname grauWennNegativ wurde gewählt, weil die abhängigen Felder der Layout-Feldgruppen auf den generierten Dokumentationsbögen eingegraut sind. Abbildung 17 zeigt die Layout-Feldgruppe DKI: IFOBTEST mit einem Filterfeld und drei abhängigen Feldern am Beispiel der Spezifikation 2021.

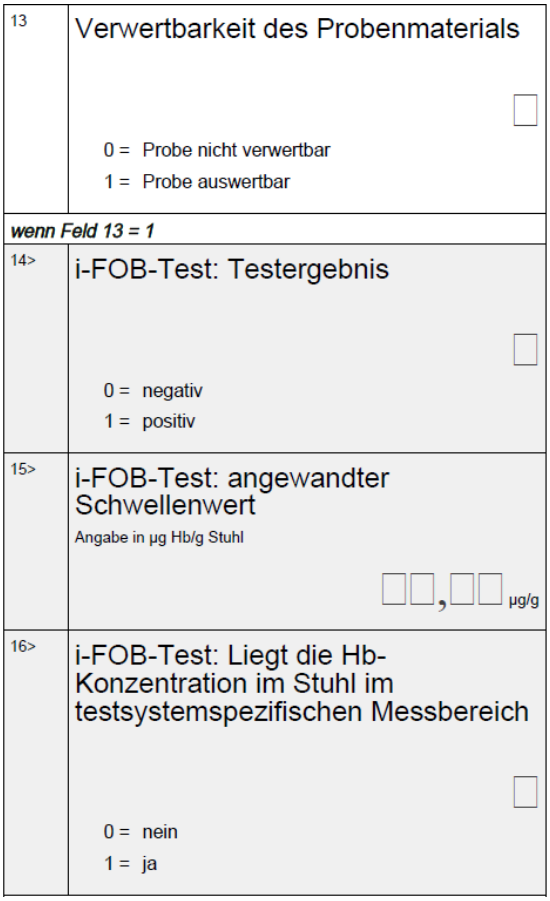


Abbildung 17: Feldgruppe DKI: IFOBTEST auf dem Dokumentationsbogen (Spezifikation 2021)

Layout-Feldgruppen haben folgende Eigenschaften:

- Sie haben mindestens ein Filterfeld.
- Jedes abhängige Feld hat die Bedingung <> LEER oder EINSNICHTIN (LEER) (Attribut bedingung in Tabelle FeldGruppeFelder).
- Das Attribut nurPositiv hat den Wert FALSCH.

Bei Vorliegen dieser drei Eigenschaften müssen die abhängigen Felder leer bleiben, wenn die negative Filterbedingung bei der Dokumentation eines Falles erfüllt ist.

Beispiel:
Wenn in Datenfeld 13 = 0 (nein) angegeben ist, so müssen die Datenfelder 14, 15 und 16 leer bleiben. Die folgenden Plausibilitätsprüfungen stellen dies sicher.

Die zugehörige Feldgruppe (Abfrage FeldgruppeFürEinModul = Zusammenschau der Tabellen Feldgruppe und FeldgruppeFelder) ist wie folgt definiert:

Tabelle 53: Definition der Feldgruppe DKI:IFOBTEST in Spezifikation 2021

Bogenfeld	Bedingung	istFilter	Feldgruppentyp	grauWennNegativ
DKI.B.13.AUSWERTBARPROBE	=1	TRUE	MF_ALLES_FILTER	TRUE
DKI.B.14.IFOBTESTERGEBNIS	<> LEER	FALSE	MF_ALLES_FILTER	TRUE
DKI.B.15.IFOBSCHWELLENWERT	<> LEER	FALSE	MF_ALLES_FILTER	TRUE
DKI.B. 16.MESSBER	<> LEER	FALSE	MF_ALLES_FILTER	TRUE

2.4.8 Prüfung von Feldeigenschaften

Die in diesem Abschnitt behandelten feldbezogenen Prüfungen ergeben sich direkt aus den Tabellen `Feld` (bzw. `ErsatzFeld` oder `ZusatzFeld`) und `BogenFeld` (bzw. `ExportFormat`) und werden vor Evaluation der in Abschnitt B 2.4.4 beschriebenen feldübergreifenden Regeln durchgeführt.

Die hier beschriebenen Prüfungen sind nur in Form von Feldeigenschaften – nicht aber in Regelsyntax – in der Datenbank für PB-Dokumentation hinterlegt.

Überprüfung des Formats

Die Formatprüfung bezieht sich auf den Exportdatensatz (Abschnitt B 2.5.2): Die PB-Dokumentations- bzw. Exportsoftware muss Daten im korrekten Format generieren, was durch die datenentgegennehmenden Stellen geprüft wird. Für Exportfelder (Tabelle `ExportFormat`), die einen Bezug zu einem Ersatzfeld (Tabelle `ErsatzFeld`) bzw. zu einem Zusatzfeld (Tabelle `ZusatzFeld`) haben, sind die standardisierten Fehlertexte anzupassen.

Die Prüfung bezieht sich insbesondere darauf, ob der Feldinhalt dem in der Spezifikation definierten Basistyp entspricht. Beispielsweise sind Buchstaben beim Basistyp GANZEZAHL nicht erlaubt.

Standardisierter Fehlertext für Formatfehler eines Ersatzfeldes

Der Wert `'<WERT>'` des Datenfeldes `<Modul.name>:<Bogen.name>:<Ersatzfeld.name> '<ErsatzFeld.bezeichnung>'` ist kein gültiger `<BasisTyp.name>` Wert (`<BasisTyp.bezeichnung> <BasisTyp.format>`).

Standardisierter Fehlertext für Formatfehler eines Bogenfeldes

Der Wert `'<WERT>'` des Datenfeldes `<Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>'` (Zeile `<BogenFeld.gliederungAufBogen>`) ist kein gültiger `<BasisTyp.name>` Wert (`<BasisTyp.bezeichnung> <BasisTyp.format>`).

Standardisierter Fehlertext für Muss-Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<ZusatzFeld.bezeichnung> ist kein gültiger <BasisTyp.name> Wert (<BasisTyp.bezeichnung> <BasisTyp.format>).

Beispiel (Bogenfeld):
Der Wert '3A.06.2020' des Datenfeldes DKK:B:DATUMUNT Datum der Untersuchung' (Zeile 11) ist kein gültiger DATUM-Wert (Zehnstelliges Datum TT.MM.JJJJ).

Überprüfung der Feldlänge

Die Feldlängenprüfung bezieht sich darauf, ob die Anzahl der Zeichen eines Wertes die spezifizierte Länge[53] (Attribut laenge in Tabelle Feld) des Feldes überschreitet.

Standardisierter Fehlertext für Längenfehler eines Ersatzfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<ErsatzFeld.name> '<ErsatzFeld.bezeichnung>' überschreitet die zulässige Feldlänge <Feld.laenge>.

Standardisierter Fehlertext für Längenfehler eines Bogenfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) überschreitet die zulässige Feldlänge <Feld.laenge>.

Standardisierter Fehlertext für Muss-Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<ZusatzFeld.bezeichnung> überschreitet die zulässige Feldlänge <ZusatzFeld.laenge>.

Beispiel (Bogenfeld):
Der Wert '31.06.20020' des Datenfeldes DKK:B: DATUMUNT Datum der Untersuchung ' (Zeile 11) überschreitet die zulässige Feldlänge 10.

Überprüfung der Schlüsselkodes

Die Überprüfung von Schlüsselkodes bezieht sich darauf, ob bei Schlüsselfeldern nur zulässige Schlüsselkodes verwendet werden.

[53] Wenn bei einem Ersatz die Länge nicht spezifiziert ist, so entfällt die Prüfung.

Standardisierter Fehlertext bei unzulässigen Schlüsselkodes eines Ersatzfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Ersatz-Feld.name> '<ErsatzFeld.bezeichnung>' ist kein zulässiger Kode des Schlüssels <Schluessel.name> (<Schluessel.bezeichnung>).

Standardisierter Fehlertext bei unzulässigen Schlüsselkodes eines Bogenfeldes

Der Wert '<WERT>' des Datenfeldes <Modul.name>:<Bogen.name>:<Feld.name> '<BogenFeld.bezeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) ist kein zulässiger Kode des Schlüssels <Schluessel.name> (<Schluessel.bezeich-nung>).

Standardisierter Fehlertext für Muss-Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<Zusatz-Feld.bezeichnung> ist kein zulässiger Kode des Schlüssels <Schluessel.name> (<Schluessel.bezeichnung>).

Bei weichen Plausibilitätsverletzungen ist dem Fehlertext das Wort „Hinweis“ voranzustellen.

Überprüfung der Muss-Felder

Ein nicht ausgefülltes Muss-Feld (Abschnitt B 2.3.3) führt zu einer Regelverletzung.

Standardisierter Fehlertext für Muss-Fehler eines Ersatzfeldes

Das Datenfeld '<Modul.name>:<Bogen.name>:<ErsatzFeld.name>'<Ersatz-Feld.bezeichnung>' muss einen gültigen Wert enthalten.

Standardisierter Fehlertext für Muss-Fehler eines Bogenfeldes

Das Datenfeld '<Modul.name>:<Bogen.name>:<Feld.name>'<BogenFeld.be-zeichnung>' (Zeile <BogenFeld.gliederungAufBogen>) muss einen gültigen Wert enthalten.

Standardisierter Fehlertext für Muss-Fehler eines Zusatzfeldes

Das Zusatzfeld <Modul.name>:<Bogen.name>:<ZusatzFeld.name> '<Zusatz-Feld.bezeichnung>' muss einen gültigen Wert enthalten.

Beispiel (Bogenfeld):
Das Datenfeld 'DKK:B: DATUMUNT ' Datum der Untersuchung ' (Zeile 11) muss einen gültigen Wert enthalten.

2.4.9 Verfahren für die Evaluation von Regeln

Grundsätzlich muss jede gemäß Abschnitt B 2.4.2 formulierte Regel evaluiert werden, wenn keine der folgenden drei Bedingungen zutrifft:

1. Für mindestens ein referenziertes Bogenfeld[54] schlägt eine harte Feldprüfung (Abschnitt B 2.4) fehl.[55]
2. Ein Feld der Regel ist nicht ausgefüllt (LEER) und **keine** der folgenden Teilbedingung trifft in Bezug auf das leere Feld zu:
   - Es ist in einer Liste enthalten, die mit einem Listenoperator (`EINSIN, KEINSIN, JEDESIN, EINSNICHTIN`) geprüft wird bzw. wird direkt gegen eine Liste geprüft (`IN, NICHTIN`).
   - Es wird in der Regel explizit mit <> oder = auf LEER geprüft.
   - Das Feld der Regel befindet sich auf einem vorhandenen Teildatensatz.
     (D.h. eine Regel wird nicht geprüft, wenn der entsprechende Teildatensatz eines leeren Feldes optional und im konkreten Fall nicht vorhanden ist).
   - Jeder Operation auf einen Wert `<> LEER` ist eine `ODER`-Verknüpfte Prüfung auf `LEER` direkt vorgeschaltet `(Feld = LEER ODER Feld Operator Operand)`.
3. Eine Funktion der Regel hat das Ergebnis LEER und wird in der Regel nicht explizit mit <> oder = auf LEER geprüft.

Der Algorithmus zur Evaluation einer Plausibilitätsregel ist in Abbildung 18 beschrieben.

Umgang mit Laufzeitfehlern

Bei der Evaluation von Regeln können, z. B. bei einem Vergleich von `LEER < WERT`, Laufzeitfehler entstehen. Solche Laufzeitfehler sind bei der Evaluation zu verhindern.

Laufzeitfehler bei der Evaluation von Regeln nach den Vorgaben sind immer ein Hinweis auf Syntaxfehler in der Regel. Das Ausbleiben von Laufzeitfehlern ist noch kein Beweis für die syntaktische Fehlerfreiheit einer Regel, da es auch fehlertolerante Parser geben könnte, die beim Verlassen des definierten Wertebereichs der Regel ein `FALSCH` zurückgeben können. Tatsächlich liegt aber hier ein undefinierter Zustand vor, der weder `FALSCH` noch `WAHR` ist.

Ein undefinierter Zustand an einem beliebigen Punkt im Evaluationsprozess muss für alle dort noch möglichen Werte durch die Struktur der Regel immer vermieden werden.

[54] Die Verbindung zwischen Regeln und Bogenfeldern geschieht über die Tabelle `RegelFelder`, siehe Regelfelder (Bogenfelder einer Regel), Abschnitt B 2.4.1.
[55] Erst bei Fehlerfreiheit der feldbezogenen Prüfungen werden die feldübergreifenden Prüfungen durchgeführt.

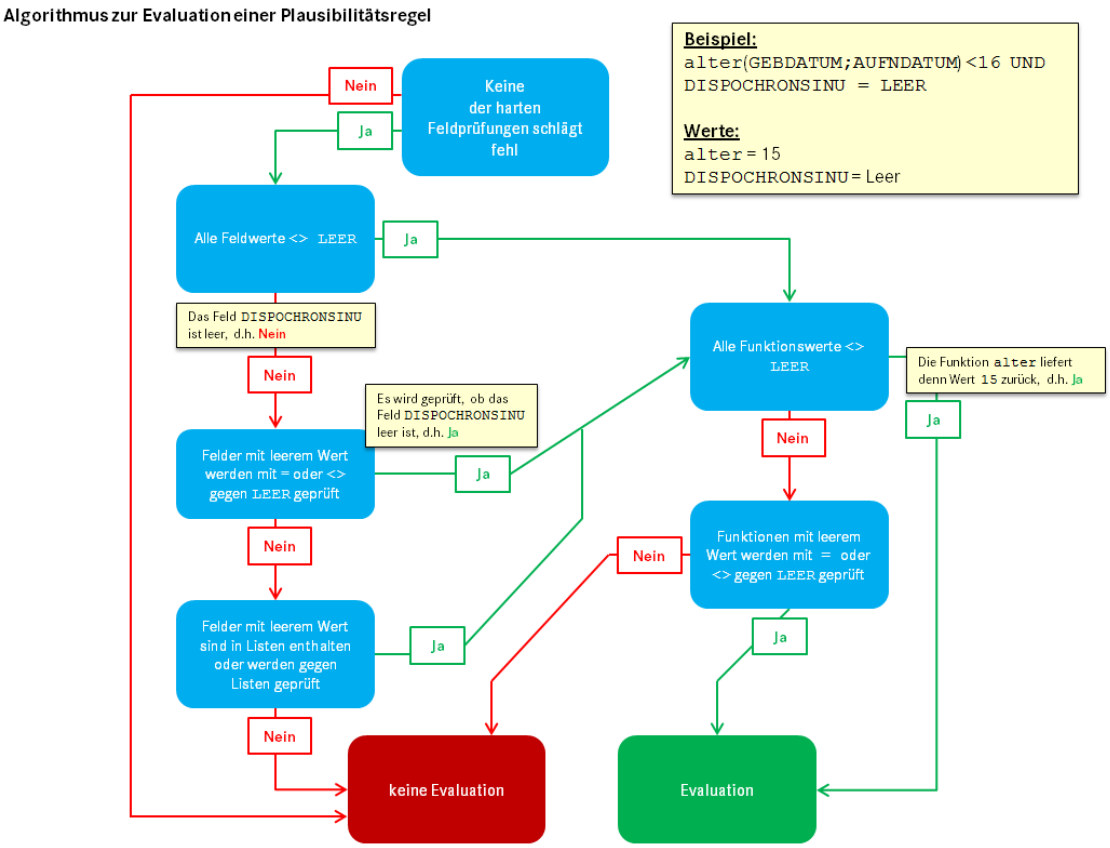


Abbildung 18: Algorithmus zur Evaluation von Plausibilitätsregeln

Teildatensatzübergreifende Regeln

Teildatensatzübergreifende Regeln (Abschnitt B 2.4.6) müssen u.U. mehrfach evaluiert werden (für jede Kombination von Teildatensätzen, die von der Regel betroffen ist).

Hinweis

In wenigen Einzelfällen beziehen sich Plausibilitätsregeln auf mehr als zwei Teildatensätze.

2.5 Exportfeldbeschreibung

Neben der Datenfeldbeschreibung (Abschnitt B 2.3) enthält die Spezifikationsdatenbank die Beschreibung der Exportfelder für ein Modul (Exportdatensatz). Diese werden zum Teil über Ersatzfelder berechnet. In der PB-Spezifikation (PBDOK) wird zwischen Dokumentationsmodulen (Abschnitt B 2.3.1) und Exportmodulen unterschieden. Für jedes Dokumentationsmodul ist genau ein Exportmodul definiert. Nur Exportmodule, die dem gleichen Pseudonymverfahren zugeordnet sind, dürfen auch gemeinsam exportiert werden.

2.5.1 Exportmodule

Um grundsätzlich unterschiedliche Abrechnungswege und davon abhängige Datenflüsse mit unterschiedlichen Datenannahmestellen für die Datenübermittlung berücksichtigen zu können, wird zwischen den Dokumentationsmodulen (Tabelle `Modul`) und den Exportmodulen (Tabelle

Exportmodul) unterschieden. In den PB-Verfahren DK und ZK ist ein Abrechnungsweg und entsprechender Datenfluss über die KVen festgelegt.

Tabelle 54: Struktur der Tabelle ExportModul

Feldname	Datentyp	Bemerkung
idExportModul	INTEGER	Primärschlüssel
fkModul	INTEGER	Bezug zum Dokumentationsmodul
name	KURZER TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	KURZER TEXT	Bezeichnung des Exportmoduls
existenzBedingung	KURZER TEXT	Definiert, unter welcher Bedingung das Modul in ein definiertes Exportmodul transformiert wird.
type_QS_data	KURZER TEXT	Datentyp im XML-Schema
type_QS_data_mds	KURZER TEXT	Datentyp im XML-Schema
fkExportzeit-raumEntlEJ	INTEGER	Definiert den Exportzeitraum für ein Exportmodul bei Entlassung im Erfassungsjahr. Die hinterlegten Zeitpunkte sind in der Tabelle Exportzeitraum definiert. Exportzeitraum.exportBis = Datenlieferfrist inkl. Korrekturfrist Exportzeitraum.exportBisQ4 = Datenlieferfrist ohne Korrekturfrist
fkExportzeit-raumEntlEJ1	INTEGER	Definiert den Exportzeitraum für ein Exportmodul bei Entlassung im Erfassungsjahr + 1 (Langlieger). Die hinterlegten Zeitpunkte sind in der Tabelle Exportzeitraum definiert. Exportzeitraum.exportBis = Datenlieferfrist inkl. Korrekturfrist Exportzeitraum.exportBisQ4 = Datenlieferfrist ohne Korrekturfrist
direkt	BOOLEAN	Handelt es sich um ein direktes Datenexportverfahren?
indirekt	BOOLEAN	Handelt es sich um ein indirektes Datenexportverfahren?

Feldname	Datentyp	Bemerkung
pid	BOOLEAN	Handelt es sich um ein Modul zur Follow-up-Erhebung?[56]
oKFE	BOOLEAN	Handelt es sich um ein Modul der oKFE-Richtlinie?
fkPseudonymVerfahren	INTEGER	Pseudonymisierung von FU-Verfahren[57]

Beispiel:
Das Modul DKK löst nach Abschluss der Dokumentation seinem Datenfluss entsprechend ein Exportmodul aus (DKK).

Das Attribut type_QS_data gibt Auskunft darüber, welchem Datentyp dies im XML-Schema entspricht.

Beispiel:
Beim Exportmodul DKK ist der Datentyp im XML-Schema qs_data_dkk_type.

Softwareanbietern soll hiermit die Integration eines Mechanismus der automatischen Datentypzuweisung ermöglicht werden, um den Aufwand zu reduzieren und Fehler zu vermeiden. Für Dokumentationsmodule, die das Anlegen eines Minimaldatensatzes zulassen, wird im Attribut ersatzBedingungMDS der Tabelle ExportModul definiert, unter welcher Bedingung das Modul MDS in ein definiertes Exportmodul transformiert wird.

2.5.2 Exportdatensatz

Der Exportdatensatz enthält die Exportfelder für ein Modul. Welche Zusatzfelder, Bogenfelder und/oder Ersatzfelder den Exportdatensatz pro Modul bilden, ist in Tabelle ExportFormat definiert.

Zusatzfelder[58]

Ein Exportfelddatensatz beginnt immer mit den folgenden Zusatzfeldern:

RegistrierNr = Registriernummer des Dokumentationssystems (Länderkode + Registrierkode)[59]
VorgangsnrMensch = Vorgangsnummer, menschenlesbar
VorgangsnrGUID = Vorgangsnummer, GUID
VersionNr = Versionsnummer

[56] Gemeint ist eine Follow-up-Erhebung mittels PID-Verfahren.
[57] Gemeint ist ein Follow-up-Verfahren anhand von PID-Daten.
[58] Dabei ist zu beachten, dass die Feldnamen beim Export durch die XML-Elemente zu ersetzen sind
[59] Für die Datenübermittlung an KVen und DAS-SV gelten andere Regelungen als bei der Datenübermittlung an die LQS/LKG.

`Storno`	= Stornierung eines Datensatzes (inkl. aller Teildatensätze)
`Modul`	= Bezeichnung des Exportmoduls
`Bogen`	= Teildatensatz (Bogen)
`DokAbschlDat`	= Dokumentationsabschlussdatum

Ein neuer Teildatensatz beginnt mit den Zusatzfeldern `RegistrierNr, Vorgangsnr-Mensch, VorgangsnrGUID` und `VersionNr`. Teildatensätze mit einem definierten Mutterteildatensatz beinhalten zusätzlich das Zusatzfeld `IdBogenFeldMutter` (= Wert des eindeutigen Bogenfeldes des Mutterteildatensatzes).

Zusatzfelder, welche nicht in der Datenfeldbeschreibung (Tabelle `BogenFeld`) eines Moduls enthalten sind, werden von der PB-Dokumentationssoftware ausgefüllt.[60]

Einige der in der Tabelle `ZusatzFeld` definierten Zusatzfelder werden nachfolgend erläutert:

- Das übertragene Speicherdatum `DokAbschlDat` (Datum des Dokumentationsabschlusses bzw. der Freigabe des Datensatzes für den Export) ist nicht Teil der Datenbank für Auswertungen und wird nur für organisatorische Zwecke verwendet. Das `DokAbschlDat` ist das Datum der letzten Änderung des gesamten Datensatzes.
- Die Versionsnummer (`VersionNr`) gibt an, welche Version des Datensatzes übertragen wird.
  In der Regel wird die Versionsnummer 1 lauten, d. h., dass der nach dem ersten Dokumentationsabschluss freigegebene Datensatz übertragen wird. Muss ein korrigierter Datensatz erneut eingesandt werden, so muss die Versionsnummer vom dokumentierenden System um eins erhöht werden. Die neue Version des Datensatzes wird bei der Entgegennahme geprüft und überschreibt bei Korrektheit die alte Version des Datensatzes.

Achtung

Wenn die entgegennehmende Stelle einen Datensatz mit derselben Versionsnummer ein zweites Mal erhält, so wird dieser zurückgewiesen.

- Der Eintrag 1 im Zusatzfeld `Storno` veranlasst die datenentgegennehmende Stelle, den übermittelten Datensatz einschließlich seiner Vorversion(en) als „storniert“ zu kennzeichnen.
- Das Zusatzfeld `IdBogenFeldMutter` wird bei Teildatensätzen eingefügt, welche einen mehrfach anlegbaren Elternteildatensatz (Attribut `fkBogenZahl = '*' oder '+'`) haben. In diesem Fall wird die identifizierende Nummer des Elternteildatensatzes (konfiguriert über `Bogen.fkEindeutigBogenFeld`) im Kindteildatensatz übermittelt.

Eine vollständige Liste der möglichen Zusatzfelder findet sich in der Tabelle `ZusatzFeld` der Spezifikationsdatenbank zur PB-Dokumentation. Zusatzfelder sind in Tabelle 55 definiert.

[60] Hier gilt also nicht der Grundsatz, dass Felder nicht vorbelegt sein dürfen.

Tabelle 55. Struktur der Tabelle ZusatzFeld

Feldname	Datentyp	Bemerkung
idZusatzFeld	INTEGER	Primärschlüssel
name	KURZER TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	KURZER TEXT	Bezeichnung des Zusatzfeldes
fkBasisTyp	INTEGER	Fremdschlüssel zur Tabelle BasisTyp
fkSchluessel	INTEGER	Fremdschlüssel zur Tabelle Schlüssel
laenge	INTEGER	Feldlänge des Zusatzfeldes
nachKommaLaenge	INTEGER	Anzahl der Nachkommastellen
istListe	BOOLEAN	Wenn istListe = WAHR, so sind die vom betreffenden Feld abgeleiteten Bogenfelder Listenfelder.
min	INTEGER	Harte Untergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional.
max	INTEGER	Harte Obergrenze des Wertebereichs eines numerischen Datenfeldes (modulübergreifend). Die Definition ist optional.
sortierNr	INTEGER	Sortiernummer
nurBasisTDS	BOOLEAN	Wenn nurBasisTDS = WAHR, so ist das Zusatzfeld im Exortdatensatz nur im Basisbogen enthalten
nurKreuzSternKindTDS	BOOLEAN	Wenn nurKreuzSternKindTDS = WAHR, so ist das Zusatzfeld nur bei Teildatensätzen mit einem definierten Mutterteildatensatz enthalten

Die Exportfelder werden pro Exportmodul exportiert. Hierfür erhält das Zusatzfeld Modul den Datentyp SCHLUESSEL und lässt nur die im Schlüssel Exportmodul definierten Kodes zu.

Hinweis

Bei den Zusatzfeldern ist zu beachten, dass die Feldnamen beim Export durch die entsprechenden XML-Elemente zu ersetzen sind (Abschnitt B 2.7.1).

Ersatzfelder für den Datenexport

Ersatzfelder werden aus einem oder mehreren Feldern der Datenfeldbeschreibung berechnet. Mit einem Ersatzfeld verknüpfte Bogenfelder werden nicht exportiert, wenn sie nicht als <bleibt> gekennzeichnet sind. Stattdessen werden ein oder mehrere Ersatzfelder exportiert.

Vorrangig dienen Ersatzfelder der Anonymisierung beim Datenexport. Die verwendeten Ersatzfelder sind in der Tabelle ErsatzFeld gespeichert.

Tabelle 56. Struktur der Tabelle ErsatzFeld

Feldname	Datentyp	Bemerkung
idErsatzFeld	INTEGER	Primärschlüssel
name	KURZER TEXT	Technischer Name (muss eindeutig sein)
bezeichnung	KURZER TEXT	Bezeichnung des Ersatzfeldes
laenge	KURZER TEXT	Zeichenlänge des Ersatzfeldes
formel	KURZER TEXT	Berechnungsformel der Ersatzfelder
fkBasisTyp	INTEGER	Obligatorischer Fremdschlüssel zum Basistyp
fkSchluessel	INTEGER	Optionaler Fremdschlüssel zum Schlüssel

Anonymisierungsvorschriften

Die Anonymisierung von Datenfeldern wird wie aus der folgenden Tabelle 57 Struktur der Tabelle ErsatzFuerFeld ersichtlich konfiguriert. Für die Programmierung der Exportfelder ist dieser Abschnitt nicht relevant, da die Exportfelder direkt über die Abfrage ExportFelderFürEinModul bzw. die Tabelle ExportFormat ermittelt werden können. Die Tabelle ErsatzFuerFeld ordnet einem Feld (Tabelle Feld) oder Bogenfeld ein oder mehrere Ersatzfelder zu. Die über das Attribut fkFeld definierte Anonymisierung ist die Standardanonymisierung für alle Module. Sie kann jedoch durch eine modulspezifische Anonymisierung überschrieben werden: Wenn ein Ersatzfeld mit einem Bogenfeld (über Attribut fkBogenFeld) verknüpft ist, wird statt des Bogenfelds das berechnete Ersatzfeld in die Exportdatei des Teildatensatzes geschrieben.

Tabelle 57. Struktur der Tabelle ErsatzFuerFeld

Feldname	Datentyp	Bemerkung
idErsatzFuerFeld	INTEGER	Primärschlüssel
fkFeld	INTEGER	Optionaler Fremdschlüssel zur Tabelle Feld
fkBogenFeld	KURZER TEXT	Optionaler Fremdschlüssel zur Tabelle BogenFeld
fkErsatzFeld	KURZER TEXT	Obligatorischer Fremdschlüssel zur Tabelle ErsatzFeld
parametrierbar	BOOL	Kennzeichen, ob das Feld parametrierbar ist

Beispiel:
Das Feld PLZ in Modul DKK (oKFE-RL) hat das Ersatzfeld PLZ3stellig.

Wenn das Ersatzfeld <entfällt> mit einem Bogenfeld verknüpft ist, entfällt das Bogenfeld in der Exportdatei. Wenn das Ersatzfeld <bleibt> mit einem Bogenfeld verknüpft ist, wird das Bogenfeld unverändert in die Exportdatei übernommen.

Wenn ein einziges Ersatzfeld sowohl über fkFeld als auch über fkBogenFeld definiert ist („doppelte Definition“), hat die spezielle Anonymisierung (über fkBogenFeld) Vorrang. Die allgemeinen Anonymisierungen (fkFeld) werden ignoriert. Die allgemeine Definition kommt nur in den Modulen zur Anwendung, in denen keine spezielle Anonymisierung vorliegt.

Parametrierung
Die Verknüpfung zwischen Feld bzw. Bogenfeld und Ersatzfeld kann parametriert werden (Attribut parametrierbar). Parametrierbare Ersatzfelder erscheinen immer als eigenes Element in der Exportdatei. Es ist aber über die Dokumentationssoftware konfigurierbar, ob die Werte auch tatsächlich exportiert werden. Auf diese Weise können spezifische Erfordernisse zum Datenschutz auf Landesebene berücksichtigt werden.

Muss-Felder des Exportdatensatzes
Verbindlich für die Muss-/Kann-Prüfung ist die Definition in der Tabelle BogenFeld. Die Muss-/Kann-Zuordnungen im Exportdatensatz werden hieraus abgeleitet:

- Für Exportfelder, die nicht pseudonymisiert werden und die keine Listenfelder sind, entspricht die Muss/Kann-Zuordnung der Definition in der Tabelle BogenFeld.
- Die Muss-/Kann-Zuordnung der pseudonymisierten Datenfelder (Ersatzfelder) ergibt sich logisch aus der Berechnungsformel (Attribut formel in Tabelle ErsatzFeld). Beispielsweise ist ein Ersatzfeld ein Muss-Feld, wenn alle an der Berechnung beteiligten Bogenfelder Muss-Felder sind.
- Bei Muss-Listenfeldern der Tabelle BogenFeld ist grundsätzlich nur das erste Element ein Muss-Feld, die weiteren Elementfelder sind Kann-Felder. Hierbei ist zu beachten, dass Exportfelder für Listenfelder seit der Spezifikation 2015 nicht mehr pro Listenelement, sondern pro Listenfeld dargestellt werden. Die Anzahl der Elemente von Listenfeldern ist den Abfragen Exportfelder und ExportfelderFürEinModul zu entnehmen (ExportfelderFürEinModul.elemente).
- Zusatzfelder der Tabelle ExportFormat sind Muss-Felder, außer wenn sie Schlüsselfelder mit einem Jleer-Schlüssel (z. B. das Zusatzfeld Storno) sind.

Achtung

Wenn ein Listenfeld als Muss-Feld deklariert ist, so ist nur das erste Exportfeld der Liste ein Muss-Feld, die restlichen Elemente sind Kann-Felder. Wenn ein Listenfeld als Kann-Feld deklariert ist, so sind alle exportierten Elemente ebenfalls Kann-Felder.

Das Nichtausführen der erforderlichen Muss-Prüfungen kann gravierende Folgen für die Auswertung haben!

Als Hilfestellung für Datenannahmestellen bei der Umsetzung gilt das Attribut fkMussKann in der Tabelle ExportFormat, deren Inhalte automatisch generiert werden.

2.6 Versionierung

Im Folgenden werden die Tabelle Version, der Abgleich zu vorherigen Versionen, die Abgrenzung zwischen Erfassungsjahren und Datensatzformaten sowie die Version von Exportverfahren und -dateien beschrieben.

2.6.1 Grundlegende Definitionen

In der Tabelle Version finden sich Informationen zur Version der Spezifikationsdatenbank. Die wichtigsten Eigenschaften einer Version sind der Versionsname (Attribut name) und die Gültigkeitszeiträume (Attribute ab und bis). Der Gültigkeitszeitraum einer Version ist in der Regel ein Erfassungsjahr (z. B. Behandlung zwischen dem 1. Januar 2021 und dem 31. Dezember 2021).

Die PB-Dokumentationssoftware eines Erfassungsjahres wird für diejenigen Behandlungsfälle verwendet, deren Behandlungsdatum (ambulante Fälle) in den oben definierten Gültigkeitszeitraum fällt.

Jedes Modul der Datenbank hat eine Version (vgl. Attribut fkVersion in Tabelle Modul). In einer Spezifikationsdatenbank haben alle Module dieselbe Version. Diese entspricht immer der in Tabelle Version als gültig gekennzeichneten Version. Über die in der Datenbank definierten Relationen sind auch für alle Bogenfelder (Tabelle BogenFeld), Exportfelder (Tabelle ExportFormat) und Plausibilitätsregeln (Tabelle Regeln) Versionen definiert.

Status der Spezifikation

Versionen können den Status in Entwicklung, final oder Update der finalen Spezifikation haben. Diese Zustände werden zum Nachschlagen in der Tabelle VersStatus verwaltet. Das Attribut gueltig zeigt die gültige Version der Datenbank an. Es darf nur eine einzige Version als gültig markiert sein.

Beispiel:
Die Version 2021 V01, einschließlich ihrer Updates, sind finale Versionen.

Hat eine Spezifikationsdatenbank den Status in Entwicklung, kann Modul.fkVersion als ungueltig markierte Versionen enthalten, um Zwischenstände abzubilden.

Historie der Versionen

Die Tabelle Version enthält auch einen Selbstbezug (Attribut fkVersion), der die Identifizierung der Vorgängerversion ermöglicht.

Die Vorgängerversion der Spezifikation 2021 V01 ist die Version 2020 V04.

2.6.2 Delta-Informationen zur vorhergehenden Version

Um den Benutzern der Spezifikation umfassende Informationen zu den jeweiligen Änderungen zur Verfügung zu stellen, enthält die Spezifikationsdatenbank Tabellen, die den Änderungsstand im Vergleich mit der letzten gültigen Version des Vorjahres und zur Vorversion der Datenbank anzeigen. Diese sogenannten Delta-Tabellen werden automatisch generiert.

2.6.3 Abgrenzung zwischen Erfassungsjahren und Datensatzformaten

Die Datenannahmestellen müssen Datensätze von Leistungserbringern entgegennehmen, falls die Eingriffsdaten ambulanter Fälle in den Gültigkeitszeitraum der Version einer finalen Datenbank fallen.

Das Abgrenzungskriterium zwischen den Erfassungsjahren[61] ist:

- das Datum des Eingriffs (OPDATUM/EBMDATUM) bei Datensätzen ambulanter Behandlungsfälle.

Achtung

Dem Erfassungsjahr 2021 zugeordnete Fälle müssen im Format der Spezifikation 2021 an die Datenannahmestellen gesandt werden, sonst ist die Datenlieferung zurückzuweisen.

Die entsprechenden Exportzeiträume sind den Attributen `Exportmodul.fkExportzeitraumEntlEJ` und `Exportmodul.fkExportzeitraumEntlEJ1` der Datenbank zur PB-Dokumentation zu entnehmen.

2.6.4 Version des Exportverfahrens

In der Spezifikationsdatenbank für die PB-Dokumentation wird in der Tabelle `Version` neben der gültigen Version (`gueltig`) auch das gültige Exportverfahren (`gueltigExportVerfahren`) angegeben. Dieses Attribut gibt an, welches Versionskürzel (Attribut `Version.name`) im XML-Dokument im Headerbereich unter `header/document/software/specification@V` verwendet werden muss.

Nur eine Version ist für das Exportverfahren als gültig markiert. Die Version des Exportverfahrens kann eine Vorversion der gültigen Version der Spezifikation sein.

2.7 Administrative Objekte

Die Datenbank für PB-Dokumentation beinhaltet neben den Dokumentationsobjekten eine Gruppe von administrativen Objekten, die Hilfestellungen für die Einhaltung eines korrekten Datenflusses und standardisierter Prüfprozesse geben.

Zu den administrativen Objekten gehören **Mapping-Informationen**, mit denen PB-Daten, die außerhalb des eigentlichen PB-Containers (dem Element `<qs-daten>`) im XML verortet wer-

[61] Das Abgrenzungskriterium definiert somit die Zuordnung des Datensatzes zu einer Version der Spezifikation bzw. das Format des Datensatzes.

den können. Außerdem gibt es eine **Auflistung und Kategorisierung von Prüfschritten** zur Implementierung eines Datenservice (Abschnitt B 2.7.3). Um eine Ansicht der administrativen Objekte zu erhalten, ist in Access (Spezifikationsdatenbank zur PB-Dokumentation) oberhalb der Objektübersicht das Drop-Down-Menü zu öffnen und der Menüpunkt „Benutzerdefiniert“ auszuwählen.

In einer separaten Datenbank ist eine Übersicht über die für ein Modul innerhalb einer Region relevanten technischen **Datenservice mit Empfängeradressen** und zu verwendenden XML-Schlüsseln(Abschnitt B 2.7.2) enthalten.

Abbildung 19 und Abbildung 20 stellen die Beziehungen der administrativen Objekte zueinander dar.[62]

Achtung

Die folgend beschriebenen Tabellen enthalten auch Einträge welche nicht für alle am Datenfluss beteiligten Stellen von Relevanz sind. Erst die Zuweisung von rollenspezifischen Zuständigkeiten (durch die Tabelle `PruefprozessPruefkategorie-Zielgruppe`) stellt entsprechend rollenspezifische Verbindlichkeiten zur Anwendung einzelner Prüfmechanismen dar.

Für eine schnelle Übersicht gültiger Prüfungen ist die Abfrage `vPruefung` zu verwenden, deren Inhalte am Ende dieses Abschnitts näher erläutert werden.

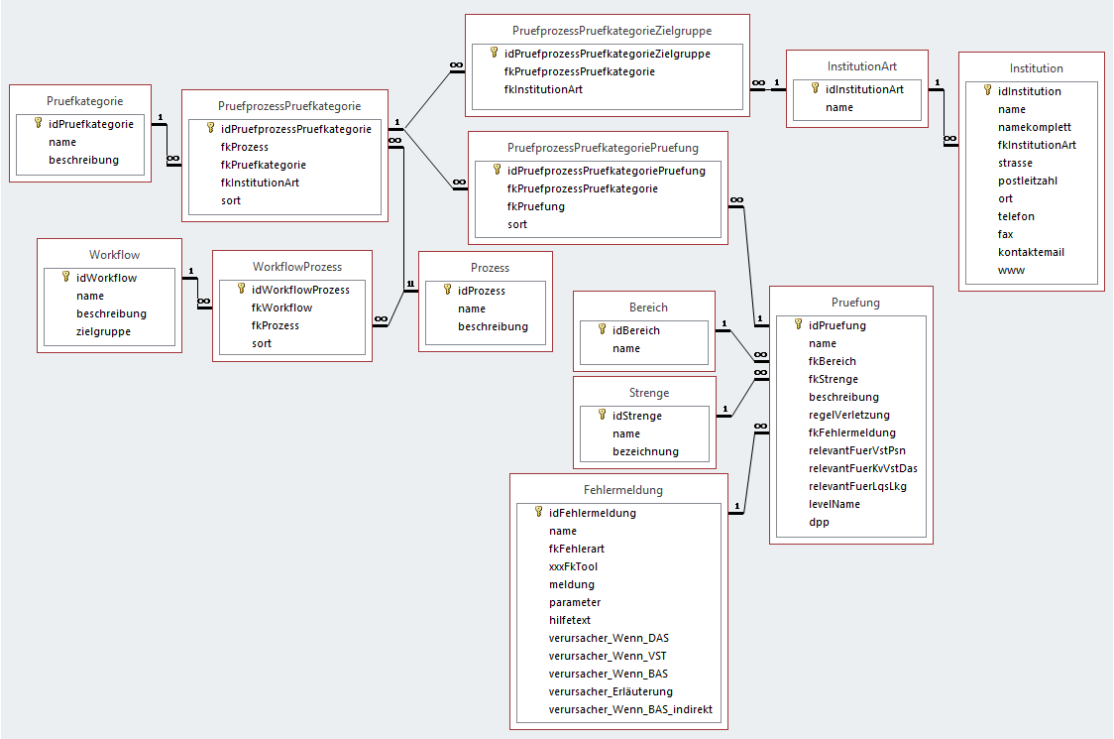


Abbildung 19: Beziehungen der administrativen Objekte (Prüfungen)

[62] Die Tabellen zu Datenserviceinformationen sind in einer separaten Datenbank abgelegt

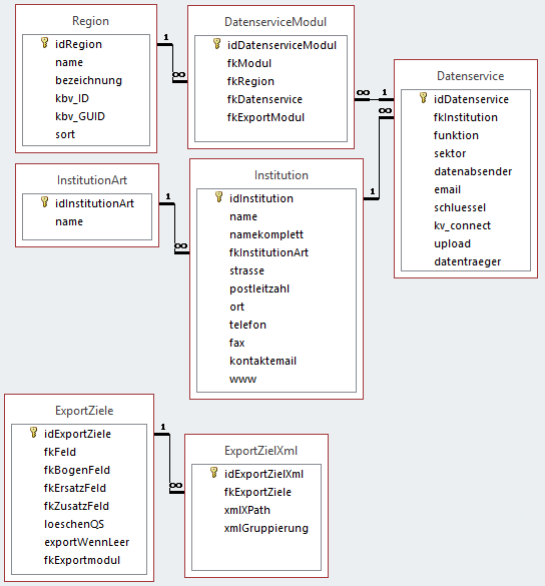


Abbildung 20: Beziehungen der administrativen Objekte (Datenservice, Mapping-Informationen)

2.7.1 CSV/XML-Mapping in der Spezifikationsdatenbank (PBDOK)

Es gibt eine Reihe von XML-Elementen, die zurzeit nicht in der Access-DB hinterlegt sind. Insgesamt gibt es vier Kategorien von Abweichungen:

- Neu eingeführte XML-Elemente
- Header-Informationen, wie die GUID des Dokuments
- Umbenannte und aus dem PB-Datensatz ausgelagerte CSV-Felder wie beispielsweise `kasseiknr2Stellen`
- Gelöschte Felder wie beispielsweise das Feld `IKNRKH`

In Abstimmung mit den Softwareanbietern werden die CSV-Felder nicht an die XML-Struktur angepasst. Stattdessen wird für betroffene Felder eine Mapping-Tabelle in der Access-DB hinterlegt.

Die durch Bögen (Teildatensätze) für das Element `<qs_data>` festgelegte Struktur bleibt durch die aufgeführten Mappings unangetastet. Die Mappings erlauben alleine eine Festlegung darüber, ob und falls ja wo ein betroffenes Datum außerhalb des `<qs_data>` Elements in das XML geschrieben wird und ob das Datum für das Element `<qs_data>` gelöscht werden muss. Ggf. nötige Umbenennungen außerhalb von `<qs_data>` werden implizit durch die Ortsangabe vorgenommen.

Die für das Mapping geführten Tabellen heißen `ExportZiele` und `ExportZielXml`. In der Abfrage `vExportZieleXml` sind diese Tabellen zu einer Übersicht zusammengefasst. Hierbei ist zu beachten, dass nicht jedem Eintrag in der Tabelle ExportZiele auch ein Eintrag in der Tabelle `ExportZielXML` zugewiesen sein muss. Soll beispielsweise ein Datum aus dem Element `<qs_data>` lediglich gelöscht werden, ohne jedoch an andere Stelle im XML verschoben zu werden, ist keine dementsprechende `xmlXPath`-Angabe erforderlich.

Es können alle Feldarten referenziert sein. Referenziert sind aber nur solche Felder, die nicht oder nicht nur im Element <qs_data> aufgeführt werden. Die Felder werden referenziert und über diese Tabelle mit zusätzlichen Informationen in Bezug auf das XML verknüpft.

Die über das Attribut fkFeld referenzierten Felder gelten für alle Module, in denen dieses Feld verwendet wird. Wird ein Feld (gleichzeitig) über unterschiedliche Feldarten referenziert, überlagern detailliertere Angaben die allgemeinen. Wird so beispielsweise in der Tabelle ExportZiele ein Feld allgemein über den Fremdschlüssel fkFeld und in einer weiteren Zeile modulspezifisch über fkBogenFeld referenziert, werden die allgemeinen Regelungen (loeschenQS, exportWennLeer, xmlXPath und xmlGruppierung) für das Feld mittels der Referenzierung über fkFeld mit den spezifischeren Regelungen für das modulbezogene Feld mittels der Referenzierung über fkBogenFeld für das konkret aus der Referenzierung hervorgehende Modul überschrieben.

Dabei ist ebenfalls eine etwaige Konkretisierung über fkExportModul zu beachten, welche Ihrerseits konkretisierenden Charakter hat. Die über das Attribut fkErsatzFeld referenzierten Ersatzfelder gelten für alle Exportmodule, in denen dieses Ersatzfeld verwendet wird. Bei Referenzierung des gleichen Feldes (insbesondere bezogen auf fkFeld, fkErsatzFeld bzw. fkZusatzfeld) in zwei Zeilen – einmal ohne und einmal mit Angabe von fkExportmodul - so überschreibt der Eintrag mit Angabe den Eintrag ohne Angabe für das konkret benannte Exportmodul.

idExportZiele	fkFeld	fkBogenFeld	fkErsatzFeld	fkZusatzFeld	loeschenQS	exportWennLeer	fkExportmodul
1				Modul	☐	☐	
2			kasseiknr2Stellen		☐	☐	
3				Bogen	☑	☐	
4	BSNRAMBULANT				☑	☐	
5	LANR				☑	☐	
6				RegistrierNr	☑	☐	
7				VorgangsnrMensch	☑	☐	
8				VorgangsnrGUID	☑	☐	
9				VersionNr	☑	☐	
10				Storno	☑	☐	
11				DokAbschlDat	☑	☐	
13			PLZ3stellig		☑	☐	
14	VERSICHERTENIDNEU				☑	☐	
(Neu)					☐	☐	

Abbildung 21: ExportZiele

idExportZielX	fkExportZiele	xmlXPath	xmlGruppierung
1	2	ancestor::data_container/care_provider/IKNR/@V	☑
2	1	ancestor::cases/@module	☐
3	1	case_admin/module/@V	☐
4	1	qs_data/@module	☐
5	2	patient/@twodigitik	☐
6	4	ancestor::data_container/care_provider/BSNRAMBULANT/@V	☑
7	5	ancestor::data_container/care_provider/LANR/@V	☑
8	6	ancestor::root/header/provider/@registration	☐
9	9	case_admin/version/@V	☐
10	7	case_admin/id/@V	☐
11	10	case_admin/action/@V="delete"	☐
12	14	patient/pid/VERSICHERTENIDNEU/@V	☐
(Neu)			☐

Abbildung 22 : ExportZieleXML

Die booleschen Spalten loeschenQS und exportWennLeer geben folgende Informationen an:

- `loeschenQS`: Das Datum wird nur an den/die alternativen Ort(e) geschrieben und taucht in den PB-Daten nicht mehr auf. Hier geht es zum Beispiel um vom Datenschutz betroffene Felder, die nur in bestimmte Bereiche des XML geschrieben werden dürfen, wo sie dann verschlüsselt werden können.
- `exportWennLeer`: Hier wird ein leeres Feld nur dann berücksichtigt und leer eingetragen, wenn dieser Wert auf `true` steht.

In der Spalte `xmlXPath` vom Typ String steht ein XPath-Ausdruck, der ein Mapping auf einen im XML liegenden Ort des betroffenen Feldes enthält. Ausgangspunkt des XPath-Ausdrucks ist immer der aktuelle Datensatz im XML, also das `<case>` Element.

Die boolesche Spalte `xmlGruppierung` gibt an, dass ein neuer Knoten nur eingesetzt wird, wenn das Datum einen neuen Wert enthalten sollte.

loeschenQS	exportWennLeer	fkExportZiel	xmlXPath	xmlGruppierung
☑	☐	1	/root/body/data_container/care_provider/IKNR/@V	☑
☐	☐	2	ancestor::cases/@module	☑
☐	☐	2	case_admin/module/@V	☐
☐	☐	2	qs_data/@module	☐
☑	☐	4	patient/pid/VERSICHERTENIDNEU/@V	☐

Abbildung 23: Beispiel für XPath-Ausdrücke in der Tabelle `ExportZielXml` in Verbindung mit weiteren Informationen

2.7.2 Datenservice

In einer separaten Datenbank sind Angaben über die beim Export relevanten technischen Datenservice und ihre verfahrensbezogene und regionale Zuordnung zu finden. Diese Datenbank zu Datenserviceinformationen ist auf der Homepage https://www.iqtig.org/ zu finden.

2.7.3 Prüfschritte

Innerhalb der administrativen Objekte der PBDOK-Datenbank befinden sich Tabellen zum Beschreiben von Prüfschritten, die Institutionen als Vorlage für die Implementierung eines Datenservice nutzen können. Diese reichen von der Prüfung einer korrekten Übermittlung der Exportdatei bis hin zur Schemaprüfung der durch die Institution modifizierten XML-Datei.

Zurzeit beschränken sich diese Prüfungen auf die PB-Dokumentation für Datenannahmestellen. Zukünftig können damit weitere Workflows abgebildet werden, wie der folgende Inhalt der Tabelle `Workflow` zeigt:

Tabelle 58: Überblick über neben der PB-Dokumentation weitere potenzielle Workflows mit definierten Prüfschritten

Name	Beschreibung	Zielgruppe	In derzeitiger Spezifikation enthalten
PB-Dokumentation	Prüfung der PB-Dokumentation	DAS	Ja
PID-Pseudonymisierung	Pseudonymisierung der PID	VST	Nein

Für jede Prüfung (Tabelle `Pruefung`) existiert eine Fehlermeldung (Tabelle `Fehlermeldung`), die als Validierungsergebnis ausgegeben werden muss, wenn diese Prüfung fehlschlägt. Um Prüfungen chronologisch, d. h. von der Ankunft einer Transportdatei bis zum Abschicken an die Vertrauensstelle zu gruppieren, sind die Prüfungen Kategorien (Tabelle `PruefprozessPruefkategorie`) zugeordnet und die Kategorien wiederum sogenannten Workflowprozessen (Tabelle `Prozess`). Über die Tabelle `PruefprozessPruefkategorie` werden die Kategorien den Workflowprozessen zugeordnet, während über die Tabelle `PruefprozessPruefkategoriePruefung` einzelne Prüfungen den Kategorien innerhalb der Workflowprozesse zugeordnet werden. Innerhalb dieser Zuordnungstabellen gibt die Spalte `sort` die oben genannte chronologische Reihenfolge wieder. Da nicht alle Prüfungen auf alle Arten von Datenannahmestellen zutreffen, ordnet die Tabelle `PruefprozessPruefkategorieZielgruppe` einzelne Kategorien unterschiedlichen Arten von Datenannahmestellen zu.

Eine aus diesen Daten generierte HTML-Ansicht, abrufbar unter https://iqtig.org/datenerfassung/datenannahmestellen/pruefschritte-qs-dokumentation/, ist dem folgenden Screenshot zu entnehmen.

In der Abbildung 26 werden die Prüfschritte der PB-Verfahren dargestellt.

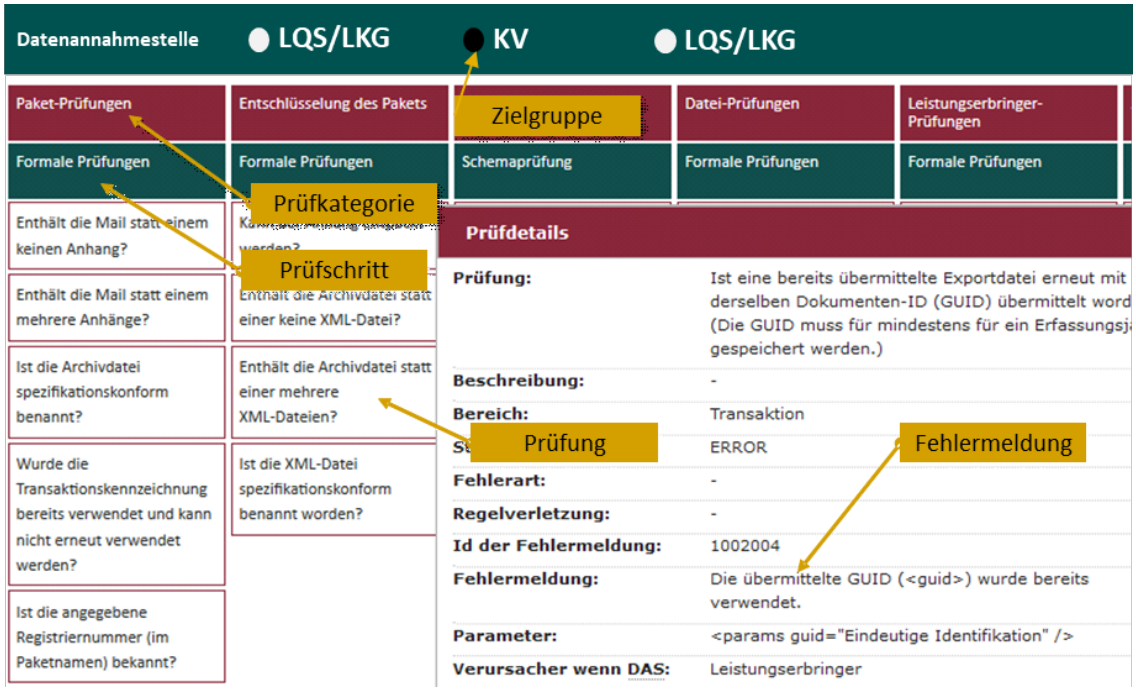


Abbildung 24: HTML-Ansicht der Prüfschritte innerhalb der PB-Dokumentation

Ein schneller Überblick über alle Prüfungen ist der Abfrage `vPruefung` zu entnehmen, deren Felder folgend kurz erläutert werden:

Tabelle 59: Felder der Abfrage vPruefung

Feldname	Bedeutung	Bezugselement im XML
Prüfung	Kurzbeschreibung der Prüfung	-
Beschreibung	ggf. erweiterte Erläuterungen zur Prüfung	-
Id-Fehlermeldung	Fehlerreferenz	validation_item/error/rule_id[@V]
Bereich	Einordnung der Prüfung	validation_item[@V]
Strenge	Konsequenz im Fehlerfall (Hinweis, Fehler)	validation_item/error/rule_type[@V]
Fehlerart	Legacy -> Bezug zu CSV	validation_item/error/error_type[@V]
Regelverletzung	Bedingungen, die zu einer Regelverletzung führen	-
Fehlermeldung	Standardisierte Fehlermeldung	validation_item/error/error_message[@V]
Parameter	Parameter für Fehlermeldung	-
Hilfetext	ggf. weitere Hinweise zum Fehler	-
Verursacher_Wenn_DAS	Verursacher wenn Regelverstoß in DAS festgestellt	validation_item/error/originator[@V]
Verursacher_Wenn_VST	Verursacher wenn Regelverstoß in VST festgestellt	validation_item/error/originator[@V]
Verursacher_Wenn_BAS	Verursacher wenn Regelverstoß in BAS festgestellt	validation_item/error/originator[@V]
Verursacher_Wenn_BAS_indirekt	Verursacher wenn Regelverstoß in BAS festgestellt	validation_item/error/originator[@V]
Verursacher_Erläuterung	ggf. Erläuterung zum Verursacher eines Fehlers	-

Feldname	Bedeutung	Bezugselement im XML
Relevant_Für_VST-PSN[63]	M=obligatorische Prüfung, K= freiwillige Prüfung	-
Relevant_Für_KV _VST-DAS[63]	M=obligatorische Prüfung, K= freiwillige Prüfung	-
Protokoll-Level	Zuordnung der Prüfung: DK-> betrifft Dokument DS-> betrifft Datensatz	-

[63] Gibt an, ob Prüfungen bei VST oder DAS ambulant vorgeschrieben (M) sind oder freiwillig (K) durchgeführt werden können.

3 XML-Schema

Die XML-Schema-Datei (XSD) ist eine Empfehlung des W3C[64] zum Definieren von Strukturen für XML-Dokumente.

In der Basisspezifikation werden die XML-Schemata aus der Spezifikationsdatenbank abgeleitet und haben die Aufgabe, die aktuellen Datenflussmodelle der G-BA-Richtlinien abzubilden:

- Abbildung der zu exportierenden PB-Daten
- Abbildung der Schnittstellen an den Institutionsübergängen (z. B. die Schnittstelle zwischen DAS und VST)
- Abbildung der vorzunehmenden Datentransformation in den jeweiligen Einrichtungen des Datenflusses
  (z. B. LE-Pseudonymisierung bei der DAS)
- Abbildung der Rückprotokollierung

Aus diesem Grund gibt es bei der Erstellung von Schemata, welche die Konformität von Richtlinie und Datenschutz sicherstellen sollen, kein „Allround-Schema“, welches alle Anforderungen an alle Beteiligten abdeckt, sondern eine „Schema-Familie“, aus der heraus gezielt für jede Schnittstelle („Interface“) eine passende Datenstruktur definiert wird.

3.1 Kompositionsmodell

Bei der Schema-Erstellung wurde auf ein Kompositionsmodell zurückgegriffen, in dem sich alle Teilschemata am Ende einen Namensraum teilen, um diese Schema-Familie besser warten zu können und gleiche Teilstrukturen nur einmal definieren zu müssen. Als Bezeichnung des Namensraums wurde „`urn:gba:sqg`“ gewählt. Zu diesem Namensraum werden die Bausteine je nach Bedarf über „`includes`“ zusammengestellt.

[64] http://www.w3.org/XML/Schema

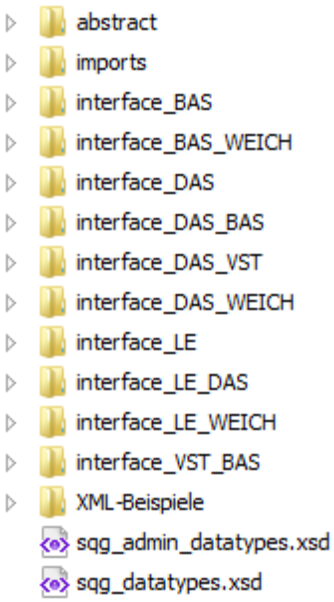


Abbildung 25: Dateiordner der Schnittstellen-Schemata

Das Kompositionsmodell macht es möglich, Konzepte aus der objektorientierten Programmierung – darunter fallen die Konzepte abstrakter Typ, Ersetzbarkeit von Typen, Wiederverwendung und Polymorphismus – zu nutzen. Dadurch können Schemata erstellt oder genutzt werden, die generische Grundtypen definieren und diese Typen so erweitern, dass sie schnittstellenspezifisch sind, ohne das ursprüngliche Schema zu beeinflussen. Dieses Kompositionsmodell wird hier näher erläutert.

Beispiel:
Die leistungserbringeridentifizierenden Daten existieren gemäß Datenflussmodell der G-BA-RL in drei Ausprägungen:

- im Klartext (Schnittstelle LE)
- pseudonymisiert (Schnittstelle DAS)
- verschlüsselt (Schnittstelle DAS-VST)

Bei dieser Konzeption werden alle drei Ausprägungen vom selben Basisdatentyp geerbt, in einem zweiten Schritt die drei Ausprägungen konkretisiert und angepasst, und über „includes“ in die jeweilige Schnittstelle integriert.

3.2 Schnittstellen

In der folgenden Tabelle werden Schema-Dateien aufgeführt, die im Rahmen der Übermittlung der PB-Daten Verwendung finden. Andere Dateien haben zwar ebenfalls die Dateiendung „.xsd“, sind aber keine vollständigen Schemata, sondern Bausteine für Schnittstellen.

Tabelle 60: Verwendbare Schemata und Ablageort

Schnittstelle	Schema	Verfahren	Absender	Empfänger
LE interface_LE	2021_kv_pid_1.0_Export	PID-Verfahren (pid)	PR/med.Labore-Kollektiv	KV
LE/DAS interface_LE_DAS	interface_LE_KV	PID-Verfahren (pid)	PR/med.Labore-Kollektiv	KV
DAS interface_DAS	interface_KV_psn	PID-Verfahren	Nach der LE-Pseudonymisierung in der DAS-KV	
DAS/VST interface_DAS_VST	interface_DAS_VST	PID-Verfahren	DAS	VST
VST/BAS interface_VST_BAS	interface_VST_BAS.xsd	PID-Verfahren	VST	BAS

Um nach einer Schemavalidierung der XML-Dateien die Weiterverarbeitung und demensprechend die spezifikationskonforme Protokollierung auf Datensatzebene weiterhin zu ermöglichen, wurden neben der oben in Tabelle 60 beschriebenen Schemavariante ein „weiches“ Schema für die Schnittstellen LE und DAS eingeführt. Diese weiche Variante wird ausschließlich mit dem Datenprüfprogramm verwendet (Abschnitt B 4.2). Es ist dabei zu beachten, dass nur die DAS, die die PB-Daten entschlüsseln, diese weiche Variante benötigen. Diese ist in den PB-Verfahren gemäß oKFE-RL die KV.

In der folgenden Tabelle werden die Schema-Dateien aufgeführt, die im Rahmen der Prüfung mit dem Datenprüfprogramm verwendet werden.

Tabelle 61: Weiche Schemavarianten für das Datenprüfprogramm

Schnittstelle	Schema	Verwendungszweck in Zusammenhang mit dem DPP
LE Interface _LE_WEICH	2021_kv_pid_1.0_Export	Validierung der Module für die KV mit dem Datenprüfprogramm (DPP)
DAS Interface _DAS_WEICH	interface_KV_pid	Validierung der PID-Verfahren mit dem DPP nach der Entschlüsselung der PB-Daten in der KV
DAS Interface _DAS_WEICH	interface_KV_psn	Validierung der PID-Verfahren mit dem DPP nach der Entschlüsselung der PB-Daten und der Pseudonymisierung der LE-Daten in der KV
BAS interface	interface_BAS_pid.xsd	Validierung der PID-Verfahren mit dem DPP

_BAS_WEICH		

In der folgenden Tabelle werden Schema-Dateien aufgeführt, die im Rahmen der Rückprotokollierung Verwendung finden.

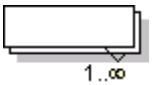
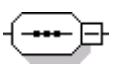
Tabelle 62: XML-Schemata für die Rückprotokollierung

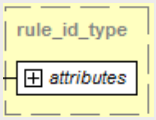
Schnittstelle	Schema	Verwendungszweck in Zusammenhang mit dem DPP
LE/DAS interface_LE_DAS	response_DAS_LE.xsd	Datenflussprotokoll der DAS
	response_receipt.xsd	Empfangsbestätigung der DAS
DAS/VST interface_DAS_VST	response_VST_DAS.xsd	Datenflussprotokoll der VST
	response_receipt.xsd	Empfangsbestätigung der VST
DAS/BAS interface_DAS_BAS	response_BAS_DAS.xsd	Datenflussprotokoll der BAS
VST/BAS interface_VST_BAS	response_BAS_VST.xsd	Datenflussprotokoll von der BAS an die VST
	response_receipt.xsd	Empfangsbestätigung

3.3 Darstellung der XML-Struktur

Zur Veranschaulichung der verwendeten XML-Schemata werden Diagramme verwendet, deren Symbole in der folgenden Tabelle kurz dargestellt und erläutert werden.

Tabelle 63: Symbole in den XML-Schema-Diagrammen

Symbol	Beschreibung
	**Optionales Element** Kardinalität 0..1 („0 oder 1“)
	**Obligatorisches Element** Kardinalität 1: das Element muss genau einmal vorkommen
	**Mehrfach wiederholbares Element** Kardinalität: die erlaubte Anzahl der Elemente wird unter dem Symbol dargestellt (Beispiel: 1..n, n..m).
	**Referenzelement** Das referenzierte globale Element ist an anderer Stelle im Schema definiert.
	**Eine Folge von Elementen** Die Elemente müssen genau in der Reihenfolge vorkommen, in der sie im Schemadiagramm angezeigt sind.

Symbol	Beschreibung
	**Eine Auswahl von Elementen** Nur ein einziges Element aus der Liste kann ausgewählt werden.
	Ein Element mit Kind-Elementen
	**Komplexer Datentyp** Der komplexe Datentyp wird mit einem Rahmen mit einem gelben Hintergrund angezeigt.

Die wesentlichen Bestandteile der XML-Schemata werden in den nachfolgenden Abschnitten dargestellt. Die Darstellung umfasst folgende Eigenschaften des betrachteten Elements:

- Grafische Abbildung der Kind-Elemente und -Attribute
- Auflistung der Kind-Elemente
- Auflistung der Kind-Attribute sowie ihre Eigenschaften wie:
  - Name
  - XML-Datentyp (technische Bezeichnung: „Type“)
  - Muss-Kann-Feld (technische Bezeichnung: „Use“)
  - Konstante (technische Bezeichnung: „Fixed“)
  - Kurze Beschreibung (technische Bezeichnung: „Annotation“)

3.4 Aufbau der XML-Exportdatei

Grundsätzlich beginnt jede XML-Exportdatei mit einer Headerzeile gefolgt vom Wurzelelement `<root>`, das den gesamten Inhalt einschließt. Als Zeichensatz wird „UTF-8“ (Unicode-Codierung) verwendet.

Beispiel:
Headerzeile

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

3.4.1 Namensräume

Die Schemata für den XML-Datenfluss sind für den Namensraum mit der id `"urn:gba:sqg"` definiert. Dieser soll ohne Präfix-Mapping im `<root>`-Element in das XML eingebunden werden.

Falls ein Ablageort des zugrundeliegenden Schemas angegeben werden soll, wird dieses im <root>-Element mit dem Attribut `"schemaLocation"` vorgenommen. Da dieses Attribut ebenfalls aus einem externen Namensraum stammt, wird dieser Namensraum dem reservierten Präfix `"xsi"` zugeordnet, was sich dann als "xsi:schemaLocation='...'" liest:

- `xmlns:xsi=`http://www.w3.org/2001/XMLSchema-instance
- `xsi:schemaLocation="urn:gba:sqg interface_LE_KV.xsd"`

Für die Verschlüsselung der XML-Elemente werden zwei externe Namensräume mit der id:

- " xmlns:ds=http://www.w3.org/2000/09/xmldsig#"
  und
- " xmlns:xenc=http://www.w3.org/2001/04/xmlenc#"

verwendet.

Diese sollen auf die reservierten Präfixe "ds" und "xenc" zugeordnet eingebunden werden. Es ist zu empfehlen, die externen Namensräume in das <root>-Element einzubinden, um lokale Wiederholungen auf Elementebene zu vermeiden.

Beispiel:

```
<root container_version="2.0" content_version="1.0"
          xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
          xmlns="urn:gba:sqg"
          xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
          xmlns:xenc="http://www.w3.org/2001/04/xmlenc#>
</root>
```

3.4.2 Wurzelelement <root>

Das Root-Element ist eine Art Umschlag oder Wurzelelement für alle XML-Typen in den PB-Verfahren. Das Wurzelelement besteht immer aus zwei Kind-Elementen (Zweige) <header> und <body>.

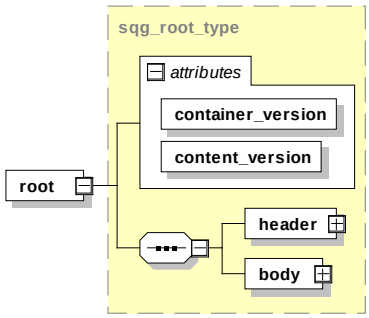


Abbildung 26: Root-Element und Kind-Elemente header und body

Das Root-Element hat zusätzlich zwei Attribute (Tabelle 64).

Tabelle 64: Root-Element – Attribute

Name	Type	Use	Fixed	Beschreibung
container_version	xs:string	required	2.0	Ist ein fixer Wert und definiert die aktuell gültige Versionsnummer des Containers. Die Versionsnummer wird erhöht, wenn Änderungen am Schema des Containers (Umschlags) gemacht werden. Bei kleinen optionalen Änderungen wird die Versionsnummer beibehalten, um die Aufwärtskompatibilität zu gewährleisten. Ein XML-Dokument, das einen alten Wert dieses Attributs enthält, muss von der Datenannahmestelle zurückgewiesen werden.
content_version	content_version_Datentyp	required	1.0	Ist ein fixer Wert und definiert die aktuell gültige Versionsnummer des Inhalts der PB-Daten. Die Versionsnummer wird erhöht, wenn unterjährig das Schema unabhängig von der zugrundliegenden Spezifikationsdatenbank geändert wird.

3.4.3 Header-Bereich

Element header

Das Element Header besteht aus Metadaten (administrative und meldebezogene Daten) zu den PB-Daten, die im <body> enthalten sind.

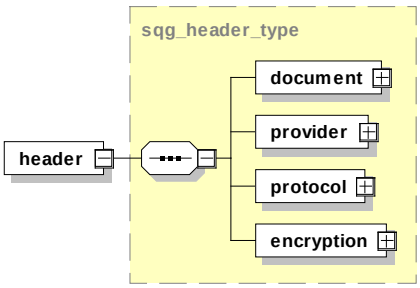


Abbildung 27: Aufbau des Elements header

Element header/document

Das Element enthält allgemeine Informationen zum erstellten Dokument. Dieses Element ist weitestgehend über den gesamten Datenfluss hinweg beständig. Nur das Element <modification_dttm> (Modifikationsdatum) wird bei jeder Bearbeitung neu gesetzt.

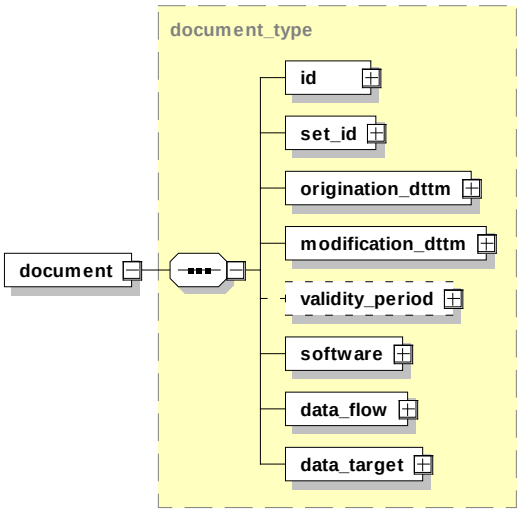


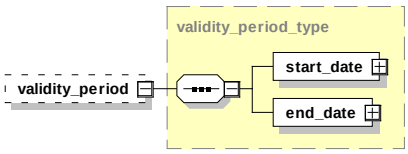
Abbildung 28: Aufbau des Elements document

Dieses Element hat weitere Kind-Elemente, die in der folgenden Tabelle beschrieben werden.

Tabelle 65: Kind-Elemente des Elements document

Kind-Elemente	Beschreibung
<id>	Nach Erstellung nicht mehr modifizierbar. Eindeutige ID des Dokuments, wird vom Dokumentenersteller als GUID erzeugt.[65]
<set_id>	Nach Erstellung nicht mehr modifizierbar. Eindeutige ID für mehrere Dokumente, die im selben Zusammenhang stehen; wird vom Leistungserbringer erzeugt. Dafür könnte z. B. die GUID vom ersten Dokument des Zusammenhangs verwendet werden.
<origination_dttm>	Das Element ist der Zeitstempel der ursprünglichen Dokumenterzeugung. Darf nach seiner Erstellung nicht mehr modifiziert werden. Format: CCYY-MM-DDThh:mm:ss
<das_receive_dttm>	Ist der Zeitstempel des Dateneingangs nach Übermittlung von LE an DAS. Nach Einfügen durch die DAS nicht mehr modifizierbar. Format: CCYY-MM-DDThh:mm:ss

[65] Ein Globally Unique Identifier oder kurz GUID ist eine global eindeutige Zahl mit 128 Bit (16 Bytes), die ein Dokument weltweit eindeutig identifiziert.

Kind-Elemente	Beschreibung
`<modification_dttm>`	Dieses Element ist ein Zeitstempel und muss bei jeder Modifikation des Dokuments aktualisiert werden. Das Modifikationsdatum darf nicht vor dem Erstelldatum liegen. Format: CCYY-MM-DDThh:mm:ss
`<validity_period>`	 Mithilfe des optionalen <validity_period>-Elements wird der Bezugszeitraum der Datei angegeben. Dies kann ein Jahr oder ein Quartal oder ein frei wählbarer Bereich sein. Zur Qualifizierung sind die Felder <start_date> und <end_date> entsprechend zu füllen.

`<origination_dttm>` und `<modification_dttm>` sind vom Datentyp `dateTime`, der einen Zeitpunkt darstellt (ISO 8601). Es handelt sich um das Format CCYY-MM-DDThh:mm:ss:

- „CC“ steht für das Jahrhundert,
- „YY“ steht für das Jahr,
- „MM“ steht für den Monat und
- „DD“ für den Tag.
- Der Buchstabe „T“ dient als Trennzeichen zwischen Datum und Zeit.
- „hh“, „mm“ und „ss“ repräsentieren jeweils Stunden, Minuten und Sekunden.

Dieser Darstellung kann direkt ein „Z“ nachgestellt werden, um anzuzeigen, dass es sich um die Universal Time Coordinated (UTC) handelt. Folgt der Zeitangabe statt eines „Z“ ein Plus- oder Minuszeichen bedeutet das, dass die darauffolgende Angabe im Format „hh:mm“ die Differenz zur UTC angibt (der Minutenanteil ist erforderlich).

Beispiele:

- `2011-11-01T21:32:52`
- `2011-11-01T21:32:52+02:00` (Zeitzonendifferenz von plus 2 Stunden)
- `2011-11-01T19:32:52Z`

Element header/document/software

Sammelelement für Angaben zur eingesetzten PB-Software.

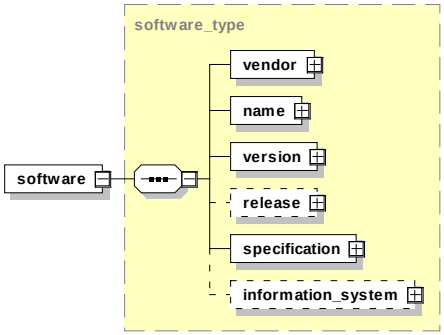


Abbildung 29: Aufbau des Elements software

Dieses Element enthält Kind-Elemente, die in der folgenden Tabelle beschrieben werden.

Tabelle 66: Kind-Elemente des Elements software

Kind-Elemente	Beschreibung
<vendor>	Enthält Informationen über den Softwarehersteller
<name>	Enthält den Softwarenamen der eingesetzten Software
<version>	Enthält die Version der eingesetzten Software
<release>	Enthält das Release der eingesetzten Software
<specification>	Enthält einen Wert aus der enumeration enum_spez_type: Version der Spezifikation, auf deren Basis die PB-Dokumentationssoftware entwickelt wurde
<information_system>	Enthält Angaben zum eingesetzten Informationssystem (AIS/LIS).

Element header/document/software/information_system

Sammelelement für Angaben zum eingesetzten Informationssystem (AIS/LIS).

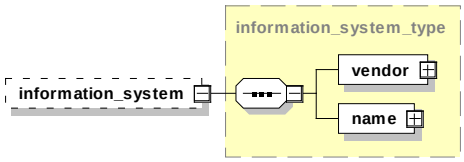


Abbildung 30: Aufbau des Elements information_system

Tabelle 67: Kind-Elemente des Elements information_system

Kind-Elemente	Beschreibung
<vendor>	Enthält Informationen über den Softwarehersteller.
<name>	Enthält den Softwarenamen der eingesetzten Software.

Dem Attribut //software/vendor/@registration wird im ambulanten Bereich die KBV-Prüfnummer der eingesetzten QS-Software oder der Dummy-Wert „000“ zugewiesen.

Element header/document/data_flow

Dieses Element gibt an, für welchen Datenfluss (Datenannahmestelle) dieses Dokument erzeugt wurde.

Tabelle 68: Angabe des betreffenden Datenflusses

Ausprägung	Beschreibung
PB-Verfahren kollektivvertraglich	für die Verfahren, die an die kassenärztlichen Vereinigungen (KVen) übermittelt werden müssen

Element header/document/data_target

Dieses Element gibt an, welches Ziel der Datenfluss hat.

Tabelle 69: Angabe des betreffenden Datenfluss-Ziels

Datenfluss	Ziel
Echtdatenpool	Echtdaten für den Echtbetrieb
Probedatenpool	Echtdaten für vorläufige Auswertungen wie der Sonderexport
Testdatenpool	Testdaten für Testzwecke

Achtung
Datenverlust

Nur Daten mit der Kennzeichnung „Echtdatenpool“ werden für die Erstellung der Auswertungen berücksichtigt.

Element header/provider

Das Element `<provider>` gibt an, welche Institution dieses Dokument zuletzt bearbeitet hat. Es wird in jeder am Datenfluss beteiligten Instanz durch diese ersetzt und so zur nächsten Instanz geschickt. Einzige Ausnahme bildet die Vertrauensstelle. Diese lässt das Provider-Element unberührt.

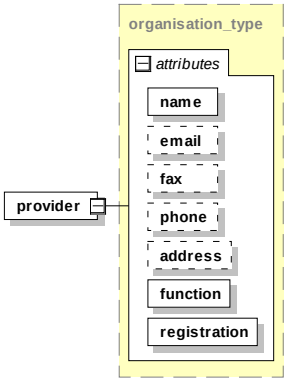


Abbildung 31: Aufbau des Elements provider

Tabelle 70: Attribute des Elements header/provider

Name	Type	Use	Beschreibung
`name`	`xs:string`	required	Name der Institution
`email`	`emailAddress_type`	optional	E-Mail-Adresse
`fax`	`xs:string`	optional	Faxnummer
`phone`	`xs:string`	optional	Telefonnummer
`address`	`xs:string`	optional	Adresse
`function`	`enum_organisa-tion_type`	required	Auswertestelle/Datenannahmestelle Vertrauensstelle/Softwarehersteller/ Undefined
`registration`	`registration_type`	required	Registrierungsnummer. Sollte für die Datenübertragung an die DAS (KV) keine Registrierungsnummer erforderlich sein, kann dieses Element für den Datenfluss vom ambulanten Leistungserbringer an die KV weggelassen werden. Für alle folgenden Datenflüsse ist die Registriernummer eine Pflichtangabe.

Element header/protocol

Das Element <protocol> nimmt Informationen zu Prüfungen auf, die im Datenfluss durchgeführt wurden. Es ist Teil der Rückprotokollierung. Dieses Element ist nicht optional und soll gemeinsam mit dem Unterelement <status_document> von Anfang an im Datenfluss vorhanden sein, um nachfolgende im Datenfluss vorgenommene Prüfergebnisse aufzunehmen.

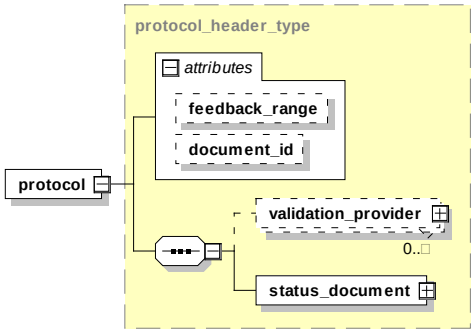


Abbildung 32: Aufbau des Elements header/protocol

Dieses Element hat zusätzlich zu den optionalen Attributen `feedback_range` und `document_id` zwei Kind-Elemente:

`<validation_provider>` und `<status_document>`

Tabelle 71: Attribute des Elements header/protocol

Name	Type	Use	Beschreibung
`feedback_range`		optional	Da die Transaktionsprotokolle durch die Empfangsbestätigungen ersetzt wurden, ist nur der Wert „`dataflow`“ zu verwenden.
`document_id`		optional	Soweit die GUID des Exportdokuments lesbar ist, muss sie in das Attribut `document_id` eingetragen werden.

Element header/protocol/validation_provider

Hier gibt sich die Stelle zu erkennen, die einen oder mehrere Prüfungsschritte durchgeführt hat. Die Ergebnisse der Prüfung werden in diesem Container abgelegt und werden Teil der Rückprotokollierung.

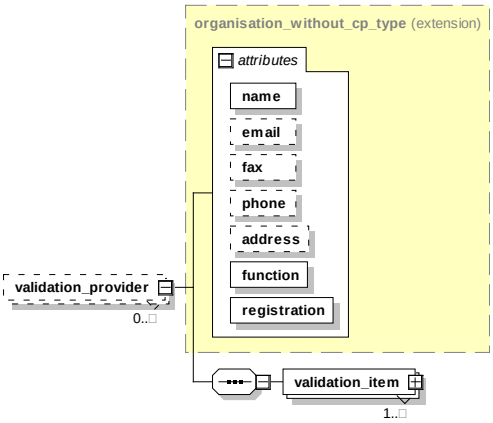


Abbildung 33: Aufbau und Kind-Elemente des Elements validation_provider

Tabelle 72: Attribute des Elements validation_provider

Name	Type	Use
name	xs:string	required
email	emailAddress_type	optional
fax	xs:string	optional
phone	xs:string	optional
address	xs:string	optional
function	enum_validation_provider_type	required
registration	registration_type	required

Element header/protocol/validation_provider/validation_item

Auf Dokumentenebene sind alle Prüfungen zu dokumentieren. Eine prüfende Einrichtung trägt sich als <validation_provider> in die entsprechende Auflistung ein und dokumentiert dann ihre durchgeführten Prüfungen in der Auflistung <validation_item>.

Es wird als Ergebnis jeder Prüfung eine der folgenden Aussagen über das geprüfte Objekt getroffen:

- OK (Keine Auffälligkeiten)
- WARNING (Auffälligkeiten, die einer Weiterverarbeitung nicht im Weg stehen)
- ERROR (Auffälligkeiten bzw. Fehler, die eine Weiterverarbeitung des Datensatzes oder des Dokumentes ausschließen)

Das Ergebnis der Prüfung wird in das Attribut @V des Elements <status> im Element <validation_item> eingetragen.

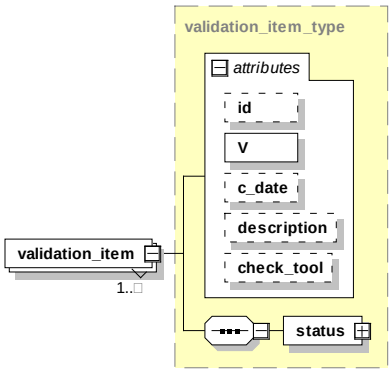


Abbildung 34: Aufbau und Kind-Elemente des Elements validation_item

Tabelle 73: Attribute des Elements validation_item

Name	Type	Beschreibung
id	xs:int	Diese ID ist dokumentweit gültig und darf im Header nur einmal vorkommen. Prüfungen auf Datensatzebene (Element <case>), die zu dieser Prüfung gehören, werden über diese ID zugeordnet. Die ID muss nur dann vergeben werden, wenn eine Prüfung auf Fallebene stattfindet.
V	enum_validation_type	Dieser Wert bezeichnet die durchgeführte Prüfung anhand einer „enumeration“, die in sqg_protocol.xsd definiert wird. Gültige Werte sind: Dechiffrierung, LE_Pseudonym, PID_Pseudonym, Protokoll, Schema, Spezifikation, Transaktion, sonstige Prüfung.
c_date	xs:dateTime	Hier kann ein Zeitstempel für die Verarbeitung angegeben werden.
description	xs:string	Prüfungsbeschreibung laut Spalte „Prüfung“ in Sicht vPruefung in PBDOK-Datenbank
check_tool	xs:string	Versionsnummer des Prüftools (z. B. das Datenprüfprogramm)

Attribut /protocol/validation_provider/validation_item/@check_tool

Hier können beim Einsatz eines Tools für die Prüfung der XML-Dateien der Name und die Versionsnummer des Tools hinterlassen werden (beim Einsatz des Datenprüfprogramms wird die Versionsnummer des XSLT-Skripts eingetragen).

Das Datenprüfprogramm trägt automatisch die Versionsnummer in das <validation_item>-Element ein. Damit das Protokoll nicht unnötig groß wird, wird die Information über das Tool nur auf Dokumentenebene aufgenommen (header/protocol/validation_provider/validation_item/).

Element header/protocol/status_document

Hier wird der Gesamtstatus des Dokuments angegeben, das Attribut V kann also auf OK, WARNING oder ERROR stehen. Dieser Status kann nur geändert werden, wenn sich der Status des Dokuments verschlechtert oder gleichbleibt. ERROR bedeutet, dass das Dokument komplett zurückgewiesen werden muss.

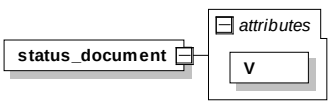


Abbildung 35: Aufbau des Elements status_document

Tabelle 74: Attribute des Elements status_document

Name	Type	Use	Beschreibung
V	`status_type`	required	Mögliche Werte: OK/WARNING/ERROR

Element header/protocol/validation_provider/validation_item/status

Das Element gibt an, ob die betroffene Testeinheit ohne Fehler (OK), mit Fehlern (WARNING) oder mit fatalem Fehler (ERROR) abgeschlossen wurde. Der Gesamtstatus des Dokuments entspricht jeweils dem schlechtesten Prüfergebnis. Bei der ersten Prüfung mit dem Ergebnis ERROR muss die Weiterverarbeitung abgebrochen werden.

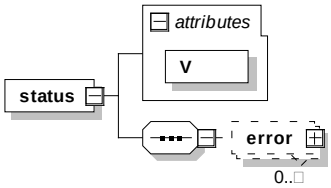


Abbildung 36: Aufbau und Kind-Elemente des Elements status

Tabelle 75: Attribut des Elements status

Name	Type	Use	Beschreibung
V	`status_type`	required	Status einer Prüfung mit folgenden, möglichen Werten: OK, WARNING oder ERROR.

Darüber hinaus gibt es die Möglichkeit, eine beliebige Anzahl von `<error>`-Elementen mit einer `<error_message>` im `<status>`-Element unterzubringen.

Element header/protocol/validation_provider/validation_item/status/error

Ein `<error>`-Element nimmt Fehlerdaten auf. Als einzig verpflichtendes Unterelement gilt das `<error_message>`-Element. Die Elemente `<rule_id>` und `<rule_type>` sind spezifisch für die Anwendung von Plausibilitätsregeln für die Spezifikation:

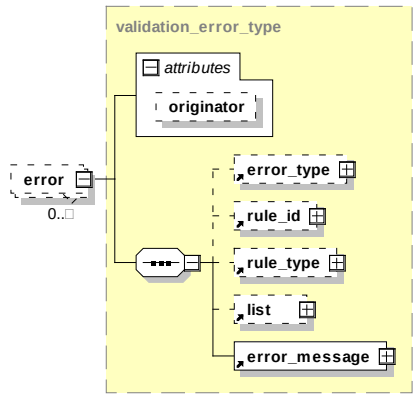


Abbildung 37: Aufbau des Elements error

Tabelle 76: Attribut des Elements error

Name	Type	Use	Beschreibung
originator	enum_organisation_type	optional	Mögliche Werte: Auswertestelle, Datenannahmestelle, Vertrauensstelle, Leistungserbringer, Softwarehersteller, undefined

Tabelle 77: Kind-Elemente des Elements error

Kind-Elemente	Beschreibung
<rule_id>	Nummer der Regel (idRegeln in Tabelle Regeln) oder Nummer der Fehlermeldung aus der Tabelle Fehlermeldung (idFehlermeldung)
<rule_type>	Werte H (=hart) oder W (=Warnung bzw. weich)
<liste>	In Abhängigkeit von der Fehlerart entweder Liste von Teildatensätzen oder von Bogenfeldern
<error_message>	Fehlermeldung als Freitext
<error_type>	Hat folgende Ausprägungen[66]: EXPORT = Formatfehler der Exportdatei DOPPELT = bereits vorhandener Datensatz wird erneut übermittelt TDS = Vollständigkeit und Version der Teildatensätze WERT = Wertebereichsverletzung REGEL = Plausibilitätsverletzung KOLLISION = Patientenpseudonym mit unterschiedlichen Alters-/Geschlechtsangaben LE = LE nicht entschlüsselbar FEHLT = Angabe fehltPID = PID nicht entschlüsselbar (in Kombination mit Dechiffrierung von „validation_item“) QS = PB-Daten nicht entschlüsselbar (in Kombination mit Dechiffrierung von „validation_item“)

[66] Die Ausprägung STEUER (= Formatfehler der Steuerdatei) wurde im Rahmen eines csv-Exports eingeführt und wird seit Überführung in das XML-Format nicht mehr verwendet

Element header/encryption

Das Element nimmt Informationen über den Schlüssel auf, mit dem die Daten verschlüsselt worden sind. Das Attribut `id` enthält den Namen des symmetrisch verschlüsselten XML-Knotens. Eine Beispielimplementierung dieser Spezifikation ist ein öffentliches Verschlüsselungsprogramm des IQTIG (XPacker).

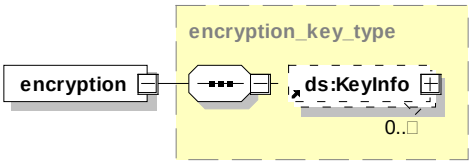


Abbildung 38: Aufbau und Attribute des Elements encryption

Das Programm dient zur Ver- und Entschlüsselung einzelner XML-Elemente (Tags) innerhalb einer XML-Datei, basierend auf einem hybriden Verfahren, das aus folgenden Einzelschritten besteht:

- Ein zufälliger symmetrischer Schlüssel wird erzeugt.
- Mit diesem Schlüssel wird ein XML-Element (z. B. `qs_data`) chiffriert.
- Der Schlüssel wird nun mit dem „public Key“ des Empfängers (KV) verschlüsselt.
- Der mit dem „public Key“ chiffrierte symmetrische Schlüssel wird dem Empfänger zusammen mit den verschlüsselten Daten übergeben.
- Der Empfänger dechiffriert den Schlüssel mit seinem „private Key“ und erhält so den symmetrischen Schlüssel.
- Mit diesem symmetrischen Schlüssel dechiffriert der Empfänger die verschlüsselten Daten.

Weitere Informationen sind der Dokumentation des Verschlüsselungsprogramms (Abschnitt B 4.3.1) zu entnehmen.

3.4.4 Body-Bereich

Im `<body>`-Element liegen die eigentlichen PID, PB- und LE-Daten. Der Body-Bereich kann einen oder mehrere `<data_container>` enthalten, die einem bestimmten Leistungserbringer zugeordnet sind.

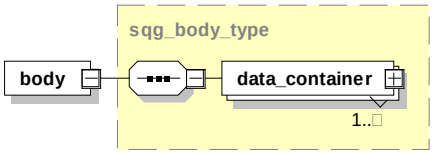


Abbildung 39: Aufbau des Elements body

Kind-Element body/data_container

Ein <data_container> ist einem bestimmten Leistungserbringer zuzuordnen. In der Regel sollte in einem Dokument nur ein <data_container> vorhanden sein. Da aber mehr als ein <data_container> erlaubt ist, können ggf. auch mehrere <data_container> für mehrere Leistungserbringer verwendet werden, wenn das Dokument z. B. von einer Stelle (z. B. einer Datenannahmestelle) erstellt wird, die Daten mehrerer Leistungserbringer sammelt.

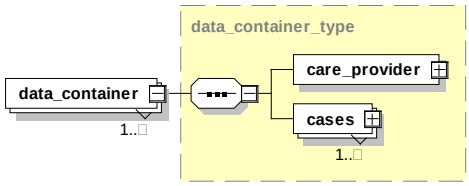


Abbildung 40: Aufbau des Elements body/data_container

Element body/data_container/care_provider

Die Zuordnung zu einem Leistungserbringer erfolgt durch das Element care_provider. Dies erfolgt in Abhängigkeit zum Status des Leistungserbringers (vertragsärztliche Praxis/med. Labor) durch einen unterschiedlichen Aufbau. Die Inhalte des <care_provider>-Elements ergeben sich aus Anwendung der Abfrage vExportZieleXml (Abschnitt 2.7.1) der PBDOK-Datenbank.

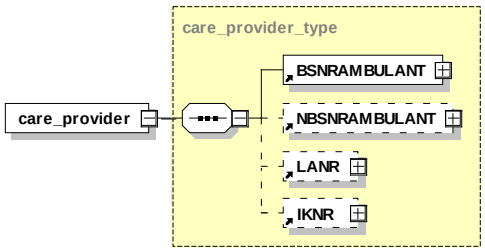


Abbildung 41: Aufbau des Elements care_provider – kollektivvertraglich

Die Kind-Elemente für die vertragsärztliche Praxis/med. Labore werden in der folgenden Tabelle beschrieben.

Tabelle 78: Leistungserbringeridentifizierende Daten im kollektiven Bereich

Kind-Elemente	Beschreibung
Leistungserbringeridentifizierende Daten im kollektiven Bereich	
<LANR>	Lebenslange Arztnummer – LANR. Für die persönliche Kennzeichnung seiner Leistungen hat jeder Vertragsarzt und -psychotherapeut zum 1. Juli 2008 eine „Lebenslange Arztnummer“ (LANR) erhalten. Diese muss er bei jeder von ihm abgerechneten Leistung und Verordnung angeben.
<BSNRAMBULANT>	Betriebsstättennummer ambulant – BSNR

Kind-Elemente	Beschreibung
	Die BSNR identifiziert die Arztpraxis als abrechnende Einheit und ermöglicht die Zuordnung ärztlicher Leistungen zum Ort der Leistungserbringung. Dabei umfasst der Begriff Arztpraxis auch Medizinische Versorgungszentren (MVZ), Institute, Notfallambulanzen sowie Ermächtigungen von am Krankenhaus beschäftigten Ärzten.
<NBSNRAMBULANT>	Nebenbetriebsstättennummer ambulant – NBSNR

Element body/data_container/cases

Container-Element für eine Liste von gleichartigen Fällen (Vorgängen). „Gleichartig“ meint hier Fälle des gleichen Primärmoduls. Das Element enthält einen oder mehrere Vorgänge[67].

Für unterschiedliche Module müssen jeweils mehrere <cases> angelegt werden. Die Ausweisung eines <cases>-Elements für Daten eines bestimmten Primärmoduls erfolgt über dessen Attribut module.

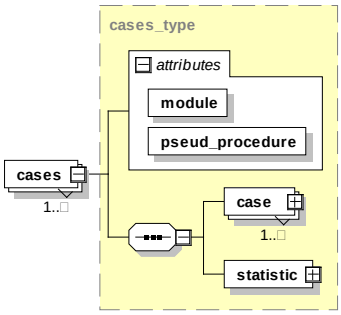


Abbildung 42: Aufbau des Elements cases

Tabelle 79: Attribute des Elements cases

Name	Type	Use	Beschreibung
module	enum_module_type	required	Das Erfassungsmodul
pseud_procedure	enum_procedure_type	required	Zuordnung des Moduls zu einem Pseudonymisierungsverfahren. Gehört dem Modul kein Pseudonymisierungsverfahren, ist das Attribut auf „undefined“ zu setzen.

Das Element <cases> enthält das Attribut „pseud_procedure“: „pseud_procedure“ ist eine Verfahrenskennung, die eindeutig zusammengehörende Exportmodule vermerkt und die es der Vertrauensstelle ermöglicht, die PID verfahrensbezogen zu pseudonymisieren.

[67] Fälle und Vorgänge werden hier als Synonyme verwendet.

Tabelle 80: Verfahrenskennung: „pseud_procedure“

Betrieb	Exportmodul	Verfahrenskennung	XML (Umsetzung)
oKFE	DKK DKI	DK	`<cases module="DKK" pseud_procedure="DK">` `<cases module="DKI" pseud_procedure="DK">`

oKFE	ZKP
	ZKA
	ZKZ
	ZKH
		ZK	`<cases module="ZKP" pseud_procedure="ZK">`
			`<cases module="ZKA" pseud_procedure="ZK">`
			`<cases module="ZKZ" pseud_procedure="ZK">`
			`<cases module="ZKH" pseud_procedure="ZK">`

Regelbetrieb Testbetrieb	Nicht-PID	undefined	`<cases module="PNEU" pseud_procedure="undefined">`

Element body/data_container/cases/case

Das Element `<case>` entspricht einem Vorgang und enthält genau einen PB-Datensatz eines Moduls und die patientenidentifizierenden Daten “`<patient>`“.

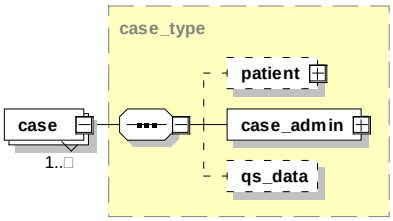


Abbildung 43: Aufbau des Elements case

Element body/data_container/case/case_admin

Das `<case_admin>`-Element enthält weitere Elemente, die einen Vorgang identifizieren. Zusätzlich legt das Element fest, was mit dem Vorgang geschehen soll. Auf Vorgangsebene (Datensatzebene) werden von jeder Prüfstelle der Status der Prüfung und ggf. die Fehler in das Element `<protocol>` eingetragen.

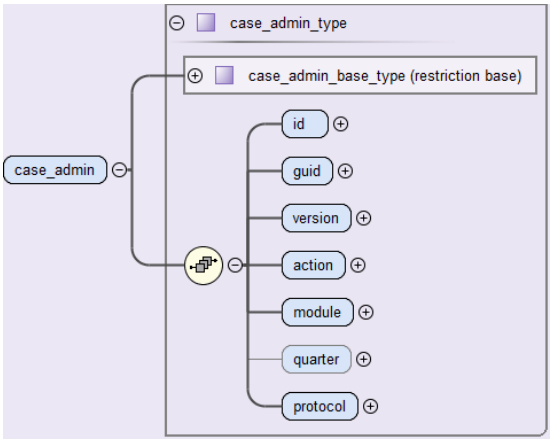


Abbildung 44: Aufbau des Elements case_admin

Im Folgenden werden die einzelnen Kind-Elemente beschrieben.

Tabelle 81: Kind-Elemente des Elements case_admin

Kind-Elemente	Beschreibung
<id>	Vorgangsnummer oder Datensatznummer. Diese Nummer kennzeichnet jeden dokumentierten Datensatz eines Dokumentationssystems eindeutig und zwar unabhängig vom angewandten Modul. Im einfachsten Fall könnte also die Vorgangsnummer um 1 erhöht werden, wenn ein neuer Datensatz angelegt wird. Insbesondere ist es falsch, einfach eine Patientenidentifikationsnummer oder die offizielle Fallnummer zu verwenden bzw. zu pseudonymisieren. Bei der Umsetzung hat der Softwareanbieter weitgehende Freiheit, vorausgesetzt die modulübergreifende Eindeutigkeit der Vorgangsnummer ist gewährleistet. Die Vorgangsnummer darf für die Datenannahmestelle nicht auf Personen zu beziehen sein.
<guid>	36-stelliger pseudozufälliger Globally Unique Identifier (GUID) zur eindeutigen bundesweiten Identifizierung des Datensatzes. Die GUID lässt keine Rückschlüsse auf den Leistungserbringer oder den Patienten zu.
<version>	Enthält eine Versionsnummer des Datensatzes. Sie gibt an, die wievielte Version des Datensatzes übertragen wird. In der Regel wird die Versionsnummer 1 lauten. D. h., dass der nach dem ersten Dokumentationsabschluss freigegebene Datensatz übertragen wird. Muss ein korrigierter Datensatz erneut eingesandt werden, so muss die Versionsnummer vom dokumentierenden System um 1 erhöht werden. Die neue Version des Datensatzes wird bei der Entgegennahme geprüft und überschreibt bei Korrektheit die alte Version des Datensatzes. Wenn die Datenannahmestelle einen Datensatz mit derselben Versionsnummer ein zweites Mal erhält, so wird dieser zurückgewiesen.
<module>	Verfahrensabkürzung. Hier ist zu beachten, dass dieser Wert identisch zu den Attributwerten im Element <cases> und <qs_data> sein muss.

Kind-Elemente	Beschreibung
	Wenn die Datenannahmestelle unterschiedliche Modulbezeichnungen innerhalb eines <cases>-Elements erhält, wird die ganze Datenlieferung zurückgewiesen.
<quarter>	Dieses Element ist optional und ermöglicht eine fallbezogene Quartalsangabe. Hiermit werden DAS (KV), welche keinen Zugriff auf die PB-Daten haben in die Lage versetzt quartalsbezogene Ist-Werte zu ermitteln.
<protocol>	Protokoll auf Vorgangsebene
<action>	Definiert die gewünschte Aktion, kann „create“, „update“ und „delete“ sein. „create“ ist beim ersten Export des Datensatzes zu verwenden, weitere Exporte des Datensatzes werden mit „update“ geliefert. Da nicht alle Datenexporte auch an die Datenannahmestelle verschickt werden (z. B. Testexporte usw.), muss die Datenannahmestelle „update“ und „create“ gleichbehandeln, wenn der erhaltene Datensatz nicht bereits im Datenpool vorhanden ist. Um den Datensatz zu stornieren, muss <action> auf „delete“ gesetzt werden. Die Datenannahmestelle wird dadurch veranlasst, den betreffenden Datensatz einschließlich aller Vorversionen und Teildatensätze als „storniert“ zu kennzeichnen. Der Stornovorgang wird in der Datenbestätigung protokolliert. Der zu stornierende Datensatz muss ebenfalls eine hochgezählte/fortgeschriebene Versionsnummer enthalten, um die Stornierung unabhängig von der Reihenfolge der Verarbeitung von Datensätzen sicherzustellen. Ein Storno mit einer bereits verwendeten Versionsnummer wird zurückgewiesen (Bestätigungsstatus ERROR, Fehlerart DOPPELT). Ein Stornoversuch eines noch nicht übermittelten Datensatzes wird ebenfalls zurückgewiesen. Zur Stornierung eines Datensatzes (Vorgang) genügt der Export der entsprechenden administrativen Daten <case>/<case_admin>. Sowohl die PID (<patient>) als auch die PB-Daten (<qs_data>) des zu stornierenden Datensatzes sind nicht erneut zu übermitteln.

Element patient (PID-Module)

Das Element enthält die patientenidentifizierenden Daten des übergeordneten Vorgangs. Das Kind-Element von <patient> ist das Element <pid> Die Inhalte des <patient>-Elements ergeben sich aus Anwendung der Abfrage vExportZieleXml (Abschnitt B 2.7.1) in der PBDOK-Datenbank. Das Attribut twodigitik ist verpflichtend und muss die ersten 2 Stellen des Institutionskennzeichens der Krankenkasse enthalten. Das Attribut ist nicht von der Verschlüsselung betroffen.

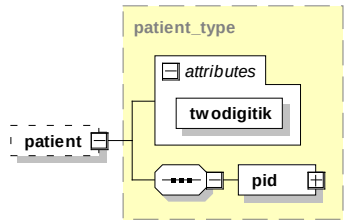


Abbildung 45: Aufbau des Elements patient

Achtung
Umgang mit PID-Daten

PB-Verfahren:

Für die PB-Verfahren gemäß oKFE-RL werden sowohl die PID- als auch die PB-Doku-Daten von nicht GKV-Versicherten bereits bei der Auslösung ausgeschlossen. Gemäß oKFE-RL können keine Verfahren entwickelt werden, die auch den Einschluss von nicht GKV-Versicherte ermöglichen.

Richtlinienübergreifend gilt:

Bei einer Stornierung ist das gesamte <patient>-Element wegzulassen.

Element patient_type/pid

Das Element <pid> dient dazu, die tatsächlichen PID aufzunehmen.

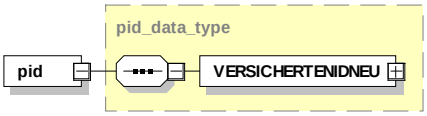


Abbildung 46: Aufbau des Elements pid

Diese PID bestehen aus dem Element <VERSICHERTENIDNEU>, welches die eGK-Versichertennummer enthält.

Element case_Datentyp/case_admin/protocol

Dieses Element hat eine auf Dokumentenebene ähnliche Struktur wie das oben beschriebene Element <protocol>.

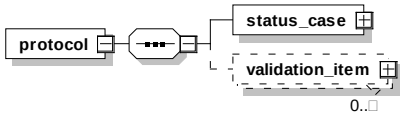


Abbildung 47: Aufbau des Elements case_admin/protocol

Die Unterschiede sind:

- Während das <protocol>-Element im Header Ergebnisse der Prüfungen, die das Dokument insgesamt betreffen, aufnimmt, nimmt das Protokoll-Element im Body-Bereich die Ergebnisse der Prüfungen auf, die auf Vorgangsebene (Datensatzebene) erfolgen.
- Für erfolgreiche Prüfergebnisse (status="OK") wird nicht explizit das Element <validation_item> erstellt. Für die Übermittlung des Status des Datensatzes dient weiterhin der implizite Wert des Elements <status_case> (<status_case V="OK">).
- <status_case> beinhaltet das schlechteste Ergebnis aller Prüfungen eines Datensatzes.
- Das Protokoll auf Vorgangsebene hat kein Element <validation_provider> (Prüfstelle). Damit auch auf dieser Ebene die Ergebnisse der durchgeführten Prüfungen einer Prüfstelle zugeordnet werden können, müssen alle Ergebnisse einer Prüfung auf Fallebene mit einer gemeinsamen, dokumentweit eindeutigen ID im Attribut ID des Elements <validation_item> eingetragen werden.

Element body/data_container/cases/statistic

Das Element <statistic> dient dazu, Statistiken über die Datenlieferung des Absenders und über deren Verarbeitung durch die Datenannahmestelle aufzunehmen.

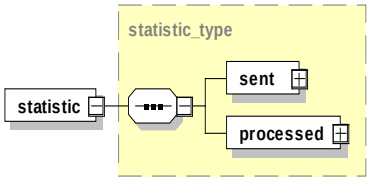


Abbildung 48: Aufbau des Elements statistic

Es ist nach der Prüfung bei der Datenannahmestelle ein Teil des Rückprotokolls und besteht aus ähnlichen Kind-Elementen.

Tabelle 82: Kind-Element des Elements statistic

Kind-Element	Beschreibung
<sent>	Statistik über die von dem Datenlieferanten exportierten Datensätze. Es muss daher vom Datenlieferanten selbst ausgefüllt werden.
<processed>	Hat dieselbe Struktur wie <sent> und enthält das Ergebnis der Verarbeitung durch die Datenannahmestelle.

Element statistic/sent

Das Element nimmt Statistiken über die von dem Datenlieferanten exportierten Datensätze auf und muss vom Datenlieferanten selbst ausgefüllt werden.

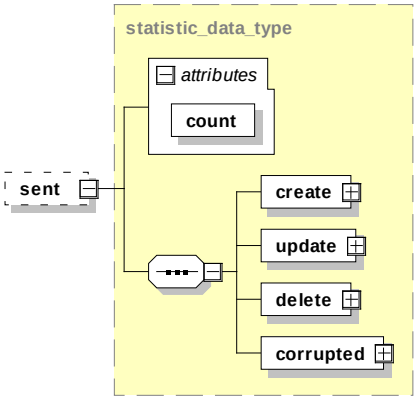


Abbildung 49: Aufbau des Elements sent

Es besteht aus vier Elementen und einem Attribut.

Tabelle 83: Attribut des Elements sent

Name	Type	Use	Beschreibung
count	non_negative_integer_type	required	Gesamtzahl von Vorgängen (Ganze Zahl >= 0): Summe von (<create>+<update>+ <delete>+<corrupted>)

Tabelle 84: Kind-Elemente des Elements statistic/sent

Kind-Element	Beschreibung
<create>	Anzahl der Datensätze, die neu importiert werden sollen.
<update>	Anzahl der Datensätze, die aktualisiert werden sollen (z. B. nach einer Korrektur).
<delete>	Anzahl der Datensätze, die von der Datenannahmestelle/Auswertungsstelle storniert werden müssen.

Kind-Element	Beschreibung
<corrupted>	Anzahl der Datensätze, die fehlerhaft sind. Der Datenabsender trägt hier „0“ ein.

Element statistic/processed

Das Element <processed> hat dieselbe Struktur wie das Element <sent> mit dem Unterschied, dass der Datenempfänger nach der Prüfung der Exportdatei in das Element <processed> eintragen soll, wie viele Datensätze er tatsächlich neu importiert, überschrieben und storniert hat und ggf. wie viele Datensätze fehlerhaft sind. Außerdem soll er im Attribut count des Elements <processed> die Gesamtsumme eintragen.

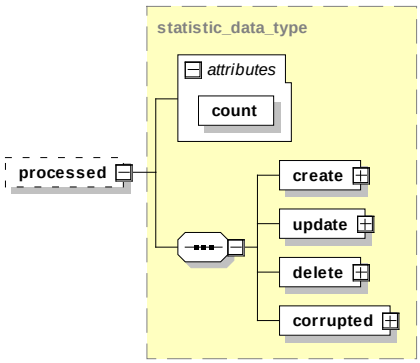


Abbildung 50: Aufbau des Elements processed

<processed> ist vom Datenlieferanten anzulegen und besteht aus vier Elementen und einem Attribut.

Tabelle 85: Attribute des Elements statistic

Name	Type	Use	Beschreibung
count	non_negative_integer_type	required	Gesamtzahl von Vorgängen (Datensätzen): Summe von (<create>+<update>+<delete>+<corrupted>)

Tabelle 86: Kind-Elemente des Elements statistic/processed

Kind-Element	Beschreibung
<create>	Anzahl der Datensätze, die der Datenempfänger nach der Prüfung tatsächlich neu importieren konnte.
<update>	Anzahl der Datensätze, die der Datenempfänger nach der Prüfung tatsächlich aktualisieren konnte.
<delete>	Anzahl der Datensätze, die der Datenempfänger tatsächlich stornieren konnte.

`<corrupted>`	Anzahl der Datensätze, die wegen Mängeln nicht entsprechend der Absicht des Datenlieferanten in den Datenpool übernommen werden konnten.

Alle Werte müssen vom Leistungserbringer mit der Zahl „0“ vorbelegt werden.

Zusammenfassend kann man die Angaben im Element `<sent>` als Absichtserklärung der Datenlieferung betrachten. Die Angaben im Element `<processed>` sind als Ergebnis der Verarbeitung in Bezug auf den Zieldatenpool zu verstehen und werden daher nur von der Datenannahmestelle gesetzt.

Element qs_data

Das `<qs_data>`-Element ist ein Container für die PB-Daten, die verfahrensspezifisch sind.

Der Datentyp der konkreten `<qs_data>`-Instanz ist jeweils als Attribut "`xsi:type`" bei der Dokumenterstellung zu definieren.

```
<qs_data xsi:type="qs_data_17n2_type" module="17/2">
```

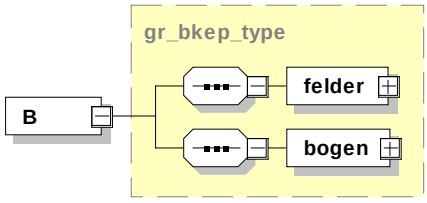
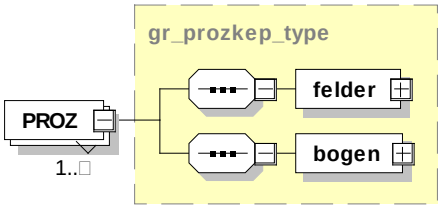
Hinweis

Liegt in der Vertrauensstelle ein Widerspruch des Patienten vor, muss die Vertrauensstelle die verschlüsselten Programmbeurteilungsdaten für diesen Patienten löschen. Um den Widerspruch im Datensatz kenntlich zu machen, ersetzt die Vertrauensstelle zusätzlich den ursprünglichen Typ im XML-Element `qs_data` mit `qs_data_widerspruch_type`. Der ursprüngliche Modulname im Attribut `qs-data/@module` bleibt unverändert.

qs_data-Beispiel bei Widerspruch:

<qs_data module="ZKA" xsi:type="qs_data_widerspruch_type"/>

Folgende Abbildungen zeigen einen komplexen und einen einfachen Bogen.

Bogen komplex	*Ein komplexer Bogen (B) mit Kinderbögen:*	
	Ein Kindbogen (PROZ) kann wiederum Kinderbögen besitzen:	

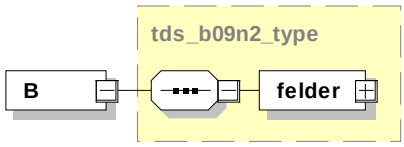
Bogen einfach	*Ein einfacher Bogen (Basisbogen) ohne Kinderbögen:*	

Abbildung 51: Diagramme „Bogen komplex“ und „Bogen einfach“

Die XML-Schemata der einzelnen Module sind in der Schnittstelle „interface_LE“ zu finden:

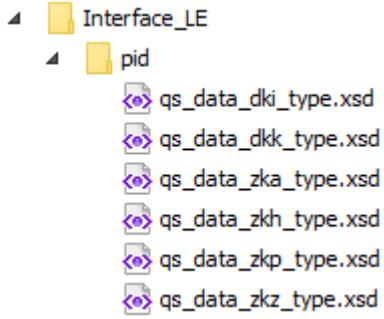


Abbildung 52: Ausprägungen des qs_data-Elements (Erfassungsmodule)

4 Tools

Das vorliegende Kapitel beschreibt Spezifikationskomponenten, die als Hilfsprogramme Prozesse im Rahmen der PB-Verfahren unterstützen. Die Hilfsprogramme basieren auf der Programmiersprache Java. Dementsprechend ist ein Abschnitt enthalten, der die Installation einer Java-Laufzeit-Umgebung (JRE) beschreibt. Die Tools selbst umfassen derzeit ein Verschlüsselungsmodul und ein Datenprüfprogramm.

4.1 Java-Installation

Zur Ausführung von Java-Programmen wird das Java Runtime Environment (JRE) zur Verfügung gestellt. Das JRE kann „online“ und „offline“ installiert werden.

Offizielle Installationspakete können von der Website
http://www.java.com/de/download/manual.jsp
heruntergeladen werden.

Anleitungen zur Installation von Java unter verschiedenen Betriebssystemen sind auf der folgenden Website detailliert beschrieben:
https://www.java.com/de/download/help/download_options.xml

Aufruf
Der Aufruf des Java-Programms erfolgt über die Konsole des Betriebssystems. Die Rückmeldungen des Programms erfolgen ebenfalls über die Konsole und können dort abgefangen werden.

Die Ein- und Ausgabe der Konsole lässt sich in der Regel in jedes Programm integrieren. Ein Exitcode von 0 ohne Ausgabe entspricht einer fehlerfreien Verarbeitung des Programms. Bei Fehlern gibt es einen Exitcode von 1 und in der Regel eine Konsolen- bzw. eine Error-Ausgabe.

Aufruf: `java –jar Beispielprogramm.jar –g –o Dateiname 2> error.txt`

- `Java`: Aufruf der JVM mit dem Befehl java
- `-jar`: Parameter –jar, mit dem der JVM mitgeteilt wird, dass ein Java-Archiv aufgerufen wird
- `Beispielprogramm.jar`: Benennung des Archivs, mit vollem Pfad, wenn es nicht im aktuellen Verzeichnis liegt
- `-g –o`: Parameter, die an das Java Programm übergeben werden sollen
- 2>: Mit 2> Ziel der Error-Ausgabe spezifizieren
- `error.txt`: Dateiname der Error-Ausgabe – hier die Datei „error.txt“

Bei längeren Pfaden oder Dateinamen, die ggf. Leerzeichen oder andere Zeichen enthalten, sind diese in Anführungszeichen zu setzen. Dies gilt sowohl für Paketnamen als auch für Parameter-Dateien.

4.2 Datenprüfprogramm

Das Datenprüfprogramm wird bereitgestellt, um nicht nur beim Leistungserbinger, sondern auch bei den Datenanahmestellen eine Datenprüfung durchzuführen. Das Datenprüfprogramm

bezieht die Plausibilitätsregeln direkt aus der Spezifikation und testet Daten vor deren Verschlüsselung.

4.2.1 Umfang der Prüfungen

Es werden zwei wesentliche Bereiche mit diesem Programm geprüft:

- Schema-Konformität (Struktur)
  Die XML-Datei wird dabei auf Konformität mit dem zugrundeliegenden Schema (XSD) überprüft.
- Regel-Konformität (Inhalte)
  Die XML-Datei wird dabei auf Einhaltung der Regeln (XSLT) überprüft.

Die eigentliche Prüfung erfolgt in der XML-Export-Datei. Deren Struktur ist der entsprechenden Dokumentation bzw. dem gültigen XML-Schema zu entnehmen.

Die inhaltliche Prüfung selbst erfolgt über ein XSLT-Stylesheet und einen XSLT-Prozessor. Das Datenprüfprogramm verwendet die freie Version eines XSLT-Version 2.0-kompatiblen Programms (XSLT2). Die Einbindung von XSLT-Stylesheet und XSLT-Prozessor erfolgt über ein Java-Programm. Prinzipiell kann jeder XSLT2-fähige XSLT-Prozessor für die Prüfung auf dieser Grundlage verwendet werden. Das Datenprüfprogramm stellt eine Referenzimplementierung dar.

4.2.2 Ausgangskontrolle vor Versand

Eine Schemavalidierung der Ausgangsdateien vor dem Versand der XML-Daten wird aus folgenden Gründen als notwendig erachtet:

- Sicherstellung der Datenintegrität nach der Verarbeitung der Daten
- Frühe Feststellung von Fehlerquellen in der eigenen Datenverarbeitung
- Entlastung des nachfolgenden Datenservice von nicht validen Daten
- Vermeidung des Versands von Daten, die gegen den Datenschutz verstoßen

Aus diesen Gründen wurde das Datenprüfprogramm um die Möglichkeit erweitert die XML-Dateien auch nach der Verschlüsselung der XML-Elemente auf Schemavalidität zu prüfen.

Der Leistungserbringer verwendet abhängig vom Datenfluss die Schemata zur Übergabe an die Datenannahmestelle (LE-DAS) unter:

`interface_LE_DAS\`

Die Datenannahmestelle verwendet das folgende Schema zur Übergabe an die Vertrauensstelle (DAS-VST):

`interface_DAS_VST\interface_DAS_VST.xsd`

Diese Schemata können einfach in den Config-Dateien im Parameter `<xsd_path>` entsprechend angegeben werden. Dazu müssen nur mehrere Config-Dateien verwendet werden (je eine pro Schema). Des Weiteren besteht die Möglichkeit, beim Programmstart mit dem Parameter `--xsd-path` das Schema zu übergeben.

Diese Config-Dateien können entweder in verschiedenen Verzeichnissen abgelegt oder mit dem Parameter `-c` oder `--config` beim Start des Datenprüfprogramms über die Konsole angegeben werden. Da in diesem Fall keine inhaltliche Prüfung erfolgen soll (nur XSD, kein XSLT) muss zusätzlich der Parameter `--no-spez-val` angegeben werden:

```
java -jar datenpruefprogramm-4.2.0-jar-with-dependencies.jar -c config_schema.xml --no-spez-val
```

Diese notwendige Ausgangskontrolle können die Softwareanbieter (bzw. die Leistungserbringer) und die Datenannahmestellen unabhängig von dem Datenprüfprogramm realisieren, indem sie eine Schemavalidierung gegen die o.g. Schemata durchführen.

Für eine Schemavalidierung gibt es zahlreiche Tools und Bibliotheken für alle bekannten Programmiersprachen (http://www.w3.org/XML/Schema).

4.2.3 Programmaufruf

Das Datenprüfprogramm erzeugt eine Ausgabe/Output-Datei, die der Eingabe/Input-Datei entspricht, die jedoch um die Ergebnisse der Prüfungen erweitert wird. Die durchgeführten Prüfungen entsprechen einer Prüfung auf Dokumenten- und Vorgangsebene (Datensatzebene).

Das Datenprüfprogramm kann mehrere Dateien in einem Aufruf prüfen. Daher gibt es entsprechende Ordner für die Ein- und Ausgabedateien. Sollten diese Ordner nach der letzten Prüfung nicht geleert worden sein, so werden die Dateien des Eingabeordners erneut geprüft und der Ausgabeordner wird parallel mit Datum und Uhrzeit gesichert und ein neues leeres Ausgabeverzeichnis angelegt.

Die Prüfungen umfassen die Schemaprüfung und die Überprüfung der Feldinhalte (auch feldübergreifend).

Parameter `–c` oder `-config`

Die Steuerung der Funktionen erfolgt über eine Konfigurationsdatei, deren Dateipfad dem Programm beim Programmstart mit dem Parameter `-c` oder `--config` beim Programmaufruf übergeben werden kann.

```
java -jar datenpruefprogramm-4.2.0-jar-with-dependencies.jar -c C:/konfiguration/config.xml
```

Wenn keine Konfigurationsdatei übergeben wird, wird die Datei ./config.xml gesucht und geladen. Wenn diese Datei nicht gefunden wird, wird eine Standard-config.xml-Datei im Start-Order angelegt.

Parameter `--no-spez-val`

Mit diesem Parameter wird das Prüfskript ausgeschaltet.

```
java -jar datenpruefprogramm-4.2.0-jar-with-dependencies.jar --no-spez-val
```

Parameter `--no-schema-val`

Mit diesem Parameter wird die Schemaprüfung ausgeschaltet.

```
java -jar datenpruefprogramm-4.2.0-jar-with-dependencies.jar --
no-schema-val
```

Batch-Dateien

Beim Datenprüfprogramm werden beispielhafte Batchdateien mitgeliefert:

- datenpruefprogramm_schema.bat
  Hier wird eine reine Schemaprüfung (Nur XSD) anhand einer Konfigurationsdatei „config_schema.xml“ durchgeführt.
- datenpruefprogramm_<Datenfluss>_<Verfahrensart>.bat
  Hier wird sowohl die Schemaprüfung (XSD) als auch die Inhaltliche Prüfung (XSLT) anhand einer Konfigurationsdatei „config_<Datenfluss>_<Verfahrensart>.xml“ durchgeführt.

Beispiel einer Konfigurationsdatei:

```
<?xml version= "1.0 " encoding= "UTF-8 " standalone= "no "?>
<config>
  <provider>
    <address>12345 Musterdorf</address>
    <email>bernd.mustermann@musterfirma.de</email>
    <fax>0123/456798</fax>
    <function>Softwarehersteller</function>
    <name>Mustermann</name>
    <phone>0123/456789</phone>
    <registration></registration>
  </provider>
  <gui>false</gui>
  <input_path recursive= "true">C:\pruefmodul\input</input_path>
  <output_path>C:\pruefmodul\output</output_path>
  <xsd_path>C:\pruefmodul\xsd\interface_LE_WEICH</xsd_path>
  <xsl_path>C:\pruefmodul\xsl</xsl_path>
</config>
```

Die Konfigurationsdatei besteht aus den folgenden Bereichen:

Provider (Softwarehersteller)

Im Element `<provider>` werden Daten benötigt, aus denen hervorgeht, wer das Prüfmodul einbindet und ausführt. In der Regel ist dies der Softwarehersteller. Zu beachten ist, dass die Auswahlmöglichkeit im Element `<function>` auf Softwarehersteller eingeschränkt ist. Die Elemente `<fax>`, `<phone>`, und `<address>` sind optional, die anderen sind Pflichtelemente.

Achtung

In den Elementen <fax>, <phone>, und <address> dürfen auf keinen Fall die Angaben des Leistungserbringers eingetragen werden!

GUI (Konsole)

Für ein vereinfachtes Debugging gibt es die Möglichkeit, eine Konsole mit detaillierten Programmausgaben während der Verarbeitung über das Element <gui> und den Wert `true` zu öffnen. Der Standard-Wert ist `false`.

Input_Path (Eingabeverzeichnis) – überschreibbar mit Parameter –`input`

Im Element <input_path> kann der Eingabeordner für die zu überprüfenden Exportdateien festgelegt werden. Das Element ist optional. Ohne diesen Parameter ist der Ordner <arbeitsverzeichnis>\input\ der Standard-Eingabe-Ordner. Es werden alle Dateien mit der Dateiendung `.xml` verarbeitet. Wenn das Attribut `recursive` auf `true` steht, werden auch alle entsprechenden Dateien in Unterordnern berücksichtigt. Die Standard- Einstellung von `recursive` ist `false`.

Output_Path (Ausgabeverzeichnis) – überschreibbar mit Parameter –`output`

Im Element <output_path> kann der Ausgabeordner festgelegt werden. Das Element ist optional. Ohne diesen Parameter ist der Standard-Ausgabeordner <arbeitsverzeichnis>\output\. Der Dateiname der Ausgabedatei ist dabei gleich dem der Eingabedatei. Ein ggf. nicht vorhandener Ordner wird angelegt.

XSD_Path (Schemaordner) – überschreibbar mit Parameter --xsd-path

Im Element <xsd_path> wird der Schemapfad gesetzt. Es wird dabei entweder auf ein Verzeichnis gezeigt, in dem genau eine Schemadatei erwartet wird, oder es wird direkt auf eine xsd-Datei gezeigt. Das Element ist optional. Ohne diesen Parameter wird das Schema im Verzeichnis <arbeitsverzeichnis>\xsd\interface_LE_WEICH gesucht. Eine Spezifikationskonforme Protokollierung kann vom Datenprüfprogramm sichergestellt werden, wenn die weiche Schemavariante verwendet wird.

Um nach einer Schemavalidierung der XML-Dateien, die Weiterverarbeitung und demensprechend die spezifikationskonforme Protokollierung auf Datensatzebene weiterhin zu ermöglichen, wurden neben der harten Schemavariante ein weiches Schema für die Schnittstellen LE und DAS eingeführt. Diese weiche Variante wird ausschließlich mit dem Datenprüfprogramm verwendet (Abbildung 53). Eine detaillierte Übersicht über die Anwendung weicher Schemata mit dem Datenprüfprogramm finden sie im Abschnitt B 3.2.

Hinweis

Neben den Leistungserbringern sind nur die Datenannahmestelle (KV) und die Auswertestelle (BAS) die Stellen, die PB- Daten gemäß den G-BA-RL entschlüsseln darf und die weiche Variante benötigt .

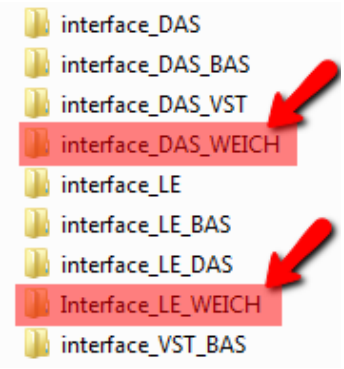


Abbildung 53: Weiche Schemavariante für das DPP

XSL_Path (XSLT-Stylesheet-Ordner) – überschreibbar mit Parameter --xsl-path

Im Element `<xsl_path>` kann der Quellordner für die XSLT-Stylesheets festgelegt werden. Das Element ist optional. Ohne diesen Parameter wird im Standard XSL-Ordner `<arbeits-verzeichnis>\xsl\` nach den XSLT–Stylesheets gesucht.

Mehrere Konfigurationsdateien können für dasselbe Datenprüfprogramm angelegt werden, um beispielsweise Dateien unterschiedlicher Spezifikationen zu validieren oder die Durchführung einer Eingangs- bzw. einer Ausgangskontrolle jeweils vor der Entschlüsselung und nach der Verschlüsselung zu ermöglichen.

4.2.4 Verzeichnisstruktur

Für das korrekte Funktionieren des Prüfprogramms ist neben den erforderlichen Dateien auch eine korrekte Verzeichnisstruktur notwendig.

In der config.xml wird der `<xsl_path>` definiert.

Wenn der Parameter `<xsl_path>` auf ein Verzeichnis zeigt, muss in diesem Verzeichnis eine Stylesheet-Datei der folgenden Art vorliegen:

`xsl\<Spezifikationsversion>.aqxsl`

Kompiliertes Haupt-XSL-Stylesheet, das die Prüfung entsprechend der Spezifikation durchführt. Es enthält alle Tests auf Regeln und Wertebereichsverletzungen. Ansonsten kann der Parameter `<xsl_path>` auf eine beliebige Stylesheet-Datei verweisen.

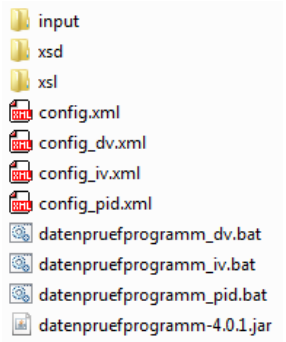


Abbildung 54: Beispiel einer typischen Verzeichnisstruktur

Die Verzeichnisstruktur kann über eine config.xml-Datei modifiziert werden. Ohne diese config.xml wird eine Standardkonfiguration angenommen, die das Prüfprogramm selbst in einer config.xml neu schreibt.

In Abbildung 54 ist eine typische Verzeichnisstruktur mit der config-Datei und dem Programm abgebildet. Unterhalb dieses Grundordners befinden sich die Verzeichnisse, die schon in der Beschreibung der Konfiguration angesprochen wurden.

4.2.5 Ausgabe

Nach dem Prüflauf wird ein Ordner `<output>` erzeugt, der die geprüften Dateien und deren Datenflussprotokolle beinhaltet.

Geprüfte Dateien

Im Order `<output>/files` liegen nun die geprüften Quell-Dateien, welche um das Ergebnis der Prüfung erweitert worden sind. Jeder Fall wird innerhalb der XML-Datei geprüft und in der XML-Struktur abgelegt. Zudem wird ein neuer Eintrag als `validation_provider` erzeugt.

Protokolle

Im Order `<output>/protocol` liegen die Datenflussprotokolle, die den entsprechenden Dateien im Ordner `<output>/files` entsprechen, in denen PB-Daten und Patientendaten entfernt wurden.

HTML-Protokolle

Im Order `<output>/html` liegt eine index.html, in der auf vereinfachte Sichten der im Ordner `<output>/protocol` erstellten Protokollen verwiesen wird.

5d29aac7-d480-4377-ad2b-070596b27dda.xml	5d29aac7-d480-4377-ad2b-070596b27dda	OK
65238192-1879-4fa2-84e6-a4323f52522c.xml	65238192-1879-4fa2-84e6-a4323f52522c	OK
8e5e8197-5478-46b5-8fd3-cb527d0141a2.xml	8e5e8197-5478-46b5-8fd3-cb527d0141a2	OK
9afe7686-7f73-43f8-a5ce-27d5139064f4.xml	9afe7686-7f73-43f8-a5ce-27d5139064f4	OK
9d6359cf-2b70-4706-8670-5dd51b67306f.xml	9d6359cf-2b70-4706-8670-5dd51b67306f	WARNING

Abbildung 55: Beispiel für eine Index.html Datei im Ordner <output>/html

Da für die Rückprotokollierung nur die Übertragung der unter `<output>/files` abgelegten Datei spezifiziert ist, wird die Darstellung außerhalb von PB-Programmen beim Lei-stungserbringer durch ein eigenes Stylesheet ermöglicht, das sich an der Darstellung des Datenprüfprogramms orientiert. Die Dokumentation in Bezug auf dieses Stylesheet und dessen Einbindung ist im Abschnitt A „Lokale Transformation (Empfehlung)“ auf S. 72 zu finden.

4.2.6 Grafische Oberfläche

Wird der Parameter GUI in der Konfigurationsdatei auf „true“ gesetzt, wird das Datenprüfprogamm mit einer einfachen grafischen Oberfläche gestartet.

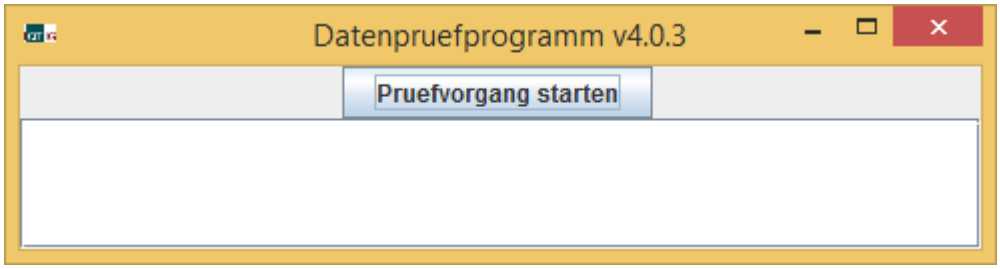


Abbildung 56: Grafische Oberfläche des Datenprüfprogramms

Bei der grafischen Oberfläche muss zum Starten die Schaltfläche „Pruefvorgang starten“ gedrückt werden.

Die grafische Oberfläche zeigt die Ausgabe im Fensterbereich direkt an. Am Inhalt des Ausgabe-Ordners ändert sich nichts; beide Laufvarianten (grafische Ausgabe oder Konsolenausgabe) erzeugen den gleichen Output.

4.2.7 Programmierschnittstelle – API

Sämtliche Funktionen des Datenprüfprogramms können über eine Programmierschnittstelle (API) aufgerufen und direkt in einem Java-Umfeld verwendet werden.

4.3 Verschlüsselungspaket

Das IQTIG stellt ein Ver- und Entschlüsselungspaket für die Anwendung zur Verfügung. Die Ver- und Entschlüsselungsalgorithmen sind entsprechend der im Abschnitt „Gesicherte PB-Datenübertragung“ beschriebenen W3C-Standards[68] implementiert. Das bedeutet, dass an jeder beliebigen Stelle im Workflow auch jede andere Implementierung, die sich an diese Standards hält, verwendet werden kann.

Das Verschlüsselungspaket besteht aus dem TPacker für die Transportverschlüsselung und dem XPacker für die XML-Verschlüsselung. Für alle Anwender, die händisch die Verschlüsselung durchführen müssen, stellt der GPacker mit seiner grafischen Oberfläche eine interaktive Alternative zur Verwendung der Programme XPacker und TPacker dar.

4.3.1 XPacker – XML-Verschlüsselung

Das Programm XPacker.jar dient zur Ver- und Entschlüsselung einzelner XML-Elemente innerhalb einer XML-Datei. Bevor diese verschlüsselt werden, wird jedes einzelne Element mit „base64“ komprimiert bzw. „gezippt“ und beim Entschlüsseln parallel wieder „entpackt“.

Eine weitere Funktion des Programms dient zur Generierung eines asymmetrischen Schlüsselpaares (privater und öffentlicher Schlüssel).

Die Verschlüsselung wird mit dem hybriden Verfahren (Abschnitt A „Die Verschlüsselung“ auf S. 55 durchgeführt:

- Die Verschlüsselung der Daten erfolgt mit einem AES-128bit-Schlüssel im CBC-Mode (aes128-

[68] RSA2048: http://www.w3.org/TR/xmlenc-core/
AES128: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

cbc).

- Die Verschlüsselung des symmetrischen Schlüssels erfolgt mit einem RSA-Schlüssel im OAEP-Mode (rsa-oaep-mgf1p).

Weitere technische Details beschreibt der W3C-Standard „XML Encryption Syntax and Processing“.[69]

Die folgende Abbildung zeigt ein XML-Dokument nach der Verschlüsselung eines XML-Elements (PB-Daten):

```
<encryption>
  <ds:KeyInfo Id="Pub_key_Bundesauswertungsstelle.pub-qs_data-2.2.0">
    <xenc:EncryptedKey Id="qs_data" xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
      <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
      <xenc:CipherData>
        <xenc:CipherValue>dybYkkepEipN48lE0mbj28aw83HDyMqJQXvT3nSof34SdLHLpZ+A==</xenc:CipherValue>
      </xenc:CipherData>
    </xenc:EncryptedKey>
  </ds:KeyInfo>
</encryption>
</header>
<body>
  <data_container>
    <care_provider>
    <cases module="KAT_FU_A" pseud_procedure="KAT">
      <case>
        <patient>
        <case_admin>
        <qs_data module="KAT_FU_A" xsi:type="qs_data_kat_fu_a_type">
          <xenc:EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Content" xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
            <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
            <xenc:CipherData>
              <xenc:CipherValue>xEO1RE59SeMKaRs1V4DIz2QJDDV</xenc:CipherValue>
            </xenc:CipherData>
          </xenc:EncryptedData>
        </qs_data>
```

Abbildung 57: Verschlüsselung eines XML-Elements (qs_data)

In diesem XML-Dokument existieren einige spezielle Elemente. Die wichtigsten sind:

- `EncryptedData` ist das einschließende Element für die XML-Verschlüsselung. Der gesamte Inhalt des übergeordneten Elements einschließlich der Attribute ist verschlüsselt.
- `CipherData` ist das verschlüsselte Element.
- `CipherValue` enthält die verschlüsselten Daten.
- `KeyInfo` enthält Informationen über den Schlüssel, mit dem die Daten verschlüsselt worden sind.
- Das Attribut `id` enthält den Namen des `PublicKeys` und den Namen des zu verschlüsselnden „Tags“ und die aktuelle Versionsnummer des jeweils aktuellen XPackers.

Syntax/Hilfe

Mit dem Befehl `–h` wird die Syntax des Programms und eine Parameterübersicht ausgegeben:

```
java –jar XPacker-4.1.2-jar-with-dependencies.jar -h
```

[69] Ebd.

Verschlüsseln

Die Verschlüsselung wird durch den Parameter –e aktiviert.

Benötigt wird der Parameter –k, gefolgt von Dateinamen des öffentlichen Schlüssels, der Parameter –t mit durch Komma getrennten Element-Namen sowie der Parameter -2 mit Tag-Name, in den der verschlüsselte Schlüssel aufgenommen wird. Optional sind die Parameter –f und –o.

–f gibt die einzulesende XML-Dateien an und –o die auszugebende Datei. Falls diese Parameter fehlen, wird die Standard Ein-/Ausgabe verwendet.

Beispiel:

```
java –jar XPacker-4.1.2-jar-with-dependencies.jar –e –f in-
file.xml –o outfile.xml
–k datenannahme.pub
–t qs_data -2 encryption
```

Entschlüsseln

Die Entschlüsselung entspricht dem Verschlüsseln, nur dass der Parameter -2 entfällt und bei dem Parameter –k der private Schlüssel angegeben werden muss und der Parameter –e durch –d ersetzt wird.

Beispiel:

```
java –jar XPacker-4.1.2-jar-with-dependencies.jar –d –f in-
file.xml –o outfile.xml
–k datenannahme.pri –t qs_data
```

Schlüsselpaare erzeugen

Für die Erzeugung eines Schlüsselpaares wird der Parameter –g verwendet. Optional kann mit –o ein Dateiname angegeben werden. An diesen Namen wird .pub für den öffentlichen Schlüssel und .pri für den privaten Schlüssel angehängt. Falls der Parameter –o fehlt, wird im aktuellen Verzeichnis ein key.pub für den öffentlichen Schlüssel und ein key.pri für den privaten Schlüssel erzeugt.

Beispiel:

```
java –jar XPacker-4.1.2-jar-with-dependencies.jar –g –k da-
tenannahme
```

Erzeugt im aktuellen Verzeichnis die Dateien datenannahme.pub und datenannahme.pri.

4.3.2 TPacker – Transportverschlüsselung

Das Programm TPacker.jar dient zur Transportverschlüsselung. Neben dem Ver- und Entschlüsseln werden die Dateien auch ge- und entpackt (.zip).

Beim Ver- und Entschlüsseln wird aus dem symmetrischen Schlüssel (Passwort) über ein MD5-Hash ein AES-Schlüssel erzeugt. Die eigentliche Verschlüsselung erfolgt im ECB-Mode. Als Padding[70] verwendet das Programm den PKCS5-Standard.

Die Steuerung der Funktionen erfolgt über Parameter beim Programmaufruf.

Syntax/Hilfe

Mit dem Befehl –h werden die Syntax des Programms und eine Parameterübersicht ausgegeben

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar -h
```

Isolierte Varianten

Das Packen und Verschlüsseln bzw. das Entschlüsseln und Entpacken können unabhängig voneinander durchgeführt werden.

Achtung
Anwender- und Übertragungsfehler

Die im Folgenden beschriebenen Aufrufe von isoliertem Ver- und Entschlüsseln sind komplex und können schnell zu Anwender- und Übertragungsfehlern führen.

Es müssen die richtigen Schritte mit den richtigen Parametern exakt in der richtigen Reihenfolge durchgeführt werden. Dazu sind die im Folgenden beschriebenen Varianten zu nutzen.

Verschlüsseln

Die Verschlüsselung wird durch die Parameter –e aktiviert. Benötig wird noch der Parameter –f mit dem Dateinamen der zu verschlüsselnden Datei und der Parameter –p mit dem Passwort.

Falls nichts anderes angegeben wird, entspricht der Name der Ausgabedatei dem mit dem Parameter –f angegebenen Eingabedateinamen.

Mit dem Parameter –o kann man den Namen der Ausgabedatei ändern. In beiden Fällen wird an den Dateinamen ein Zeitstempel und .aes hinzugefügt. Falls der Zeitstempel nicht gewünscht ist, wird dieser mit dem Parameter –t deaktiviert.

Beispiel:

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar     -e -f Datei
-o Dateiname -p geheim
```

[70] Als „Padding“ wird hier das Auffüllen eines unvollständigen Blocks am Ende des Datenstroms bezeichnet.

würde die Datei Dateiname_2012_01_01_121212.aes erzeugen, wenn das Programm am 01.01.2012 um 12:12 Uhr und 12 Sekunden ausgeführt wird. Die Verschlüsselung würde mit dem Schlüssel „privat“ durchgeführt.

Entschlüsseln

Beim Entschlüsseln geht man genau wie beim Verschlüsseln vor, nur dass man den Parameter –e durch den Parameter–d ersetzt.

Der Dateiname der Ausgabedatei entspricht der von Parameter –f. Falls ein Suffix .aes enthalten ist, wird dieses entfernt.

Beispiel:

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar -t -d -f Datei.aes –p geheim
```

würde die Datei „Datei“ erzeugen.

Packen

Beim Packen geht man genau wie beim Verschlüsseln vor, nur dass man die Parameter –e durch –z ersetzt und der Parameter –p überflüssig ist. Das Suffix der gebildeten Datei lautet .zip.

Der Parameter –f kann eine oder mehrere, durch Komma(ta) getrennte Dateinamen (ohne Leerzeichen) enthalten. Bei Angabe eines Verzeichnisses gibt es einen Fehler.

Beispiel:

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar –t -z -f Datei -o Dateiname
```

würde die Datei „Dateiname.zip“ erzeugen.

Entpacken

Beim Entpacken geht man genau wie beim Packen vor, nur dass man den Parameter –z durch den Parameter–u ersetzt.

Die Ausgabe über Parameter –o ist immer ein Verzeichnis. Falls kein Verzeichnis angegeben wird, wird das aktuelle Verzeichnis als Ausgabeverzeichnis verwendet.

Beispiel:

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar -t -u -f Datei.zip -o Verzeichnis
```

würde die Datei „Datei(en)“ im Verzeichnis „Verzeichnis“ entpacken.

Integrierte Varianten

Das Packen und Verschlüsseln bzw. das Entschlüsseln und Entpacken kann in einem Schritt zusammengefasst werden.

Verschlüsseln mit integriertem Verpacken

Es ist auch möglich, in einem Schritt zu packen und zu verschlüsseln. Hier kombiniert man die Parameter zu `-ze`. Es wird immer zuerst gepackt und dann verschlüsselt. Als Suffix wird dann .zip.aes angehängt.

Beispiel:

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar –t -ze -f Datei
-o Dateiname -p geheim
```

würde die Datei „Dateiname.zip.aes“ erzeugen.

Entschlüsseln mit integriertem Entpacken

So wie das Packen und Verschlüsseln, kann auch das Entschlüsseln und Entpacken in einem Schritt zusammengefasst werden. Hier kombiniert man die Parameter zu `–ud`. Es wird immer zuerst entschlüsselt und dann entpackt. Die Ausgabe erfolgt auch hier immer in ein Verzeichnis.

Beispiel:

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar –t -ud -f Da-
tei.zip.aes -p geheim
```

würde die Datei „Datei.zip.aes“ im aktuellen Verzeichnis entschlüsseln und entpacken.

Der Modus „unsafe“

Der TPacker ist standardmäßig so eingestellt, dass vor der Verschlüsselung das zu verschlüsselnde Dokument darauf untersucht wird, ob ein Element `VERSICHERTENID` bzw. `VERSICHERTENIDNEU` vorhanden ist. Falls diese Prüfung positiv ausfällt oder kein wohlgeformtes XML gefunden wird, unterbricht eine entsprechende Fehlermeldung die Verschlüsselung. Durch diese Maßnahme soll verhindert werden, dass versehentlich eine Transportdatei VOR der XML-Verschlüsselung erzeugt werden kann.

Um diesen Mechanismus auszuschalten, kann der Parameter

```
--unsafe
```

gemeinsam mit dem Verschlüsselungsparameter -e verwendet werden.

Beispiel:

```
java –jar TPacker-4.1.2-jar-with-dependencies.jar --unsafe -e -f
Datei -o Dateiname -p geheim
```

würde beim Erzeugen der verschlüsselten Datei keine Prüfung auf den Inhalt vornehmen

---

4.3.3 TPacker und XPacker ohne externe Abhängigkeiten

XPacker und TPacker werden vom IQTIG bei Bedarf ohne externe Abhängigkeiten (ohne fremde Bibliotheken) zur Verfügung gestellt. Die aktuelle Version ist auf der Kommunikationsplattform des IQTIG zu finden.

Hinweis

Die Programmvarianten ohne externe Abhängigkeiten sind nur für die Softwareanbieter relevant, die selber die Programme in ein JAVA-Umfeld integrieren und dabei Konflikte mit bereits installierten Bibliotheken vermeiden wollen.

4.3.4 Programmierschnittstelle – API

Sämtliche Funktionen der Verschlüsselungsprogramme können über je eine Programmierschnittstelle (API) aufgerufen und direkt in einem Java-Umfeld verwendet werden.

4.3.5 GPacker

Für alle Anwender, die händisch die Verschlüsselung durchführen müssen, stellt der GPacker mit seiner grafischen Oberfläche eine interaktive Alternative zur Verwendung der Programme XPacker und TPacker dar. Der GPacker integriert dabei den XPacker und TPacker über die GUI, die als einzelne Programme damit nicht benötigt werden. Er ermöglicht die Verschlüsselung und Komprimierung von PB-Dateien im XML-Format, die im Rahmen der oKFE-RL erstellt worden sind. Eine ausführliche Dokumentation liegt der Komponente bei.

C Anhang

Glossar

Begriff	Beschreibung
Abrechnungsdaten	Daten, die von Leistungserbringern über von ihnen erbrachte Leistungen zum Zweck der Abrechnung mit den Kostenträgern dokumentiert und übermittelt wurden.
Anonymisierung	Verändern personenbezogener Daten derart, dass die Einzelangaben über persönliche oder sachliche Verhältnisse nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand an Zeit, Kosten und Arbeitskraft einer bestimmten oder bestimmbaren natürlichen Person zugeordnet werden können. (§ 3 Abs. 6a BDSG)
Auslösekriterien	Algorithmus zur Auslösung der Dokumentationspflicht (PB-Filter).
PB-Spezifikation	Gemeinsame Spezifikation für PB-Dokumentationen.
Bundesdatenpool	Zusammenführung aller bundesweit dokumentierten PB-Daten.
Datenannahmestellen	Stellen, an die die Leistungserbringer oder andere Daten liefernde Stellen (z. B. Krankenkassen) die erhobenen Daten übermitteln. Sie prüfen die übermittelten Daten auf Plausibilität, Vollständigkeit und Vollzähligkeit.
Datenbasis	Im Einzelfall festzulegende bzw. festgelegte Gesamtmenge von auszuwertenden bzw. ausgewerteten Daten.
Datenfeld	Kleinste Einheit eines Datensatzes (z. B. Angabe des Geschlechts im Dokumentationsbogen).
Datenfluss	Übermittlung der Daten der PB-Verfahren in einem festgelegten Format und Inhalt, die vom Leistungserbringer über eine Datenannahmestelle, Vertrauensstelle (nur einrichtungs- und sektorenübergreifende PID-Verfahren) bis zur Datenauswertungsstelle gelangen. Die Datenflüsse sind grundsätzlich in der oKFE-RL des G-BA festgelegt.
Datensatz	Eine zusammenhängende Menge von PB-Daten, die einem Fall (beispielsweise einem Patienten) zugeordnet wird.
Datenvalidierung	Verfahren zur Überprüfung von PB-Daten einerseits auf Vollzähligkeit, Vollständigkeit und Plausibilität (statistische Basisprüfung), andererseits ihre Übereinstimmung (Konkordanz) mit einer Referenzquelle wie bspw. der Krankenakte (Stichprobenverfahren mit Datenabgleich).
Dokumentation	siehe: PB-Dokumentation

Einheitlicher Bewertungsmaßstab (EBM)	Vergütungssystem für die Abrechnung von Leistungen im vertragsärztlichen Bereich.
Einrichtung	siehe: Leistungserbringer
Erfassungsjahr	Das Jahr, in dem die Daten erhoben werden. Hierauf beruhen die Ergebnisse der Indikatoren.
Gemeinsamer Bundesausschuss (G-BA)	Oberstes Beschlussgremium der gemeinsamen Selbstverwaltung der Ärzte, Zahnärzte, Psychotherapeuten, Krankenhäuser und Krankenkassen in Deutschland. Er bestimmt in Form von Richtlinien den Leistungskatalog der Gesetzlichen Krankenversicherung (GKV) für etwa 70 Millionen Versicherte und legt damit fest, welche Leistungen der medizinischen Versorgung von der GKV erstattet werden.
Kostenträger	Personen und Institutionen, die die Kosten für medizinische Versorgungsleistungen tragen. Im Rahmen der gesetzlich verankerten Qualitätssicherung sind dies die gesetzlichen Krankenkassen.
Leistungserbringer	Personen und Einrichtungen, die medizinische Versorgungsleistungen erbringen bzw. bereitstellen. Der Begriff wird im SGB V auch für Ärzte und ärztliche Einrichtungen sowie für zugelassene Krankenhäuser gemäß § 108 SGB V genutzt.
Leistungserbringeridentifizierende Daten (LID)	Daten, die eindeutig einen bestimmten Leistungserbringer identifizieren (z. B. Institutionskennzeichen oder Betriebsstättennummer).
Missing Values	„Fehlende Werte“, z. B. fehlende Antworten und nicht auswertbare Antworten bei der Auswertung eines Fragebogens.
Patientenidentifizierende Daten (PID)	Daten, die eindeutig einen bestimmten Versicherten identifizieren (z. B. Versichertennummer).
Plausibilitätsprüfung	Statistisches Verfahren, mit dem die Dokumentationsdaten auf erlaubte und/oder fehlende Werte, Widerspruchsfreiheit, Werteverteilung und bekannte Korrelationen geprüft werden.
Probebetrieb	Erprobung eines PB-Verfahrens in einer begrenzten Anzahl von Einrichtungen. Ziel ist die Prüfung, ob die für das PB-Verfahren benötigten Daten entsprechend der vom Auftragnehmer vorgesehenen Planungen für die vorgelegten Indikatoren und Instrumente erhebbar und die Ergebnisse umsetzbar sowie für die Durchführung der Programmbeurteilung verwertbar sind.
Pseudocode	Programmcode, der das zugrundeliegende Prinzip eines Algorithmus beschreibt, selbst aber nicht lauffähig ist. Er dient zur Veranschaulichung, unabhängig von der konkret zu verwendenden Programmiersprache.

Pseudonymisierung	Ersetzen des Namens und anderer Identifikationsmerkmale durch ein Kennzeichen zu dem Zweck, die Bestimmung des Betroffenen auszuschließen oder wesentlich zu erschweren (§ 3 Abs. 6a BDSG).
PB-Auslösung	Initiierung einer Dokumentationspflicht zu Zwecken der Qualitätssicherung (PB-Dokumentation). Bei einer Erhebung vorhandener Daten (z. B. Sozialdaten bei den Krankenkassen) analog das Kriterium, das die Lieferung eines bestimmten Datensatzes auslöst.
PB-Daten	Sammelbegriff für alle Daten, die im Zuge eines PB-Verfahrens erhoben und ausgewertet werden.
PB-Dokumentation	Gesonderte Erhebungen der Leistungserbringer zu Diagnose- und Behandlungsdaten der Patienten durch die Leistungserbringer für die Qualitätssicherung.
PB-Filter	Algorithmus, der auf Grundlage festgelegter Kriterien die für die Qualitätssicherung durch die Leistungserbringer zu dokumentierenden Patienten und deren Daten „filtert“. Die Kriterien hierzu werden in einer Spezifikation definiert.
PB-Filter-Software	Implementierung der Spezifikation für den PB-Filter.
PB-Verfahren	siehe: Verfahren
Qualität	Bezogen auf die Gesundheitsversorgung: Grad, in dem versorgungsrelevante Ergebnisse, Prozesse und Strukturen bestimmte, definierte Anforderungen erfüllen.
Regelbetrieb	auch: Routinebetrieb oder Echtbetrieb. Verpflichtende und flächendeckende Umsetzung eines PB-Verfahrens.
Routinedaten	hier: Daten, die wesentlich zur Abwicklung von Geschäfts- und Verwaltungsabläufen erhoben werden (z. B. Abrechungsdaten, personenbezogene administrative Daten). Abseits des uneinheitlichen Sprachgebrauchs stehen die Sozialdaten bei den Gesetzlichen Krankenkassen (auch: GKV-Routinedaten) im Vordergrund des Interesses, da sie gemäß § 299 Abs. 1a SGB V zu Zwecken der Qualitätssicherung verwendet werden dürfen. Diese beinhalten insbesondere die abrechnungsrelevanten Daten für ambulante und stationäre Versorgungsleistungen (§§ 295 und 301 SGB V), für Arznei-, Heil- und Hilfsmittel (§§ 300 und 302 SGB V) sowie die Versichertenstammdaten (§ 284 SGB V).
Sozialdaten	Einzelangaben über die persönlichen und sachlichen Verhältnisse (personenbezogene Daten), die von den sozialrechtlichen Leistungsträgern zur Erfüllung ihrer gesetzlichen Aufgaben gesammelt und gespeichert werden.
Spezifikation	Datensatzbeschreibung. Festlegung, welche Daten für die Qualitätssicherung erhoben bzw. übermittelt werden müssen, welche

	Prüfalgorithmen zur Anwendung kommen (z. B. für Plausibilitätsprüfungen) und wie die PB-Auslösung operationalisiert ist. Im Rahmen der Neuentwicklung von PB-Verfahren ist die Spezifikation als das Ergebnis der informationstechnischen Aufbereitung zu betrachten.
Systempflege	Routinemäßige und kontinuierliche Evaluation und Anpassung der Qualitätsindikatoren, der Softwarespezifikation usw.
Verfahren	Bereich der Früherkennung von Krebserkrankungen der im Rahmen der oKFE-RL dokumentationspflichtig ist.
Vertrauensstelle	Institution, die im Rahmen der einrichtungs- und sektorenübergreifenden Qualitätssicherung erhobene patientenidentifizierende Daten pseudonymisiert. Näheres regeln die die entsprechenden Richtlinien (oKFE-RL) sowie themenspezifische Bestimmungen.
Vollständigkeit	Erfassung aller zu einem einzelnen Behandlungsfall erforderlichen Angaben (Daten).
XSLT	Extensible Stylesheet Language Transformations. Programmiersprache zur Transformation von XML-Dokumenten in andere XML-Dokumente oder andere Dokumentformate wie HTML. Im PB-Kontext kann es auch für Datenprüfung und Protokollerstellung verwendet werden.