

# **Erstellung von XML-Signaturen für Dokumente nach Clinical Documents Architecture – R2**

**(Elektronische Signatur von Arztbriefen)**



**Bundesärztekammer    Ärztekammer Nordrhein    Ärztekammer Westfalen-Lippe**

**Ärztekammern in NRW  
im Auftrag der  
Bundesärztekammer**

Version 1.3

Status: Final

Stand: 30. Januar 2008

## Inhaltsverzeichnis

1	Ziele und Anforderungen .....	4
2	Beteiligte .....	6
3	XML-Namensräume .....	7
4	Typ und Positionierung von Signaturelementen .....	8
5	Schema CDocumentPayload .....	10
5.1	Root-Element ContentPackage .....	10
5.2	ContentContainerType .....	11
5.3	Anhänge zu XML-strukturierten Dokumenten .....	13
5.3.1	Allgemeine Behandlung von Anhängen .....	13
5.3.2	Anhänge, die aus CDA-R2-Dokumenten referenziert werden .....	15
5.3.3	ContentPackage(s) als Anhang .....	17
5.4	Hinweise zu verschlüsselten ContentContainern .....	18
5.5	StructuredContentType .....	19
6	Belegung von Id-Attributen .....	20
6.1	Sicherung der Eindeutigkeit .....	20
6.2	ID-Attribute in CDA-R2-Dokumenten .....	20
7	Profilierung von XML-Signature .....	23
7.1	Signature Element .....	23
7.2	SignedInfo Element .....	24
7.2.1	CanonicalizationMethod .....	24
7.2.2	SignatureMethod Element .....	24
7.2.3	Reference, DigestMethod Element .....	25
7.3	KeyInfo Element .....	25
7.3.1	RetrievalMethod Element .....	26
7.3.2	X509Data Element .....	26
7.4	Reference: URI .....	27
7.5	Object Elemente: qualifizierende Angaben zur Signatur .....	27
7.5.1	Signaturzeitpunkt (Systemzeit) .....	28
7.5.2	Optional qualifizierter Zeitstempel .....	28
7.5.3	Verweis auf das Signaturzertifikat .....	30
7.5.4	Signature Policy .....	30
7.5.5	Attributzertifikate und Rollen .....	31
7.5.6	Gesamtübersicht: SignedSignatureProperties .....	34
7.6	Object Element, Referenz auf das zugehörige Stylesheet .....	34
7.7	Zusammenfassung: Weitere Reference-Elemente .....	36
7.7.1	Referenz auf das Manifest-Element .....	36
7.7.2	Referenz auf XAdES-Erweiterungen .....	36
8	Verzeichnisse .....	37
8.1	Abbildungen .....	37
8.2	Tabellen .....	37
8.3	Listings .....	37
8.4	Referenzen .....	38
9	Anhang .....	40
9.1	Schemaeinschränkungen zu XML Signature .....	40
9.2	Hinweise zu den Schemata XAdES und XML Encryption .....	43
9.3	Schemaeinschränkungen XAdES .....	44
9.4	Schema CDocumentPayload .....	45
9.5	Beispiel (Ausschnitt) .....	47

## Änderungshistorie

Version	Stand	Bearbeiter	Änderungen / Kommentar
1.3	30.1.2008	Apitzsch	<p>Einarbeitung Vorschläge gematik zu V 1.:</p> <ul style="list-style-type: none"> <li>• In ContentContainerType "&lt;xs:sequence&gt;" ersetzt durch "&lt;xs:choice maxOccurs='unbounded'&gt;"</li> <li>• cdp:StructuredContent Kardinalität 1 im Schema (wie in 1.2 schon in Spec)</li> <li>• Keine Restriction zu xades:SignaturePolicyType</li> </ul>
1.2	19.1.2008	Apitzsch	<p>Hinweis auf neuen Algorithmenkatalog BNetzA vom Dezember 2007</p> <p>Korrektur Schema CDP:</p> <ul style="list-style-type: none"> <li>• Import aller Namespaces für CDA-Dokumente und XAdES (mit Restrictions)</li> <li>• Element &lt;StructuredContent&gt; jetzt Kardinalität 1 (nicht 1..n)</li> </ul> <p>Kap. 5.2: Hinweise zum Import von Namespaces</p> <p>Kap 5.3.2: Überarbeitung Table 2, zulässige Ausprägungen für Mime-Types von Attachments im Attribut &lt;xmime:contentType&gt; für Elemente &lt;UnstructuredContent&gt; (siehe Tabelle 2)</p> <p>Neues Kapitel 5.3.3: Hinweise zur Aufnahme von Teilbäumen &lt;ContentPackage&gt; in Elemente &lt;UnstructuredContent&gt;</p> <p>Kap. 7.5: Detaillierungen zu XAdES-Erweiterungen</p> <p>Kap. 7.6: Neuer Name des Stylesheets für das gesamte ContentPackage</p> <p>Neu: Kap. 9.3: Hinweise zu Schemata xenc, xades</p> <p>Neu: Kap. 9.4: XAdES Schema-restrictions</p> <p>Kap. 9.5: Beispiel-Dokument an neuen Stand angepasst</p>
1.1	23.11.2007	Apitzsch	<p>Korrektur Tippfehler</p> <p>Kap. 5.3.2: Anmerkung zur Aufnahme des Hash von Attachments zur Integritätssicherung</p>

# 1 Ziele und Anforderungen

Mit dieser Spezifikation werden Datenstrukturen und Mechanismen zur Erstellung von elektronischen Signaturen für XML-Dokumente definiert, die dem Standard „HL7 Clinical Document Architecture, Release 2.0“ [CDA2] entsprechen (im Folgenden als CDA-R2-Dokumente bezeichnet).

Die strikte Umsetzung der Spezifikation von im Gesundheitswesen genutzten Signaturanwendungskomponenten (SAKs) soll Basis der Verkehrsfähigkeit und Handhabbarkeit von signierten CDA-R2-Dokumenten sein und die Interoperabilität unterschiedlicher Implementierungen sicherstellen.

Mit Hinsicht auf die Handhabung von CDA-R2-Dokumenten sind in dieser Spezifikation über Anforderungen an reine SAK-Funktionalitäten auch weitere Anwendungsspezifika formuliert, die über die reine Erstellung qualifizierter elektronischer Signaturen hinausgehen. Dieses Dokument erhebt dabei nicht den Anspruch, eine trennscharfe Unterscheidung von Anforderungen an die „Signaturanwendungskomponente“ im Sinne von SigG/SigV und an die vor- bzw. nachgeschaltete „Anwendung“ zu treffen. So wird in diesem Dokument oft von funktionalen Anforderungen an die „SAK“ gesprochen (z.B. Möglichkeiten der Verschlüsselung) obwohl das entsprechende Software-Modul nicht Gegenstand der Evaluierung und Bestätigung ist und eigentlich die „Anwendung“ gemeint ist. Es ist Aufgabe des SAK-Herstellers ggf. in Kooperation mit einem PVS-Hersteller im Rahmen eines Gesamtpaketes („SAK“ aus Anwendersicht) den genauen Umfang der SAK zu definieren und nur die relevanten Module nach SigG zu bestätigen bzw. in einer Herstellererklärung aufzunehmen.

Anerkannte internationale Standards bilden die Basis. Diese werden profiliert gem. den spezifischen Anforderungen des vorliegenden Anwendungskontextes sowie Anforderungen, die sich für rechtsgültige elektronische Signaturen aus dem deutschen Signaturgesetz<sup>1</sup>, der Signaturverordnung ([SigG],[SigV]) und der daraus resultierenden Anwendung sicherer kryptographischer Algorithmen ([BNetzA\_Alg]) ergeben.

Entsprechend dem XML-Format der zu signierenden Dokumente ist die Empfehlung des W3C „XML-Signature Syntax and Processing“ [xmldsig] Grundlage dieser Spezifikation. Das Schema von [xmldsig] wird redefiniert gemäß Anforderungen, wie sie sich aus dem deutschen Signaturgesetz für qualifizierte Signaturen sowie Interoperabilitätsgesichtspunkten ergeben und in den ISIS-MTT-Spezifikationen [isis\_mtt], [isis\_opt] konkretisiert sind.

Diese Version fokussiert auf die parallele Mehrfachsignatur von CDA-R2-Dokumenten. Eine Signatur umfasst jeweils das gesamte Dokument und schließt ggf. für dieses Dokument nach dieser Spezifikation bereits schon applizierte XML-Signaturen **nicht** ein.

Intention dieser Spezifikation ist eine Erweiterbarkeit hinsichtlich

- Übertragbarkeit auf andere medizinische Dokumente im XML-Format,
- Signatur von Anhängen zu solchen Dokumenten in beliebigen (binär-) Formaten,
- Countersignatures („Gegenzeichnung“),
- Signaturen von Fragmenten eines CDA-R2-Dokuments,
- Skalierbarkeit bzgl. zukünftiger CDA-Versionen.

Im Fokus des Entwurfs stand weiter die Interoperabilität mit entsprechenden Spezifikationen der gematik<sup>2</sup> [gematik] für die Telematik-Infrastruktur des deutschen Gesundheitswesens.

---

<sup>1</sup> „qualifizierte elektronische Signaturen“ (QES)

<sup>2</sup> Gesellschaft für Telematikanwendungen der Gesundheitskarte; <http://www.gematik.de>

Die in diesem Dokument getroffenen Festlegungen sind bindend für die Umsetzung entsprechender Anwendungen; optionale Bestandteile sind jeweils im Kontext hervorgehoben.

Die Nutzung der Spezifikation ist für die Signatur von Dokumenten im Deutschen Gesundheitswesen bzw. für die Signatur mit dem elektronischen Arztausweis unentgeltlich.

## 2 Beteiligte

Die Eckpfeiler dieser Spezifikation wurden von einer interessierten Fachöffentlichkeit im ersten Halbjahr 2007 durch Abgleich von Konzepten sowie in mehreren Workshops abgestimmt.

Im Rahmen der Erstellung waren beteiligt:

Jörg Apitzsch, bremen online services GmbH & Co. KG (Editor);

Thomas Althoff, Ärztekammer Westfalen-Lippe; Hendrick Brockhaus, Siemens; Dr. Bernd Dressler, Deutsche Rentenversicherung Bund; Stefan Engelbert, Wrocklage; Dr. Erich Gehlen, Duria EG; Frank Jeschka, OpenLimit; Andrea Kassner, VHiTG; Viktor Krön, Ärztekammer Nordrhein; Hans-Joachim Marschall, KV Nordrhein; Dr. Volker Paul, Fraunhoferinstitut IBMT; Georgios Raptis, Bundesärztekammer; Arthur Steinell, Telemed-Online; Danny Zadach, DocExpert.

Für fachkundige Beratung wird gedankt:

Michael Bartkowiak, gematik

Nils Büngener, bremen online services GmbH & Co. KG

Dr. Jörg Caumanns, Fraunhofer-Institut für Software- und Systemtechnik ISST

Dr. Christian Geuer-Pollmann, Microsoft Deutschland GmbH

Marc Horstmann, bremen online services GmbH & Co. KG

Dr. Kai U. Heitmann, Heitmann Consulting & Services

Dr. Thomas Meischner, gematik

### 3 XML-Namensräume

Folgende XML Namespaces werden referenziert:

Präfix	XML Namespace	Referenzen
cda	urn:hl7-org:v3	[CDA2]
cdp	http://ws.gematik.de/fa/cds/CDocumentPayload/v1.0	dieses Dokument
ds	http://www.w3.org/2000/09/xmldsig#	[xmldsig]
xenc	http://www.w3.org/2001/04/xmlenc#	[xenc]
xades	http://uri.etsi.org/01903/v1.3.2#	[XAdES]
xmime	http://www.w3.org/2005/05/xmlmime	[xmime]
xs, xsd	http://www.w3.org/2001/XMLSchema	[xmlschema]

Tabelle 1: Namespaces

## 4 Typ und Positionierung von Signaturelementen

Das XML-Schema für CDA-R2-Dokumente sieht keine konkreten Signaturelemente vor und bietet auch keine Erweiterungspunkte für Elemente aus fremden Namensräumen entsprechend den Konstrukten von [xmldsig] an.

Mit Blick auf eine Übertragbarkeit der hier dargestellten Lösung auf klinische Dokumente in XML-Formaten mit beliebigen Schemata und um auf eine Schemaerweiterung in [CDA2] um Signaturelemente verzichten zu können, werden die Signaturelemente außerhalb der Struktur `<cda:ClinicalDocument>` positioniert. Bezüglich des CDA-R2-Schemas handelt es sich damit um eine „Detached Signature“ gemäß [xmldsig], die in ein umhüllendes Schema eingebettet wird, welches lediglich als „Container“ für CDA-R2-Dokumente, zugehörige Signaturen incl. optionaler Anhänge dient.



**Hinweis:** Dieses umhüllende Schema dient lediglich der Sicherung des Zusammenhangs der Signaturelemente mit dem jeweiligen CDA-R2-Dokument in einem strukturierten Dateiobjekt; im Fokus steht weder der elektronische Transport dieser Objekte noch ihre Ablage im Sinne von Archivierung. Die im Schema vorgesehene Möglichkeit, den Inhalt des Containers zu verschlüsseln, ist optional.

Mit diesem Ansatz ist es möglich, Signaturelemente sowohl für parallele Mehrfachsignaturen als auch Übersignaturen für das CDA-R2-Dokument als Ganzes wie auch als Fragmente aufzubauen. Weiter erlaubt die Wahl dieses Typs neben der Aufnahme von Signaturen für Dokumente, die auf anderen Schemata basieren, auch den Aufbau von Signaturelementen für beliebige Nicht-XML-Formate<sup>3</sup>.

Um eine Referenzierung der signierten Inhalte aus den Signaturelementen heraus zu vereinfachen, ist die Gesamtstruktur so ausgelegt, dass jeweils über den Id-Mechanismus gem. [xmldsig] referenziert werden kann (siehe [xmldsig], `<ds:Reference URI>`-Attribut).

[Abbildung 1] gibt eine Übersicht zur Einbettung der genannten Elemente in diese Metastruktur `<cdp:ContentPackage>`; das Element `<cdp:StructuredContent>` dient dabei der Aufnahme der eigentlichen medizinischen Dokumente im XML-Format – Inhalt ist z.B. das aktuelle CDA-R2-Dokument. Das Schema selbst ist in Abschnitt [Schema CDocumentPayload] im Detail erläutert.

<sup>3</sup> Grundsätzlich können damit auch bereits signierte Dokumente – z.B. PKCS7-signierte Dokumente – Teil eines ContentPackages sein, sofern die SAK die Lesbarkeit sicherstellt.



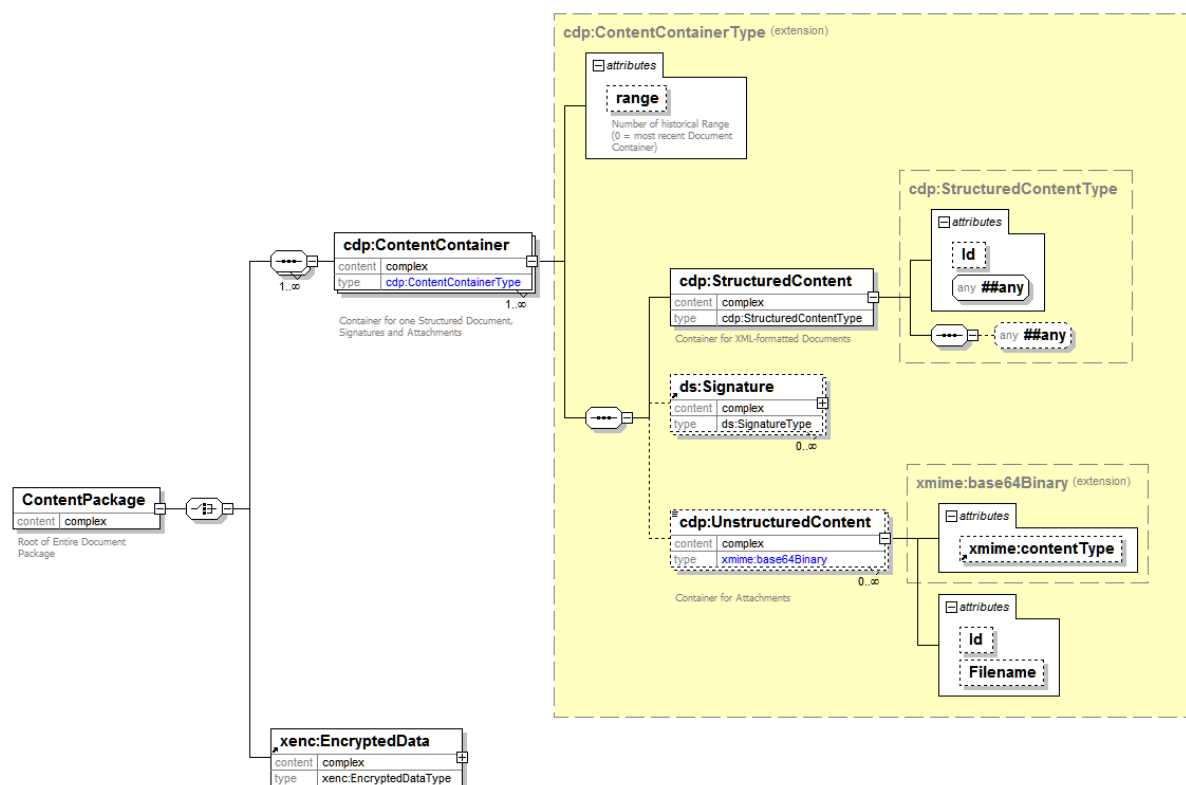


Abbildung 1: Übersicht der Metastruktur cdp:ContentPackage

## 5 Schema CDocumentPayload

Die Version des Schemas ist dem Namensraum zu entnehmen; die aktuelle Version führt:

<http://ws.gematik.de/fa/cds/CDocumentPayload/v1.0>

### 5.1 Root-Element ContentPackage

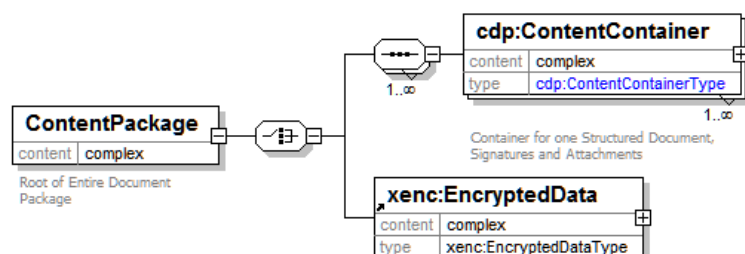


Abbildung 2: Root-Element cdp:ContentPackage

Das Root-Element enthält alternativ eine Sequenz von Elementen

- `<cdp:ContentContainer>` zur Aufnahme von strukturierten Dokumenten, Signaturen und Attachments

oder

- `<xenc:EncryptedData>` – verschlüsselte `<cdp:ContentContainer>` gem. [xenc]; diese werden im Rahmen dieser Spezifikation nicht näher betrachtet.

► **Hinweis:** Eine SAK kann die Ver- und Entschlüsselung unterstützen; ansonsten muss diese Funktionalität im Kontext der gewählten Telematik-Infrastruktur von den Transportkomponenten abgedeckt werden. Siehe auch Abschnitt [5.4].

```

<xs:element name="ContentPackage">
  <xs:annotation>
    <xs:documentation>Root of Entire Document Package</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:choice>
      <xs:sequence>
        <xs:element name="ContentContainer" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Container for one Structured Document,
              Signatures and Attachments</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:complexContent>
              <xs:extension base="cdp:ContentContainerType"/>
            </xs:complexContent>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:element name="EncryptedData" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>Encrypted Content Container</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="xenc:EncryptedDataType"/>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
    </xs:choice>
  </xs:complexType>
</xs:element>
  
```

```

        </xs:complexType>

        </xs:element>

    </xs:sequence>

    <xs:sequence>

        <xs:element ref="enc:EncryptedData" />

    </xs:sequence>

</xs:choice>

</xs:complexType>

</xs:element>

```

Listing 1: cdp:ContentPackage

## 5.2 ContentContainerType

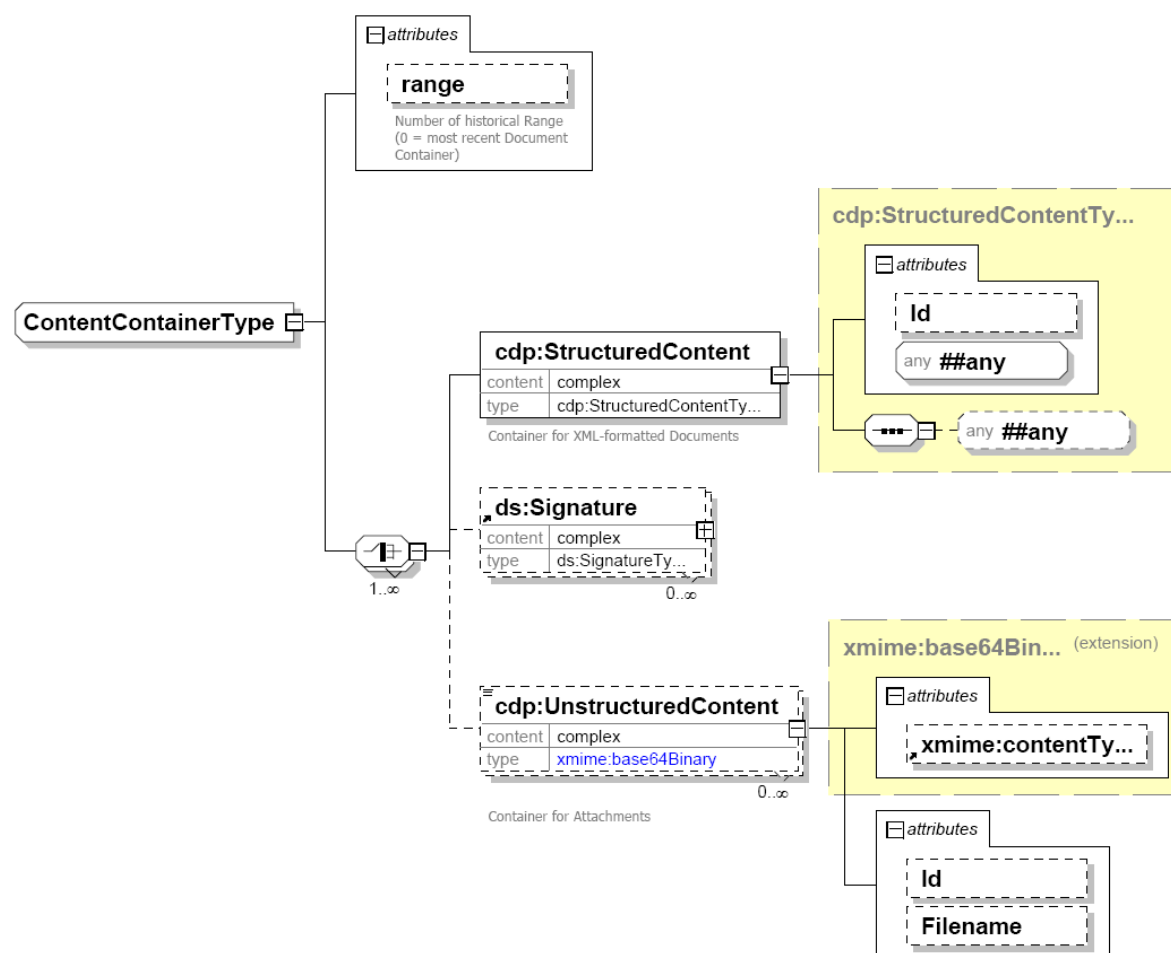


Abbildung 3: cdp:ContentContainerType

Die Elemente `<cdp:ContentContainer>` nehmen in `<cdp:StructuredContent>` die strukturierten XML-Dokumente (siehe Abschnitt [5.4]), die optionalen Signaturelemente in `<ds:signature>` und/oder Attachments in Elementen `<cdp:UnstructuredContent>` auf. `<cdp:ContentContainer>` hat die Kardinalität 1..n, da es möglich sein soll, zum Paket `<cdp:ContentPackage>` neben dem aktuellen Dokument auch solche der Episode des Falls hinzuzufügen.

Das Attribut `<cdp:range>` vom Typ `<xs:int>` dient zur Kennzeichnung der zeitlichen Reihenfolge des jeweiligen Pakets in der Episode in aufsteigender Nummerierung, beginnend bei null.

► **Hinweis:** Eine SAK bzw. Anwendung muss zum Aufbau der Gesamtstruktur

`<cdp:ContentPackage>` beim Zusammenstellen der `<cdp:ContentContainer>` das Attribut `<cdp:Range>` setzen. Wenn möglich, sollte es entsprechend der zeitlichen Reihenfolge der Pakete gesetzt werden. In jedem Fall muss der aktuellste `<cdp:ContentContainer>` in `<Range>` mit einem Wert „0“ belegt werden; diese Belegung für `<cdp:Range>` darf innerhalb einer Instanz eines `<cdp:ContentPackage>` nur einmal vorkommen. I.d.R. wird davon ausgegangen, dass diese Information auch mit geeigneten Datumsfeldern oder anderen Informationen in den zugrunde liegenden strukturierten klinischen Dokumenten automatisiert korreliert werden kann; in CDA-R2 sind dies z.B. die Header-Attribute `<cda:setId>` und `<cda:VersionNumber>` (siehe [VHitG-Lf], S. 49).

► **Hinweis:** Für SAKs bzw. Anwendungen zur Anzeige der Gesamtstruktur

`<cdp:ContentPackage>` und Prüfung der Signaturen wird empfohlen, die Bestandteile der Gesamtstruktur zu visualisieren und dem Nutzer die Möglichkeit selektiven Zugriffs auf die einzelnen `<cdp:ContentContainer>` zu geben. Ergebnisse zu Signaturprüfungen sollten sich auf die jeweils ausgewählten `<cdp:ContentContainer>` beziehen.

► **Hinweis:** Um evt. Probleme mit XSLT-und XPATH-Prozessoren zu vermeiden, sollten im Kopfelement einer Instanz eines `<cdp:ContentPackage>` bereits alle XML-Namespaces deklariert werden, die innerhalb des `<cdp:ContentPackage>` genutzt werden. Insbesondere macht der Wechsel von Default-Namespaces innerhalb eines komplexen XML-Dokuments Probleme und sollte daher ausgeschlossen werden. Dies impliziert, dass auch nur ein Dokumententyp – in diesem Fall CDA-R2 – mit dem Default-Namespace belegt werden kann. Aus Gründen der Komplexitätsminderung wird empfohlen, innerhalb der Teilbäume `<cdp:ContentContainer>` immer Dokumente gleichen Typs aufzunehmen; beim einem Mix sind die Elemente der Dokumente, die auf anderen XML-Schemata basieren mit einem entsprechend Namespace-Prefix zu belegen.

► **Hinweis:** Um Probleme mit XSLT-und XPATH-Prozessoren zu vermeiden, werden im aktuellen CDP-Schema bereits alle benötigten Subschemata importiert, insbesondere auch das CDA-R2-Schema für Arztbriefe. Sollten in ein `<cdp:ContentPackage>` andere Dokumente aufgenommen werden, sollten deren Schemata entsprechend im CDP-Schema bereits importiert werden – auch wenn sie erst innerhalb eines konkreten `<cdp:StructuredContent>` - der vom Typ `<##any>` ist – benötigt werden.

Der Aufbau der optionalen Signaturelemente `<ds:Signature>` wird im Abschnitt [7] beschrieben.

Optional können zusätzlich Anhänge in beliebigen Binärformaten in dieser Sequenz liegen, die in den Elementen `<cdp:UnstructuredContent>` im Format `<xs:base64Binary>` gehalten werden.

```
<xs:complexType name="ContentContainerType">
  <xs:sequence>
    <xs:element name="StructuredContent"
      type="cdp:StructuredContentType">
      <xs:annotation>
        <xs:documentation>Container for XML-formatted Documents
      </xs:documentation>
      </xs:annotation>
    </xs:element>
```

```

<xs:element ref="ds:Signature" minOccurs="0"
              maxOccurs="unbounded"/>

<xs:element name="UnstructuredContent" minOccurs="0"
              maxOccurs="unbounded">

  <xs:annotation>
    <xs:documentation>Container for attachments
    </xs:documentation>
  </xs:annotation>

  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="xmime:base64Binary">
        <xs:attribute name="Id" type="xs:ID" use="optional"/>
        <xs:attribute name="Filename" type="xs:string"
          use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="range" type="xs:int">
  <xs:annotation>
    <xs:documentation>Number of historical Range (0 = most recent
      Document Container)</xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:complexType>

```

Listing 2: cdp:ContentContainerType

## 5.3 Anhänge zu XML-strukturierten Dokumenten

<cdp:UnstructuredContent> ist eine Extension von <xmime:base64Binary> gem. der W3C-Note „Describing Media Content of Binary Data in XML“ [xmime]. Dieser Typ erweitert <xs:base64Binary> um ein Attribut <xmime:contentType>; zulässige Mime-Types und die entsprechenden Einträge in <xmime:contentType> sind [Tabelle 2] zu entnehmen.

In dieser Version der Spezifikation ist die Signatur von Anhängen nicht vorgesehen.

### 5.3.1 Allgemeine Behandlung von Anhängen

<cdp:UnstructuredContent> erweitert <xmime:base64Binary> um ein optionales Attribut <Id> vom Typ <xs:ID> (benötigt für die in späteren Versionen dieser Spezifikation vorgesehene Referenzierung aus Signaturelementen für Attachments) und ein optionales Attribut <Filename> vom Typ <xs:string>, in dem bei Bedarf der Dateiname eines Attachments geführt werden kann, um so eine gewisse Unterstützung bei der Visualisierung und/oder gezielten Trennung in Bestandteile eines <cdp:ContentPackage> geben zu können.

► **Hinweis:** Der Wert des Attributs `<Filename>` sollte nach Möglichkeit einen Dateinamen ohne Directory-Hierarchie enthalten, soweit die Eindeutigkeit der Dateinamen von Attachments innerhalb eines `<cdp:ContentPackage>` sichergestellt werden kann. Bei der Dekomposition eines `<cdp:ContentPackage>` sind die Attachments in das gleiche Directory einzustellen wie die XML-Datei des `<cdp:StructuredContent>`. Enthält der Dateiname des Attachments eine Directory-Angabe, ist diese unterhalb des Directories anzulegen, in den die XML-Datei des `<cdp:StructuredContent>` eingestellt wird.

Folgende Mime-Types werden in Elementen `<cdp:UnstructuredContent>` zugelassen<sup>4</sup>:

Name	MIME-Type	Beschreibung
AUDIO	audio/basic	Basisformat für Audiodaten
CSV	text/csv	einfach strukturierte Daten
DICOM	application/dicom	DICOM-Objekte
EXCEL	application/vnd.ms-excel	Microsoft Excel
GIF	image/gif	Basisformat für eingescannte Dokumente
HL7-CDA	application/x-hl7-cda-level-one+xml	HL7-CDA-Dokumente
HTML	text/html	Texte in HTML-Format
JPEG	image/jpeg	Basisformat für Bilder, die von allen Systemen angezeigt werden können
MP3	audio/mpeg	MPEG Layer 3 komprimierte Audiodaten
MPEG	video/mpeg	Filme
ODT	application/vnd.oasis.opendocument.text	Texte im OpenDocument-Format
PCX	image/x-pcx	Bilder im PCX-Format
PDF	application/pdf	Basisformat für seitenorientierte Darstellungen
PNG	Image/png	Bilder im PNG-Format
RTF	application/x-rtf	Texte in Rich-Text-Format
TEXT	text/plain	Texte ohne Formatierung (ASCII)
TIFF	image/tiff	Basisformat für eingescannte Dokumente
WORD	application/msword	Microsoft Word
xDT	application/xdtd	strukturierte ASCII-Text-Dateien im xDT-Format
XML	application/xml	strukturierte Daten im XML-Format

Tabelle 2: Zulässige Mime-Types für Attachments

► **Hinweis:** Die gewählte Telematik-Infrastruktur kann Größenbeschränkungen für den Transport von Dokumenten vorgeben, die bei der Erstellung von Paketen von CDA-R2-Dokumenten zu beachten

<sup>4</sup> Registrierungen sind für sämtlich Mime-Typen <http://www.iana.org/assignments/media-types/> zu entnehmen; Ausnahme: für den Type „xDT“ liegen nähere Informationen unter <http://www.kbv.de/ita/4274.html> vor.

sind; dies gilt natürlich insbesondere bei Aufnahme binärer Attachments. Bitte hier insbesondere die aktuelle Dokumentation der gematik<sup>5</sup> beachten.

► **Hinweis:** Nicht alle aufgeführten Mime-Types sind wg. aktiver bzw. auch verborgener Inhalte geeignet für eine sinnvolle Applizierung von Signaturen über diese Container. In einer zukünftigen Version dieser Spezifikation können die Typen eingeschränkt werden, die sich für qualifizierte Signaturen als bestätigbar herauskristallisieren.

### 5.3.2 Anhänge, die aus CDA-R2-Dokumenten referenziert werden

Das Schema von CDA-R2-Dokumenten sieht seinerseits bereits die Referenzierung extern geführter Anhänge vor, wie in folgendem Ausschnitt eines CDA-R2-Dokuments verdeutlicht:

---

<sup>5</sup> Stand September 2007:  
[http://www.gematik.de/\(S\(bnxgwy551efsvs45lgw4by45\)\)/Uebersichtsseite\\_\\_\\_Release\\_2\\_3\\_1.Gematik](http://www.gematik.de/(S(bnxgwy551efsvs45lgw4by45))/Uebersichtsseite___Release_2_3_1.Gematik)

```

.....
<component>

  <!-- Bildbefund Komponente mit externen Referenzen -->

  <section>

    <code code="8709-8" codeSystem="2.16.840.1.113883.6.1"
      codeSystemName="LOINC" />

    <title>30.11.2005: Untersuchung der Haut</title>

    <text>15.12.2005: Bildbefund<renderMultiMedia
      referencedObject="MM200511301" />

    </text>

    <entry>

      <observationMedia classCode="OBS" moodCode="EVN"
        ID="MM200511301">

        <id root="10.23.4567.345" />

        <value xsi:type="ED" mediaType="image/jpeg">

          <reference value="BBDA4D01.JPG" />

        </value>

      </observationMedia>

    </entry>

  </section>

</component>

...

```

Listing 3: Referenz aus einem CDA-R2-Dokument auf einen extern geführten Bildbefund

Das HL7-Basischema `datatypes.base.xsd` sieht für die Aufnahme von Anhängen in CDA-R2-Dokumente folgenden Datentyp `<ED>` vor:

```

<xsd:complexType name="ED" mixed="true">
  <xsd:complexContent mixed="true">
    <xsd:extension base="BIN">
      <xsd:sequence>
        <xsd:element name="reference" type="TEL" minOccurs="0"/>
        <xsd:element name="thumbnail" type="thumbnail" minOccurs="0"/>
      </xsd:sequence>
      <xsd:attribute name="mediaType" type="cs" use="optional"
        default="text/plain"/>
      <xsd:attribute name="language" type="cs" use="optional"/>
      <xsd:attribute name="compression" type="cs_CompressionAlgorithm"
        use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</complexType>

```



```

<xsd:attribute name="integrityCheck" type="bin" use="optional"/>
<xsd:attribute name="integrityCheckAlgorithm"
  type="cs_IntegrityCheckAlgorithm" use="optional" default="SHA-1"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

```

Listing 4: Complex Type &lt;ED&gt; aus datatypes-base.xsd

Für solche Anhänge gilt:

1. Der Inhalt des Attachments wird im Format base64binary in <cdp:UnstructuredContent> eingestellt.
2. Der Inhalt des ED-Attributs <mediaType> wird in das Attribut <contentType> von <cdp:UnstructuredContent> übernommen.
3. Der Inhalt des ED-Attributs <reference> wird in das Attribut <Filename> von <cdp:UnstructuredContent> übernommen.

Für eine Überprüfbarkeit der Unversehrtheit der Anhänge sind weiter die ED-Attribute <integrityCheck> und <integrityCheckAlgorithm> aufzubauen:

- <integrityCheck> nimmt den Hashwert des Attachments im Format base64binary auf
- <integrityCheckAlgorithm> kennzeichnet den verwendeten Algorithmus. Abweichend von der Default-Belegung „SHA-1“ ist hier „SHA-256“ in dieses Attribut einzutragen und für die Digest-Bildung der Algorithmus „http://www.w3.org/2001/04/xmlenc#sha256“ anzuwenden.

► **Hinweis:** Anwendungen zur Signaturerstellung und -prüfung von CDA-R2-Dokumenten müssen solche Elemente vom Typ <ED> lokalisieren und wie beschrieben vervollständigen können. Die Vervollständigung aller solcher Elemente muss vor Applikation der ersten Signatur geschehen. Weiter müssen sie dem Nutzer die Möglichkeit anbieten, bei Bedarf die Unversehrtheit der hier referenzierten Anhänge zu überprüfen. Die Anwendungen zur Signaturerstellung und -prüfung müssen dabei deutlich kenntlich machen, was genau signiert werden soll bzw. wurde, d.h. dass diese Anhänge (trotz Aufnahme des Hashwertes zur Integritätssicherung der Referenz) **nicht rechtsgültig signiert sind**.

### 5.3.3 ContentPackage(s) als Anhang

Es soll insbesondere auch möglich sein, bei Bedarf einem aktuellen Arztbrief in <cdp:StructuredContent> innerhalb seines <cdp:ContentContainer> ein zuvor schon einmal aufgebautes <cdp:ContentPackage> beizufügen. Dieses ist dann wie jedes Attachments im Format base64binary in einen Container <cdp:UnstructuredContent> einzustellen und mit dem Mime-Type application/xml zu kennzeichnen.

► **Hinweis:** Anwendungen müssen Attachments von diesem Typ erkennen und bei der Anzeige / Signaturprüfung als gesonderte, selbstständige Einheiten vom Typ <cdp:ContentPackage> behandeln.

► **Hinweis:** Beim Aufbau und dem Parsen solcher Packages ist zu beachten, dass das CDP-Schema an dieser Stelle beliebige Verschachtelungen zulässt. Beim Aufbau muss mehrfache Base64-Codierung vermieden werden um die Paketgröße nicht unnötig aufzublähen. Anwendungen müssen beim Aufbau solcher Pakete eine maximale Rekursionstiefe von 2 sicherstellen, um exzessive Speicheranforderungen bei der Dekomposition solcher Pakete zu vermeiden.

Alternativ können Arztbriefe der Episode inkl. ihrer Anhänge – wie in Kapitel [5.2] dargestellt – als weitere Instanzen von `<cdp:ContentContainer>` in einem `<cdp.ContentPackage>` aufgenommen werden.

► **Hinweis:** In diesem Fall müssen Anwendungen sicherstellen, dass die Regel eindeutiger Belegungen der Attribute vom Type `<xsd:ID>` innerhalb eines XML-Dokuments nicht verletzt werden. Sollten in einer weiteren `<cdp:ContentContainer>`-Instanz identische Belegungen von Id-Attributen auftreten wie in schon im aktuellen `<cdp:ContentPackage>` vorkommend, ist ein solches in ein separates `<cdp:ContentPackage>` einzustellen; dieses kann sodann im Format `base64binary` in einen Container `<cdp:UnstructuredContent>` mit dem Mime-Type `application/xml` eingestellt werden. Es muss für die spätere Dekomposition ein im Gesamtpaket eindeutiges Attribut `<Filename>` vergeben werden.

## 5.4 Hinweise zu verschlüsselten ContentContainern

Die Sequenz der `<cdp:ContentContainer>` können im Kontext anderer, hier nicht betrachteter Infrastrukturen gem. [xenc] verschlüsselt werden:

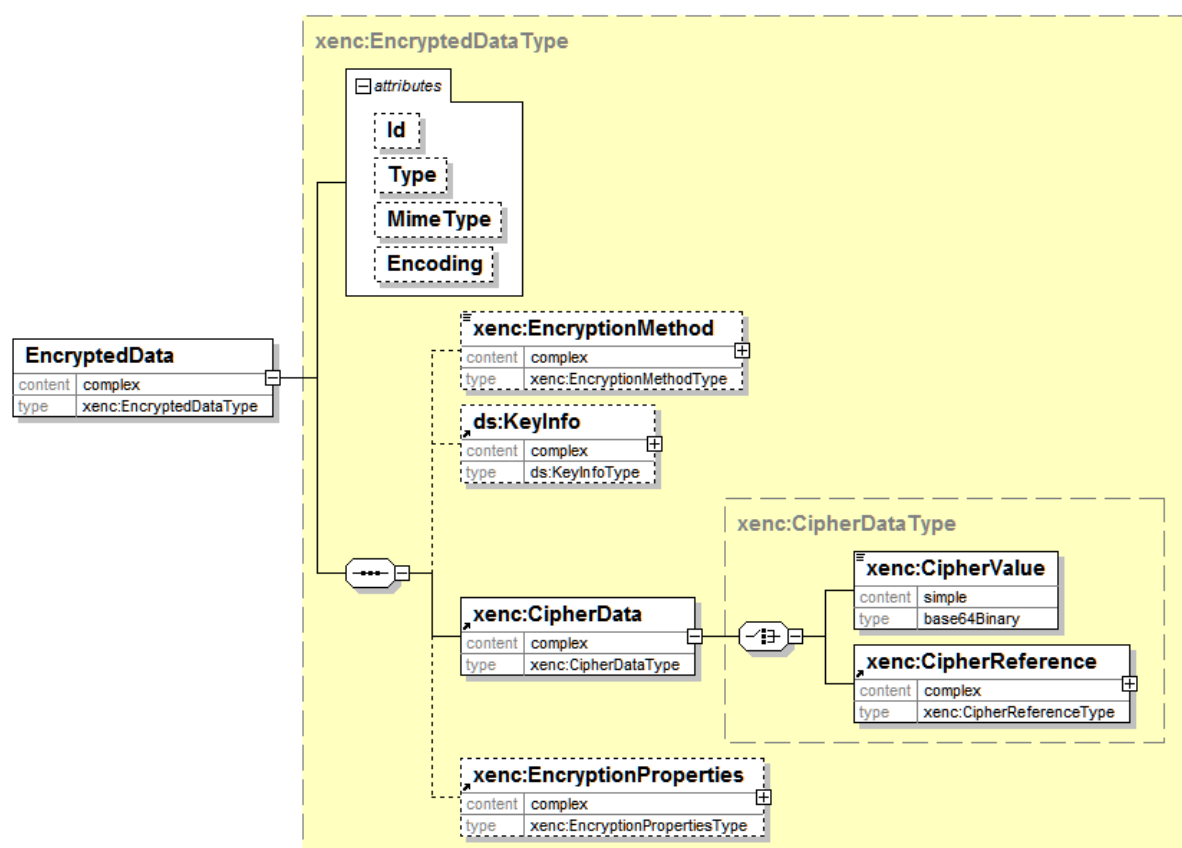


Abbildung 4: Übersicht xenc:EncryptedData

► **Hinweis:** [xenc] sieht im Element `<xenc:EncryptedData>` die Angaben zu Schlüssel und Methode der Verschlüsselung als optionale Elemente vor. Es wird empfohlen – wie in Abbildung 4 dargestellt – die Elemente `<ds:KeyInfo>` und `<xenc:EncryptionMethod>` in jedem Fall bei der Verschlüsselung aufzubauen, damit diese für die Entschlüsselung von `<xenc:CipherData>` benötigten Informationen zur Verfügung stehen. **Es bleibt entsprechenden Anwendungen überlassen, ob dieser Empfehlung gefolgt wird.**

Es wird jeweils die gesamte Sequenz `<cdp:ContentContainer>` verschlüsselt. Aufgrund der hybriden Verschlüsselung (die Inhalte selbst werden symmetrisch verschlüsselt, der symmetrische

Schlüssel sodann asymmetrisch mit dem eingesetzten Cipher-Key) ist es möglich, die Inhalte auch mehrfach zu verschlüsseln. Näheres ist der Spezifikation [xenc] zu entnehmen.

## 5.5 StructuredContentType

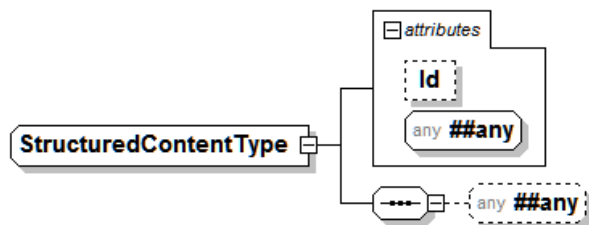


Abbildung 5: cdp:StructuredContentType

Ein Element `<cdp:StructuredContent>` kann XML-Dokumente aus beliebigen Namensräumen aufnehmen; im Fall CDA-R2-Dokumente ist dies eine Struktur vom Typ `<POCD_MT000040.ClinicalDocument>` aus dem Namensraum „urn:hl7-org:v3“.

Optional wird ein Attribut `<Id>` vom Typ `<xs:ID>` geführt, welches dazu dienen kann, den gesamten Teilbaum aus Signaturelementen heraus zu referenzieren.

Im Falle von CDA-R2-Dokumenten ist dies nicht nötig, da das Root-Element `<ClinicalDocument>` selbst ein Id-Attribut führt. Das Id-Attribut von `<cdp:StructuredContent>` soll dann - und nur dann - belegt werden, wenn das Root-Element des einzuhängenden XML-Dokuments selbst kein Id-Attribut zulässt. Die den Container generierende Komponente muss sicherstellen, dass das zu referenzierende XML-Element genau ein eindeutiges Id-Attribut hat.

```
<xs:complexType name="StructuredContentType">
  <xs:sequence>
    <xs:any namespace="##any" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="Id" type="xs:ID"/>
  <xs:anyAttribute namespace="##any" />
</xs:complexType>
```

Listing 5: cdp:StructuredContentType

## 6 Belegung von Id-Attributen

### 6.1 Sicherung der Eindeutigkeit

[xmld-schema] fordert die Eindeutigkeit der Werte von Id-Attributen innerhalb eines XML-Dokuments. Wie in Kapitel [7.4] ausgeführt, werden diese aus den Signaturelementen heraus referenziert, um die Teilbäume der XML-Dokumente zu adressieren, die jeweils in die Signatur einbezogen werden. In einer späteren Version dieser Spezifikation gilt dies auch für Signaturen von Attachments.

► **Hinweis:** *Auf Referenzierungsmechanismen über XPATH-Ausdrücke [xpath] wird abweichend zu Spezifikationen der gematik verzichtet. Der Referenzierungsmechanismus über Id-Attribute stellt sicher, dass Referenzen auch der der Zusammenführung von selbstständigen XML-Dokumenten in einem <cdp:ContentPackage> problemlos aufgelöst werden können.*

Da in der Gesamtstruktur <cdp:ContentPackage> auch ursprünglich selbstständige XML-Dokumente zusammengeführt werden können, ist sicherzustellen, dass die Eindeutigkeit der Werte der Id-Attribute auch dann gewährleistet ist. Kann dieses nicht gewährleistet werden, weil z.B. Dokumente der Episode in ein <cdp:ContentPackage> aufgenommen werden, die nicht mehr geändert werden können, sind die Hinweise am Ende des Kapitels [5.3.3] zu beachten.

Algorithmen zur Erzeugung von eindeutigen Identifiern stellt die Spezifikation „A Universally Unique Identifier (UUID) URN Namespace“ zur Verfügung [RFC4122]. Erzeugt wird dabei jeweils ein 128-Bit-String. SAK-Implementierungen müssen diesen Mechanismus unterstützen. Es muss hier die OID-Produktionsregel der gematik unterstützt werden, die in [gematikGArch], Kapitel 12.8 .1 detailliert dargestellt ist. Weiter darf die Produktionsregel von Id-Attributen gem. [xmld-schema] nicht verletzt werden.

► **Hinweis:** *Eine SAK bzw. Anwendung zum Aufbau der Gesamtstruktur muss sicherstellen, dass keine XML-Dokumente in das Gesamtpaket einbezogen werden können, die die Id-Attribute nicht gem. dieser Vorschrift aufgebaut haben – d.h. die Eindeutigkeit der Werte der Id-Attribute muss bei der Integration von XML-Dokumenten der Episode überprüft werden. Ist dies nicht der Fall, können solche XML-Dokumente als Attachment behandelt werden, d.h. sie müssen in Base64-Codierung in ein Element <cdp:UnstructuredContent> eingestellt werden.*

► **Hinweis:** *Alle Id-Attribute müssen – wie alle Inhalte von <cdp:StructuredContent> - vor Applikation der ersten Signatur Vollständigkeit aufgebaut sein. Eine SAK bzw. Anwendung zum Aufbau der Gesamtstruktur muss sicherstellen, dass Inhalte signierte Teilbäume der Gesamtstruktur nach Applikation einer Signatur nicht mehr verändert werden können.*

### 6.2 ID-Attribute in CDA-R2-Dokumenten

Folgende Teilbäume des Schemas von CDA-R2-Dokumenten werden aus Signaturelementen über ihre Id-Attribute referenziert. Diese müssen daher mit einer UUID belegt werden:

**Root <cda:ClinicalDocument>**, aus jeder Signatur des Gesamtdokuments referenziert:

```
<xs:complexType name="POCD_MT000040.ClinicalDocument">
  <xs:sequence> ....(Header- und Body-Elemente von ClinicalDocument) </xs:sequence>
  <xs:attribute name="nullFlavor" type="NullFlavor" use="optional"/>
  <xs:attribute name="classCode" type="ActClinicalDocument" use="optional"
    fixed="DOCCLIN"/>
  <xs:attribute name="moodCode" type="ActMood" use="optional"
    fixed="EVN"/>
  <xs:attribute name="Id" type="xs:ID"/>
```

```
</xs:complexType>
```

Listing 6: cda:ClinicalDocumentType

**<cda:LegalAuthenticator>**, spezifische ergänzende Informationen zum jeweiligen Unterzeichner, der eine Signatur auf das Gesamtdokument appliziert:

```
<xs:complexType name="POCD_MT000040.LegalAuthenticator">
  <xs:sequence>
    ... (Subelemente von LegalAuthenticator)
  </xs:sequence>
  <xs:attribute name="nullFlavor" type="NullFlavor" use="optional"/>
  <xs:attribute name="typeCode" type="ParticipationType" use="optional"
    fixed="LA"/>
  <xs:attribute name="contextControlCode" type="ContextControl"
    use="optional" fixed="OP"/>
  <xs:attribute name="Id" type="xs:ID"/>
</xs:complexType>
```

Listing 7: cda:LegalAuthenticatorType

In einer zukünftigen Version sollen auch Signaturen möglich sein, die sich nur auf fachliche Fragmente des Dokuments erstrecken, für die ein bestimmter Autor verantwortlich zeichnet. Diese Fragmente werden in CDA-R2-Dokumenten durch **<cda:Sequence>**-Elemente geklammert, denen im zugehörigen Signaturelement jeweils ein **<cda:Authenticator>**-Element zugeordnet werden kann:

**<cda:Section>**, klammert einen fachlichen Abschnitt des CDA-R2-Dokuments:

```
<xs:complexType name="POCD_MT000040.Section">
  <xs:sequence>
    ... Subelemente von Section
  </xs:sequence>
  <xs:attribute name="ID" type="xs:ID"/>
  <xs:attribute name="nullFlavor" type="NullFlavor" use="optional"/>
  <xs:attribute name="classCode" type="ActClass" use="optional"
    fixed="DOCSECT"/>
  <xs:attribute name="moodCode" type="ActMood" use="optional"
    fixed="EVN"/>
</xs:complexType>
```

Listing 8: cda:SectionType

**<cda:Authenticator>**, spezifische ergänzende Informationen zum jeweiligen Unterzeichner, der eine Signatur auf ein oder mehrere **<cda:Section>**-Elemente appliziert:

```
<xs:complexType name="POCD_MT000040.Authenticator">
  <xs:sequence>
```

```
    ...Subelemente von Authenticator  
  
  </xs:sequence>  
  
  <xs:attribute name="nullFlavor" type="NullFlavor" use="optional"/>  
  
  <xs:attribute name="typeCode" type="ParticipationType" use="optional"  
    fixed="AUTHEN"/>  
  
  <xs:attribute name="ID" type="xs:Id"/>  
  
</xs:complexType>
```

Listing 9: cda:AuthenticatorType

## 7 Profilierung von XML-Signature

Die folgende Profilierung der Spezifikation [xmldsig] folgt der Spezifikation [isis\_mtt], Part 8: „XML Signature and Encryption Message Formats“ vom März 2004, die insbesondere auch die Anforderungen aufnimmt, die sich aus dem deutschen Signaturgesetz ergeben [SigG], [SigV].

Zur Zeit der Erstellung dieser Spezifikation wird die Profilierung insbesondere hinsichtlich der aktuellen Empfehlungen der Bundesnetzagentur bezüglich geeigneter kryptografischer Algorithmen bei der Signaturerzeugung überarbeitet; zur derzeit aktuellsten verfügbaren Version siehe [BNetzA\_Alg]. Insbesondere wird die Verwendung des Hash-Algorithmus SHA-1 für die Erstellung qualifizierter Signaturen nur noch bis Ende 2007 als geeignet angesehen. Zur Verwendung von SHA-256, SHA-512 und Algorithmen, die auf elliptischen Kurven basieren, siehe [BNetzA\_Alg].

Dabei ist jedoch zu berücksichtigen, dass Anwendungen zur Validierung von (älteren) signierten Dokumenten auch mit Algorithmen umgehen können müssen, die aktuell nicht mehr als ausreichend sicher für die Signaturerstellung angesehen werden. Eine Verankerung ausschließlich „aktueller“ Algorithmen direkt im Schema würde in diesen Fällen zu Problemen bei der Schemavalidierung führen. Die hier festgelegten Restriktionen zu Algorithmen gelten daher aus aktueller Sicht als Anforderung an SAKs für die Signaturerstellung; sie müssen gem. zukünftiger Erkenntnisse zur Stärke kryptografischer Algorithmen jeweils zeitnah aktualisiert werden.

Ergänzend zu [isis\_mtt] sind hier auch Festlegungen übernommen aus ETSI „XML Advanced Electronic Signatures“ [XAdES] zur Aufnahme von Attributzertifikaten und ergänzenden Informationen in die Signatur.

### 7.1 Signature Element

Ein X509-V3-Signaturzertifikat mit dem öffentlichen Schlüssel muss in die Signatur aufgenommen werden; die Kardinalität von `<ds:KeyInfo>` ist 1. Mindestens zwei Elemente `<ds:Object>` müssen vorhanden sein, da zusätzliche Informationen in die Signatur einzubeziehen sind (Erläuterungen zu den `<ds:Object>`-Elementen siehe Abschnitte [7.5], [7.6]).

```
<xsd:complexType name="SignatureType">
  <xsd:complexContent>
    <xsd:restriction base="ds:SignatureType">
      <xsd:sequence>
        <xsd:element ref="ds:SignedInfo"/>
        <xsd:element ref="ds:SignatureValue"/>
        <xsd:element ref="ds:KeyInfo"/>
        <xsd:element ref="ds:Object"
          minOccurs="2" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

Listing 10: Restriktion ds:SignatureType

## 7.2 SignedInfo Element

Folgende Einschränkungen gelten für Kindelemente von <ds:SignedInfo>:

### 7.2.1 CanonicalizationMethod

Zur Vermeidung von Namespace-Problemen bei der Verarbeitung signierter XML-Dokumente (z.B. innerhalb von SOAP-Nachrichten) ist ausschließlich die exklusive Kanonisierung zugelassen.

```
<xsd:complexType name="CanonicalizationMethodType">
  <xsd:complexContent>
    <xsd:restriction base="ds:CanonicalizationMethodType">
      <xsd:attribute name="Algorithm" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:anyURI">
            <xsd:enumeration
              value="http://www.w3.org/2001/10/xml-exc-c14n#" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

Listing 11: Restriktion ds:CanonicalizationMethodType

### 7.2.2 SignatureMethod Element

Zulässige Algorithmen bei der Signaturerzeugung müssen den jeweils aktuellen Empfehlungen der Bundesnetzagentur folgen. Auf dem Stand von April 2007 ergibt sich folgende Liste:

```
<xsd:complexType name="SignatureMethodType">
  <xsd:complexContent>
    <xsd:restriction base="ds:SignatureMethodType">
      <xsd:attribute name="Algorithm" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:anyURI">
            <xsd:enumeration value=
              "http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" />
            <xsd:enumeration value=
              "http://www.w3.org/2001/04/xmldsig-more#rsa-sha512" />
            <xsd:enumeration value=
              "http://www.w3.org/2001/04/xmldsig-more/rsa-ripemd160" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```



```

    </xsd:attribute>
  </xsd:restriction>
</xsd:complexContent>
</xsd:complexType>

```

Listing 12: Restriktion ds:SignatureMethodType

### 7.2.3 Reference, DigestMethod Element

Zulässige Hash-Algorithmen müssen den jeweils aktuellen Empfehlungen der Bundesnetzagentur folgen. Auf dem Stand von April 2007 ergibt sich folgende Liste:

```

<xsd:complexType name="DigestMethodType">
  <xsd:complexContent>
    <xsd:restriction base="ds:DigestMethodType">
      <xsd:attribute name="Algorithm" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:anyURI">
            <xsd:enumeration
              value="http://www.w3.org/2001/04/xmlenc#sha256"/>
            <xsd:enumeration
              value="http://www.w3.org/2001/04/xmlenc#sha512"/>
            <xsd:enumeration
              value="http://www.w3.org/2001/04/xmlenc#ripemd160"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

```

Listing 13: Restriktion ds:DigestMethodType

## 7.3 KeyInfo Element

Die Erzeugung qualifizierter elektronischer Signaturen basiert auf X509v3-Zertifikaten gem. Profilierung von [isis\_mtt] und [isis\_opt]; ausschließlich ISIS-MTT-konforme Zertifikate werden zugelassen.

```

<xsd:complexType name="KeyInfoType">
  <xsd:complexContent>
    <xsd:restriction base="ds:KeyInfoType">
      <xsd:sequence maxOccurs="unbounded">
        <xsd:element ref="ds:RetrievalMethod"/>
        <xsd:element ref="ds:X509Data"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

```

```

    <!-- (1,1) elements from (0,unbounded) namespaces -->
    </xsd:sequence>
  </xsd:restriction>
</xsd:complexContent>
</xsd:complexType>

```

Listing 14: Restriktion ds:KeyInfoType

### 7.3.1 RetrievalMethod Element

Entsprechend [isis\_mtt] darf nur X509Data genutzt werden.

```

<xsd:complexType name="RetrievalMethodType">
  <xsd:complexContent>
    <xsd:restriction base="ds:RetrievalMethodType">
      <xsd:attribute name="URI" type="xsd:anyURI"
        use="required"/>
      <xsd:attribute name="Type">
        <xsd:simpleType>
          <xsd:restriction base="xsd:anyURI">
            <xsd:enumeration
              value="http://www.w3.org/2000/09/xmldsig#X509Data"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

```

Listing 15: Restriktion ds:RetrievalMethodType

### 7.3.2 X509Data Element

Es muss jeweils das Signaturzertifikat mit dem öffentlichen Schlüssel in das Signaturelement aufgenommen werden.

Entsprechend [isis\_mtt] werden die Elemente <ds:X509SKI>; <ds:X509SubjectName> und der Extensionpoint („any-Element“) aus der Choice-Klausel in <ds:X509Data> ausgeschlossen. Darüber hinausgehend wird hier auch <ds:X509IssuerSerial> nicht zugelassen (ist dem Zertifikat zu entnehmen), weiter auch <ds:X509CRL> (keine Überfrachtung des Signaturelements mit ggf. sehr großen Sperrlisten).

```

<xsd:complexType name="X509DataType">
  <xsd:complexContent>
    <xsd:restriction base="ds:X509DataType">
      <xsd:sequence maxOccurs="unbounded">

```

```

        <xsd:choice>
            <xsd:element name="X509Certificate"
                type="xsd:base64Binary"/>
        </xsd:choice>
    </xsd:sequence>
</xsd:restriction>

</xsd:complexContent>
</xsd:complexType>

```

Listing 16: Restriktion ds:X509DataType

## 7.4 Reference: URI

In Version 1.0 dieser Spezifikation werden ausschließlich 0..n Signaturen auf das gesamte CDA-R2-Dokument appliziert. Es wird hier das Id-Attribut von `<cda:ClinicalDocument>` referenziert und somit sind alle „Header-“ und „Component-“ Abschnitte in die Signatur einbezogen.

Der Wert des Attributs `<URI>` ist eine lokale Referenz im Dokument und gem. [RFC2396] bzw. [xmldsig], 4.3.3.3 „Same-Document URI-References“ aufzubauen (URI=#xxx).

Neben dem eigentlichen Dokument werden weitere Informationen in die Signatur einbezogen, die in den Folgeabschnitten beschrieben werden. Dafür sind weitere `<ds:Reference>`-Elemente aufzubauen. In diesen `<ds:Reference>`-Elementen wird auch das `<ds:type>`-Attribut jeweils mit dem Typ der Referenz gesetzt (siehe Abschnitte [7.7.1],[7.7.2]).

## 7.5 Object Elemente: qualifizierende Angaben zur Signatur

Das `<ds:Object>`-Element aus [xmldsig] ist vom Typ `<any>` und kann entsprechend unter einem eigenen Namensraum mit individuell benötigten Strukturen belegt werden. Erweiterungen von [xmldsig] definiert [XAdES] für „Qualifying Properties“. Das Element `<xades:QualifyingProperties>` wird unmittelbar unter `<ds:Object>` positioniert; hier relevante Elemente sind jeweils **fett** markiert):

```

<xsd:element name="QualifyingProperties"
    type="QualifyingPropertiesType"/>
<xsd:complexType name="QualifyingPropertiesType">
    <xsd:sequence>

```

```

        <xsd:element name="SignedProperties"
            type="SignedPropertiesType" minOccurs="1"/>

```

```

        <xsd:element name="UnsignedProperties"
            type="UnsignedPropertiesType" minOccurs="0"/>
    </xsd:sequence>

```

```

    <xsd:attribute name="Target" type="xsd:anyURI"
        use="required"/>

```

```

    <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>

```

Listing 17: xades:QualifyingProperties

Das <Target>-Attribut muss hier das Attribut <Id> des Elements <ds:Signature> selbst referenzieren.

XAdES legt die hier relevanten „SignedProperties“ wie folgt fest:

```
<xsd:element name="SignedProperties" type="SignedPropertiesType"/>
<xsd:complexType name="SignedPropertiesType">
  <xsd:sequence>
    <xsd:element name="SignedSignatureProperties"
      type="SignedSignaturePropertiesType"/>
    <xsd:element name="SignedDataObjectProperties"
      type="SignedDataObjectPropertiesType" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="Id" type="xsd:ID" use="optional"/>
</xsd:complexType>
```

Listing 18: xades:SignedProperties

Im Sub-Element <xades:SignedSignatureProperties> sind wiederum folgende fett hervorgehobenen Elemente mandatorisch zu belegen:

```
<xsd:element name="SignedSignatureProperties"
  type="SignedSignaturePropertiesType"/>
<xsd:complexType name="SignedSignaturePropertiesType">
  <xsd:sequence>
    <xsd:element name="SigningTime" type="xsd:dateTime"/>
    <xsd:element name="SigningCertificate" type="CertIDListType"/>
    <xsd:element name="SignaturePolicyIdentifier"
      type="SignaturePolicyIdentifierType"/>
    <xsd:element name="SignatureProductionPlace"
      type="SignatureProductionPlaceType" minOccurs="0"/>
    <xsd:element name="SignerRole" type="SignerRoleType"
      minOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

Listing 19: xades:SignedSignatureProperties

### 7.5.1 Signaturzeitpunkt (Systemzeit)

In <xades:SigningTime> ist die Systemzeit des Systems einzustellen, auf dem die Signatur erstellt wird. <xades:SigningTime> ist vom Typ <xs:dateTime>.

### 7.5.2 Optionaler qualifizierter Zeitstempel

SAKs können Nutzern die optionale Anbringung qualifizierter Zeitstempel anbieten. Die [XAdES]-Erweiterungen von [xmldsig] sehen zwei Varianten für Zeitstempel vor:

1. `<xades:AllDataObjectsTimeStamp>`, Zeitstempel vor der Signaturbildung über alle in `<ds:Reference>` adressierten Daten; dieses Zeitstampelement wird dann in die Signaturbildung einbezogen.
2. `<xades:SignatureTimeStamp>`, Zeitstempel über das Signaturelement; wird **nach** der Signatur appliziert.

1.) wird in die `<xades:SignedDataObjectProperties>` eingestellt, während sich 2.) unterhalb von `<xades:QualifyingProperties>` (siehe [Listing 15]) als Kind-Element von `<xades:UnsignedProperties>`, `<xades:UnsignedSignatureProperties>` befindet.

Im Falle der Einbettung von Zeitstempeln in die Signatur von CDA-R2-Dokumenten ist die Variante 2.) verbindlich. Sie bietet die Vorteile, dass auch mögliche Empfänger signierter Dokumente solch einen Zeitstempel nachträglich applizieren können und Übersignaturen ermöglicht werden. Damit ist die Nachweisfähigkeit gesichert, dass die Signatur **vor** dem Zeitpunkt erstellt wurde, den der qualifizierte Zeitstempel ausweist – unabhängig vom „behaupteten“ Signaturzeitpunkt in `<xades:SigningTime>`.

Das Element `<xades:SignatureTimeStamp>` hat folgende Typdefinition:

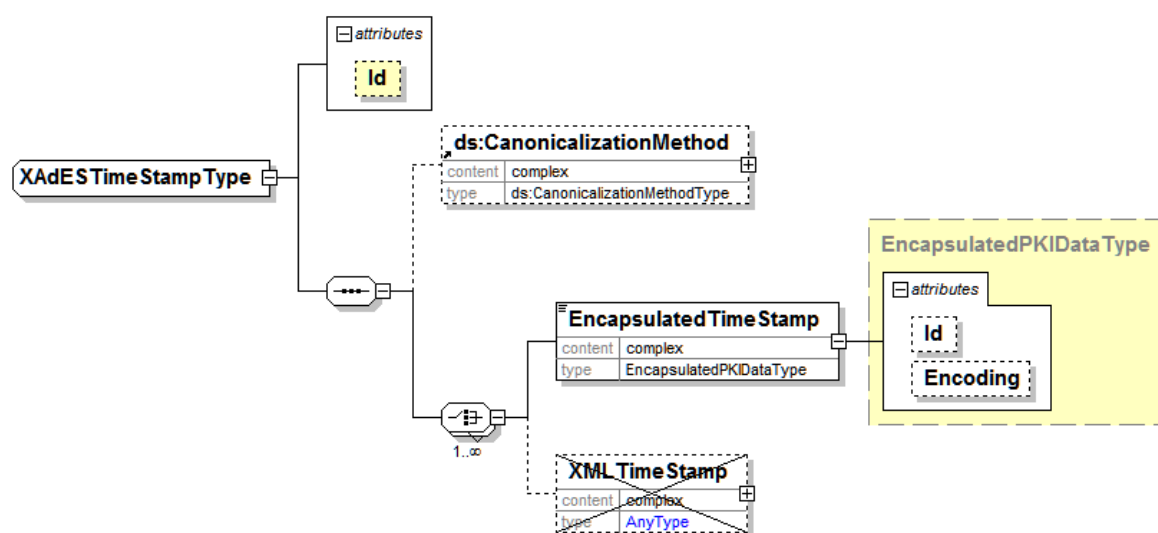


Abbildung 6: xades:XAdESTimeStampType

Das Element `<xades:XMLTimeStamp>` wird nicht genutzt; qualifizierte Zeitstempel nach [RFC3161] sind in das Element `<xades:EncapsulatedTimeStamp>` einzustellen; der Typ ist eine Extension von `<xs:base64Binary>`:

```
<xs:complexType name="EncapsulatedPKIDataType">
  <xs:simpleContent>
    <xs:extension base="xs:base64Binary">
      <xs:attribute name="Id" type="xs:ID" use="optional"/>
      <xs:attribute name="Encoding" type="xs:anyURI" use="optional"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

Listing 20: xades:EncapsulatedPKIDataType

### 7.5.3 Verweis auf das Signaturzertifikat

`<xades:SigningCertificate>` enthält einen Verweis auf das zur Signatur eingesetzte Zertifikat in der Form:

```
<xs:complexType name="CertIDListType">
  <xs:sequence>
    <xs:element name="Cert" type="CertIDType"
      maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CertIDType">
  <xs:sequence>
    <xs:element name="CertDigest" type="DigestAlgAndValueType"/>
    <xs:element name="IssuerSerial" type="ds:X509IssuerSerialType"/>
  </xs:sequence>
  <xs:attribute name="URI" type="xs:anyURI" use="optional" />
</xs:complexType>
```

Listing 21: xades:CertIDListType, CertType

Das Signaturzertifikat selbst ist – wie beschrieben – als X509-Data-Element mandatorischer Bestandteil des Signaturelements. Mit diesem Mechanismus wird die Referenz auf das Zertifikat in die Signaturbildung selbst aufgenommen.

### 7.5.4 Signature Policy

Die Signatur von medizinischen Dokumenten folgt unterschiedlichen Policies, die u.a. Festlegungen treffen bzgl. mandatorischer und optionaler Elemente, einzusetzender Algorithmen etc. Auch die vorliegende Spezifikation wird durch eine entsprechende Policy adressiert. Die Aufnahme eines Verweises auf die Policy in die Signaturbildung macht überprüfbar, welcher Policy bei Signaturerstellung gefolgt werden sollte.

Hierzu dient die XAdES-Erweiterung `<SignaturPolicyIdentifier>`, wobei in der aktuellen Version dieser Spezifikation die Policy implizit die vorliegende Spezifikation ist. Es ist daher in der Choice nur das leere Element `<SignaturePolicyImplied>` aufzubauen.

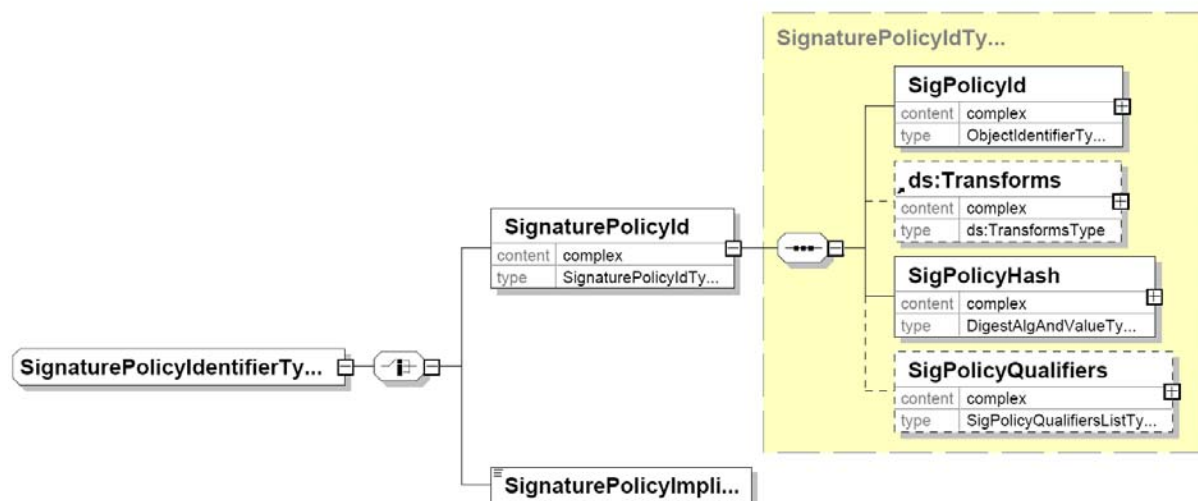


Abbildung 7: Restriktion xades:SignaturePolicyIdentifierType

► **Hinweis:** Liegt für eine zukünftige Version dieser Spezifikation eine Signature-Policy in maschinenlesbarer Form als XML-Datei vor, wird diese dann aus `<xades:SignaturePolicyId>` referenziert werden (siehe dazu [XAdES], Kapitel 7.2.3).

Für die Telematik-Infrastruktur werden SignaturePolicyIdentifier für die Erstellung und Prüfung von Signaturen benötigt. Dieser Identifier ist für CDA-R2-Dokumente wie folgt festgelegt:

[http://www.e-arztausweis.de/sigpolicies/earztbrief-1\\_0/sigpolicyv1.xml](http://www.e-arztausweis.de/sigpolicies/earztbrief-1_0/sigpolicyv1.xml)

► **Hinweis:** Die genannte URL wird in der vorliegenden Version der Spezifikation nicht in die Signaturelemente aufgenommen; sie ist hier lediglich ein Hinweis für Realisierungen der vorliegenden Spezifikation innerhalb der Konnektoren der Telematik-Infrastruktur.

## 7.5.5 Attributzertifikate und Rollen

In den Elementen von `<xades:SignerRole>` können Informationen hinterlegt werden, die Auskunft geben über spezifische Rollen des jeweiligen Unterzeichners.

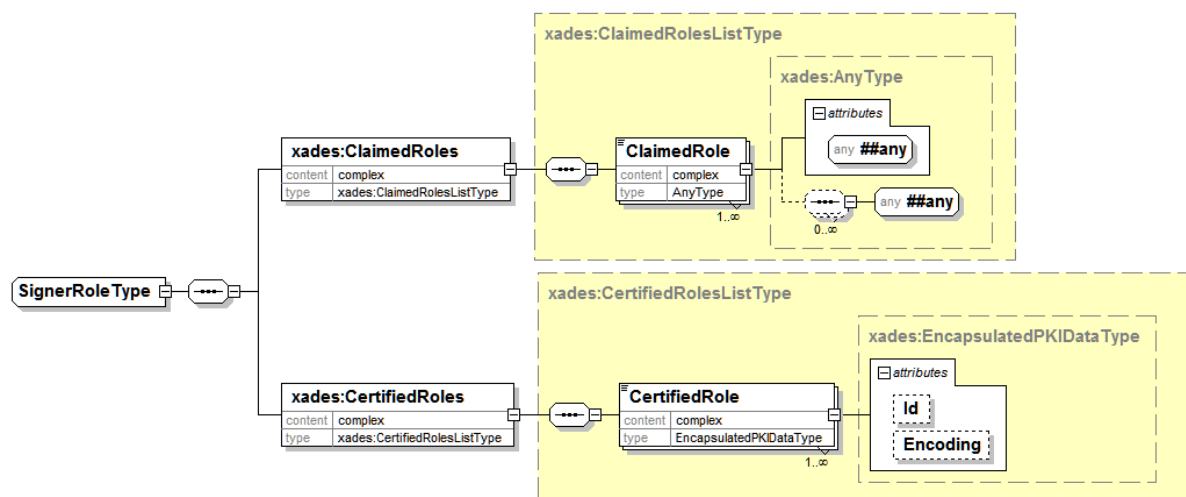


Abbildung 8: xades:SignerRoleType

Das Sub-Element `<xades:SignerRole>` kann sowohl „zertifizierte“ als auch „behauptete“ Rollen aufnehmen:

```

<xsd:element name="SignerRole" type="SignerRoleType"/>
<xsd:complexType name="SignerRoleType">
  <xsd:sequence>
    <xsd:element name="ClaimedRoles" type="ClaimedRolesListType"
      minOccurs="0"/>
    <xsd:element name="CertifiedRoles" type="CertifiedRolesListType"
      minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ClaimedRolesListType">
  <xsd:sequence>
    <xsd:element name="ClaimedRole" type="AnyType"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CertifiedRolesListType">
  <xsd:sequence>
    <xsd:element name="CertifiedRole" type="EncapsulatedPKIDataType"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>

```

Listing 22: xades:SignerRole

<xades:EncapsulatedPKIDataType> ist vom Typ <xsd:base64binary> und kann somit X509-Zertifikate aufnehmen.

### 7.5.5.1 XAdES CertifiedRoles-Element

Attributzertifikate werden in die Signatur eingeschlossen, sofern der Unterzeichner dies nicht explizit ausschließen will (Funktionsanforderung an eine SAK). Im Falle der Signatur mit dem elektronischen Arztausweis wird in der Regel das auf der Karte befindliche Attributzertifikat „Arzt“ mitsigniert. Attributzertifikate sind in das Element <xades:CertifiedRole> innerhalb einer <xades:SignedProperties>-Struktur gem. [XAdES] im <ds:Object>-Element einzustellen.

### 7.5.5.2 XAdES ClaimedRoles-Element

Bei CDA-R2-Dokumenten muss ein Bezug von der Signatur zu den zuzuordnenden Elementen <cda:LegalAuthenticator> bzw. <cda:Authenticator> hergestellt werden.

► **Hinweis:** Es ist Anforderung an eine Signaturanwendungskomponente sowie das Stylesheet für die Visualisierung, diesen Bezug korrekt herzustellen.

In einer kurzfristig vorgesehenen Fortschreibung des CDA-R2-Schemas sind für die Elemente <cda:LegalAuthenticator> und <cda:Authenticator> Id-Attribute vorgesehen.

Bei CDA-R2-Dokumenten wird der Wert des ID-Attributs der Elemente <cda:LegalAuthenticator> bzw. <cda:Authenticator> direkt in das Element <xades:ClaimedRole> innerhalb einer <xades:SignedProperties>-Struktur gemäß [XAdES]



im `<ds:Object>`-Element übernommen. Eine SAK muss Mechanismen unterstützen, bei Signaturerstellung dem Signaturkarteninhaber die korrekte Zuordnung zu den Elementen `<cda:LegalAuthenticator>` bzw. `<cda:Authenticator>` zu ermöglichen.

**Hinweis:** *Die Festlegung gilt für CDA-R2-Dokumente und ist nicht übertragbar auf andere medizinische Dokumente, bei denen entsprechende Elemente im Schema nicht vorhanden sind!*

## 7.5.6 Gesamtübersicht: SignedSignatureProperties

Die folgende Abbildung fasst die geforderten Belegungen von `<xades:SignedSignatureProperties>` zusammen:

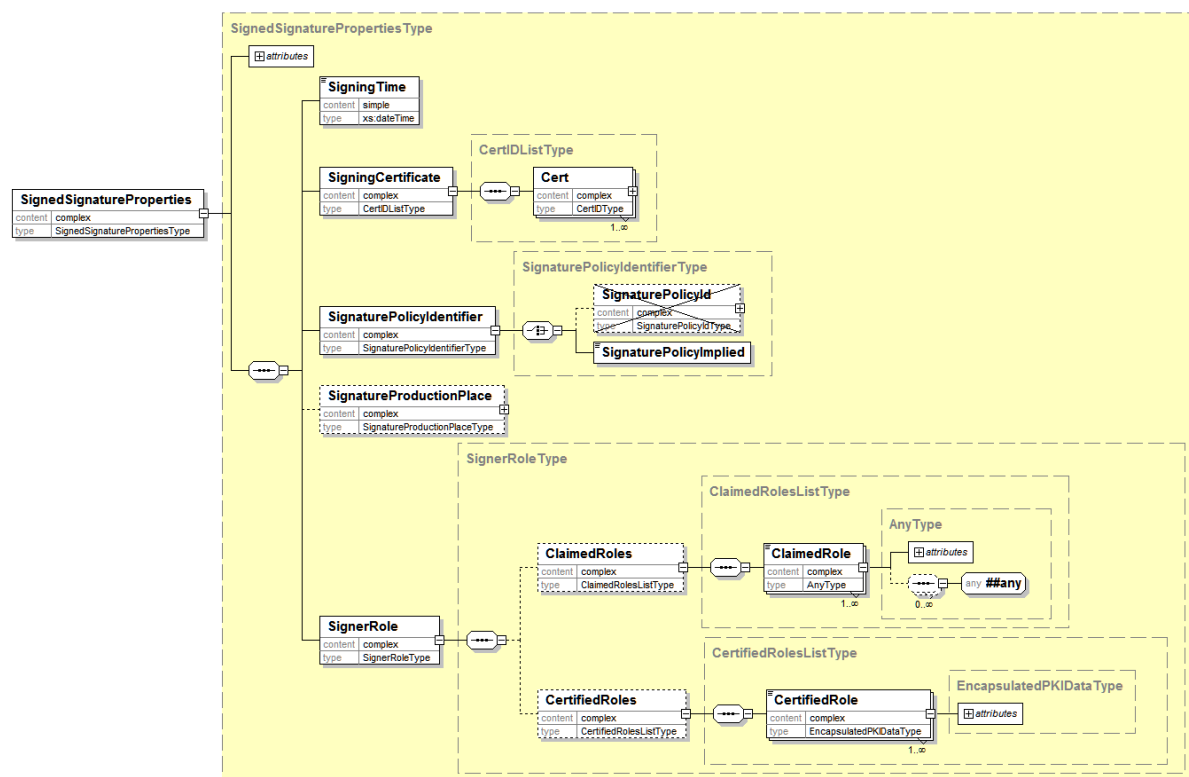


Abbildung 9: Übersicht xades:SignedSignatureProperties

► **Hinweis:** In Abweichung von [XAdES] sind dabei die Elemente `<xades:SigningTime>`, `<xades:SigningCertificate>`, `<xades:SignaturePolicyIdentifier>` und `<xades:SignerRole>` mandatorisch.<sup>6</sup>

## 7.6 Object Element, Referenz auf das zugehörige Stylesheet

Für die Visualisierung von CDA-R2-Dokumenten kommt ein Stylesheet zum Einsatz. Dieses Stylesheet wird in den XML-Dokumenten referenziert.

Die Verlässlichkeit der Darstellung („you sign what you see“) kann von dieser Spezifikation nur als Anforderungen an eine SAK formuliert werden. Entscheidend für die Verlässlichkeit der Präsentation ist die Vertrauenswürdigkeit der eingesetzten SAK und der von dieser genutzten XSLT-Prozessoren.

Die gematik hat in [KonnSpec] spezifiziert, wie Referenzen zu Stylesheets in die Signaturelemente aufgenommen werden. Diese Policy, die lediglich nachweisbar machen soll, welches Stylesheet bei Signaturerstellung zum Einsatz kam, wird hier übernommen.

Dazu ist ein weiteres `<ds:object>`-Element aufzubauen, welches ein in [xmldsig] definiertes `<ds:Manifest>`-Element enthält:

```
<xsd:complexType name="ManifestType">
  <xsd:complexContent>
    <xsd:restriction base="ds:ManifestType">
```

<sup>6</sup> Kardinalität dieser Elemente ist 1 und nicht obligatorisch 0..1, wie in [XAdES] definiert

```

<xsd:sequence>
  <xsd:element ref="ds:Transforms"/>
  <xsd:element ref="ds:DigestMethod"/>
  <xsd:element ref="ds:DigestValue"/>
</xsd:sequence>

<xsd:attribute name="URI" type="xsd:anyURI" use="required"/>
<xsd:attribute name="Id" type="ID" use="required"/>

</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>

```

#### Listing 23: Restriktion <ds:Manifest>

Die Restriktion gegenüber der Manifest-Typdefinition in [xmldsig] ist, dass beide Attribute *required* und nicht optional sind.

In das Kindelement <ds:reference>, <URI>-Attribut ist die URI des eingesetzten Stylesheets aufzunehmen; <ds:DigestValue> und <ds:DigestMethod> nehmen den Hashwert des Stylesheets respektive des eingesetzten Hash-Algorithmus auf (zulässige Algorithmen siehe Abschnitt [7.2.3]).

Die (geplante) URI für das Stylesheet für CDP-Content-Packages mit darin enthaltenen (ggf. signierten) CDA-R2-Dokumenten ist:

[http://www.e-arztausweis.de/stylesheets/earztbrief-1\\_0/vhitg\\_cdp\\_v1.xsl](http://www.e-arztausweis.de/stylesheets/earztbrief-1_0/vhitg_cdp_v1.xsl)

Signaturen müssen auch ohne existierende Internetverbindung erzeugt werden können. Das Stylesheet wird daher auch lokal zum einem CDP-Content-Package im identischen Verzeichnis gehalten; die Belegung des Attributs <URI> ist damit:

./vhitg\_cdp\_v1.xsl

► **Hinweis:** In der Testphase wird zunächst ein vorläufige Stylesheet

./CDP\_V1\_3.xsl

zur Verfügung gestellt!

► **Hinweis:** Der Inhalt des Manifests ist in dieser Version NICHT in die Signaturprüfung einzubeziehen<sup>7</sup>.

Das Id-Attribut dieses <ds:Manifest>-Elements ist aus einem entsprechenden <ds:Reference>-Element des Signaturelements zu referenzieren. Somit sind die hier aufgebauten Referenzen zum eingesetzten Stylesheet in die Digest-Berechnung der Signatur einzubeziehen. (Die Ansicht des XML-Dokuments im „Trusted-View-Modus“ unter Verwendung des referenzierten Stylesheets soll dem Nutzer bei Erstellung bzw. Prüfung der Signatur von der SAK dringend nahe gelegt werden. In anderen Situationen kann die Verwendung anderer Stylesheets ermöglicht werden.)

Anwendungen müssen Nutzer die Möglichkeit zu Verfügung stellen, die Unversehrtheit der lokal Kopie des Stylesheets durch Abgleich mit der Referenz zu Prüfen.

Die SHA-1 Hashwert des Referenzstylesheets hat folgende URI:

**[ToDo]**

<sup>7</sup> Entsprechend Festlegung aus [KonnSpec].

## 7.7 Zusammenfassung: Weitere Reference-Elemente

Zusätzlich zum `<ds:Reference>`-Element zur Referenzierung des eigentlichen CDA-R2-Dokuments sind folgende `<ds:Reference>`-Elemente in den Signaturelementen aufzubauen:

### 7.7.1 Referenz auf das Manifest-Element

Das `<ds:Manifest>`-Element ist mit einem eindeutigen `<Id>`-Attribut zu belegen (z.B. „\_3577B5EF-523F-4946-9734-C974CEA6C646“), welches referenziert wird aus dem URI-Attribut des weiteren `<ds:Reference>`-Elements, in diesem Beispiel:

```
<Reference URI="#_3577B5EF-523F-4946-9734-C974CEA6C646"
Type="http://www.w3.org/2000/09/xmldsig#Manifest">
.....
</Reference>
```

Listing 24: Beispiel - Reference URI zum Element `ds:Manifest`

Das `<ds:Type>`-Attribut dieses `<ds:Reference>`-Elements kennzeichnet den Typ der Referenz, in diesem Fall

`"http://www.w3.org/2000/09/xmldsig#Manifest"`.

### 7.7.2 Referenz auf XAdES-Erweiterungen

Das `<xades:SignedProperties>`-Element ist mit einem eindeutigen `<Id>`-Attribut zu belegen (z.B. „\_3577B5EF-523F-4946-9734-C974CEA6C6E9“), welches referenziert wird aus dem URI-Attribut des weiteren `<ds:Reference>`-Elements, in diesem Beispiel:

```
<Reference URI="#_3577B5EF-523F-4946-9734-C974CEA6C6E9"
Type="http://uri.etsi.org/01903/v1.3.2#SignedProperties">
.....
</Reference>
```

Listing 25: Beispiel - Reference URI zum Element `xades:SignedProperties`

Das `<ds:Type>`-Attribut dieses `<ds:Reference>`-Elements kennzeichnet den Typ der Referenz, in diesem Fall

`"http://uri.etsi.org/01903/v1.3.2#SignedProperties"`.

## 8 Verzeichnisse

### 8.1 Abbildungen

Abbildung 1: Übersicht der Metastruktur cdp:ContentPackage .....	9
Abbildung 2: Root-Element cdp:ContentPackage .....	10
Abbildung 3: cdp:ContentContainerType .....	11
Abbildung 4: Übersicht xenc:EncryptedData .....	18
Abbildung 5: cdp:StructuredContentType .....	19
Abbildung 6: xades:XAdESTimeStampType .....	29
Abbildung 7: Restriktion xades:SignaturePolicyIdentifierType .....	31
Abbildung 8: xades:SignerRoleType .....	31
Abbildung 9: Übersicht xades:SignedSignatureProperties .....	34

### 8.2 Tabellen

Tabelle 1: Namespaces .....	7
Tabelle 2: Zulässige Mime-Types für Attachments .....	14

### 8.3 Listings

Listing 1: cdp:ContentPackage .....	11
Listing 2: cdp:ContentContainerType .....	13
Listing 3: Referenz aus einem CDA-R2-Dokument auf einen extern geführten Bildbefund .....	16
Listing 4: Complex Type <ED> aus datatypes-base.xsd .....	17
Listing 5: cdp:StructuredContentType .....	19
Listing 6: cda:ClinicalDocumentType .....	21
Listing 7: cda:LegalAuthenticatorType .....	21
Listing 8: cda:SectionType .....	21
Listing 9: cda:AuthenticatorType .....	22
Listing 10: Restriktion ds:SignatureType .....	23
Listing 11: Restriktion ds:CanonicalizationMethodType .....	24
Listing 12: Restriktion ds:SignatureMethodType .....	25
Listing 13: Restriktion ds:DigestMethodType .....	25
Listing 14: Restriktion ds:KeyInfoType .....	26
Listing 15: Restriktion ds:RetrievalMethodType .....	26
Listing 16: Restriktion ds:X509DataType .....	27
Listing 17: xades:QualifyingProperties .....	27
Listing 18: xades:SignedProperties .....	28

Listing 19: xades:SignedSignatureProperties .....	28
Listing 20: xades:EncapsulatedPKIDataType .....	29
Listing 21: xades:CertIDListType, CertType.....	30
Listing 22: xades:SignerRole.....	32
Listing 23: Restriktion <ds:Manifest> .....	35
Listing 24: Beispiel - Reference URI zum Element ds:Manifest.....	36
Listing 25: Beispiel - Reference URI zum Element xades:SignedProperties.....	36
Listing 26: Restriktionen XML Signature .....	43
Listing 27: Restriktionen XAdES .....	45
Listing 28: Schema CDocumentPayload.....	47
Listing 29: Beispiel.....	52

## 8.4 Referenzen

[BNetzA_Alg]	Bekanntmachung zur elektronischen Signatur nach Signaturgesetz und Signaturverordnung (Übersicht über geeignete Algorithmen), Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen, 17. Dezember 2007; <a href="http://www.bundesnetzagentur.de/media/archive/12198.pdf">http://www.bundesnetzagentur.de/media/archive/12198.pdf</a> .
[CDA2]	HL7 Clinical Document Architecture, Release 2.0; HL7® Version 3 Standard, © 2004 Health Level Seven®, Inc. All Rights Reserved.
[gematik]	gematik: "Konnektorspezifikation", Version 2.2.0, und "Verwendung kryptographischer Algorithmen in der Telematik-Infrastruktur", Version 1.0.0; beide vom 24.8.2007; <a href="http://www.gematik.de/(S(uaneue311hbgxjnmhcolc45))/Uebersichtsseite___Release_2_3_1.Gematik">http://www.gematik.de/(S(uaneue311hbgxjnmhcolc45))/Uebersichtsseite___Release_2_3_1.Gematik</a>
[gematikGArch]	gematik: Einführung der Gesundheitskarte, Gesamtarchitektur, Version 1.0.0 vom 15.7.1007; <a href="http://www.gematik.de/(S(53ohxv3w0wr4t1qoer30uo45))/Detailseite___Architektur___Gesamtarchitektur.Gematik">http://www.gematik.de/(S(53ohxv3w0wr4t1qoer30uo45))/Detailseite___Architektur___Gesamtarchitektur.Gematik</a>
[hl7crs]	HL7 v3, CDA Rel. 2 „Implementation Guide for CDA Release 2 – Level 1 and 2 – Care Record Summary (US realm)“, 2005-11-17, 3rd normative ballot.
[isis_mtt]	Common ISIS-MTT Specifications for interoperable PKI Applications, Version 1.1, 16.03.2004; <a href="http://www.isis-mtt.t7-isis.org/uploads/media/ISIS-MTT_Core_Specification_v1.1_03.pdf">http://www.isis-mtt.t7-isis.org/uploads/media/ISIS-MTT_Core_Specification_v1.1_03.pdf</a> .
[isis_opt]	Common ISIS-MTT Specifications for interoperable PKI Applications, Optional Profile SigG-Profile Version 1.1, 16.03.2004; <a href="http://www.isis-mtt.t7-isis.org/uploads/media/ISIS-MTT_Optional_Profiles_v1.1_02.pdf">http://www.isis-mtt.t7-isis.org/uploads/media/ISIS-MTT_Optional_Profiles_v1.1_02.pdf</a> .
[KonnSpec]	Gesellschaft für Telematikanwendungen der Gesundheitskarte. Konnektorspezifikation, Version 2.2.0 vom 24.08.2007.
[RFC2396]	T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax. IETF RFC 2396; <a href="http://www.ietf.org/rfc/rfc2396.txt">http://www.ietf.org/rfc/rfc2396.txt</a> .
[RFC3161]	D. Pinkas, R. Zuccherato, Time-Stamp Protocol (TSP), IETF RFC 1661, <a href="http://www.ietf.org/rfc/rfc3161.txt">http://www.ietf.org/rfc/rfc3161.txt</a>

[RFC3986]	T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax. IETF RFC 3986; <a href="http://www.ietf.org/rfc/rfc3986.txt">http://www.ietf.org/rfc/rfc3986.txt</a> .
[RFC4122]	P. Leach, M. Mealing, R. Salz. A Universally Unique IDentifier (UUID) URN Namespace, The Internet Society, July 2005; <a href="http://www.ietf.org/rfc/rfc4122.txt">http://www.ietf.org/rfc/rfc4122.txt</a>
[SigG]	Gesetz über Rahmenbedingungen für elektronische Signaturen, 16.05.2001; <a href="http://bundesrecht.juris.de/sigg_2001/BJNR087610001.html">http://bundesrecht.juris.de/sigg_2001/BJNR087610001.html</a> .
[SigV]	Verordnung zur elektronischen Signatur, 16.11.2006; <a href="http://bundesrecht.juris.de/sigv_2001/BJNR307400001.html">http://bundesrecht.juris.de/sigv_2001/BJNR307400001.html</a> .
[VHitG-Lf]	Arztbrief auf Basis der HL7 Clinical Document Architecture Release 2, Implementierungsleitfaden; Version 1.5 vom 12.05.2006; vorgelegt vom VHitG.
[XAdES]	European Telecommunications Standards Institute. ETSI TS 101 903: XML Advanced Electronic Signatures, V1.3.2 2006-03; <a href="http://webapp.etsi.org/action/PU/20060307/ts_101903v010302p.pdf">http://webapp.etsi.org/action/PU/20060307/ts_101903v010302p.pdf</a> .
[xmime]	W3C Working Group Note, Describing Media Content of Binary Data in XML, 4 May 2005, <a href="http://www.w3.org/TR/xml-media-types/#contentType">http://www.w3.org/TR/xml-media-types/#contentType</a>
[xmldsig]	World Wide Web Consortium. XML-Signature Syntax and Processing, W3C Recommendation, 12 February 2002; <a href="http://www.w3.org/TR/xmldsig-core/">http://www.w3.org/TR/xmldsig-core/</a> .
[xenc]	World Wide Web Consortium. XML Encryption Syntax and Processing, W3C Recommendation, 10.12.2002; <a href="http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/">http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/</a> .
[xml-exc]	World Wide Web Consortium. Exclusive XML Canonicalization, W3C Recommendation, 18.07.2002; <a href="http://www.w3.org/TR/xml-exc-c14n/">http://www.w3.org/TR/xml-exc-c14n/</a> .
[xpath]	World Wide Web Consortium. XML Path Language (XPath), Version 1.0, W3C Recommendation 16 November 1999, <a href="http://www.w3.org/TR/xpath">http://www.w3.org/TR/xpath</a> .
[xmlschema]	World Wide Web Consortium. XML Schema, Parts 0, 1, and 2 (Second Edition). W3C Recommendation, 28 October 2004; <a href="http://www.w3.org/TR/xmlschema-0/">http://www.w3.org/TR/xmlschema-0/</a> , <a href="http://www.w3.org/TR/xmlschema-1/">http://www.w3.org/TR/xmlschema-1/</a> , and <a href="http://www.w3.org/TR/xmlschema-2/">http://www.w3.org/TR/xmlschema-2/</a> .

## 9 Anhang

► **Hinweis:** In folgenden Schema-Definitionen werden lokale Schema-Lokationen auch für referenzierte Schemata geführt.

### 9.1 Schemaeinschränkungen zu XML Signature

Die Redefinition von `xmldsig-core-schema.xsd` wird mit dem Schema `CDA_redefine_xdsig.xsd` bereitgestellt.

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
targetNamespace="http://www.w3.org/2000/09/xmldsig#"
elementFormDefault="qualified">

  <xsd:redefine schemaLocation="xmldsig-core-schema.xsd">

    <xsd:complexType name="KeyInfoType">
      <xsd:complexContent>
        <xsd:restriction base="ds:KeyInfoType">
          <xsd:sequence maxOccurs="unbounded">
            <xsd:element ref="ds:RetrievalMethod"/>
            <xsd:element ref="ds:X509Data"/>
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="RetrievalMethodType">
      <xsd:complexContent>
        <xsd:restriction base="ds:RetrievalMethodType">
          <xsd:attribute name="URI" type="xsd:anyURI" use="required"/>
          <xsd:attribute name="Type">
            <xsd:simpleType>
              <xsd:restriction base="xsd:anyURI">
                <xsd:enumeration value="http://www.w3.org/2000/09/xmldsig#X509Data"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:attribute>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>
```



```
<xsd:complexType name="CanonicalizationMethodType">
  <xsd:complexContent>
    <xsd:restriction base="ds:CanonicalizationMethodType">
      <xsd:attribute name="Algorithm" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:anyURI">
            <xsd:enumeration value="http://www.w3.org/TR/2001/10/xml-exc-c14n#" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="DigestMethodType">
  <xsd:complexContent>
    <xsd:restriction base="ds:DigestMethodType">
      <xsd:attribute name="Algorithm" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:anyURI">
            <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#sha256" />
            <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#sha512" />
            <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#ripemd160" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="SignatureMethodType">
  <xsd:complexContent>
    <xsd:restriction base="ds:SignatureMethodType">
      <xsd:attribute name="Algorithm" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:anyURI">
```

```
<xsd:enumeration value=
    "http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<xsd:enumeration value=
    "http://www.w3.org/2001/04/xmldsig-more#rsa-sha512"/>
<xsd:enumeration value=
    "http://www.w3.org/2001/04/xmldsig-more/rsa-ripemd160"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="X509DataType">
    <xsd:complexContent>
        <xsd:restriction base="ds:X509DataType">
            <xsd:sequence maxOccurs="unbounded">
                <xsd:choice>
                    <xsd:element name="X509Certificate" type="xsd:base64Binary"/>
                </xsd:choice>
            </xsd:sequence>
        </xsd:restriction>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="SignatureType">
    <xsd:complexContent>
        <xsd:restriction base="ds:SignatureType">
            <xsd:sequence>
                <xsd:element ref="ds:SignedInfo"/>
                <xsd:element ref="ds:SignatureValue"/>
                <xsd:element ref="ds:KeyInfo"/>
                <xsd:element ref="ds:Object" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:restriction>
    </xsd:complexContent>
</xsd:complexType>
```

```

<xsd:complexType name="ReferenceType">
  <xsd:complexContent>
    <xsd:restriction base="ds:ReferenceType">
      <xsd:sequence>
        <xsd:element ref="ds:Transforms" minOccurs="0"/>
        <xsd:element ref="ds:DigestMethod"/>
        <xsd:element ref="ds:DigestValue"/>
      </xsd:sequence>
      <xsd:attribute name="URI" type="xsd:anyURI" use="required"/>
      <xsd:attribute name="Type" type="xsd:anyURI" use="optional"/>
      <xsd:attribute name="Id" type="xsd:ID" use="required"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
</xsd:redefine>

<xsd:annotation>
  <xsd:documentation xml:lang="de">
    Einschränkungen von XML-Signature gem. SigG/SigV und
      Algorithmenkatalog BNetzA 2007
  </xsd:documentation>
</xsd:annotation>
</xsd:schema>

```

Listing 26: Restriktionen XML Signature

## 9.2 Hinweise zu den Schemata XAdES und XML Encryption

Mit der Spezifikation werden auch die w3c-Schemata XAdES132.xsd und xenc-schema.xsd zur Verfügung gestellt.

Beide Schemata importieren im Original `xmldsig-core-schema.xsd`. In den lokalen Versionen wurden diese Importe ersetzt durch das Schema `CDA_redefine_xdsig.xsd`, da sonst Doppeldefinition von Schemaelementen aus XML Signature vorliegen.

### 9.3 Schemaeinschränkungen XAdES

[XAdES] sieht die hier benötigten Erweiterungen für XML-Signaturen als optionale Elemente vor. Die vorgenommene Redefinition in `CDA_redefine_xades.xsd` kennzeichnet die mandatorisch zu belegenden Elemente.

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:xades="http://uri.etsi.org/01903/v1.3.2#"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  targetNamespace="http://uri.etsi.org/01903/v1.3.2#" elementFormDefault="qualified">

  <xs:redefine schemaLocation="XAdES132.xsd">

    <xs:complexType name="QualifyingPropertiesType">
      <xs:complexContent>
        <xs:restriction base="xades:QualifyingPropertiesType">
          <xs:sequence>
            <xs:element name="SignedProperties" type="xades:SignedPropertiesType"/>
            <xs:element name="UnsignedProperties"
              type="xades:UnsignedPropertiesType" minOccurs="0"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>

    <xs:complexType name="SignedSignaturePropertiesType">
      <xs:complexContent>
        <xs:restriction base="xades:SignedSignaturePropertiesType">
          <xs:sequence>
            <xs:element name="SigningTime" type="xs:dateTime"/>
            <xs:element name="SigningCertificate" type="xades:CertIDListType"/>
            <xs:element name="SignaturePolicyIdentifier"
              type="xades:SignaturePolicyIdentifierType"/>
            <xs:element name="SignatureProductionPlace"
              type="xades:SignatureProductionPlaceType" minOccurs="0"/>
            <xs:element name="SignerRole" type="xades:SignerRoleType"/>
          </xs:sequence>
        </xs:restriction>
      </xs:complexContent>
    </xs:complexType>

    <xs:complexType name="SignerRoleType">
      <xs:complexContent>
```

```

    <xs:restriction base="xades:SignerRoleType">
      <xs:sequence>
        <xs:element name="ClaimedRoles" type="xades:ClaimedRolesListType"/>
        <xs:element name="CertifiedRoles" type="xades:CertifiedRolesListType"/>
      </xs:sequence>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
</xs:redefine>
</xs:schema>

```

Listing 27: Restriktionen XAdES

## 9.4 Schema CDocumentPayload

```

<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:enc="http://www.w3.org/2001/04/xmlenc#"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
  xmlns:cdp="http://ws.gematik.de/fa/cds/CDocumentPayload/v1.0"
  targetNamespace="http://ws.gematik.de/fa/cds/CDocumentPayload/v1.0"
  elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0">

  <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
    schemaLocation="CDA_redefine_xdsig.xsd"/>

  <xs:import namespace="http://www.w3.org/2001/04/xmlenc#"
    schemaLocation="xenc-schema.xsd"/>

  <xs:import namespace="http://www.w3.org/2005/05/xmlmime"
    schemaLocation="xmlmime.xsd"/>

  <xs:complexType name="ContentContainerType">
    <xs:choice maxOccurs="unbounded">
      <xs:element name="StructuredContent"
        type="cdp:StructuredContentType">
        <xs:annotation>
          <xs:documentation>Container for XML-formatted Documents</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element ref="ds:Signature" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="UnstructuredContent" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>Container for Attachments</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:choice>
  </xs:complexType>

```

```

</xs:annotation>

<xs:complexType>
  <xs:complexContent>
    <xs:extension base="xmime:base64Binary">
      <xs:attribute name="Id" type="xs:ID" use="optional"/>
      <xs:attribute name="Filename" type="xs:string" use="optional"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
</xs:element>
</xs:choice>
<xs:attribute name="range" type="xs:int">
  <xs:annotation>
    <xs:documentation>Number of historical Range
      (0 = most recent Document Container)</xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:complexType>

<xs:element name="ContentPackage">
  <xs:annotation>
    <xs:documentation>Root of Entire Document Package</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:choice>
      <xs:sequence>
        <xs:element name="ContentContainer" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Container for one Structured Document,
              Signatures and Attachments</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:complexContent>
              <xs:extension base="cdp:ContentContainerType"/>
            </xs:complexContent>
          </xs:complexType>
        </xs:element>

```

```

        </xs:sequence>

        <xs:sequence>

            <xs:element ref="enc:EncryptedData"/>

        </xs:sequence>

    </xs:choice>

</xs:complexType>

</xs:element>

<xs:complexType name="StructuredContentType">

    <xs:sequence>

        <xs:any namespace="##any" minOccurs="0"/>

    </xs:sequence>

    <xs:attribute name="Id" type="xs:ID"/>

    <xs:anyAttribute namespace="##any"/>

</xs:complexType>

</xs:schema>

```

Listing 28: Schema CDocumentPayload

## 9.5 Beispiel (Ausschnitt)

Folgendes fiktives Beispiel bildet folgenden Fall ab:

- Ein <ContentContainer> im <ContentPackage>
- mit einer Signatur, die zu einem <LegalAuthenticator>-Element gehört
- mit einem Attachment im <ContentPackage>.

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="./CDP_V1_3.xsl"?>
<cdp:ContentPackage xmlns:cdp="http://ws.gematik.de/fa/cds/CDocumentPayload/v1.0"
xmlns="urn:hl7-org:v3" xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:sciphox="urn:sciphox-org/sciphox"
xmlns:xades="http://uri.etsi.org/01903/v1.3.2#"
xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ws.gematik.de/fa/cds/CDocumentPayload/v1.0
./schemas/CDA_Package_V01.xsd">

    <cdp:ContentContainer range="0">

        <cdp:StructuredContent>

<!--*****
CDA Header mit ID-Attribut 1
*****-->

        <ClinicalDocument xmlns="urn:hl7-org:v3"

            xmlns:sciphox="urn:sciphox-org/sciphox"

            Id="_80CC2BC3-10EC-8048-9C1F-6B88BC251404"

            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

```

```

.....
<!--*****
LegalAuthenticator mit Id-Attribut 2
*****-->

<legalAuthenticator Id="_F436B5F5-59CE-F2DA-502C-8ACD38DD3DF3">
  <time value="20051222"/>
.....
</legalAuthenticator>
.....
</ClinicalDocument>
</cdp:StructuredContent>
<!--*****
Signature Element mit Id 3 (wird aus QualifyingProperties, Target referenziert)
*****-->

<Signature xmlns="http://www.w3.org/2000/09/xmldsig#"
  Id="_6B5A7A1E-01B1-B729-F406-015888803C9B">
  <SignedInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
    <CanonicalizationMethod Algorithm=
      "http://www.w3.org/2001/10/xml-exc-c14n#"
      xmlns="http://www.w3.org/2000/09/xmldsig#" />
    <SignatureMethod Algorithm=
      "http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"
      xmlns="http://www.w3.org/2000/09/xmldsig#" />
  <--*****
Referenz auf ClinicalDocument
*****-->

  <Reference URI="#_80CC2BC3-10EC-8048-9C1F-6B88BC251404"
    xmlns="http://www.w3.org/2000/09/xmldsig#">
    <Transforms xmlns="http://www.w3.org/2000/09/xmldsig#">
      <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"
        xmlns="http://www.w3.org/2000/09/xmldsig#" />
    </Transforms>
    <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"
      xmlns="http://www.w3.org/2000/09/xmldsig#" />
    <DigestValue xmlns="http://www.w3.org/2000/09/xmldsig#">
      6KSuFTCezLSbYQruytuZvHtj7BffeMl8jfpWReesZ3w=</DigestValue>
    </Reference>
  <--*****
Referenz auf Object-Element 1, Signed Properties

```



```

*****-->

<Reference Id="_2651E293-A58E-D71D-4561-69A5DFB53E7E"
  Type="http://uri.etsi.org/01903/v1.1.1/#SignedProperties"
  URI="#_A00DD61-8964-46C4-8FD2-8E7CFC9B84C9"
  xmlns="http://www.w3.org/2000/09/xmldsig#">

  <Transforms xmlns="http://www.w3.org/2000/09/xmldsig#">

    <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"
      xmlns="http://www.w3.org/2000/09/xmldsig#" />

  </Transforms>

  <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"
    xmlns="http://www.w3.org/2000/09/xmldsig#" />

  <DigestValue xmlns="http://www.w3.org/2000/09/xmldsig#">

    1HRODHJulkkr3nqn07cshrORdE888m7UAS7NPkNaOPY=</DigestValue>

  </Reference>

<--*****

Referenz auf Object-Element 2, Manifest

*****-->

<Reference Id="_0DC744EB-A71F-236A-AB36-FF1A90F52D43"
  Type="http://www.w3.org/2000/09/xmldsig#Manifest"
  URI="#_9080F14F-6936-E2FF-0F5D-A022F573129D"
  xmlns="http://www.w3.org/2000/09/xmldsig#">

  <Transforms xmlns="http://www.w3.org/2000/09/xmldsig#">

    <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"
      xmlns="http://www.w3.org/2000/09/xmldsig#" />

  </Transforms>

  <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"
    xmlns="http://www.w3.org/2000/09/xmldsig#" />

  <DigestValue xmlns="http://www.w3.org/2000/09/xmldsig#">

    /eeKDHC9jMipvnzL/V6u2g5oECEQCkHIXbbOD697Rwk=</DigestValue>

  </Reference>

</SignedInfo>

<--*****

Signature Value / Key

*****-->

<SignatureValue xmlns="http://www.w3.org/2000/09/xmldsig#">

  Kplu73LcmVYli081.....BgRTPTyilpjlw==</SignatureValue>

<KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">

  <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">

    <X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">

      MIIDRjCCA...xs4tIMeVHaUF1q40y16xpbi96vayTfnGs=

    </X509Certificate>

```

```

    </X509Data

  </KeyInfo>

  <!-- *****

  Erstes Object-Element, nimmt xades:SignedProperties auf

  *****-->

  <Object xmlns="http://www.w3.org/2000/09/xmldsig#"
    xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">

  <!-- *****

  Target verweist auf das Signaturelement selbst (Id 3)

  *****-->

    <xades:QualifyingProperties

      Target="#_6B5A7A1E-01B1-B729-F406-015888803C9B"
      xmlns:xades="http://uri.etsi.org/01903/v1.3.2#"

  <!-- *****

  Signed Properties: SigningTime, Verweis auf Signaturzertifikat,
  SignaturePolicy, Verweis auf zugeordneten LegalAuthenticator,
  Attributzertifikat

  *****-->

    <xades:SignedProperties
      xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">

      <xades:SignedSignatureProperties

        Id="_A00DD61-8964-46C4-8FD2-8E7CFC9B84C9"
        xmlns:xades="http://uri.etsi.org/01903/v1.3.2#"

        <xades:SigningTime
          xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">
          2007-12-20T14:00:16</xades:SigningTime>

        <!-- *****

        Referenz auf das Signaturzertifikat

        *****-->

        <xades:SigningCertificate
          xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">

          <xades:Cert
            xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">

            <xades:CertDigest

              xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">

              <ds:DigestMethod
                Algorithm="http://www.w3.org/2001/04/xmldsig#sha256"
                xmlns:ds="http://www.w3.org/2000/09/xmldsig#" />

              <ds:DigestValue
                xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
                jvvyoVjXJ7qAhtdt7kL3HuIK+IV3FraKrDEhehkjiAI=

```

```

        </ds:DigestValue>
    </xades:CertDigest>
    <xades:IssuerSerial
        xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">
        <ds:X509IssuerName
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            CN=Test Signer, OU=TestLab, O=bos, L=Bremen, ST=Bremen, C=DE
        </ds:X509IssuerName>
        <ds:X509SerialNumber
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            1197199567</ds:X509SerialNumber>
        </xades:IssuerSerial>
    </xades:Cert>
</xades:SigningCertificate>

<!-- *****
Signature Policy
*****-->

    <xades:SignaturePolicyIdentifier
        xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">
        <xades:SignaturePolicyImplied
            xmlns:xades="http://uri.etsi.org/01903/v1.3.2#" />
        </xades:SignaturePolicyIdentifier>

<!-- *****
Referenz auf zugeordnetes legalAuthenticator-Element (Id 2)
*****-->

    <xades:SignerRole
        xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">
        <xades:ClaimedRoles
            xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">
            <xades:ClaimedRole
                xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">
                _F436B5F5-59CE-F2DA-502C-8ACD38DD3DF3
            </xades:ClaimedRole>
        </xades:ClaimedRoles>

<!-- *****
Attributzertifikat
*****-->

    <xades:CertifiedRoles
        xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">
        <xades:CertifiedRole
            xmlns:xades="http://uri.etsi.org/01903/v1.3.2#">

```

```

MIIDaDCCAtGgaDBmMFJhOLv...r5wnopSu3lyEMe7wFX4WrJo

</xades:CertifiedRole>

</xades:CertifiedRoles>

</xades:SignerRole>

</xades:SignedSignatureProperties>

</xades:SignedProperties>

</xades:QualifyingProperties>

</Object>

<!--*****
2. Object-Element mit Manifest (Referenz auf Stylesheet; Id 5)
*****-->

<Object xmlns="http://www.w3.org/2000/09/xmldsig#">

  <Manifest Id="_9080F14F-6936-E2FF-0F5D-A022F573129D"
    xmlns="http://www.w3.org/2000/09/xmldsig#">

    <Reference Id="_5C18BE44-E091-E5FA-5567-8F3D9FCDFBEA"

      URI="./CDP_V1_3.xsl" xmlns="http://www.w3.org/2000/09/xmldsig#">

        <Transforms xmlns="http://www.w3.org/2000/09/xmldsig#">

          <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"
            xmlns="http://www.w3.org/2000/09/xmldsig#" />

        </Transforms>

        <DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"
          xmlns="http://www.w3.org/2000/09/xmldsig#" />

        <DigestValue xmlns="http://www.w3.org/2000/09/xmldsig#">

          vI+JkLBhicxUCfkklolWlwxUANztkA3XLudlaXxRLVo=</DigestValue>

        </Reference>

      </Manifest>

    </Object>

  </Signature>

<!--*****
Beispielhaftes Attachment
*****-->

<cdp:UnstructuredContent Filename="BBDA4D01.JPG" xmime:contentType="Image/jpeg">
/9j/4AAQSkZJRgABAQAAAQABAAD//gAzQ. ....FRob21hcyBXLiBMaXBwAP/bAEMABQMEBAQDBQQEBAUFBQY
HDAGHBwCHDw</cdp:UnstructuredContent>

</cdp:ContentContainer>

</cdp:ContentPackage>

```

Listing 29: Beispiel