Elektronische Gesundheitskarte und Telematikinfrastruktur

Errata 4 zum Konnektor PTV1 (VSDM) und PTV2 Online-Produktivbetrieb (Stufe 1)

Version:	1.0.0
Stand:	15.11.2018
Status:	freigegeben
Klassifizierung:	öffentlich
Referenzierung:	[gemErrata_4_Kon_PTV1_PTV2]

Betroffene Produkttypen

gemProdT_Kon_PTV2

ID	Dokument	Quelle	Beschreibung der Änderung	**Anpassung** an Afos; TUCs, Tabellen, Korrekturen	**von Änderung betroffene Dokumente**
C_6574	Dokumentenlandkarte		**Redaktionelle Korrektur - Publikation importierter DTD datatypes.dtd und XMLSchema.dtd** Die Dateien datatypes.dtd und XMLSchema.dtd werden durch externe Schemadateien referenziert, sind aber bisher nicht Teil des von der gematik veröffentlichten Schnittstellendefinitionspakets. Um eine einheitliche Umsetzung sicherzustellen, werden die Dateien in das Dokumentenpaket aufgenommen.	Die Dateien datatypes.dtd und XMLSchema.dtd werden in das Schnittstellendefinitionspaket, Verzeichnis "ext", aufgenommen. Änderung in xenc-schema.xsd, xenc-schema-11.xsd, xmldsig-core-schema.xsd: <!DOCTYPE schema PUBLIC "-//W3C//DTD XMLSchema 200102//EN" Alt: "http://www.w3.org/2001/XMLSchema.dtd" [Neu: "XMLSchema.dtd"	Dokumentenlandkarte xenc-schema.xsd xenc-schema-11.xsd xmldsig-core-schema.xsd
C_6576	gemSpec_Kon	TAB_KON_777	**Korrektur - Fehlender Event-Parameter "Value" bei BNetzA_VL-Betriebszuständen** Der Parameter "Value=true/false" fehlt bei den OPERATIONAL_STATEs EC_BNetzA_VL_Update_Not_Successful und EC_BNetzA_VL_not_valid.	siehe C_6576_Anlage	gemSpec_Kon gemProdT_Kon_PTV2 gemProdT_Kon_PTV3
C_6577	gemSpec_Kon	TIP1-A_5034	**Korrektur - Neuer Fehlercode bei VerifyDocument: Keine Signatur im Aufruf** Wenn im Aufruf von VerifyDocument im Parameter SIG:Document bzw. SignatureObject ein Dokument übergeben wurde, das keine Signatur enthält, und auch keine weitere detached Signatur übergeben wird, fehlt ein passender Fehlercode.	siehe C_6577_Anlage	gemSpec_Kon gemProdT_Kon_PTV2 gemProdT_Kon_PTV3

gematik
Gesellschaft für Telematikanwendungen der Gesundheitskarte mbH

Änderungen in gemSpec_Kon

1 Anhang F – Übersicht Events

TAB_KON_777 Events Interne Mechanismen

Topic Ebene1 /Topic Ebene2 /Topic Ebene3	Typ	Schwere	Prot	An Clients	Parameter	Bedeutung	Auslöser (TUC/Op)
[...]							
BOOTUP /BOOTUP_COMPLETE	Op	Info	x	x		Änderung des Betriebszustandes	
OPERATIONAL_STATE /EC_CardTerminal_Software_Out_Of_Date($ctId)	Op	Info	x	x	Value=true/false; CtID=$ctId; Bedeutung=$EC.description	Änderung des Betriebszustandes durch Änderung im Fehlerzustand (Änderung im Value).	
[...]							
OPERATIONAL_STATE /EC_BNetzA_VL_Update_Not_Successful	Op	Info	x	x	Value=true/false; LastUpdate BNetzAVL=$lastUpdate BNetzAVL Timestamp; Bedeutung=$EC.description;	"	
OPERATIONAL_STATE /EC_BNetzA_VL_not_valid	Sec	Warning	x	x	Value=true/false; NextUpdate BNetzAVL=$NextUpdate-Element der BNetzA-VL; Bedeutung=$EC.description;	"	
[...]							

Änderungen in gemSpec_Kon

4.1.8 Signaturdienst

4.1.8.4.2 TUC_KON_161 „nonQES Dokumentsignatur prüfen”

TIP1-A_4654 - TUC_KON_161 „nonQES Dokumentsignatur prüfen”
Der Konnektor MUSS den technischen Use Case TUC_KON_161 „nonQES Dokumentsignatur prüfen” umsetzen.

TAB_KON_121 - TUC_KON_161 „nonQES Dokumentsignatur prüfen“

Element	Beschreibung
Name	TUC_KON_161 „nonQES Dokumentsignatur prüfen”
Beschreibung	Es wird die nicht-qualifizierte elektronische Signatur (nonQES) eines Dokuments geprüft. Dabei werden die Signaturverfahren laut Tabelle 183: TAB_KON_582 – Signaturverfahren unterstützt. Sind mehrere Signaturen vorhanden, so werden alle geprüft. Auch Parallel- und Gegensignaturen MÜSSEN unterstützt werden.
Auslöser	Aufruf durch ein Clientsystem (Operation VerifyDocument) oder ein Fachmodul
Vorbedingungen	keine
Eingangsdaten	• signedDocument (Signiertes Document vom Typ `nonQES_DocFormate`) • signature – *optional/falls detached Signatur* (Signatur. Es werden Parallel- und Gegensignaturen unterstützt.) • optionalInputs (optionale Eingabeparameter, siehe Operation VerifyDocument, Parameter SIG:OptionalInputs) • certificate – *optional/verpflichtend, wenn das Zertifikat nicht im signierten Dokument enthalten ist* (X.509-Zertifikat, gegen das die Signatur geprüft werden soll) ocspGracePeriod (OCSP-Grace Period: maximal zulässiger Zeitraum, den die letzte OCSP-Antwort aus dem Cache bezüglich des Referenzzeitpunkts zurückliegen darf) • xmlSchemas – *optional/nur für XML-Dokumente* (XMLSchema und ggf. weitere vom Hauptschema benutzte Schemata) • includeRevocationInfo: – *optional; Default = false* (Dieser optionale Parameter steuert die Einbettung von OCSP Antworten in die Signatur)
Komponenten	Konnektor
Ausgangsdaten	• verificationResult [VerificationResult] (Ergebnis der Signaturprüfung) • optionalOutput – *optional* (weitere Ausgabedaten gemäß SIG:OptionalOutput)
Standardablauf	1. **„DocumentValidation“**: Falls die Signatur im Dokument eingebettet ist, wird das signierte

	Dokument validiert durch Aufruf TUC_KON_080 „Dokument validieren“ { CheckDisplayability=false; … } Treten dabei Fehler bei Validierung der Typkonformität auf, wird die Prüfung mit einem Fehler abgebrochen. 2. **„CoreValidation“**: Es erfolgt die mathematische Prüfung der Signatur bestehend aus der Prüfung der Hash-Kette bis zum signierten Hashwert und der Prüfung der kryptographischen Signatur unter Verwendung des öffentlichen Schlüssels, des Signaturwertes und des signierten Hashwertes. XML-Signatur: Die Core Validation erfolgt entsprechend [XMLDSig] Kapitel 3.2 Core Validation. CMS-Signatur: Die Core Validation erfolgt entsprechend Cryptographic Message Syntax (CMS) Kapitel 5.6 Signature Verification Process [RFC5652]. PDF-Signatur: Die Core Validation erfolgt entsprechend [PAdES-3] Kapitel 4.6 Signature Validation aus PAdES-BES Part 3. Auch wenn die Validierung fehlschlägt, werden die folgenden Prüfschritte durchgeführt, so dass ein vollständiges Prüfprotokoll erstellt werden kann. 3. **„CheckSignatureCertificate“**: **Teil 1: Signaturzertifikat ermitteln** XML-Signatur: Das Signaturzertifikat ist im XMLDSig Element `ds:KeyInfo/ds:X509Data` gespeichert [XMLDSig] oder wird als Eingangsparameter übergeben. CMS-Signatur: Das Signaturzertifikat für CAdES ist im Feld `certificates` im `SignedData` Container gespeichert [CAdES] oder wird als Eingangsparameter übergeben. PDF-Signatur: Das PDF Signaturzertifikat für PAdES ist im Feld SignedData.certificates entsprechend Kapitel 6.1.1 „Placements of the signing certificate“ [PAdES Baseline Profile] gespeichert oder wird als Eingangsparameter übergeben. **Teil 2: Signaturzeitpunkt bestimmen** Der Signaturzeitpunkt Ermittelter_Signaturzeitpunkt_Eingebettet wird wie folgt selektiert: XML-Signatur: Das XML element `SigningTime` spezifiziert den Signaturzeitpunkt entsprechend Kapitel 7.2.1 XAdES [XAdES]. CMS-Signatur: Das Attribut `SigningTime` spezifiziert den Signaturzeitpunkt entsprechend Kapitel 11.3 CMS [CMS]. PDF-Signatur:

	Der Signaturzeitpunkt kann dem M Eintrag des Signature
	Dictionary entnommen werden [PAdES Baseline Profile]
	Kapitel 6.2.1 Signing time.
	Der Signaturzeitpunkt `Benutzerdefinierter_Zeitpunkt` liegt gegebenenfalls als Aufrufparameter vor. Der Signaturzeitpunkt `Ermittelter_Signaturzeitpunkt_System` wird ermittelt.
	Teil 3: Signaturzertifikatsprüfung:
	Bei der folgenden Signaturzertifikatsprüfung sind die Signaturzeitpunkte gemäß [TIP1-A_5545] zu berücksichtigen.
	Die Signaturzertifikatsprüfung erfolgt durch Aufruf von TUC_KON_037 „Zertifikat prüfen“, und zwar:
	Wenn es sich um das X.509-Zertifikat einer eGK handelt (PolicyList = oid_egk_aut bzw. oid_egk_autn), dann:
	TUC_KON_037 „Zertifikat prüfen“ {
	certificate;
	qualifiedCheck = not_required;
	baseTime = Signaturzeitpunkt;
	offlineAllowNoCheck = true;
	policyList = [oid_egk_aut | oid_egk_autn];
	intendedKeyUsage = digitalSignature&keyEncipherment;
	intendedExtendedKeyUsage = id-kp-clientAuth;
	ocspResponses = OCSP-Response;
	gracePeriod = ocspGracePeriod;
	validationMode = OCSP;
	getOCSPResponses = includeRevocationInfo }
	Wenn es ein X.509-Zertifikat der SM-B ist (PolicyList = oid_smc_b_osig), dann:
	TUC_KON_037 „Zertifikat prüfen“ {
	certificate;
	qualifiedCheck = not_required;
	baseTime = Signaturzeitpunkt;
	offlineAllowNoCheck = true;
	policyList = oid_smc_b_osig;
	intendedKeyUsage = nonRepudiation;
	ocspResponses = OCSP-Response;
	gracePeriod = ocspGracePeriod;
	validationMode = OCSP ;
	getOCSPResponses = includeRevocationInfo }
	Sind OCSP-Responses in der Signatur eingebettet, ist die jüngste OCSP-Response, die für die Zertifikatsprüfung notwendig ist, beim Aufruf von TUC_KON_037 zu übergeben.
	Sofern der Aufruf von TUC_KON_037 ocspResponsesRenewed zurückgibt, wird die Liste der OCSP-Responses in die Signatur eingebettet.

	Auch wenn die Zertifikatsprüfung fehlschlägt, werden die folgenden Prüfungen durchgeführt. 4. **"CheckPolicyConstraints"** In diesem Schritt wird das signierte Dokument entsprechend der Profilierung der Signaturformate (siehe Anhang B.2) geprüft. Es sind die Vorgaben für die Prüfung von Signaturen aus den Standards für AdES [XAdES], [XAdES Baseline], [CAdES], [CAdES Baseline], [PAdES-3] und [PAdES Baseline] umzusetzen. Dabei sind die Vorgaben aus Tabelle TAB_KON_779 „Profilierung der Signaturformate" zu erfüllen. Auch wenn nicht alle Anforderungen an das Format des signierten Dokuments erfüllt werden, wird die Prüfung mit den folgenden Schritten fortgesetzt, um ein vollständiges Prüfungsprotokoll zu erhalten. 5. Das **Prüfergebnis** (verificationResult, optionalOutput wird an den Aufrufer zurückgegeben (siehe TAB_KON_754 Übersicht Status für Prüfung einer Dokumentensignatur).
Varianten/ Alternativen	Im Fall, dass die Online-Prüfung des Sperrzustands des Signaturzertifikats nicht möglich ist und eine möglicherweise gecachte OCSP-Response nicht vorhanden ist oder nicht mehr verwendet werden darf, wird das Prüfergebnis mit der entsprechenden Warnung zurückgegeben. Im Fall einer PKCS#1-Signatur ist das verwendete Signaturverfahren, RSASSA-PSS bzw. RSASSA-PKCS1-v1_5, aus der Signatur zu bestimmen.
Fehlerfälle	Das Verhalten des TUCs bei einem Fehlerfall ist in TAB_KON_124 Fehlercodes TUC_KON_161 „nonQES Dokumentensignatur prüfen" beschrieben. (->1) keine Signatur in signedDocument und signature vorhanden: 4253 (→2 **„CoreValidation"**) Interner Fehler: 4001, Signatur des Dokument ungültig: 4115. (→3 „**CheckSignatureCertificate"**) Interner Fehler: 4001, Signaturzertifikat ermitteln fehlgeschlagen: 4206. (→4 **„CheckPolicyConstraints"**) Interner Fehler: 4001, Dokument nicht konform zu Regeln für nonQES: 4112.
Nichtfunktionale Anforderungen	keine
Zugehörige Diagramme	keine

TAB_KON_124 Fehlercodes TUC_KON_161 „nonQES Dokumentensignatur prüfen"

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen technischen Use Cases können folgende weiteren Fehlercodes auftreten.			
4001	Technical	Error	Interner Fehler
4206	Technical	Error	Signaturzertifikat ermitteln ist fehlgeschlagen

4112	Technical	Error	Dokument nicht konform zu Regeln für nonQES
4115	Security	Error	Signatur des Dokuments ungültig. Der SignatureValue des Dokuments ist falsch oder für mindestens eine Reference ist der DigestValue falsch.
4253	Technical	Error	Keine Signatur im Aufruf

Das Gesamtergebnis (VerificationResult) für die Prüfung einer Dokumentensignatur fasst die Ergebnisse aller Prüfungsschritte in einem einzelnen Statuswert zusammen.

TAB_KON_754 Übersicht Status für Prüfung einer Dokumentensignatur

VerificationResult für gesamtes Dokument (VerificationResult/HighLevelResult)	
Wert	**Bedeutung**
VALID	Wenn VerificationResult für alle Signaturen zum Dokument VALID
INVALID	Wenn VerificationResult für eine Signatur zum Dokument INVALID
INCONCLUSIVE	in allen anderen Fällen
VerificationResult pro Signatur (VerificationReport/IndividualReport/Result)	
Wert	**Bedeutung** **mögliche Ausprägungen im VerificationReport**
VALID	Die Signatur wurde gemäß den Regeln für die nonQES geprüft und für gültig befunden.
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:OnAllDocuments
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:HasManifestResults
INVALID	Die Signatur ist ungültig oder aufgrund eines Fehlers konnte die Signaturprüfung nicht durchgeführt werden.
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:invalid:IncorrectSignature
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidSignatureTimestamp
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:RequesterError
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:ResponderError
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation ResultMinor =

	urn:oasis:names:tc:dss:1.0:resultminor:invalid:IncorrectSignature
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:CertificateChainNotComplete
INCONCLUSIVE	Die Signatur wurde gemäß den Regeln für die nonQES geprüft. Allerdings konnten eine oder mehrere Prüfungen nicht vollständig durchgeführt werden. Einzelheiten finden sich in Result-Detail. Die Prüfungen, die durchgeführt werden konnten, waren erfolgreich.
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:OcspNotAvailiable Hinweis: Das Erreichen dieses Zustandes hängt davon ab, ob eine OCSP-Abfrage nicht durchgeführt werden konnte, unabhängig davon, ob die Ursache dafür die Offlineschaltung des Konnektors (MGM_LU_ONLINE = Disabled) oder die Nichterreichbarkeit des OCSP-Responders im Online-Betrieb (MGM_LU_ONLINE = Enabled) ist.

[<=]

4.1.8.4.6 TUC_KON_151 „QES-Dokumentensignatur prüfen“

TIP1-A_4672 - TUC_KON_151 „QES-Dokumentensignatur prüfen“
Der Konnektor MUSS den technischen Use Case TUC_KON_151 „QES-Dokumentensignatur prüfen” umsetzen.

TAB_KON_591 - TUC_KON_151 „QES-Dokumentensignatur prüfen“

Element	Beschreibung
Name	TUC_KON_151 „QES-Dokumentensignatur prüfen”
Beschreibung	Es wird die QES eines Dokuments geprüft. Dabei werden die Signaturverfahren laut Tabelle 183: TAB_KON_582 – Signaturverfahren unterstützt. Sind mehrere Signaturen vorhanden, so werden alle geprüft. Auch Parallel- und Gegensignaturen MÜSSEN unterstützt werden.
Eingangsanforderung	keine
Auslöser	Aufruf durch ein Clientsystem (Operation VerifyDocument) oder durch ein Fachmodul im Konnektor
Vorbedingungen	keine
Eingangsdaten	• signedDocument – *optional* (QES-signiertes Dokument vom Typ `QES_DocFormate` -> siehe Definition in Operation `VerifyDocument` mit `SIG:Document`) • signatureObject – *optional* (-> siehe Definition in Operation VerifyDocument mit

	dss:SignatureObject. Es werden Parallel- und Gegensignaturen unterstützt.) • optionalInputParams (optionale Eingabeparameter, siehe Operation VerifyDocument, Parameter SIG:OptionalInputs) • certificates – *optional/falls diese nicht im signierten Dokument enthalten sind, sondern nur referenziert werden* (X.509-Zertifikate). • xmlSchemas – *optional/nur für XML-Dokumente* (XMLSchema und ggf. weitere vom Hauptschema benutzte Schemata) • includeRevocationInfo [Boolean]: – *optional; Default: false* (Dieser optionale Parameter steuert die Einbettung von OCSP Antworten in die Signatur)
Komponenten	Konnektor
Ausgangsdaten	• verificationResult [VerificationResult] (Ergebnis der Signaturprüfung) • optionalOutput – *optional* (weitere Ausgabedaten gemäß SIG:OptionalOutput)
Standardablauf	1. **„DocumentValidation“**: Das signierte Dokument wird validiert mit Aufruf TUC_KON_080 „Dokument validieren“{ … }. Treten Fehler bei der Validierung der Typkonformität auf, wenn die Signatur im Dokument eingebettet ist, wird die Prüfung mit einem Fehler abgebrochen. Treten bei der Typkonformität, wenn die Signatur nicht im Dokument eingebettet ist, Fehler auf, so bricht der TUC nicht ab, sondern führt die folgenden Schritte soweit sinnvoll möglich durch. (Die Entscheidung über das sinnvoll Durchführbare liegt beim Hersteller des Konnektors.) 2. **„CoreValidation“**: Es erfolgt die mathematische Prüfung der Signatur, bestehend aus der Prüfung der Hash-Kette bis zum signierten Hashwert und der Prüfung der Signatur unter Verwendung des öffentlichen Schlüssels, des Signaturwertes und des signierten Hashwertes. XML-Signatur: Die Core Validation erfolgt entsprechend [XMLDSig] Kapitel 3.2 Core Validation. CMS-Signatur: Die Core Validation erfolgt entsprechend Cryptographic Message Syntax (CMS) Kapitel 5.6 Signature Verification Process [RFC5652]. PDF-Signatur: Die Core Validation erfolgt entsprechend [PAdES-3] Kapitel 4.6 Signature Validation aus PAdES-BES Part 3. Auch wenn die Validierung fehlschlägt, werden die folgenden Prüfschritte durchgeführt, so dass ein vollständiges Prüfprotokoll erstellt werden kann. 3. **„CheckSignatureCertificate“:** **Teil 1: Signaturzertifikat ermitteln** XML-Signatur: Das Signaturzertifikat ist im XMLDSig Element `ds:KeyInfo/ds:X509Data` gespeichert [XMLDSig] oder wird als Eingangsparameter übergeben. CMS-Signatur:

	Das Signaturzertifikat für CAdES ist im Feld `certificates` im `SignedData` Container gespeichert [CAdES] oder wird als Eingangsparameter übergeben.
	PDF-Signatur:
	Das PDF Signaturzertifikat für PAdES ist im Feld SignedData.certificates entsprechend Kapitel 6.1.1 „Placements of the signing certificate“ [PAdES Baseline Profile] gespeichert oder wird als Eingangsparameter übergeben.
	
	Teil 2: Signaturzeitpunkt bestimmen
	Der Signaturzeitpunkt `Ermittelter_Signaturzeitpunkt_Eingebettet` wird wie folgt selektiert:
	XML-Signatur:
	Das XML element `SigningTime` spezifiziert den Signaturzeitpunkt entsprechend Kapitel 7.2.1 XAdES [XAdES].
	CMS-Signatur:
	Das Attribut `SigningTime` spezifiziert den Signaturzeitpunkt entsprechend Kapitel 11.3 CMS [CMS].
	PDF-Signatur:
	Der Signaturzeitpunkt kann dem M Eintrag des Signature Dictionary entnommen werden [PAdES Baseline Profile] Kapitel 6.2.1 Signing time.
	Der Signaturzeitpunkt `Ermittelter_Signaturzeitpunkt_Qualifiziert` wird wie folgt selektiert:
	XML-Signatur:
	Das XML element `SignatureTimeStamp` spezifiziert den Signaturzeitpunkt entsprechend Kapitel 4.4.3.1 XAdES [XAdES].
	CMS-Signatur und PDF-Signatur:
	Das Attribut `signature-time-stamp` spezifiziert den Signaturzeitpunkt entsprechend Kapitel 6.1 CAdES [CAdES].
	Der Signaturzeitpunkt `Benutzerdefinierter_Zeitpunkt` liegt gegebenenfalls als Aufrufparameter vor. Der Signaturzeitpunkt `Ermittelter_Signaturzeitpunkt_System` wird ermittelt.
	
	Teil 3: Signaturzertifikatsprüfung:
	Bei der folgenden Signaturzertifikatsprüfung sind die Signaturzeitpunkte gemäß [TIP1-A_5540] zu berücksichtigen.
	Die Signaturzertifikatsprüfung erfolgt durch Aufruf von TUC_KON_037 „Zertifikat prüfen“ {
	certificate = C.HP.QES;
	qualifiedCheck = required;
	baseTime = Signaturzeitpunkt;
	offlineAllowNoCheck = true;
	validationMode = OCSP;
	ocspResponses = OCSP-Response;
	getOCSPResponses = includeRevocationInfo
	}.
	Sind OCSP-Responses in der Signatur eingebettet, ist die jüngsten OCSP-Response des EE-Zertifikats, die für die Zertifikatsprüfung notwendig ist, beim Aufruf von TUC_KON_037 zu übergeben.
	Sofern der Aufruf von TUC_KON_037 ocspResponses zurückgibt, wird die OCSP-Response des EE-Zertifikats in die Signatur eingebettet.
	Auch wenn die Zertifikatsprüfung fehlschlägt, werden die folgenden Prüfungen durchgeführt.

	4. **„CheckPolicyConstraints“**: In diesem Schritt wird das signierte Dokument entsprechend der Profilierung der Signaturformate (siehe Anhang B.2) geprüft. Es sind die Vorgaben für die Prüfung von Signaturen aus den Standards für AdES [XAdES], [XAdES Baseline], [CAdES], [CAdES Baseline], [PAdES-3] und [PAdES Baseline] umzusetzen. Dabei sind die Vorgaben aus Tabelle TAB_KON_779 „Profilierung der Signaturformate“ zu erfüllen. Auch wenn nicht alle Anforderungen an das Format des signierten Dokuments erfüllt werden, wird die Prüfung mit den folgenden Schritten fortgesetzt, um ein vollständiges Prüfungsprotokoll zu erhalten. 5. Das **Prüfergebnis** (VerificationResult, OptionalOutput) wird an den Aufrufer zurückgegeben (siehe TAB_KON_593 Übersicht Status für Prüfung einer Dokumentensignatur).
Varianten/Alternativen	Keine
Fehlerfälle	Das Verhalten des TUCs bei einem Fehlerfall ist in TAB_KON_592 Fehlercodes TUC_KON_151 „QES Dokumentensignatur prüfen“ beschrieben. (->1) keine Signatur in signedDocument und signatureObject vorhanden: 4253. (→ 2 **„CoreValidation“**) Interner Fehler: 4001, Signatur des Dokuments ungültig: 4115 (→3 **„CheckSignatureCertificate“**) Interner Fehler: 4001, Signaturzertifikat ermitteln ist fehlgeschlagen: 4206. (→4 **„CheckPolicyConstraints“**) Interner Fehler: 4001, Dokument nicht konform zu Regeln für QES: 4124, Dokument nicht konform zu Profilierung der Signaturformate: 4208.
Nichtfunktionale Anforderungen	Keine
Zugehörige Diagramme	keine

TAB_KON_592 Fehlercodes TUC_KON_151 „QES Dokumentensignatur prüfen“

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen technischen Use Cases können folgende weitere Fehlercodes auftreten:			
4001	Technical	Error	interner Fehler
4115	Security	Error	Signatur des Dokuments ungültig. Prüfung der Hashwertkette bzw. Prüfung der kryptographischen Signatur fehlgeschlagen.
4124	Technical	Error	Dokument nicht konform zu Regeln für QES
4206	Technical	Error	Signaturzertifikat ermitteln ist fehlgeschlagen
4208	Technical	Error	Dokument nicht konform zu Profilierung der Signaturformate
4253	Technical	Error	Keine Signatur im Aufruf

Das Gesamtergebnis (VerificationResult) für die Prüfung einer Dokumentensignatur fasst die Ergebnisse
aller Prüfungsschritte in einem einzelnen Statuswert zusammen.

TAB_KON_593 Übersicht Status für Prüfung einer Dokumentensignatur

VerificationResult für gesamtes Dokument (VerificationResult/HighLevelResult)		
Wert	Bedeutung	
VALID	Wenn VerificationResult für alle Signaturen zum Dokument VALID	
INVALID	Wenn VerificationResult für eine Signatur zum Dokument INVALID	
INCONCLUSIVE	in allen anderen Fällen	
VerificationResult pro Signatur (VerificationReport/IndividualReport/Result)		
Wert	Bedeutung mögliche Ausprägungen im VerificationReport	
VALID	Die Signatur wurde gemäß den Regeln für die QES geprüft und für gültig befunden.	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:OnAllDocuments	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:HasManifestResults	
INVALID	Die Signatur ist ungültig oder aufgrund eines Fehlers konnte die Signaturprüfung nicht durchgeführt werden.	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:invalid:IncorrectSignature	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:Success ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:InvalidSignatureTimestamp	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:RequesterError	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:ResponderError	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:invalid:IncorrectSignature	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:CertificateChainNotComplete	
INCONCLUSIVE	Die Signatur wurde gemäß den Regeln für die QES geprüft. Allerdings konnten eine oder mehrere Prüfungen nicht vollständig durchgeführt werden. Einzelheiten finden sich in Result-Detail. Die Prüfungen, die durchgeführt werden konnten, waren erfolgreich.	
	ResultMajor = urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation ResultMinor = urn:oasis:names:tc:dss:1.0:resultminor:OcspNotAvailiable Hinweis: Das Erreichen dieses Zustandes hängt davon ab, ob eine OCSP-Abfrage nicht durchgeführt werden konnte, unabhängig davon, ob die Ursache dafür die Offlineschaltung des Konnektors (MGM_LU_ONLINE = Disabled) oder die Nichterreichbarkeit des OCSP-	

	Responders im Online-Betrieb (MGM_LU_ONLINE = Enabled) ist.

[<=]

4.1.8.5.2 VerifyDocument (nonQES und QES)

TIP1-A_5034 - Operation VerifyDocument (nonQES und QES)
Der Signaturdienst des Konnektors MUSS an der Clientschnittstelle eine an [OASIS-DSS] angelehnte Operation VerifyDocument (nonQES und QES) anbieten**.**

[…]

TAB_KON_760 Ablauf Operation VerifyDocument (nonQES und QES)

Nr.	Aufruf Technischer Use Case oder Interne Operation	Beschreibung
1.	checkArguments	Die übergebenen Werte werden auf Konsistenz und Gültigkeit überprüft. Treten hierbei Fehler auf, so bricht die Operation mit Fehler 4000 ab.

2.	TUC_KON_000 „Prüfe Zugriffs- berechtigung“	Die Prüfung erfolgt durch den Aufruf
		TUC_KON_000 {
		mandantId = $context.mandantId;
		clientsystemId = $context.clientsystemId;
		workplaceId = $context.workplaceId;
		needCardSession= false;
		}
		Tritt bei der Prüfung ein Fehler auf, bricht die Operation mit Fehlercode aus TUC_KON_000 ab.

3.	prüfe, ob QES oder nonQES	Ist im jeweiligen Signaturzertifikat mindestens ein QCStatement mit dem OID id-etsi-qcs-QcCompliance (0.4.0.1862.1.1) enthalten, handelt es sich um eine QES-Signatur, andernfalls liegt eine nonQES-Signatur vor.
Für QES-Signaturen wird Schritt 4 ausgeführt. Für nonQES-Signaturen wird Schritt 5 ausgeführt.		
4.a	Prüfe Signaturdienst-Modul	Prüfe, ob MGM_LU_SAK=Enabled. Ist dies nicht der Fall, so bricht die Operation mit Fehler 4125 ab.
4.b	TUC_KON_151 „QES Dokumentensignatur prüfen“	Die QES wird geprüft. Tritt hierbei ein Fehler auf, bricht die Operation ab.
5.	TUC_KON_161 „nonQES Dokumentensignatur prüfen“	Die nonQES wird geprüft. Tritt hierbei ein Fehler auf, bricht die Operation ab.

TAB_KON_761 Fehlercodes „VerifyDocument (nonQES und QES)“

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen TUCs (siehe Tabelle 220: TAB_KON_760 Ablauf Operation VerifyDocument) können folgende weiteren Fehlercodes auftreten:			
4000	Technical	Error	Syntaxfehler
4125	Technical	Error	LU_SAK nicht aktiviert

[…]

[<=]