gematik
Gesellschaft für Telematikanwendungen der Gesundheitskarte mbH

Elektronische Gesundheitskarte und Telematikinfrastruktur

Spezifikation Wrapper

Version:	1.8.0
Revision:	18497
Stand:	24.08.16
Status:	freigegeben
Klassifizierung:	öffentlich
Referenzierung:	gemSpec_COS_Wrapper

Dokumentinformationen

Änderungen zur Vorversion

Überarbeitung der Dokumente für den Online-Produktivbetrieb (Stufe 1), als Grundlage für Produktivzulassungen und den bundesweiten Rollout.

Dokumentenhistorie

Version	Stand	Kap./ Seite	Grund der Änderung, besondere Hinweise	Bearbeitung
0.0.1	28.06.13		Erstversion des Dokumentes	ITS/SPE
0.1.0	26.07.13		Überarbeitung gemäß Industriekommentaren	ITS/SPE
1.0.0	02.08.13		In 1.6 wurde die neue Darstellung Date aufgenommen. Diese wird in 6.2.19 und 6.2.46 verwendet	gematik
1.1.0	15.08.13		Einarbeitung Kommentare	gematik
1.1.1	10.10.13		Überarbeitung wegen Kommentaren	gematik
1.3.0	21.02.14		Angaben in 6.2.51 korrigiert	gematik
1.4.0	01.04.14		Kap. 3.6 und Anh B ergänzt	gematik
1.5.0	06.06.14		Fehlerkorrekturen (Iteration 3)	gematik
1.6.0	26.08.14		• pointInTime MUSS geliefert werden • Wrapperfunktion prepareFingerprint • Schlüsselreferenzierung im objectLocator	gematik
1.7.0	17.07.15		Folgende Errata eingearbeitet: R1.4.2, R1.4.3	gematik
1.7.9	18.12.15		Anpassungen zum Online-Produktivbetrieb (Stufe 1)	Technik / SPE
1.8.0	24.08.16		freigegeben	gematik

Inhaltsverzeichnis

1 Einordnung des Dokumentes

1.1 Zielsetzung

Die vorliegende Spezifikation definiert die Anforderungen zu Herstellung, Test und Betrieb eines so genannten „Wrappers“. Hauptziel dieses Dokumentes ist es, einen Wrapper zu spezifizieren, der dem Auslesen von Objektinformationen aus Karten der G2 dient. Der Wrapper ist ein Liefergegenstand im Zulassungsobjekt COS und unterstützt das Prüftool bei der Beantwortung der Frage ob das Objektsystem eines Prüflings spezifikationskonform umgesetzt wurde. Darüber hinaus wird die Einsatzumgebung des Wrappers betrachtet. Zusätzlich findet sich eine Spezifikation zur Beschreibung von Objektsystemen in XML.

1.2 Zielgruppe

Da dieses Dokument im Wesentlichen die Schnittstelle zwischen Prüftool und Wrapper beschreibt, richtet es sich an Implementierer dieser Komponenten.

1.3 Geltungsbereich

Dieses Dokument enthält normative Festlegungen zur Telematikinfrastruktur des deutschen Gesundheitswesens. Der Gültigkeitszeitraum der vorliegenden Version und deren Anwendung in Zulassungs- oder Abnahmeverfahren wird durch die gematik GmbH in gesonderten Dokumenten (z.B. Dokumentenlandkarte, Produkttypsteckbrief, Leistungsbeschreibung) festgelegt und bekannt gegeben.

Schutzrechts-/Patentrechtshinweis

Die nachfolgende Spezifikation ist von der gematik allein unter technischen Gesichtspunkten erstellt worden. Im Einzelfall kann nicht ausgeschlossen werden, dass die Implementierung der Spezifikation in technische Schutzrechte Dritter eingreift. Es ist allein Sache des Anbieters oder Herstellers, durch geeignete Maßnahmen dafür Sorge zu tragen, dass von ihm aufgrund der Spezifikation angebotene Produkte und/oder Leistungen nicht gegen Schutzrechte Dritter verstoßen und sich ggf. die erforderlichen Erlaubnisse/Lizenzen von den betroffenen Schutzrechtsinhabern einzuholen. Die gematik GmbH übernimmt insofern keinerlei Gewährleistungen.

1.4 Abgrenzungen

Spezifiziert werden in dem Dokument die vom Wrapper bereitgestellten (angebotenen) Schnittstellen. Durch den Wrapper verwendete Schnittstellen sind herstellerspezifisch und werden deshalb hier nicht behandelt.

Dieses Dokument behandelt ausschließlich solche Attribute, die sich aus [gemSpec_COS] ergeben. Es ist möglich, dass sich im Rahmen der Evaluierung der Kartenplattform die Notwendigkeit ergibt, zur Unterstützung der sicherheitstechnischen

Bewertung eines Prüflings weitere sicherheitsrelevante Attribute aus einem Prüfling auszulesen und über die Wrapper-Schnittstelle zu transportieren. Daraus folgt, dass Anforderungen an die Implementierung des Wrappers und an den Rückgabewert des Wrappers nicht nur in diesem Dokument enthalten sind, sondern sich auch aus der Sicherheitsevaluierung ergeben.

1.5 Methodik

Anforderungen als Ausdruck normativer Festlegungen werden durch die dem RFC 2119 [RFC2119] entsprechenden, in Großbuchstaben geschriebenen deutschen Schlüsselworte MUSS, DARF NICHT, SOLL, SOLL NICHT, KANN gekennzeichnet. Abwandlungen von „**MUSS**“ zu „**MÜSSEN**“ etc. sind der Grammatik geschuldet.

Da im Beispielsatz „*Eine leere Liste DARF NICHT ein Element besitzen*.“ die Phrase „DARF NICHT“ semantisch irreführend wäre (wenn nicht ein, dann vielleicht zwei?), wird in diesem Dokument stattdessen „*Eine leere Liste DARF KEIN Element besitzen*.“ verwendet.

1.6 Nomenklatur

Darstellung	Bedeutung
??	Zeichenkette, die als Wildcard in HEX-Darstellungen verwendet wird um anzuzeigen, dass der konkrete Wert eines Oktetts irrelevant ist
`AttributeNotSet`	Diese Zeichenkette ist sowohl in einer Objektsystemspezifikation als auch für die Schnittstelle des Wrappers relevant und wird verwendet um anzuzeigen, dass ein Attribut noch nicht mit einem Wert belegt ist. Typischerweise werden derartige Attribute personalisiert. Beispiele dafür sind Schlüsseldaten in privaten Schlüsselobjekten oder PIN und PUK Werte.
DATE	HEX{6}-Zeichenkette, wobei jedes der zwölf Zeichen eine dezimale Ziffer ist und alle zwölf Zeichen ein Datum der Art YYMMDD im Format 0Y0Y0M0M0D0D angeben. Beispiel: Der 30. Dezember 2045 wird wie folgt codiert: 040501020300
HEX	Zeichenkette, die aus den Zeichen {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f} aufgebaut ist. Die Länge einer solchen Zeichenkette ist eine beliebige, gerade, nicht negative ganze Zahl, d.h. 0, 2, 4, 6, 8, 10, 12, … Beispiele: 0123 ist eine gültige HEX-Zeichenkette mit 4 Zeichen die zwei Oktette repräsentieren 123 ist keine HEX-Zeichenkette, da die Länge nicht gerade ist Ab ist keine HEX-Zeichenkette, da A nicht zum Definitionsbereich gehört ab ist eine gültige HEX-Zeichenkette mit 2 Zeichen, die ein Oktett repräsentieren
HEX{n}	Ein HEX-Zeichenkette, die aus genau n Oktetten besteht. Die Anzahl der Zeichen in der HEX-Zeichenkette ist dann $2n$.
HEX{x .. y}	Eine HEX-Zeichenkette, die aus mindestens x Oktetten und nicht mehr als y Oktetten besteht. Der kleinste Wert für x ist 0, der größte Wert für y ist ∞. Es gilt $x < y$.
HEX{x, y, z}	Eine HEX-Zeichenkette, die aus x, oder y, oder z Oktetten besteht.
INTEGER	HEX-Zeichenkette, die eine ganze Zahl repräsentiert. Die Umwandlung geschieht so, wie für das Tag ´02´ in [ISO/IEC 8825–1] definiert. Beispiele: INTEGER(0) = 00 INTEGER(127) = 7F INTEGER(255) = 00FF INTEGER(1) = 01 INTEGER(128) = 0080 INTEGER(256) = 0100
SE#	Security Environment Nummer: Die SE# wird als dezimale Zahl mit einem Wertebereich von [1, 254] dargestellt. Der Wert SE#=?? ist eine Wildcard für alle Security Environments, die das COS unterstützt und für die nicht anderweitig explizite

	Angaben vorliegen.
`Wildcard`	Diese Zeichenkette ist in Objektsystemspezifikationen immer dann als Wert zulässig, wenn der konkrete Wert von einem Hersteller beliebig gewählt werden kann. An der Schnittstelle zwischen Wrapper und Prüftool ist diese Zeichenkette nicht zulässig.

2 Systemüberblick

Im Rahmen der Zulassung von G2-Karten ist an mehreren Stellen nachzuweisen, dass die G2-Karten korrekt konfiguriert wurden. Dies betrifft sowohl den Test des Betriebssystems, der nur an korrekt konfigurierten Karten sinnvoll ist, als auch Tests des Objektsystems etwa im Rahmen der Anwendungstests. Stets ist unter anderem zu prüfen, ob das Objektsystem des Prüflings alle vorgegebenen Objekte und diese die korrekten Attributswerte enthalten.

Diesbezüglich gibt es zwei grundsätzliche Aspekte unter denen eine Prüfung auf Konformität vorgenommen wird. An dieser Stelle wird nur rudimentär auf die Aspekte eingegangen. Weitere Informationen finden sich in [Zertifizierungskonzept]:

1. Zunächst ist es aus funktionaler Sicht wichtig, dass ein Prüfling konform zu den Vorgaben der entsprechenden gematik-Dokumente ist. Bei dieser Prüfung stehen ausschließlich die in den gematik-Dokumenten festgelegten Objekte und deren Attribute im Vordergrund und herstellerspezifische, zusätzliche Objekte und Attribute, die der gewählten Implementierungsvariante geschuldet sind, sind irrelevant. Die funktionale Sicht auf den Prüfling wird stets im personalisierten Zustand vorgenommen.
2. Der zweite Aspekt ist die sicherheitstechnische Bewertung des Prüflings. Dabei ist nachzuweisen, dass der Prüfling alle sicherheitstechnischen Anforderungen erfüllt. Für diesen Aspekt sind Objekte und sicherheitsrelevante Attribute interessant, die sich aus der gewählten Implementierung ergeben (etwa EF.Rule oder EF.Pwd) und die über Festlegungen in gematik-Dokumenten hinausgehen. Relevant sind aber auch Erweiterungen in Zugriffsregeln gegenüber den gematik-Dokumenten etwa um herstellerspezifische Kommandos oder Kommandovarianten. Gemäß dem gegenwärtigen Stand von [Zertifizierungskonzept] wird die sicherheitstechnische Bewertung stets im initialisierten Zustand des Prüflings, das heißt vor der Personalisierung, vorgenommen.

Für die Prüfung eines Objektsystems sind folgende Artefakte relevant:

1. **Sollwert:** Der Sollwert beschreibt das Objektsystem mit sämtlichen darin enthaltenen Objekten, deren (maßgeblichen) Attributen und der Struktur des hierarchischen Objektsystems.
2. **Prüftool:** Das Prüftool vergleicht das Objektsystem eines Prüflings mit dem Sollwert.
3. **Wrapper:** Der Wrapper ist ein herstellerspezifisches Stück Software mit zwei Schnittstellen und wird vom Hersteller beigestellt. Das Prüftool kommuniziert nicht direkt mit dem Prüfling, sondern bedient sich eines Wrappers, der die Attributswerte herstellerspezifisch aus dem Prüfling ausliest und standardisiert an das Prüftool übermittelt.
4. **Prüfling:** Der Prüfling wird daraufhin untersucht, ob dessen Objektsystem den Sollwertvorgaben entspricht.

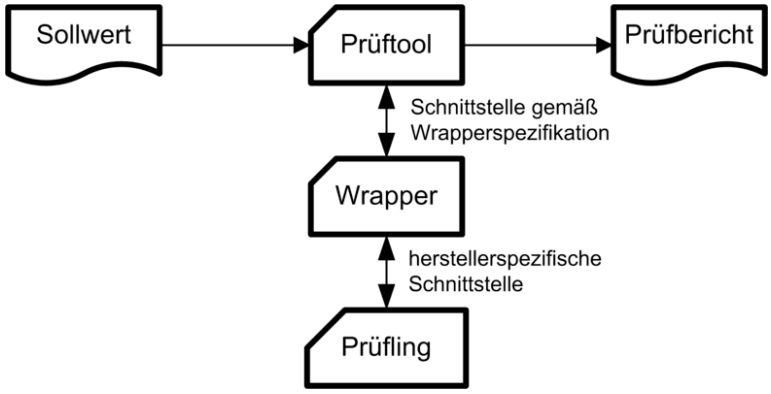

Abbildung 1: Artefakte des Imagetests

3 Implementierung des Wrappers

Der Hersteller liefert den Wrapper als jar-Datei, die in einer Java Standard Edition Laufzeitumgebung 7 oder höher lauffähig ist.

Das Interface IApduLayer.java stellt eine Schnittstelle bereit, mit deren Hilfe die Wrapper-Implementierung mit dem Prüfling kommuniziert.

Das Prüftool als Nutzer des Wrappers ist für die Implementierung des Interfaces IApduLayer verantwortlich.

Zur abstrakten Klasse „Wrapper“ erstellt der COS-Hersteller eine Unterklasse, mit deren Hilfe sich die Kartenkonfiguration aus einem Prüfling auslesen lässt.

3.1 Anforderungen an die Funktion getInformation

Anforderung „Exklusiver Zugriff“
Der Nutzer des Wrappers MUSS sicherstellen, dass nach Aufruf der Funktion `getInformation(...)` dem Wrapper die Schnittstelle zum Prüfling so lange exklusiv zur Verfügung steht, bis diese Funktion einen Returnwert liefert.

Anforderung „Nicht exklusiver Zugriff“
Der Implementierer des Wrappers MUSS damit rechnen, dass beliebige APDUs zum Prüfling gesendet werden, falls die Funktion `getInformation(...)` nicht aktiv ist, d.h. bevor sie benutzt wird, oder nachdem sie einen Returnwert lieferte und bis zu ihrem nächsten Aufruf.

Anforderung „Codierung `objectLocator`“
Der Aufrufer der Funktion `getInformation(...)` MUSS den Parameter `objectLocator` gemäß 3.4 codieren.

Anforderung „Codierung `return`-Wert“
Der Wrapper MUSS die Attribute der angefragten Objekte als XML-Struktur gemäß den übrigen Festlegungen dieses Dokumentes codieren. Es ist möglich, dass ein Knoten im `return`-Wert neben den in diesem Dokument genannten Attributen, die sich aus [gemSpec_COS] ergeben, weitere, herstellerspezifische Attribute enthält. Die Notwendigkeit solcher zusätzlichen Attribute ergibt sich typischerweise aus Anforderungen für die sicherheitstechnische Bewertung eines Prüflings.

Anforderung „Inhalt `return`-Wert“
Der Wrapper MUSS für jedes Objekt, welches durch den `objectLocator` referenziert wird, ein Element in das Array des Rückgabewertes einstellen.

Anforderung "Lesender Zugriff"
Der Wrapper DARF mit Ausnahme der weiter unten genannten Attribute KEINE persistenten Änderungen an Objekten oder deren Attributen vornehmen, die in Spezifikationen der gematik festgelegt sind. Die Ausnahmen sind:

1. Objektsystem → *persistentPublicKeyList*,
2. Objektsystem → *publicKeyList*.

3.2 Anforderungen an die Funktion prepareFingerprint

Anforderung "Exklusiver Zugriff“
Der Nutzer des Wrappers MUSS sicherstellen, dass nach Aufruf der Funktion `prepareFingerprint(...)` dem Wrapper die Schnittstelle zum Prüfling so lange exklusiv zur Verfügung steht, bis diese Funktion einen Returnwert liefert.

Anforderung "Nicht exklusiver Zugriff"
Der Implementierer des Wrappers MUSS damit rechnen, dass beliebige APDUs zum Prüfling gesendet werden, falls die Funktion `prepareFingerprint(...)` nicht aktiv ist, d.h. bevor sie benutzt wird, oder nachdem sie einen Returnwert lieferte und bis zu ihrem nächsten Aufruf.

Anforderung "Codierung`return`-Wert"
Der Wrapper MUSS den Wert True zurückliefern, wenn der Prüfling so vorbereitet ist, dass es möglich ist, ein ungesichertes Fingerprint Kommando an den Prüfling zu schicken. Andernfalls MUSS der Rückgabewert entweder False sein oder die Funktion wirft eine Exception.

Anforderung "Lesender Zugriff"
Die Anforderung „Lesender Zugriff“ aus 3.1 MUSS uneingeschränkt auch für die Funktion `prepareFingerprint` gelten.

3.3 Anforderungen an die Einsatzumgebung

Anforderung "Integre Kommunikation"
Die Einsatzumgebung des Wrappers MUSS gewährleisten, dass die Kommunikation zwischen Wrapper und Prüfling integer ist in dem Sinne, dass Nachrichten

1. während der Übertragung nicht verändert werden und
2. nicht unterdrückt werden und
3. nicht mehrfach geschickt werden (Replay).

Anforderung "Offene Kommunikation"
Der Implementierer des Wrappers DARF NICHT davon ausgehen, dass die Einsatzumgebung des Wrapper eine vertrauliche Kommunikation gewährleistet.

Hinweis (1): Gemäß den Anforderungen dieses Unterkapitels ist es für den Implementierer eines Wrappers nicht erforderlich weitere Maßnahmen zum Schutz der Kommunikation zwischen Wrapper und Prüfling vorzusehen.

3.4 Codierung objectLocator

Je nach Type wird ein angefragtes Objekt wie folgt in ein DER-codiertes TLV-Objekt gemäß [ISO/IEC 8825–1] mit Tag 'E0' codiert, wobei die hier angegebenen Referenzen sich auf das Dokument [gemSpec_COS] beziehen, sofern nicht anders angegeben:

1. Das Objektsystem (siehe (N019.900)) wird mit einem leeren DO'E0' referenziert.
2. Eine Applikation (siehe 8.3.1.1) wird mittels *applicationIdentifier* (siehe (N010.200)) in einem DO'4F' referenziert.
3. Ein Applikation Dedicated File (siehe 8.3.1.3) wird mittels *applicationIdentifier* (siehe (N010.700) → (N010.200)) in einem DO'4F' referenziert.

4. Ein Dedicated File (siehe 8.3.1.2) wird mittels *applicationIdentifier* eines Vorfahren, gefolgt von einem relativen Pfad in einem DO'51' referenziert.
5. Ein Elementary File (siehe 8.3.2) wird über seinen Ordner, gefolgt von seinem *fileIdentifier* (siehe (N010.800)) in einem DO'D1' referenziert.
6. Ein reguläres Passwortobjekt (siehe 8.4) wird über seinen Ordner, gefolgt von seinem *pwdIdentifier* in einem DO'83' referenziert, wobei das Wertfeld des DO'83' lediglich den*pwdIdentifier* enthält und insbesondere keine Zusatzinformationen zu „global“ oder „DF-spezifisch“.
7. Ein Multireferenz Passwortobjekt (siehe 8.4) wird über seinen Ordner, gefolgt von seinem *pwdIdentifier* in einem DO'83' referenziert, wobei das Wertfeld des DO'83' lediglich den*pwdIdentifier* enthält und insbesondere keine Zusatzinformationen zu „global“ oder „DF-spezifisch“.
8. Ein symmetrisches Authentisierungsobjekt (siehe 8.6.1) wird über seinen Ordner, gefolgt von seinem *keyIdentifier* in einem DO’C3’ referenziert.
9. Ein symmetrisches Kartenverbindungsobjekt (siehe 8.6.2) wird über seinen Ordner, gefolgt von seinem *keyIdentifier* in einem DO’C3’ referenziert.
10. Ein privates Schlüsselobjekt (siehe 8.6.3) wird über seinen Ordner, gefolgt von seinem *keyIdentifier* in einem DO’C3’ referenziert.
11. Ein öffentliches Schlüsselobjekt (siehe 8.6.4) wird über seinen Ordner, gefolgt von seinem *keyIdentifier* in einem passenden CRT referenziert.

3.5 Beispiel für ein Objektsystem

Die Beispielcodierungen von `objectLocator` in 3.6 basieren auf folgendem Objektsystem:

```
object system

MF, type=application: applicationIdentifier='F123456789AB'

+-- EF.DIR,          type=EF: fileIdentifier='2F00'

+-- PIN.CH,          type=regular password: pwdIdentifier='03'

+-- PrK.AUT,         type=private key object: keyIdentifier='02',
|                    algorithmIdentifier=elcRoleAuthentication='00'

+-- DF.a,             type=DF: fileIdentifier='DF0A'

|    +-- PrK.ENC,     type=private key object: keyIdentifier='06',
|    |                algorithmIdentifier=rsaDecipherOAEP='85'

|    +-- MRP.df-specific,     type=multireference password: pwdIdentifier='04'

|    +-- DF.b,         type=DF: fileIdentifier='DF0B'

|        +-- EF.c,     type=EF: fileIdentifier='EF0C'

|        +-- AES.AUT, type=symmetrical authentication object: keyIdentifier='0A',
|        |            algorithmIdentifier=aesSessionkey4SM='54'

|        +-- PrK.SIG, type=private key object: keyIdentifier>='0B',
|                    algorithmIdentifier=signPSS='05'

+-- PuK.RCA,         type=public signature verification object:
|                    keyIdentifier='1122334455667788',

|                    algorithmIdentifier    =ecdsa-with-SHA256
|                                           ={1.2.840.10045.4.3.2}
|                                           =´2A8648CE3D040302´

+-- CAN,             type=symmetrical card connection object: keyIdentifier='0E',
```

```
algorithmIdentifier    =id-PACE-ECDH-GM-AES-CBC-CMAC-128
                       ={0.4.0.127.0.7.2.2.4.2.2}
                       =´04007f00070202040202´
```

3.6 Beispiele zur Codierung des objectLocator

Die Objekte aus dem Beispiel in 3.5 werden wie folgt adressiert:

Tabelle 1: Beispielcodierungen objectLocator

Objekt	objectLocator
Objektsystem	e0 00
MF	e0 08 4f 06 f123456789ab
EF.DIR	e0 0c 4f 06 f123456789ab d1 02 2f00
PIN.CH	e0 0b 4f 06 f123456789ab 83 01 03
PrK.AUT	e0 0b 4f 06 f123456789ab c3 01 02
DF.a	e0 0c 4f 06 f123456789ab 51 02 df0a
PrK.ENC	e0 0f 4f 06 f123456789ab 51 02 df0a c3 01 06
MRP.df-specific	e0 0f 4f 06 f123456789ab 51 02 df0a 83 01 04
DF.b	e0 0e 4f 06 f123456789ab 51 04 df0adf0b
EF.c	e0 12 4f 06 f123456789ab 51 04 df0adf0b d1 02 ef0c
AES.AUT	e0 11 4f 06 f123456789ab 51 04 df0adf0b c3 01 0a
PrK.SIG	e0 11 4f 06 f123456789ab 51 04 df0adf0b c3 01 0b
PuK.RCA	e0 17 4f 06 f123456789ab b6 0d 95 01 80 83 08 1122334455667788
CAN	e0 0b 4f 06 f123456789ab c3 01 0e

3.7 Anbindung Konsistenz-Prüftool an Wrapper

Die Anforderungen in diesem Unterkapitel sind relevant für die Entwicklung von Konsistenz-Prüftool und Wrapper sowie für deren Anbindung.

Anforderung „Auslieferung Wrapper“
Die Wrapper-Implementierung MUSS als JAR-Archiv ausgeliefert werden. Die Wrapper-Implementierung MUSS in einer Java Standard Edition Laufzeitumgebung 7 lauffähig sein. Die Wrapper-Implementierung KANN in anderen Laufzeitumgebungen ausführbar sein.

Anforderung „Versionskennung“
Die Manifestdatei des JAR-Archives MUSS mindestens folgende Versionsangaben enthalten:

1. Aus der Manifestdatei von iwrapper.jar MÜSSEN folgende Informationen übernommen werden:
   a. Specification-Title
   b. Specification-Vendor
   c. Specification-Version
   d. Specification-iWrapper
2. Zulassung-Schluessel
3. Zulassung-Produkt
4. Zulassung-Version

Anforderung „Abhängigkeiten Wrapper“

Alle Abhängigkeiten des Wrapper JAR-Archivs MÜSSEN in dessen MANIFEST.MF innerhalb des Class-Path: Elements mit Angabe des relativen Pfades (z.B.: lib/<wrapper1-abhängigkeit1>.jar) spezifiziert sein und mitgeliefert werden, sodass sie vom Konsistenz-Prüftool automatisch beim Einlesen gefunden werden.

Anforderung „externe Abhängigkeit“

1. Die Wrapper-Implementierung MUSS gegen ein von der gematik bereitgestelltes JAR-Archiv entwickelt und kompiliert werden.
2. Dieses JAR-Archiv DARF beim Ausliefern der Wrapper-Implementierung WEDER mitgeliefert NOCH in der MANIFEST.MF-Datei eingetragen sein, da die enthaltenen Klassen zur Laufzeit vom Konsistzenz-Prüftool bereitgestellt werden.
3. Das JAR-Archiv MUSS genau folgende Klassen enthalten:
   a. ApduWrapperException, siehe B.1
   b. IApduLayer, siehe B.3
   c. IWrapper, siehe B.2
   d. WrapperException, siehe B.4

Anforderung „Schnittstelle Wrapper“

   e. Jede Wrapper-Implementierung MUSS folgende Klasse anbieten: `de.gematik.smartcard.g2.wrapper.Wrapper`
   f. Die Klasse `de.gematik.smartcard.g2.wrapper.Wrapper` MUSS folgende Methode anbieten, die vom Wrapper-Implementierer auszuimplementieren ist:

```
public static IWrapper getInstance() throws WrapperException
{
```

```
        . . .
}
```

4 Klassen

4.1 Rekord gemäß [gemSpec_COS#8.1.5]

Rekords werden als Elemente im Attribut *recordList* von strukturierten Dateien verwendet.

Tabelle 2: Attribute Rekord

Attributsname	[gemSpec_COS]	XML	Wrapper
number	(N007.600)	irrelevant	irrelevant
data	(N007.700)	siehe 6.2.16	irrelevant
lifeCycleStatus	(N007.800)	siehe 6.2.33	irrelevant

Hinweis (2): Die Rekordnummer und damit der Wert des Attributes number *ergibt sich implizit aus der Position eines Rekords in der* recordList*. Deshalb ist dieses Attribut für die Objektsystemspezifikation nicht erforderlich.*

Hinweis (3): Die Attribute eines Rekords sind an der Schnittstelle des Wrappers als irrelevant gekennzeichnet, weil sie Teil von recordList *sind. Diesbezüglich wird hier auf 5.1.10 verwiesen.*

4.2 Ordner

Diese generische Klasse und deren Unterklassen werden in [gemSpec_COS#8.3.1] behandelt.

4.2.1 Applikation gemäß [gemSpec_COS#8.3.1.1]

Diese Klasse wird in 4.2.3 behandelt wobei zu beachten ist, dass

1. in einer Objektsystemspezifikation kein *fileIdentifier* vorgegeben wird, aber ein *fileIdentifier* außerhalb des Intervalls [gemSpec_COS#(N006.700)] zugelassen ist.
2. in der Antwort eines Wrappers kein *fileIdentifier* vorhanden ist.

4.2.2 DF gemäß [gemSpec_COS#8.3.1.2]

Diese Klasse wird in 4.2.3 behandelt, wobei zu beachten ist, dass

1. in einer Objektsystemspezifikation kein *applicationIdentifier* vorhanden, aber ein beliebiger *applicationIdentifier* zugelassen wird.
2. in der Antwort des Wrappers kein *applicationIdentifier* vorhanden ist.

4.2.3 ADF gemäß [gemSpec_COS#8.3.1.3]

Diese Klasse wird hier als Generalisierung von Applikation und DF behandelt.

Tabelle 3: Attribute eines Application Dedicated File

Attributsname	[gemSpec_COS]	XML	Wrapper

accessRules	(N009.900)	siehe 6.2.2	MUSS
applicationIdentifier	(N010.200), (N014.300)	siehe 6.2.8	MUSS
children	(N010.000)	siehe 6.2.13	MUSS
fileIdentifier	(N010.800), (N014.200)	siehe 6.2.20	MUSS
lifeCycleStatus	(N009.800), (N014.500)	siehe 6.2.32	MUSS
shareable	(N009.850), (N014.100)	siehe 6.2.59	MUSS

4.3 Transparentes Elementary File

Diese Klasse wird in [gemSpec_COS#8.3.2.1] behandelt.

Tabelle 4: Attribute eines transparenten Elementary File

Attributsname	[gemSpec_COS]	XML	Wrapper
fileIdentifier	(N010.800), (N014.200)	siehe 6.2.20	MUSS
shortFileIdentifier	(N010.900), (N014.400)	siehe 6.2.60	MUSS
lifeCycleStatus	(N011.000), (N014.500)	siehe 6.2.32	MUSS
shareable	(N011.050), (N014.100)	siehe 6.2.59	MUSS
accessRules	(N011.100)	siehe 6.2.2	MUSS
flagTransactionMode	(N011.200)	siehe 6.2.24	MUSS
flagChecksum	(N011.300)	siehe 6.2.21	MUSS
numberOfOctet	(N011.500), (N014.000)	siehe 6.2.41	MUSS
positionLogicalEndOfFile	(N011.510), (N014.700)	siehe 6.2.47	MUSS
body	(N011.700)	siehe 6.2.9	DARF NICHT, 5.1.1

4.4 Linear variables Elementary File

Diese Klasse wird in [gemSpec_COS#8.3.2.2.1] behandelt.

Tabelle 5: Attribute eines linear variablen Elementary File

Attributsname	[gemSpec_COS]	XML	Wrapper
fileIdentifier	(N010.800), (N014.200)	siehe 6.2.20	MUSS
shortFileIdentifier	(N010.900), (N014.400)	siehe 6.2.60	MUSS
lifeCycleStatus	(N011.000), (N014.500)	siehe 6.2.32	MUSS
shareable	(N011.050), (N014.100)	siehe 6.2.59	MUSS
accessRules	(N011.100)	siehe 6.2.2	MUSS
flagTransactionMode	(N011.200)	siehe 6.2.24	MUSS
flagChecksum	(N011.300)	siehe 6.2.21	MUSS
recordList	(N012.200)	siehe 6.2.55	DARF NICHT, 5.1.10
maximumNumberOfRecords	(N012.400), (N014.100)	siehe 6.2.37	MUSS
maximumRecordLength	(N012.500), (N014.100)	siehe 6.2.39	MUSS

flagRecordLifeCycleStatus	(N012.600), (N014.600)	siehe 6.2.23	MUSS
numberOfOctet	(N011.500), (N014.000)	siehe 6.2.41	MUSS

4.5 Linear fixes Elementary File

Diese Klasse wird in [gemSpec_COS#8.3.2.2.2] behandelt.

Tabelle 6: Attribute eines linear fixen Elementary File

Attributsname	[gemSpec_COS]	XML	Wrapper
fileIdentifier	(N010.800), (N014.200)	siehe 6.2.20	MUSS
shortFileIdentifier	(N010.900), (N014.400)	siehe 6.2.60	MUSS
lifeCycleStatus	(N011.000), (N014.500)	siehe 6.2.32	MUSS
shareable	(N011.050), (N014.100)	siehe 6.2.59	MUSS
accessRules	(N011.100)	siehe 6.2.2	MUSS
flagTransactionMode	(N011.200)	siehe 6.2.24	MUSS
flagChecksum	(N011.300)	siehe 6.2.21	MUSS
recordList	(N012.200)	siehe 6.2.55	DARF NICHT, 5.1.10
maximumNumberOfRecords	(N012.400), (N014.100)	siehe 6.2.37	MUSS
maximumRecordLength	(N012.500), (N014.100)	siehe 6.2.39	MUSS
flagRecordLifeCycleStatus	(N012.600), (N014.600)	siehe 6.2.23	MUSS

4.6 Zyklisches Elementary File

Diese Klasse wird in [gemSpec_COS#8.3.2.2.3] behandelt.

Tabelle 7: Attribute eines zyklischen Elementary File

Attributsname	[gemSpec_COS]	XML	Wrapper
fileIdentifier	(N010.800), (N014.200)	siehe 6.2.20	MUSS
shortFileIdentifier	(N010.900), (N014.400)	siehe 6.2.60	MUSS
lifeCycleStatus	(N011.000), (N014.500)	siehe 6.2.32	MUSS
shareable	(N011.050), (N014.100)	siehe 6.2.59	MUSS
accessRules	(N011.100)	siehe 6.2.2	MUSS
flagTransactionMode	(N011.200)	siehe 6.2.24	MUSS
flagChecksum	(N011.300)	siehe 6.2.21	MUSS
recordList	(N012.200)	siehe 6.2.55	DARF NICHT, 5.1.10
maximumNumberOfRecords	(N012.400), (N014.100)	siehe 6.2.37	MUSS
maximumRecordLength	(N012.500), (N014.100)	siehe 6.2.39	MUSS
flagRecordLifeCycleStatus	(N012.600), (N014.600)	siehe 6.2.23	MUSS

4.7 Reguläres Passwort

Diese Klasse wird in [gemSpec_COS#8.4] behandelt.

Tabelle 8: Attribute eines regulären Passwortes

Attributsname	[gemSpec_COS]	XML	Wrapper
pwdIdentifier	(N015.000)	siehe 6.2.54	MUSS
lifeCycleStatus	(N015.050)	siehe 6.2.32	MUSS
accessRules	(N015.100)	siehe 6.2.2	MUSS
secret	(N015.200)	siehe 6.2.58	DARF NICHT, 5.1.11
minimumLength	(N015.300)	siehe 6.2.40	MUSS
maximumLength	(N015.310)	siehe 6.2.38	MUSS
startRetryCounter	(N015.400)	siehe 6.2.61	MUSS
retryCounter	(N015.500)	siehe 6.2.56	MUSS
transportStatus	(N015.600)	siehe 6.2.63	MUSS
flagEnabled	(N015.700)	siehe 6.2.22	MUSS
startSsecList	(N015.800)	siehe 6.2.62	MUSS
PUK	(N015.900)	siehe 6.2.52	DARF NICHT, 5.1.9
pukUsage	(N016.000)	siehe 6.2.53	MUSS

4.8 Multireferenz-Passwort

Diese Klasse wird in [gemSpec_COS#8.5] behandelt.

Tabelle 9: Attribute eines Multireferenz-Passwortes

Attributsname	[gemSpec_COS]	XML	Wrapper
pwdIdentifier	(N016.320)a	siehe 6.2.54	MUSS
lifeCycleStatus	(N016.320)b	siehe 6.2.32	MUSS
accessRules	(N016.320)c	siehe 6.2.2	MUSS
startSsecList	(N016.320)d	siehe 6.2.62	MUSS
flagEnabled	(N016.320)e	siehe 6.2.22	MUSS
passwordReference	(N016.320)f	siehe 6.2.45	MUSS

4.9 Symmetrisches Authentisierungsobjekt

Diese Klasse wird in [gemSpec_COS#8.6.1] behandelt.

Tabelle 10: Attribute eines symmetrischen Authentisierungsobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N016.400)a	siehe 6.2.28	MUSS
lifeCycleStatus	(N016.450)	siehe 6.2.32	MUSS

accessRules	(N016.500)	siehe 6.2.2	MUSS
keyType	(N016.590)	siehe 6.2.30	MUSS
encKey	(N016.600)	siehe 6.2.18	DARF NICHT, 5.1.4
macKey	(N016.700)	siehe 6.2.36	DARF NICHT, 5.1.5
numberScenario	(N016.710)	siehe 6.2.42	konditional MUSS irrelevant
algorithmIdentifier	(N016.800)	siehe 6.2.6	MUSS
accessRulesSessionkeys	(N016.820)	siehe 6.2.5	konditional MUSS irrelevant

4.10 Symmetrisches Kartenverbindungsobjekt

Diese Klasse wird in [gemSpec_COS#8.6.2] behandelt.

Tabelle 11: Attribute eines symmetrischen Kartenverbindungsobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N017.020)a	siehe 6.2.28	MUSS
lifeCycleStatus	(N017.024)	siehe 6.2.32	MUSS
accessRules	(N017.026)	siehe 6.2.2	MUSS
can	(N017.028)	siehe 6.2.10	DARF NICHT, 5.1.2
algorithmIdentifier	(N017.030)	siehe 6.2.6	MUSS

4.11 Privates ELC-Schlüsselobjekt

Diese Klasse wird in [gemSpec_COS#8.6.3] behandelt, wobei als Attribut *privateKey* eine Instanz der Klasse *privateElcKey* gemäß [gemSpec_COS#8.2.3.2] zum Tragen kommt.

Tabelle 12: Attribute eines privaten ELC-Schlüsselobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N017.100)a	siehe 6.2.28	MUSS
lifeCycleStatus	(N017.150)	siehe 6.2.32	MUSS
accessRules	(N017.200)	siehe 6.2.2	MUSS
privateElcKey → *domainParameter*	(N017.300), (N008.900)	siehe 6.2.48	MUSS
privateElcKey → *keyData*	(N017.300), 8.2.3.2	siehe 6.2.48	DARF NICHT, 5.1.7
numberScenario	(N017.430)	siehe 6.2.42	konditional MUSS irrelevant
listAlgorithmIdentifier	(N017.400)	siehe 6.2.34	MUSS
accessRulesSessionkeys	(N017.420)	siehe 6.2.5	konditional MUSS

			irrelevant
keyAvailable	(N018.200)	siehe 6.2.27	MUSS

4.12 Privates RSA-Schlüsselobjekt

Diese Klasse wird in [gemSpec_COS#8.6.3] behandelt, wobei als Attribut *privateKey* eine Instanz der Klasse *privateRsaKey* gemäß [gemSpec_COS#8.2.3.1] zum Tragen kommt.

Tabelle 13: Attribute eines privaten RSA-Schlüsselobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N017.100)a	siehe 6.2.28	MUSS
lifeCycleStatus	(N017.150)	siehe 6.2.32	MUSS
accessRules	(N017.200)	siehe 6.2.2	MUSS
privateRsaKey → *modulusLength*	(N017.300), (N008.710)	siehe 6.2.49	MUSS
privateRsaKey → *keyData*	(N017.300), 8.2.3.1	siehe 6.2.49	DARF NICHT, 5.1.8
listAlgorithmIdentifier	(N017.400)	siehe 6.2.34	MUSS
accessRulesSessionkeys	(N017.420)	siehe 6.2.5	konditional MUSS irrelevant
keyAvailable	(N018.200)	siehe 6.2.27	MUSS

4.13 Öffentliches ELC-Signaturprüfobjekt

Diese Klasse wird in [gemSpec_COS#8.6.4.1] behandelt, wobei als Attribut *publicKey* eine Instanz der Klasse *publicElcKey* gemäß [gemSpec_COS#8.2.4.2] zum Tragen kommt.

Tabelle 14: Attribute eines öffentlichen ELC-Signaturprüfobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N018.500), (N019.100)	siehe 6.2.29	MUSS
lifeCycleStatus	(N018.550)	siehe 6.2.32	MUSS
publicElcKey	(N018.600), 8.2.4.2	siehe 6.2.50	MUSS
oid	(N018.700), (N019.200)a.2	siehe 6.2.43	MUSS
accessRules	(N018.800)	siehe 6.2.2	MUSS
accessRulesPublicSignatureVerificationObject	(N019.110)a	siehe 6.2.4	MUSS
accessRulesPublicAuthenticationObject	(N019.110)b	siehe 6.2.3	MUSS

CHAT	(N019.210)a	siehe 6.2.12	MUSS
expirationDate	(N019.210)b	siehe 6.2.19	MUSS

4.14 Öffentliches RSA-Signaturprüfobjekt

Diese Klasse wird in [gemSpec_COS#8.6.4.1] behandelt, wobei als Attribut *publicKey* eine Instanz der Klasse *publicRsaKey* gemäß [gemSpec_COS#8.2.4.1] zum Tragen kommt.

Tabelle 15: Attribute eines öffentlichen RSA-Signaturprüfobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N018.500), (N019.100)	siehe 6.2.29	MUSS
lifeCycleStatus	(N018.550)	siehe 6.2.32	MUSS
publicRsaKey	(N018.600), 8.2.4.1	siehe 6.2.51	MUSS
oid	(N018.700), (N019.200)a.1	siehe 6.2.43	MUSS
accessRules	(N018.800)	siehe 6.2.2	MUSS
accessRulesPublicSignatureVerificationObject	(N019.110)a	siehe 6.2.4	MUSS
accessRulesPublicAuthenticationObject	(N019.110)b	siehe 6.2.3	MUSS

4.15 Öffentliches ELC-Authentisierungsobjekt

Diese Klasse wird in [gemSpec_COS#8.6.4.2] behandelt, wobei als Attribut *publicKey* eine Instanz der Klasse *publicElcKey* gemäß [gemSpec_COS#8.2.4.2] zum Tragen kommt.

Tabelle 16: Attribute eines öffentlichen ELC-Authentisierungsobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N018.500), (N019.500)	siehe 6.2.29	MUSS
lifeCycleStatus	(N018.550)	siehe 6.2.32	MUSS
publicElcKey	(N018.600), 8.2.4.2	siehe 6.2.50	MUSS
oid	(N018.700), (N019.600)a.2	siehe 6.2.43	MUSS
accessRules	(N018.800)	siehe 6.2.2	MUSS
CHAT	(N019.700)b.1	siehe 6.2.12	MUSS
expirationDate	(N019.700)b.2	siehe 6.2.19	MUSS

4.16 Öffentliches RSA-Authentisierungsobjekt

Diese Klasse wird in [gemSpec_COS#8.6.4.2] behandelt, wobei als Attribut *publicKey* eine Instanz der Klasse *publicRsaKey* gemäß [gemSpec_COS#8.2.4.1] zum Tragen kommt.

Tabelle 17: Attribute eines öffentlichen RSA-Authentisierungsobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N018.500), (N019.500)	siehe 6.2.29	MUSS
lifeCycleStatus	(N018.550)	siehe 6.2.32	MUSS
publicRsaKey	(N018.600), 8.2.4.1	siehe 6.2.51	MUSS
oid	(N018.700), (N019.600)a.1	siehe 6.2.43	MUSS
accessRules	(N018.800)	siehe 6.2.2	MUSS
CHA	(N019.700)a	siehe 6.2.11	MUSS

4.17 Öffentliches ELC-Verschlüsselungsobjekt

Diese Klasse wird in [gemSpec_COS#8.6.4.3] behandelt, wobei als Attribut *publicKey* eine Instanz der Klasse *publicElcKey* gemäß [gemSpec_COS#8.2.4.2] zum Tragen kommt.

Tabelle 18: Attribute eines öffentlichen ELC-Verschlüsselungsobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N018.500), (N019.822)	siehe 6.2.29	MUSS
lifeCycleStatus	(N018.550)	siehe 6.2.32	MUSS
publicElcKey	(N018.600), 8.2.4.2	siehe 6.2.50	MUSS
oid	(N018.700), (N019.824)a.2	siehe 6.2.43	MUSS
accessRules	(N018.800)	siehe 6.2.2	MUSS

4.18 Öffentliches RSA-Verschlüsselungsobjekt

Diese Klasse wird in [gemSpec_COS#8.6.4.3] behandelt, wobei als Attribut *publicKey* eine Instanz der Klasse *publicRsaKey* gemäß [gemSpec_COS#8.2.4.1] zum Tragen kommt.

Tabelle 19: Attribute eines öffentlichen RSA-Verschlüsselungsobjektes

Attributsname	[gemSpec_COS]	XML	Wrapper
keyIdentifier	(N018.500), (N019.500)	siehe 6.2.29	MUSS
lifeCycleStatus	(N018.550)	siehe 6.2.32	MUSS
publicRsaKey	(N018.600), 8.2.4.1	siehe 6.2.51	MUSS
oid	(N018.700), (N019.824)a.1	siehe 6.2.43	MUSS
accessRules	(N018.800)	siehe 6.2.2	MUSS

4.19 Objektsystem

Diese Klasse wird in [gemSpec_COS#9.1] behandelt.

Tabelle 20: Attribute eines Objektsystems

Attributsname	[gemSpec_COS]	XML	Wrapper
root	(N019.900)a	siehe 6.2.57	MUSS
coldAnswerToReset	(N019.900)b, (N024.100)	siehe 6.2.14	SOLL, 5.1.3
warmAnswerToReset	(N019.900)b, (N024.100)	siehe 6.2.64	SOLL, 5.1.13
iccsn8	(N019.900)c	siehe 6.2.25	MUSS
persistentPublicKeyList	Anhang I.1	siehe 6.2.44	MUSS
volatileCache	(N019.900)g	irrelevant	DARF NICHT, 5.1.12
lifeCycleStatus	(N019.900)i	siehe 6.2.32	MUSS
pointInTime	(N019.900)h	siehe 6.2.46	MUSS
listOfApplication	nicht enthalten	siehe 6.2.35	MUSS

Hinweis (4): Das Attribut volatileCache *ist sowohl für die Objektsystemspezifikation als auch für die Schnittstelle des Wrappers irrelevant. Begründung:*

a. *Das Attribut* volatileCache *ist ungeeignet um in einer Objektsystemspezifikation persistent zu speichernde Schlüsselobjekte zu spezifizieren, weil eine persistente Speicherung gemäß [gemSpec_COS#(N019.900)g.3] ausgeschlossen ist.*

b. *An der Schnittstelle des Wrappers sind volatil gespeicherte öffentliche Schlüsselobjekte irrelevant und persistent gespeicherte Schlüsselobjekte werden von* persistentPublicKeyList *erfasst, siehe 6.2.44.*

Hinweis (5): Die Frage nach dem Graphen des Dateisystems wird in 5.2 behandelt.

5 Prüfen von Objektsystemanforderungen ohne Wrapper

Kapitel 4 zeigt die Klassen und Attribute, soweit in [gemSpec_COS] spezifiziert und legt fest, welche Attribute der Wrapper zurückzumelden hat. Falls ein Attribut vom Wrapper nicht zurückgemeldet wird und für dieses Attribut Vorgaben existieren, dann ist eine Nutzung des Wrapper für ein derartiges Attribut nicht möglich. Dieses Kapitel zeigt Möglichkeiten zur Prüfung derartiger Attribute auf.

5.1 Attributsprüfung ohne Wrapper

Gemäß Kapitel 4 ist es für einige Klassen und Attribute zulässig, dass der Wrapper die spezifizierten Eigenschaften nur teilweise an seiner Schnittstelle zur Verfügung stellt. Für andere, insbesondere geheime Attribute, ist ein Transport über die Schnittstelle des Wrappers nicht zulässig. In diesem Unterkapitel wird dargelegt, wie für fehlende Attribute deren spezifikationskonforme Umsetzung zu prüfen ist.

5.1.1 Überprüfen von body

Das Attribut *body* ist für folgende Klassen definiert:

Tabelle 21: Prüfen des Attributes body

Klasse	Wrapper	Referenz
Transparentes Elementary File	DARF NICHT	Tabelle 4

Falls Anforderungen für das Attribut *body* getroffen werden, dann MUSS dieses Attribut aus dem Prüfling mittels Read Binary Kommando ausgelesen werden.

5.1.2 Überprüfen von can

Das Attribut *can* ist für folgende Klassen definiert:

Tabelle 22: Prüfen des Attributes can

Klasse	Wrapper	Referenz
Symmetrisches Kartenverbindungsobjekt	DARF NICHT	Tabelle 11

Falls Anforderungen für das Attribut *can* getroffen werden, dann MUSS dieses Attribut mittels General Authenticate Kommando geprüft werden.

5.1.3 Überprüfen von coldAnswerToReset

Das Attribut *coldAnswerToReset* ist für folgende Klassen definiert:

Tabelle 23: Prüfen des Attributes coldAnswerToReset

Klasse	Wrapper	Referenz
Objektsystem	SOLL	Tabelle 20

Falls Anforderungen für das Attribut *coldAnswerToReset* getroffen werden und der Wrapper dieses Attribut nicht zur Verfügung stellt, dann MUSS dieses Attribut durch einen "cold Reset" gemäß [gemSpec_COS#(N023.920)a, b)] überprüft werden.

5.1.4 Überprüfen von encKey

Das Attribut *encKey* ist für folgende Klassen definiert:

Tabelle 24: Prüfen des Attributes encKey

Klasse	Wrapper	Referenz
Symmetrisches Authentisierungsobjekt	DARF NICHT	Tabelle 10

Falls Anforderungen für das Attribut *encKey* getroffen werden, dann MUSS dieses Attribut durch ein Kommando überprüft werden, was den Schlüssel erfolgreich verwendet.

5.1.5 Überprüfen von macKey

Das Attribut *macKey* ist für folgende Klassen definiert:

Tabelle 25: Prüfen des Attributes macKey

Klasse	Wrapper	Referenz
Symmetrisches Authentisierungsobjekt	DARF NICHT	Tabelle 10

Falls Anforderungen für das Attribut *macKey* getroffen werden, dann MUSS dieses Attribut durch ein Kommando überprüft werden, was den Schlüssel erfolgreich verwendet.

5.1.6 Überprüfen von pointInTime

Da das Attribut *pointInTime* gemäß Tabelle 20 vom Wrapper zur Verfügung gestellt wird, ist es möglich diesbezügliche Anforderungen ohne weiteres zu prüfen.

5.1.7 Überprüfen von privateElcKey

Das Attribut *privateElcKey* ist für folgende Klassen definiert:

Tabelle 26: Prüfen des Attributes privateElcKey

Klasse	Wrapper	Referenz
Privates Schlüsselobjekt, ELC	DARF NICHT	Tabelle 12

Falls Anforderungen für das Attribut *privateElcKey* getroffen werden, dann MUSS dieses Attribut durch ein Kommando überprüft werden, was den Schlüssel erfolgreich verwendet.

5.1.8 Überprüfen von privateRsaKey

Das Attribut *privateRsaKey* ist für folgende Klassen definiert:

Tabelle 27: Prüfen des Attributes privateRsaKey

Klasse	Wrapper	Referenz

Privates Schlüsselobjekt, RSA	DARF NICHT	Tabelle 13

Falls Anforderungen für das Attribut *privateRsaKey* getroffen werden, dann MUSS dieses Attribut durch ein Kommando überprüft werden, was den Schlüssel erfolgreich verwendet.

5.1.9 Überprüfen von PUK

Das Attribut *PUK* ist für folgende Klassen definiert:

Tabelle 28: Prüfen des Attributes PUK

Klasse	Wrapper	Referenz
Reguläres Passwort	DARF NICHT	Tabelle 8

Falls Anforderungen für das Attribut *PUK* getroffen werden, dann MUSS dieses Attribut mittels Reset Retry Counter Kommando geprüft werden.

5.1.10 Überprüfen von recordList

Das Attribut *recordList* ist für folgende Klassen definiert:

Tabelle 29: Prüfen des Attributes recordList

Klasse	Wrapper	Referenz
Linear variables Elementary File	DARF NICHT	Tabelle 5
Linear fixes Elementary File	DARF NICHT	Tabelle 6
Zyklisches Elementary File	DARF NICHT	Tabelle 7

Falls Anforderungen für das Attribut *recordList* oder die darin enthaltenen Elemente getroffen werden, dann MUSS dieses Attribut wie folgt geprüft werden:

1. Mittels einer Schleife werden ausgehend von *number* = 1 so lange mittels Read Record Daten aus dem Prüfling gelesen, bis '6A83' = RecordNotFound das Ende von*recordList* anzeigt.
2. Jeder Schleifendurchlauf liefert zusätzlich zum Attribut *lifeCycleStatus* eines Rekords für aktivierte Rekords auch das Attribut *data*.
3. Falls *recordList* deaktivierte Elemente enthält, für die Vorgaben für das Attribut *data* existieren, dann sind die Attribute *data* dieser Elemente wie folgt zu prüfen:
   a. Die Elemente sind mittels Activate Record Kommando zu aktivieren.
   b. Die Attribute *data* sind mittels Read Record Kommando zu prüfen.

5.1.11 Überprüfen von secret

Das Attribut *secret* ist für folgende Klassen definiert:

Tabelle 30: Prüfen des Attributes secret

Klasse	Wrapper	Referenz
Reguläres Passwort	DARF NICHT	Tabelle 8

Falls Anforderungen für das Attribut *secret* getroffen werden, dann MUSS dieses Attribut mittels Verify Kommando geprüft werden.

5.1.12 Überprüfen von volatileCache

Das Attribut *volatileCache* ist für folgende Klassen definiert:

Tabelle 31: Prüfen des Attributes volatileCache

Klasse	Wrapper	Referenz
Objektsystem	DARF NICHT	Tabelle 20

Im Rahmen einer Objektsystemspezifikation ist es nicht möglich Anforderungen an dieses Attribut zu stellen, weil gemäß [gemSpec_COS#(N019.900)g.3] eine persistente Speicherung der in diesem Attribut möglicherweise enthaltenen öffentlichen Schlüsselobjekte ausgeschlossen ist.

5.1.13 Überprüfen von warmAnswerToReset

Das Attribut *warmAnswerToReset* ist für folgende Klassen definiert:

Tabelle 32: Prüfen des Attributes warmAnswerToReset

Klasse	Wrapper	Referenz
Objektsystem	SOLL	Tabelle 20

Falls Anforderungen für das Attribut *warmAnswerToReset* getroffen werden und der Wrapper dieses Attribut nicht zur Verfügung stellt, dann MUSS dieses Attribut durch einen "warm Reset" gemäß [gemSpec_COS#(N023.920)c)] überprüft werden.

5.2 Strukturprüfung

Die Strukturprüfung wird vom Prüftool vorgenommen. Sie ist für die Wrapper-Spezifikation uninteressant.

6 XML-Darstellung der Attribute eines Objektes

6.1 Verwendungszwecke der XML-Darstellung

Es ist möglich die XML-Darstellung eines Objektsystems in folgenden Fällen zu benutzen:

1. **Image-Editor:** Das bedeutet das Objektsystem wird mit all seinen Attributen und den darin enthaltenen Objekten in XML gemäß Kapitel 4 im Sinne einer Objektsystemspezifikation dargestellt. Das besondere an dieser Darstellung ist, dass für einige Attribute der Implementierer herstellerspezifische Werte festlegen darf. Dazu zählen auch Attribute, die personalisiert werden. Die Imagespezifikation sollte das berücksichtigen.
2. **Kartensimulator:** Das Objektsystem wird in XML für das Zielsystem Kartensimulator angegeben. Das besondere an dieser Darstellung ist, dass es hier keine Wahlmöglichkeiten gibt und keine herstellerspezifischen Erweiterungen.
3. **Sollwert:** Das Objektsystem wird in XML als Sollwert für einen Test der Kartenkonfiguration angegeben. Es ist möglich, dass der Sollwert weniger beinhaltet als von einer Objektsystemspezifikation gefordert ist, wenn nur Teilbereiche der Konfiguration getestet werden. Ebenso ist es möglich, dass der Sollwert mehr beinhaltet als von einer Objektsystemspezifikation plus ggf. Personalisierungsdaten umfasst, etwa wenn die Prüfstelle eines COS im Rahmen der Evaluierung Werte für die Attribute einer herstellerspezifischen Implementierung vorgibt.
4. **Wrapper:** Die Eigenschaften eines Objektes, welches real in einer Karte existiert, werden mittels eines Wrappers ausgelesen und von diesem an den Nutzer weitergereicht. Das Besondere an dieser Darstellung ist, dass
   a. einerseits nicht alle Attribute in der Darstellung vorhanden sind und
   b. andererseits weitere, herstellerspezifische Attribute vorhanden sein können.

6.2 Darstellung einzelner Attribute

Die Attribute sind in diesem Kapitel alphabetisch sortiert.

In den folgenden Unterkapiteln wird häufig folgende Phrase verwendet (wobei … den Namen eines Attributes enthält): An der Schnittstelle des Wrappers gilt für … dieselbe Darstellung wie in der Objektsystemspezifikation.

Diese Phrase ist wie folgt zu verstehen: Falls nicht anderweitig ausgeschlossen oder erlaubt, wird an der Schnittstelle des Wrappers für das Attribut dieselbe Darstellung verwendet, wie in der Objektsystemspezifikation ergänzt um herstellerspezifische Werte.

Beispielsweise in 6.2.31 *LCS_SE_dependendAccessRule* ist es möglich, dass es Einträge für LCS-Werte gibt, die gemäß 6.2.32 über die drei Werte `ACTIVATED`, `DEACTIVATED` oder `TERMINATED` hinausgehen. Ebenso ist es in 6.2.31 möglich, dass für SE# ein Wert verwendet wird, der außerhalb des Wertebereiches von

[gemSpec_COS#(N007.900)a] liegt, was nach [gemSpec_COS#(N007.900)b] zulässig ist.

6.2.1 XML-Darstellung von accessCondition

Im Rahmen einer Objektsystemspezifikation wird *accessCondition* wie folgt dargestellt:

```
condition
```

wobei *condition* genau einen Wert aus Tabelle 33 enthält.

Tabelle 33: XML-Darstellung accessCondition

[gemSpec_COS#10.2]	XML-Darstellung
ALWAYS	`ALW`
NEVER	siehe Hinweis (6):
PWD(*passwordReference*)	`PWD(HEX{1})`
AUT(*keyReference*)	`AUT(HEX{1})`
AUT(CHAT)	`AUTCHAT(OID : HEX{7})`
AUT(CHA)	`AUTC(HEX{7})`
SmMac(*keyInformation*)	• falls *keyInformation* einen symmetrischen Schlüssel referenziert: `SMMAC(HEX{1})` • falls *keyInformation* die Rolle eines RSA-Schlüssels referenziert: `SMMAC(HEX{7})` • falls *keyInformation* die Flags eines ELC-Schlüssels referenziert: `SMMAC(OID : HEX{7})`
SmCmdEnc	`SMCMDENC`
SmRspEnc	`SMRSPENC`
AND-Operator	`AND{...}`
OR-Operator	`OR{...}`

Hinweis (6): Eine Zugriffsart, für die keine commandDescription *besteht, ist gemäß [gemSpec_COS#(N023.100)a.1] implizit mit der Zugriffsbedingung NEVER verknüpft. Deshalb ist die Darstellung von NEVER in XML nicht erforderlich.*

Die Elemente AND und OR enthalten eine Liste mit mehr als einem Element aus Tabelle 33 die gemäß dem Operator als boolsche Verknüpfung aufgefasst werden. Die einzelnen Listenelemente werden durch Kommata getrennt.

An der Schnittstelle des Wrappers gilt für *accessCondition* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.2 XML-Darstellung von accessRules

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *accessRules* wie folgt dargestellt:

```
<attribute id="accessRules">interfaceDependentAccessRules</attribute>
```

wobei *interfaceDependentAccessRules* in 6.2.26 genauer beschrieben ist.

An der Schnittstelle des Wrappers gilt für das Attribut *accessRules* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.3 XML-Darstellung von accessRulesPublicAuthenticationObject

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *accessRulesPublicAuthenticationObject* wie folgt dargestellt:

```
<attribute id="accessRulesAuthentication>
  interfaceDependentAccessRules
</attribute>
```

wobei *interfaceDependentAccessRules* in 6.2.26 genauer beschrieben ist.

An der Schnittstelle des Wrappers gilt für das Attribut *accessRules* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.4 XML-Darstellung von accessRulesPublicSignatureVerificationObject

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *accessRulesPublicSignatureVerificationObject* wie folgt dargestellt:

```
<attribute id="accessRulesVerification">
  interfaceDependentAccessRules
</attribute>
```

wobei *interfaceDependentAccessRules* in 6.2.26 genauer beschrieben ist.

An der Schnittstelle des Wrappers gilt für das Attribut *accessRules* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.5 XML-Darstellung von accessRulesSessionkeys

Falls für ein Objekt das Attribut *accessRulesSessionkeys* vorgeschrieben ist (siehe [gemSpec_COS#(N016.820), (N017.420)]) dann wird in einer Objektsystemspezifikation das Attribut *accessRulesSessionkeys* wie folgt dargestellt:

```
<attribute id="accessRulesSessionkeys">
  interfaceDependentAccessRules
</attribute>
```

wobei *interfaceDependentAccessRules* in 6.2.26 genauer beschrieben ist.

Falls für ein Objekt das Attribut *accessRulesSessionkeys* nicht vorgeschrieben ist (siehe [gemSpec_COS#(N016.820), (N017.420)]) dann fehlt es in einer Objektsystemspezifikation.

Falls für ein Objekt in einer Objektsystemspezifikation ein Attribut *accessRulesSessionkeys* angegeben ist, dann MUSS dieses Attribut an der Schnittstelle des Wrappers vorhanden sein.

Falls für ein Objekt in einer Objektsystemspezifikation kein Attribut *accessRulesSessionkeys* angegeben ist, dann DARF dieses Attribut an der Schnittstelle des Wrappers NICHT vorhanden sein.

6.2.6 XML-Darstellung algorithmIdentifier

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *algorithmIdentifier* wie folgt dargestellt:

```
<attribute id="algorithmIdentifier"> ... >/attribute>
```

wobei als Wert genau ein Element aus Tabelle 34 verwendet wird.

An der Schnittstelle des Wrappers gilt für das Attribut *algorithmIdentifier* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.7 XML-Darstellung algorithmIdentifier in listAlgorithmIdentifier

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *algorithmIdentifier* wie folgt dargestellt:

Tabelle 34: XML-Darstellung algorithmIdentifier

Algorithmus	XML-Darstellung
aesSessionkey4SM	aesSessionkey4SM
aesSessionkey4TC	aesSessionkey4TC
desSessionkey4SM	desSessionkey4SM
desSessionkey4TC	desSessionkey4TC
elcAsynchronAdmin	elcAsynchronAdmin
elcRoleAuthentication	elcRoleAuthentication
elcSessionkey4SM	elcSessionkey4SM
elcSessionkey4TC	elcSessionkey4TC
elcSharedSecretCalculation	elcSharedSecretCalculation
id-PACE-ECDH-GM-AES-CBC-CMAC-128	id-PACE-ECDH-GM-AES-CBC-CMAC-128
id-PACE-ECDH-GM-AES-CBC-CMAC-192	id-PACE-ECDH-GM-AES-CBC-CMAC-192
id-PACE-ECDH-GM-AES-CBC-CMAC-256	id-PACE-ECDH-GM-AES-CBC-CMAC-256
rsaDecipherPKCS1_V1_5	rsaDecipherPKCS1_V1_5
rsaDecipherOaep	rsaDecipherOaep
rsaClientAuthentication	rsaClientAuthentication
rsaRoleAuthentication	rsaRoleAuthentication
rsaSessionkey4SM	rsaSessionkey4SM
rsaSessionkey4TC	rsaSessionkey4TC
sign9796_2_DS2	sign9796_2_DS2
signECDSA	signECDSA
signPKCS1_V1_5	signPKCS1_V1_5
signPSS	signPSS

An der Schnittstelle des Wrappers gilt für das Attribut *algorithmIdentifier* dieselbe Darstellung wie in der Objektsystemspezifikation. Falls das COS weitere Algorithmen unterstützt (siehe [gemSpec_COS#(N016.800)c, (N017.030)b], ist es möglich, dass das Attribut für herstellerspezifische Objekte Werte annimmt, die in der obigen Tabelle nicht genannt sind.

6.2.8 XML-Darstellung von applicationIdentifier

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *applicationIdentifier* wie folgt dargestellt:

```
<attribute id="applicationIdentifier"> ... </attribute>
```

Das Attribut *applicationIdentifier* enthält eine Menge mit einem oder mehreren HEX{5..16} Werten gemäß:

```
<attribute id="applicationIdentifier">{HEX{5..16}, ...}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *applicationIdentifier* dieselbe Darstellung wie in der Objektsystemspezifikation, wobei für jedes Element der Menge eine Darstellung gemäß HEX{1..∞} möglich ist, gemäß:

```
<attribute id="applicationIdentifier">{HEX{1..∞}, ...}</attribute>
```

Die Länge null wird hier explizit ausgeschlossen, weil ein fehlendes Attribut *applicationIdentifier* nicht durch einen leeren Wert, sondern durch die Abwesenheit des Attributes anzuzeigen ist. Längen größer 16 sieht [gemSpec_COS] nicht vor, könnten aber vom COS unterstützt werden.

6.2.9 XML-Darstellung von body

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *body* wie folgt dargestellt:

```
<attribute id="body">HEX</attribute>
```

Für die Schnittstelle des Wrappers ist das Attribut *body* irrelevant.

6.2.10 XML-Darstellung von can

Für die Objektsystemspezifikation gilt: Falls das Attribut *can* zu personalisieren ist, dann fehlt es in der Objektsystemspezifikation. Falls das Attribut *can* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann wird es wie folgt dargestellt:

```
<attribute id="can">HEX{8}</attribute>
```

Dabei muss der Oktettstring ein gültiger Format-2-PIN-Block gemäß [gemSpec_COS#(N008.100)] sein.

Für die Schnittstelle des Wrappers ist das Attribut *can* irrelevant.

6.2.11 XML-Darstellung con CHA

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *CHA* wie folgt dargestellt:

```
<attribute id="accessRights">HEX{7}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *CHA* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.12 XML-Darstellung con CHAT

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *CHAT* wie folgt dargestellt:

```
<attribute id="CHAT">oid : HEX{7}</attribute>
```

wobei *oid* aus der Menge {cvc_FlagList_TI, cvc_FlagList_CMS} zu wählen ist und der sieben Oktett lange Wert, der hinter einem Doppelpunkt folgt, die Flagliste repräsentiert.

An der Schnittstelle des Wrappers gilt für das Attribut *CHAT* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.13 XML-Darstellung von children

6.2.13.1 Objektsystemspezifikation

Im Rahmen der Objektsystemspezifikation gibt es für Ordner einen Abschnitt mit der Liste der Kindobjekte:

```
<children>
  ...
</children>
```

wobei für jedes Listenelement ein XML-Knoten gemäß 6.3 zulässig ist.

6.2.13.2 Wrapper-Schnittstelle

Weder in [gemSpec_COS] noch in [ISO/IEC 7816–4] ist ein Kommando enthalten mit dessen Hilfe sich alle Kinder eines Ordners effizient ermitteln lassen. Abhängig vom Objekttyp sind auf Basis von [gemSpec_COS] folgende Strategien denkbar eine solche Liste zusammenzustellen. Herstellerspezifisch mag es performantere Strategien geben. Da die Implementierung des Wrappers herstellerspezifisch ist, wird dies hier nicht weiter betrachtet, sondern es geht hier lediglich um eine Machbarkeitsanalyse: Gemäß [gemSpec_COS#(N010.000)a] sind folgende Objekttypen als Elemente in der Liste *children* zu berücksichtigen:

1. **Applikation:** Sofern der *applicationIdentifier* einer Applikation nicht im EF.DIR genannt oder aus der Objektsystemspezifikation bekannt ist, ist es praktisch unmöglich mit den normativen Mitteln aus [gemSpec_COS] eine vollständige Liste aller Applikationen zu erstellen. Zusatzapplikationen mit unbekanntem *applicationIdentifier* lassen sich praktisch nicht durch Ausprobieren aller möglichen Werte für *applicationIdentifier* finden. Falls pro Sekunde 1.000 Werte geprüft werden könnten (der reale Wert dürfte geringer sein), dauerte selbst das Durchprobieren aller fünf Oktett langen Werte 256^5 Werte / (1000 Werte/s) > 34 Jahre. Falls keine herstellerspezifischen Erweiterungen (etwa die nicht normative Variante des Select-Kommandos aus [gemSpec_COS#(N042.200)]) im COS enthalten sind, dann KÖNNEN Kindobjekte vom Typ Applikation im Attribut*children* fehlen, welches der Wrapper zurückliefert.
2. **Dedicated File:** Kindobjekte vom Typ Dedicated File lassen sich beispielsweise mittels einer Schleife über alle vom COS unterstützten *fileIdentifier* und der Select-Variante gemäß [gemSpec_COS#(N044.900)] ermitteln.
3. **Application Dedicated File:** Für diesen Objekttyp gelten dieselben Aussagen, wie für den Typ Dedicated File.
4. **Datei:** Kindobjekte vom Typ Datei lassen sich beispielsweise mittels einer Schleife über alle vom COS unterstützten *fileIdentifier* und der Select-Variante gemäß [gemSpec_COS#(N046.700)] ermitteln.
5. **Reguläres Passwort:** Kindobjekte vom Typ reguläres Passwort lassen sich beispielsweise mittels einer Schleife über alle vom COS unterstützten *pwdIdentifier* und der Get Pin Status Variante gemäß [gemSpec_COS#(N077.900)] ermitteln.

6. **Multireferenz Passwort:** Für diesen Objekttyp gelten dieselben Aussagen, wie für den Typ reguläres Passwort.
7. **Symmetrisches Authentisierungsobjekt:** Kindobjekte vom Typ symmetrisches Authentisierungsobjekt lassen sich beispielsweise mittels einer Schleife über alle vom COS unterstützten *keyIdentifier* und der MSE-Set-Variante gemäß [gemSpec_COS#(N100.900)] ermitteln. Dabei zeigen die Trailer NoError und UnsupportedFunction an, dass ein Schlüsselobjekt mit dem im Kommando enthaltenen *keyIdentifier* existiert. Der Trailer KeyNotFound zeigt an, dass ein solches Schlüsselobjekt nicht existiert.
8. **Privates Schlüsselobjekt:** Für diesen Objekttyp gelten dieselben Aussagen, wie für den Typ "symmetrisches Authentisierungsobjekt".
9. **Symmetrisches Kartenverbindungsobjekt:** Für diesen Objekttyp gelten dieselben Aussagen, wie für den Typ "symmetrisches Authentisierungsobjekt".
10. **Öffentliches Schlüsselobjekt:** Öffentliche Schlüsselobjekte werden im Zusammenhang mit den Attributen des Objektsystem betrachtet, siehe 4.19.

An der Schnittstelle des Wrappers wird die Liste der Kindobjekte wie folgt dargestellt:

Tabelle 35: XML-Darstellung children an der Wrapper-Schnittstelle

XML-Darstellung	Bemerkung
`<children>`	

`<DF_Identifier>`
`  <fileIdentifier>HEX</fileIdentifier>`
`...`
`</DF_Identifier>`
	Liste mit einem oder mehreren *fileIdentifier* gemäß 6.2.20, die zu einem Dedicated File oder Application Dedicated File gehören. Falls ein Ordner keine solchen Objekte enthält, dann fehlt diese Liste.

`<EF_Identifier>`
`  <fileIdentifier>HEX</fileIdentifier>`
`  ...`
`</EF_Identifier>`
	Liste mit einem oder mehreren *fileIdentifier* gemäß 6.2.20, die zu einer Datei gehören. Falls ein Ordner keine solchen Objekte enthält, dann fehlt diese Liste.

`<Key_Identifier>`
`  <keyIdentifier>HEX</keyIdentifier>`
`  ...`
`</Key_Identifier>`
	Liste mit einem oder mehreren *keyIdentifier* gemäß 6.2.28, die zu einem symmetrischen Authentisierungsobjekt, oder einem symmetrischen Kartenverbindungsobjekt, oder einem privaten Schlüsselobjekt gehören. Falls ein Ordner keine solchen Objekte enthält, dann fehlt diese Liste.

`<Password_Identifier>`
`  <pwdIdentifier>HEX</pwdIdentifier>`
`  ...`
`</Password_Identifier>`
	Liste mit einem oder mehreren pwd*Identifier* gemäß 6.2.54, die zu einem regulären Passwortobjekt oder einem Multireferenz Passwortobjekt gehören. Falls ein Ordner keine solchen Objekte enthält, dann fehlt diese Liste.

`</children>`	

6.2.14 XML-Darstellung von coldAnswerToReset

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *coldAnswerToReset* wie folgt dargestellt:

```
<attribute id="coldAnswerToReset">HEX{9..33}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *coldAnswerToReset* dieselbe Darstellung wie in der Objektsystemspezifikation.

Hinweis (7): Der kürzest mögliche ATR, der konform zu [gemSpec_COS] ist, besteht aus den acht Zeichen: T0, TA1, TD1, TD2, TA3, TD3, TA4 und TCK. Typischerweise wird auch das Zeichen TS (siehe [ISO/IEC 7816-3#8.1) zum ATR gezählt, was zur unteren Schranke für die Länge des Attributes coldAnswerToReset *führt.*

Hinweis (8): Die obere Schranke für die Länge des Attributes coldAnswerToReset *basiert auf der gemäß [ISO/IEC 7816-3] maximal möglichen Länge von 32 Zeichen plus dem TS-Zeichen.*

6.2.15 XML-Darstellung von commandDescription

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *commandDescription* wie folgt dargestellt:

```
HEX{1}||HEX{1}||HEX{1}||HEX{1}
```

Es handelt sich dabei also um eine Zeichenkette von vier Oktetten, die durch die beiden Konkatenationszeichen || voneinander getrennt sind. Für jedes Oktett ist als Wert auch die Zeichenkette ?? zulässig, wodurch angezeigt wird, dass dieser Teil der Kommandobeschreibung für die Beurteilung der Zugriffsart irrelevant ist.

Das erste Oktett repräsentiert das CLA-Byte gemäß [gemSpec_COS#(N026.510)] wobei als Kanalnummer 0 zu verwenden ist und kein Secure Messaging angezeigt wird.

Das zweite Oktett repräsentiert ein INS-Byte gemäß [gemSpec_COS#(N026.600).

Das dritte Oktett repräsentiert ein P1-Byte gemäß [gemSpec_COS#(N026.700)].

Das vierte Oktett repräsentiert ein P2-Byte gemäß [gemSpec_COS#(N026.800)].

An der Schnittstelle des Wrappers gilt für *commandDescription* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.16 XML-Darstellung von data

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *data* wie folgt dargestellt:

```
<attribute id="recordValue">HEX</attribute>
```

Für die Schnittstelle des Wrappers ist das Attribut *data* irrelevant.

6.2.17 XML-Darstellung von elementaryAccessRule

Im Rahmen einer Objektsystemspezifikation wird *elementaryAccessRule* wie folgt dargestellt:

```
({accessMode}, accessCondition)
```

wobei gilt:

1. Der *accessMode* ist eine Menge von kommaseparierten Elementen. Jedes Element ist eine *commandDescription* gemäß 6.2.15.
2. Der Knoten *accessCondition* wird gemäß 6.2.1 codiert.

An der Schnittstelle des Wrappers gilt für *elementaryAccessRule* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.18 XML-Darstellung von encKey

Für die Objektsystemspezifikation gilt: Falls das Attribut *encKey* zu personalisieren ist, dann fehlt es in der Objektsystemspezifikation. Falls das Attribut *encKey* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann wird es in Abhängigkeit vom Attribut *keyType* (siehe 6.2.30) wie folgt dargestellt:

Tabelle 36: XML-Darstellung encKey

***encKey* gemäß [gemSpec_COS#(N016.600)]**	**XML-Darstellung**
3TDES	`<attribute id="encKey">HEX{24}</attribute>`
AES-128	`<attribute id="encKey">HEX{16}</attribute>`
AES-192	`<attribute id="encKey">HEX{24}</attribute>`
AES-256	`<attribute id="encKey">HEX{32}</attribute>`

Für die Schnittstelle des Wrappers ist das Attribut *encKey* irrelevant.

6.2.19 XML-Darstellung von expirationDate

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *expirationDate* wie folgt dargestellt:

```
<attribute id="expirationDate">Date</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *expirationDate* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.20 XML-Darstellung von fileIdentifier

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *fileIdentifier* wie folgt dargestellt:

```
<attribute id="fileIdentifier">HEX{2}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *fileIdentifier* die Darstellung:

```
<attribute id="fileIdentifier">HEX{1..∞}</attributre>
```

Die Länge null wird hier explizit ausgeschlossen, weil ein fehlendes Attribut *fileIdentifier* nicht durch einen leeren Wert, sondern durch die Abwesenheit des Attributes anzuzeigen ist. Längen größer zwei sieht [gemSpec_COS] nicht vor, könnten aber vom COS unterstützt werden.

6.2.21 XML-Darstellung von flagChecksum

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *flagChecksum* wie folgt dargestellt:

Tabelle 37: XML-Darstellung flagChecksum

[gemSpec_COS#(N011.300)]	**XML-Darstellung**
True	`<attribute id="flagChecksum">TRUE</attribute>`
False	`<attribute id="flagChecksum">FALSE</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *flagChecksum* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.22 XML-Darstellung von flagEnabled

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *flagEnabled* wie folgt dargestellt:

Tabelle 38: XML-Darstellung flagEnabled

[gemSpec_COS# (N015.700), (N016.320)e]	XML-Darstellung
True	`<attribute id="flagEnabled">TRUE</attribute>`
False	`<attribute id="flagEnabled">FALSE</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *flagEnabled* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.23 XML-Darstellung von flagRecordLifeCycleStatus

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *flagRecordLifeCycleStatus* wie folgt dargestellt:

Tabelle 39: XML-Darstellung flagRecordLifeCycleStatus

[gemSpec_COS#(N012.600)]	XML-Darstellung
True	`<attribute id="flagRecordLifeCycleStatus>TRUE</attribute>`
False	`<attribute id="flagRecordLifeCycleStatus>FALSE</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *flagRecordLifeCycleStatus* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.24 XML-Darstellung von flagTransactionMode

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *flagTransactionMode* wie folgt dargestellt:

Tabelle 40: XML-Darstellung flagTransactionMode

[gemSpec_COS#(N011.200)]	XML-Darstellung
True	`<attribute id="flagTransactionMode">TRUE</attribute>`
False	`<attribute id="flagTransactionMode">FALSE</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *flagTransactionMode* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.25 XML-Darstellung von iccsn8

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *iccsn8* wie folgt dargestellt:

```
<attribute id="iccsn8">HEX{8}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *iccsn8* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.26 XML-Darstellung von interfaceDependentAccessRules

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *interfaceDependentAccessRules* wie folgt dargestellt:

```
InterfaceType{...}, ...
```

wobei *InterfaceType* gemäß Tabelle 41 codiert wird. Mit anderen Worten: Schnittstellenabhängige Zugriffsregeln werden als kommaseparierte Liste dargestellt und jedes Listenelement setzt sich zusammen aus einer Beschreibung für den Typ der physikalischen Schnittstelle gefolgt von einer Menge und jedes Element dieser Menge ist eine *LCS_SE_dependendAccessRule*, die in 6.2.31 genauer beschrieben wird.

Tabelle 41: Codierung der physikalischen Schnittstelle in Zugriffsregeln

physikalische Schnittstelle	XML-Darstellung
kontaktbehaftete Schnittstelle gemäß [gemSpec_COS#11.2.1, 11.2.2]	CB
kontaktlose Schnittstelle gemäß [gemSpec_COS#11.2.3]	CL

An der Schnittstelle des Wrappers gilt für *interfaceDependentAccessRules* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.27 XML-Darstellung von keyAvailable

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *keyAvailable* wie folgt dargestellt:

Tabelle 42: XML-Darstellung keyAvailable

[gemSpec_COS#(N018.200)]	XML-Darstellung
True	<attribute id="keyAvailable">TRUE</attribute>
False	<attribute id="keyAvailable">FALSE</attribute>

An der Schnittstelle des Wrappers gilt für das Attribut *keyAvailable* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.28 XML-Darstellung von keyIdentifier für private und geheime Schlüssel

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *keyIdentifier* für private oder geheime Schlüsselobjekte wie folgt dargestellt:

```
<attribute id="keyIdentifier">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *keyIdentifier* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.29 XML-Darstellung von keyIdentifier für öffentliche Schlüssel

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *keyIdentifier* für öffentliche Schlüsselobjekte wie folgt dargestellt:

```
<attribute id="keyIdentifier">HEX{8, 12}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *keyIdentifier* die Darstellung:

```
<attribute id="keyIdentifier">HEX{1..∞}</attributre>
```

Die Länge null wird hier explizit ausgeschlossen, weil ein Schlüssel ohne *keyIdentifier* nicht selektierbar ist. Längen ungleich acht oder zwölf sieht [gemSpec_COS] nicht vor, könnten aber vom COS unterstützt werden.

6.2.30 XML-Darstellung von keyType

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *keyType* wie folgt dargestellt:

Tabelle 43: XML-Darstellung keyType

[gemSpec_COS#(N016.590)a]	XML-Darstellung
3TDES	`<attribute id="keyMode">3DES</attribute>`
AES-128	`<attribute id="keyMode">AES_128</attribute>`
AES-192	`<attribute id="keyMode">AES_192</attribute>`
AES-256	`<attribute id="keyMode">AES_256</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *keyType* dieselbe Darstellung wie in der Objektsystemspezifikation. Falls das COS weitere Schlüsseltypen unterstützt (siehe [gemSpec_COS#(N016.590)b], ist es möglich, dass das Attribut für herstellerspezifische Objekte Werte annimmt, die in der obigen Tabelle nicht genannt sind.

6.2.31 XML-Darstellung von LCS_SE_dependendAccessRule

Im Rahmen einer Objektsystemspezifikation wird *LCS_SE_dependendAccessRule* wie folgt dargestellt:

`[`*`LCS`*`, SE#, {...}]`

wobei gilt:

1. *LCS* zeigt an, für welchen *lifeCycleStatus* diese *LCS_SE_dependendAccessRule* gilt. *LCS* ist ein Wert aus der Menge `{ACTIVATED, DEACTIVATED, TERMINATED}`.
2. SE# zeigt an, für welches Security Environment diese *LCS_SE_dependendAccessRule* gilt.
   a. Die Codierung der Security Environment Nummer wird gemäß 1.6 vorgenommen.
   b. Im Rahmen einer Objektsystemspezifikation wird für *LCS_SE_dependendAccessRule* mit einem *lifeCycleStatus* ungleich "`ACTIVATED`" die Security Environment Nummer 0 verwendet, d.h. SE# wird mit ?? codiert.
   c. An der Schnittstelle des Wrappers sind für *LCS* und SE# beliebige Kombinationen zulässig.
3. Die Menge enthält ein oder mehrere kommaseparierte Elemente. Jedes Element ist vom Typ *elementaryAccessRule* gemäß 6.2.17.

An der Schnittstelle des Wrappers gilt für *LCS_SE_dependendAccessRule* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.32 XML-Darstellung von lifeCycleStatus für Objekte außer Rekords

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *lifeCycleStatus* wie folgt dargestellt:

Tabelle 44: XML-Darstellung lifeCycleStatus für Objekte außer Rekords

[gemSpec_COS#(N007.100)a]	XML-Darstellung
„Operational state (active)“	`<attribute id="lifeCycleStatus">ACTIVATED</attribute>`
„Operational state (deactivated)“	`<attribute id="lifeCycleStatus">DEACTIVATED</attribute>`
„Termination state"	`<attribute id="lifeCycleStatus">TERMINATED</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *lifeCycleStatus* dieselbe Darstellung wie in der Objektsystemspezifikation. Falls das COS weitere Status unterstützt (siehe [gemSpec_COS#(N007.100)b], ist es möglich, dass das Attribut für herstellerspezifische Objekte Werte annimmt, die in der obigen Tabelle nicht genannt sind.

6.2.33 XML-Darstellung von lifeCycleStatus für Rekords

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *lifeCycleStatus* für Rekords wie folgt dargestellt:

Tabelle 45: XML-Darstellung lifeCycleStatus für Rekords

[gemSpec_COS#(N007.100)a]	XML-Darstellung
„Operational state (active)“	`<attribute id="recordLifeCycleStatus">ACTIVATED</attribute>`
„Operational state (deactivated)“	`<attribute id="recordLifeCycleStatus">DEACTIVATED</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *lifeCycleStatus* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.34 XML-Darstellung von listAlgorithmIdentifier

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *listAlgorithmIdentifier* wie folgt dargestellt:

```
<attribute id="listAlgorithmIdentifier"> ... </attribute>
```

Das Attribut *listAlgorithmIdentifier* enthält eine Menge von 2-Tupeln, wobei jedes dieser 2-Tupel in eckigen Klammern eingeschlossen ist. Das erste Element dieser 2-Tupel ist eine Security Environment Nummer gemäß 1.6 und das zweite Element jedes 2-Tupels ist ein n-Tupel mit einem oder mehreren *algorithmIdentifier* gemäß 6.2.7

An der Schnittstelle des Wrappers gilt für das Attribut *listAlgorithmIdentifier* dieselbe Darstellung wie in der Objektsystemspezifikation.

Tabelle 46: Beispiele zur Codierung von listAlgorithmIdentifier

listAlgorithmIdentifier	XML-Darstellung

SE#2 → rsaDecipherOaep	`<attribute id="listAlgorithmIdentifier">`
	`  {[2, (rsaDecipherOaep)]}`
	`</attribute>`

SE#1 → signPSS + sign9796_2_DS2,
SE#3 → signPKCS1_V1_5
	`<attribute id="listAlgorithmIdentifier">`
	`  {`
	`  [1, (signPSS, sign9796_2_DS2)],`
	`  [3, (signPKCS1_V1_5)]`
	`  }`
	`</attribute>`

SE#1 → rsaDecipherPKCS1_V1_5,
alle anderen SE → rsaDecipherOaep
	<attribute id="listAlgorithmIdentifier">
	{
	[1, (rsaDecipherPKCS1_V1_5)],
	[??, (rsaDecipherOaep)]
	}
	</attribute>

6.2.35 XML-Darstellung von listOfApplication

Im Rahmen einer Objektsystemdarstellung wird ein Attribut *listOfApplication* nicht verwendet, weil es sich implizit aus den übrigen Objekten des Objektsystems ergibt.

An der Schnittstelle des Wrappers gilt für das Attribut *listOfApplication* folgende Darstellung:

```
<listOfApplication>
  <applicationIdentifier>HEX{1..∞}</applicationIdentifier>
  <applicationIdentifier>   ...   </applicationIdentifier>
</listOfApplication>
```

Die Länge null wird hier explizit ausgeschlossen, weil ein fehlendes Attribut *applicationIdentifier* nicht durch einen leeren Wert, sondern durch die Abwesenheit des Attributes anzuzeigen ist. Längen größer 16 sieht [gemSpec_COS] nicht vor, könnten aber vom COS unterstützt werden.

Anforderung „AID in *listOfApplication*“
Der Wrapper MUSS in *listOfApplication* von jedem im Prüfling vorhandenen Ordner, der sich per *applicationIdentifier* selektieren lässt (siehe [gemSpec_COS#(N10.300), (N010.700)]) mindestens einen *applicationIdentifier* einstellen.

6.2.36 XML-Darstellung von macKey

Für die Objektsystemspezifikation gilt: Falls das Attribut *encKey* zu personalisieren ist, dann fehlt es in der Objektsystemspezifikation. Falls das Attribut *encKey* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann wird es in Abhängigkeit vom Attribut *keyType* (siehe 6.2.30) wie folgt dargestellt:

Tabelle 47: XML-Darstellung macKey

***macKey* gemäß [gemSpec_COS#(N016.700)]**	**XML-Darstellung**
3TDES	<attribute id="macKey">HEX{24}</attribute>
AES-128	<attribute id="macKey">HEX{16}</attribute>
AES-192	<attribute id="macKey">HEX{24}</attribute>
AES-256	<attribute id="macKey">HEX{32}</attribute>

Für die Schnittstelle des Wrappers ist das Attribut *macKey* irrelevant.

6.2.37 XML-Darstellung von maximumNumberOfRecords

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *maximumNumberOfRecords* wie folgt dargestellt:

```
<attribute id="maximumNumberOfRecords">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *maximumNumberOfRecords* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.38 XML-Darstellung von maximumLength

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *maximumLength* wie folgt dargestellt:

```
<attribute id="maximumLength">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *maximumLength* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.39 XML-Darstellung von maximumRecordLength

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *maximumRecordLength* wie folgt dargestellt:

```
<attribute id="maximumRecordLength">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *maximumRecordLength* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.40 XML-Darstellung von minimumLength

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *minimumLength* wie folgt dargestellt:

```
<attribute id="minimumLength">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *minimumLength* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.41 XML-Darstellung von numberOfOctet

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *numberOfOctet* wie folgt dargestellt:

```
<attribute id="numberOfOctet">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *numberOfOctet* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.42 XML-Darstellung von numberScenario

Falls in einer Objektsystemspezifikation

1. das Attribut
   a. *algorithmIdentifier* den Wert aesSessionkey4SM besitzt, oder
   b. das Attribut *listAlgorithmIdentifer* den Wert elcAsynchronAdmin enthält, dann MUSS das Attribut *numberScenario* vorhanden sein,
2. sonst MUSS das Attribut *numberScenario* fehlen.

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *numberScenario* wie folgt dargestellt:

```
<attribute id="numberScenario">INTEGER</attribute>
```

Falls ein Authentisierungsobjekt einen der Algorithmen *aesSessionkey4SM* oder *elcAsynchronAdmin*

3. entweder ausschließlich, oder neben weiteren Algorithmen unterstützt, dann MUSS das Attribut *numberScenario* an der Schnittstelle des Wrappers vorhanden sein,
4. sonst DARF das Attribut *numberScenario* NICHT vorhanden sein.

An der Schnittstelle des Wrappers gilt für das Attribut *numberScenario* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.43 XML-Darstellung von oid

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *oid* wie folgt dargestellt:

Tabelle 48: XML-Darstellung algorithmIdentifier

Algorithmus	XML-Darstellung
authS_gemSpec-COS-G2_ecc-with-sha256 {1.3.36.3.5.3.1}	`<attribute id="oid"> authS_gemSpec-COS-G2_ecc-with-sha256 </attribute>`
authS_gemSpec-COS-G2_ecc-with-sha384 {1.3.36.3.5.3.2}	`<attribute id="oid"> authS_gemSpec-COS-G2_ecc-with-sha384 </attribute>`
authS_gemSpec-COS-G2_ecc-with-sha512 {1.3.36.3.5.3.3}	`<attribute id="oid"> authS_gemSpec-COS-G2_ecc-with-sha512 </attribute>`
authS_ISO9796-2Withrsa_sha256_mutual {1.3.36.3.5.2.4}	`<attribute id="oid"> authS_ISO9796-2Withrsa_sha256_mutual </attribute>`
ecdsa-with-SHA256 {1.2.840.10045.4.3.2}	`<attribute id="oid"> ecdsa-with-SHA256 </attribute>`
ecdsa-with-SHA384 {1.2.840.10045.4.3.3}	`<attribute id="oid"> ecdsa-with-SHA384 </attribute>`
ecdsa-with-SHA512 {1.2.840.10045.4.3.4}	`<attribute id="oid"> ecdsa-with-SHA512 </attribute>`
id-ELC-shared-secret-calculation	`<attribute id="oid"> id-ELC-shared-secret-calculation </attribute>`
id-RSAES-OAEP {1.2.840.113549.1.1.7}	`<attribute id="oid"> id-RSAES-OAEP </attribute>`
rsaEncryption {1.2.840.113549.1.1.1}	`<attribute id="oid"> rsaEncryption </attribute>`
sigS_ISO9796-2Withrsa_sha256 {1.3.36.3.4.2.2.4}	`<attribute id="oid"> sigS_ISO9796-2Withrsa_sha256 </attribute>`

6.2.44 XML-Darstellung von persistentPublicKeyList

Im Rahmen der Objektsystemspezifikation fehlt dieses Attribut. Öffentliche Schlüsselobjekte, die persistent zu speichern sind, werden als `child` im Attribut `children` eines Ordners abgelegt (siehe 6.2.13.1).

An der Schnittstelle des Wrappers gilt für das Attribut *persistentPublicKeyList* folgende Darstellung:

```
<attribute id="persistentPublicKeyList">
  <objectLocator>HEX</objectLocator>
  ...
</attribute>
```

Das Attribut *persistentPublicKeyList* enthält keinen, einen oder mehrere Knoten `objectLocator` und jeder Knoten `objectLocator` enthält eine DER TLV codierte Referenz auf ein öffentliches Schlüsselobjekt, welche unverändert als Parameter in der Methode `getInformation` (...) aus 3.1 verwendbar ist.

Das Attribut *persistentPublicKeyList* enthält für jedes persistent gespeicherte öffentliche Schlüsselobjekt einen Knoten `objectLocator`.

6.2.45 XML-Darstellung von passwordReference

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *passwordReference* wie folgt dargestellt:

```
<attribute id="passwordReference">HEX{1}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *passwordReference* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.46 XML-Darstellung von pointInTime

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *pointInTime* wie folgt dargestellt:

```
<attribute id="pointInTime">Date</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *pointInTime* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.47 XML-Darstellung von positionLogicalEndOfFile

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *positionLogicalEndOfFile* wie folgt dargestellt:

```
<attribute id="positionLogicalEndOfFile">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *positionLogicalEndOfFile* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.48 XML-Darstellung von privateElcKey

Für die Objektsystemspezifikation gilt: Falls das Attribut *privateElcKey* nicht festgelegt wird, etwa weil es zu personalisieren ist, dann gilt folgende Darstellung:

```
<attribute id="domainParameterOid">domainParameter</attribute>
<attribute id="privateKey">AttributeNotSet</attribute>
```

Falls das Attribut *privateElcKey* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann gilt die obige Darstellung.

```
<attribute id="domainParameterOid">domainParameter</attribute>
<attribute id="privateKey">INTEGER</attribute>
```

Für die Schnittstelle des Wrappers gilt: An der Schnittstelle des Wrappers DARF ein Knoten mit der `id="privateKey"` NICHT enthalten sein. Ansonsten MUSS an der Schnittstelle des Wrappers dieselbe Darstellung verwenden, wie in der Objektsystemspezifikation.

Mit den Definitionen aus [gemSpec_COS#8.2.2] werden die Domainparameter der elliptischen Kurve wie folgt dargestellt:

Tabelle 49: XML-Darstellung der Domainparameter

domainParameter	XML-Darstellung
ansix9p256r1	`<attribute id="domainParameterOid">ansix9p256r1</attribute>`
ansix9p384r1	`<attribute id="domainParameterOid">ansix9p384r1</attribute>`
brainpoolP256r1	`<attribute id="domainParameterOid">brainpoolP256r1</attribute>`
brainpoolP384r1	`<attribute id="domainParameterOid">brainpoolP384r1</attribute>`
brainpoolP512r1	`<attribute id="domainParameterOid">brainpoolP512r1</attribute>`

6.2.49 XML-Darstellung von privateRsaKey

Anders als in [gemSpec_COS#8.2.3.1], wird hier für die Daten des privaten Schlüssels eine Darstellung mit CRT-Parametern gewählt.

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *privateRsaKey* wie folgt dargestellt, wobei Längenangaben in Bit erfolgen:

Tabelle 50: Private RSA-Schlüssel in der Objektsystemdarstellung

Parameter	XML-Darstellung	Erläuterung

Moduluslänge / [bit]	`<attribute id="modulusLength">`
	`  INTEGER`
	`</attribute>`
		gemäß [gemSpec_COS#(N002.100)a] entweder 2048 oder 3072

Modulus n	`<attribute id="n">INTEGER</attribute>`	$n = p\,q$
Öffentlicher Exponent e	`<attribute id="e">INTEGER</attribute>`	gcd(e, $(p-1)(q-1)$) = 1
Geheimer Exponent d	`<attribute id="d">INTEGER</attribute>`	$e\,d \equiv 1 \bmod \mathrm{lcm}(p-1, q-1)$
Primzahl p	`<attribute id="p">INTEGER</attribute>`	beliebige Primzahl
Primzahl q	`<attribute id="q">INTEGER</attribute>`	beliebige Primzahl
CRT Exponent zu p	`<attribute id="dP">INTEGER</attribute>`	$d_P = d \bmod (p-1)$
CRT Exponent zu q	`<attribute id="dQ">INTEGER</attribute>`	$d_Q = d \bmod (q-1)$
c	`<attribute id="c">INTEGER</attribute>`	$c = q^{-1} \bmod p$

Für die Objektsystemspezifikation gilt: Falls das Attribut *privateRsaKey* nicht festgelegt wird, etwa weil es zu personalisieren ist, dann wird als Wert für `n`, `e`, `d`, `p`, `q`, `dp`, `dQ` und `c` die Zeichenkette `AttributeNotSet` verwendet.

Für die Schnittstelle des Wrappers gilt: An der Schnittstelle des Wrappers DÜRFEN Angaben zum privaten Schlüssel NICHT enthalten sein. Daraus ergibt sich für die Schnittstelle des Wrappers folgende Darstellung für *privateRsaKey*:

Tabelle 51: Private RSA-Schlüssel an der Schnittstelle des Wrappers

Parameter	XML-Darstellung	Erläuterung

Moduluslänge / [bit]	<attribute id="modulusLength"> INTEGER </attribute>	gemäß [gemSpec_COS#(N002.100)a] entweder 2048 oder 3072

6.2.50 XML-Darstellung von publicElcKey

Für die Objektsystemspezifikation gilt: Falls das Attribut *publicElcKey* nicht festgelegt wird, etwa weil es zu personalisieren ist, dann gilt folgende Darstellung:

```
<attribute id="domainParameterOid">domainParameter</attribute>
<attribute id=P">AttributeNotSet</attribute>
```

Falls das Attribut *publicElcKey* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann wird es wie folgt dargestellt.

```
<attribute id="domainParameterOid">domainParameter</attribute>
<attribute id=P">HEX</attribute>
```

wobei der Wert von P eine unkomprimierte Darstellung der Punktkoordinaten gemäß [gemSpec_COS#(N000.400)] ist.

Für die Schnittstelle des Wrappers gilt: An der Schnittstelle des Wrappers wird dieselbe Darstellung verwendet, wie in der Objektsystemspezifikation.

6.2.51 XML-Darstellung von publicRsaKey

Für die Objektsystemspezifikation gilt: Falls das Attribut *publicRsaKey* nicht festgelegt wird, etwa weil es zu personalisieren ist, dann gilt folgende Darstellung:

```
<attribute id="modulusLength">INTEGER</attribute>
<attribute id="n">AttributeNotSet</attribute>
<attribute id="e">AttributeNotSet</attribute>
```

Falls das Attribut *publicRsaKey* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann wird es wie folgt dargestellt.

```
<attribute id="modulusLength">INTEGER</attribute>
<attribute id=”n”>INTEGER</attribute>
<attribute id=”e”>INTEGER</attribute>
```

Für die Schnittstelle des Wrappers gilt: An der Schnittstelle des Wrappers wird dieselbe Darstellung verwendet, wie in der Objektsystemspezifikation.

6.2.52 XML-Darstellung von PUK

Für die Objektsystemspezifikation gilt: Falls das Attribut *PUK* zu personalisieren ist, dann fehlt es in der Objektsystemspezifikation. Falls das Attribut *PUK* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann wird es wie folgt dargestellt:

```
<attribute id="PUK">HEX{8}</attribute>
```

Dabei muss der Oktettstring ein gültiger Format-2-PIN-Block gemäß [gemSpec_COS#(N008.100)] sein.

Für die Schnittstelle des Wrappers ist das Attribut *PUK* irrelevant.

6.2.53 XML-Darstellung von pukUsage

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *pukUsage* wie folgt dargestellt:

```
<attribute id="pukUsage">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *pukUsage* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.54 XML-Darstellung von pwdIdentifier

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *pwdIdentifier* wie folgt dargestellt:

```
<attribute id="pwdIdentifier">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *pwdIdentifier* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.55 XML-Darstellung von recordList

Im Rahmen der Objektsystemspezifikation gibt es für strukturierte Dateien einen Abschnitt mit der Liste der enthaltenen Rekords. Diese Liste ist stets vorhanden. Eventuell ist diese Liste leer:

```
<records>
  ...
</records>
```

Das Attribut *recordList* enthält keinen, einen oder mehrere Knoten <record> … </record> gemäß 6.3.

An der Schnittstelle des Wrappers fehlt diese Liste.

6.2.56 XML-Darstellung von retryCounter

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *retryCounter* wie folgt dargestellt:

```
<attribute id="retryCounter">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *retryCounter* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.57 XML-Darstellung von root

Im Rahmen der Objektsystemspezifikation wird ein Attribut *root* wie folgt dargestellt, wobei die Applikation gemäß 6.3 dargestellt wird:

```
<attribute id="root">
  <child> MF </child>
</attribute>
```

wobei MF die XML-Darstellung der Applikation ist, die an der Wurzel der Baumstruktur steht.

An der Schnittstelle des Wrappers wird ein Attribut *root* wie folgt dargestellt, wobei ein beliebiger *applicationIdentifier* des Wurzelverzeichnisses gemäß 6.2.8 zurückgemeldet wird:

```
<attribute id="root">HEX{1.. ∞}</attribute>
```

Die Länge null wird hier explizit ausgeschlossen, weil ein leeres Attribut *applicationIdentifier* nicht zulässig ist. Längen größer 16 sieht [gemSpec_COS] nicht vor, könnten aber vom COS unterstützt werden.

6.2.58 XML-Darstellung von secret

Für die Objektsystemspezifikation gilt: Falls das Attribut *secret* zu personalisieren ist, dann fehlt es in der Objektsystemspezifikation. Falls das Attribut *secret* in der Objektsystemspezifikation auf einen konkreten Wert festgelegt wird, dann wird es wie folgt dargestellt:

```
<attribute id="secret">HEX{8}</attribute>
```

Dabei muss der Oktettstring ein gültiger Format-2-PIN-Block gemäß [gemSpec_COS#(N008.100)] sein.

Für die Schnittstelle des Wrappers ist das Attribut *secret* irrelevant.

6.2.59 XML-Darstellung von shareable

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *shareable* wie folgt dargestellt:

Tabelle 52: XML-Darstellung shareable

[gemSpec_COS#(N009.850)], [gemSpec_COS#(N011.050)]	XML-Darstellung
True	`<attribute id="shareable">TRUE</attribute>`
False	`<attribute id="shareable">FALSE</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *shareable* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.60 XML-Darstellung von shortFileIdentifier

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *shortFileIdentifier* wie folgt dargestellt:

```
<attribute id="shortFileIdentifier">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *shortFileIdentifier* dieselbe Darstellung wie in der Objektsystemspezifikation.

Ein fehlendes Attribut *shortFileIdentifier* wird sowohl in einer Objektsystemspezifikation als auch an der Schnittstelle des Wrappers nicht durch einen leeren Wert, sondern durch die Abwesenheit des Attributes angezeigt.

Hinweis (9): Gemäß [gemSpec_COS#(N007.000)] nimmt ein shortFileIdentifier *ganzzahlige Werte aus Intervall [1, 30] an. Gemäß [gemSpec_COS#(N014.400)] wird ein* shortFileIdentifier *innerhalb der File Control Parameter FCP im Wertfeld eines DO´88´ in den oberen fünf Bit codiert.*

Tabelle 53: Beispiele zur Codierung eines shortFileIdentifier

shortFileIdentifier	DO´88´ in FCP	XML-Darstellung
1 = ´01´	´88 01 08´	`<attribute id="shortFileIdentifier">01</attribute>`
2 = ´02´	´88 01 10´	`<attribute id="shortFileIdentifier">02</attribute>`
30 = ´1E´	´88 01 F0´	`<attribute id="shortFileIdentifier">1E</attribute>`

6.2.61 XML-Darstellung von startRetryCounter

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *startRetryCounter* wie folgt dargestellt:

```
<attribute id="startRetryCounter">INTEGER</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *startRetryCounter* dieselbe Darstellung wie in der Objektsystemspezifikation.

6.2.62 XML-Darstellung von startSsecList

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *startSsecList* wie folgt dargestellt:

```
<attribute id="startSSecList"> .. </attribute>
```

Das Attribut *startSsecList* enthält eine Menge von einem oder mehreren 2-Tupeln. Der erste Wert jedes 2-Tupels ist eine Security Environment Nummer gemäß 1.6 und der zweite Wert jedes 2-Tupels ist eine ganze Zahl gemäß INTEGER oder der Wert INFINITY.

An der Schnittstelle des Wrappers gilt für das Attribut *startSsecList* dieselbe Darstellung wie in der Objektsystemspezifikation.

Tabelle 54: Beispiele zur Codierung von startSsecList

startSsecList	XML-Darstellung

SE#2 → 1	`<attribute id="startSSecList">`
	`  {(2, 01)}`
	`</attribute>`

SE#1 → ∞, SE#3 → 30	`<attribute id="startSSecList">`
	`  {(1, INFINITY), (3, 1E)}`
	`</attribute>`

SE#1 → 6, SE#2 → 5, SE#3 → 250	`<attribute id="startSSecList">`
	`  {(1, 06), (2, 05), (3, 00FA)}`
	`</attribute>`

SE#1 → ∞, alle anderen SE → 4	`<attribute id="startSSecList">`
	`  {(1, INFINITY), (??, 04)}`
	`</attribute>`

6.2.63 XML-Darstellung von transportStatus

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *transportStatus* wie folgt dargestellt:

Tabelle 55: XML-Darstellung transportStatus

[gemSpec_COS#(N009.600)a]	XML-Darstellung
regularPassword	`<attribute id="transportStatus">` `  regularPassword`

	`</attribute>`

Leer-PIN	`<attribute id="transportStatus">`
	`  leerPIN`
	`</attribute>`

Transport-PIN	`<attribute id="transportStatus">`
	`  transportPIN`
	`</attribute>`

An der Schnittstelle des Wrappers gilt für das Attribut *transportStatus* dieselbe Darstellung wie in der Objektsystemspezifikation. Falls das COS weitere Status unterstützt (siehe [gemSpec_COS#(N009.600)b], ist es möglich, dass das Attribut für herstellerspezifische Objekte Werte annimmt, die in der obigen Tabelle nicht genannt sind.

6.2.64 XML-Darstellung von warmAnswerToReset

Im Rahmen einer Objektsystemspezifikation wird ein Attribut *warmAnswerToReset* wie folgt dargestellt:

```
<attribute id="warmAnswerToReset">HEX{9..33}</attribute>
```

An der Schnittstelle des Wrappers gilt für das Attribut *warmAnswerToReset* dieselbe Darstellung wie in der Objektsystemspezifikation.

Hinweis (10): Die Längenbeschränkungen von warmAnswerToReset werden in 6.2.14 erläutert.

6.3 Generische Darstellung von Klassen

Dieser Abschnitt beschreibt die Klassen generisch. Die hier beschriebenen Knoten enthalten in der Regel weitere Knoten, welche das sind ergibt sich mittels der in der Tabelle angegebenen Referenz.

Tabelle 56: Tags von Klassen

Klasse	objectType
Rekord gemäß [gemSpec_COS#8.1.5]	siehe 6.4.1
Applikation gemäß [gemSpec_COS#8.3.1.1]	`Application`
DF gemäß [gemSpec_COS#8.3.1.2]	`DF`
ADF gemäß [gemSpec_COS#8.3.1.3]	`ADF`
Transparentes Elementary File	`TransparentElementaryFile`
Linear variables Elementary File	`LinearVariableElementaryFile`
Linear fixes Elementary File	`LinearFixedElementaryFile`
Zyklisches Elementary File	`CyclicElementaryFile`
Reguläres Passwort	`PWD`
Multireferenz-Passwort	`MultireferencePWD`
Symmetrisches Authentisierungsobjekt	`SymmetricAutKey`
Symmetrisches Kartenverbindungsobjekt	`SymmetricConnectionKey`
Privates ELC-Schlüsselobjekt	`PrivateElcKey`
Privates RSA-Schlüsselobjekt	`PrivateRsaKey`

Öffentliches ELC-Signaturprüfobjekt	PublicElcVerificationKey
Öffentliches RSA-Signaturprüfobjekt	PublicRsaVerificationKey
Öffentliches ELC-Authentisierungsobjekt	PublicElcAutKey
Öffentliches RSA-Authentisierungsobjekt	PublicRsaAutKey
Öffentliches ELC-Verschlüsselungsobjekt	PublicElcEncKey
Öffentliches RSA-Verschlüsselungsobjekt	PublicRsaEncKey
Objektsystem	siehe 6.4.5

Generisch betrachtet werden Klassen in XML wie folgt dargestellt, wobei der Wert von `objectType` durch einen Wert aus Tabelle 56 zu ersetzen ist:

```
<child   id="..."   objectType="...">
    <attributes>
        ...
    </attributes>
    <ManufacturerSpecificExtension>
        ...
    </ManufacturerSpecificExtension>
</child>
```

Das Attribut `id` hilft eine Verbindung zwischen diesem Objekt und der Bezeichnung des Objektes im allgemeinen Sprachgebrauch herzustellen.

Im Rahmen der Objektsystemspezifikation MUSS das Attribut `id` vorhanden sein.

An der Schnittstelle des Wrappers SOLL das Attribut `id` fehlen.

Die obligatorischen XML-Knoten in `child.attributes` ergeben sich für jeden Objekttyp aus Kapitel 4. Herstellerspezifische Attribute liefert der Wrapper in einem zusätzlichen Knoten, dessen Inhalt herstellerspezifisch codiert ist. So ein Knoten fehlt, falls keine herstellerspezifischen Attribute über den Wrapper transportiert werden.

Es ist möglich, dass ein herstellerspezifisches symmetrisches Authentisierungsobjekt mehr als einen Algorithmus unterstützt und/oder der unterstützte Algorithmus SE abhängig ist. In diesem Fall MUSS auch für symmetrische Authentisierungsobjekte an der Schnittstelle des Wrappers die Darstellung analog zu 6.2.34 gewählt werden.

6.4 Beispiele

6.4.1 Rekord gemäß [gemSpec_COS#8.1.5]

```
<record   id="Explanation of Record content, hopefully this is meaningful to a user">
    <attribute id="recordValue">0040051234</attribute>
    <attribute id="recordLifeCycleStatus">ACTIVATED</attribute>
</record>
```

6.4.2 Transparentes Elementary File

Die folgende Struktur zeigt ein transparentes EF, wie es in einer Objektsystemspezifikation vorhanden sein könnte.

```
<child id="EF.C.CA_eGK.CS.E256" objectType="TransparentElementaryFile">
  <attributes>
    <attribute id="fileIdentifier">2f07</attribute>
    <attribute id="shortFileIdentifier">07</attribute>
    <attribute id="lifeCycleStatus">ACTIVATED</attribute>
    <attribute id="shareable">TRUE</attribute>
    <attribute id="accessRules">
    CB{                                <!-- begin CB AR-->
        [                             <!-- begin LCS_SE_AR -->
            ACTIVATED,                     <!-- LCS -->
            ??,                       <!-- all SE -->
            {                         <!-- begin List AR -->
                ({00||b0||??||??}, ALW),    <!-- Read Binary ALWAYS -->
                (                     <!-- begin 2nd elem. AR -->
                    {                 <!-- begin accessMode -->
                        00||e4||??||??,    <!-- Delete -->
                        00||d6||??||??,    <!-- Update Binary -->
                    },                 <!-- end accessMode -->
                    AND{                 <!-- begin accessCon. -->
                        OR{             <!-- begin OR -->
                            AUT(13),    <!-- SK.CMS.AES128 -->
                            AUT(18)    <!-- SK.CMS.AES256 -->
                        },             <!-- end OR -->
                        SMCMDENC,     <!-- SmCmdEnc -->
                        SMRSPENC        <!-- SmRspEnc -->
                    }                 <!-- end accessCon. -->
                )                     <!-- end 2nd elem. AR -->
            }                         <!-- end List AR -->
        ]                             <!-- end LCS_SE_AR -->
    },                                 <!-- end CB AR-->
    CL{                                <!-- begin CL AR-->
        [                             <!-- begin LCS_SE_AR -->
            ACTIVATED,                     <!-- LCS -->
            ??,                       <!-- all SE -->
            {                         <!-- begin List AR -->
                (                     <!-- begin 1st elem. AR -->
                    {00||b0||??||??},    <!-- Read Binary -->
                    AND{                 <!-- begin accessCon. -->
                        SMMAC(02),    <!-- SmMac(SK.CAN) -->
                        SMRSPENC        <!-- SmRspEnc -->
                    }                 <!-- end accessCon. -->
```

```
                ),                      <!-- end 1st elem. AR -->
                (                       <!-- begin 2nd elem. AR -->
                    {                   <!-- begin accessMode -->
                        00||e4||??||??,     <!-- Delete -->
                        00||d6||??||??,     <!-- Update Binary -->
                    },                  <!-- end accessMode -->
                    AND{                  <!-- begin accessCon. -->
                        OR{               <!-- begin OR -->
                            AUT(13),      <!-- SK.CMS.AES128 -->
                            AUT(18)       <!-- SK.CMS.AES256 -->
                        },                <!-- end OR -->
                        SMCMDENC,       <!-- SmCmdEnc -->
                        SMRSPENC          <!-- SmRspEnc -->
                    }                   <!-- end accessCon. -->
                )                       <!-- end 2nd elem. AR -->
            }                           <!-- end List AR -->
        ]                               <!-- end LCS_SE_AR -->
    }                                   <!-- end CL AR-->
    </attribute>
    <attribute id="flagTransactionMode">FALSE</attribute>
    <attribute id="flagChecksum">FALSE</attribute>
    <attribute id="numberOfOctet">00dc</attribute>
    <attribute id="positionLogicalEndOfFile">00dc</attribute>
    <attribute id="body">Wildcard</attribute>
  </attributes>
</child>
```

6.4.3 Linear fixes Elementary File

Die folgende Struktur zeigt ein linear fixes EF, wie es in einer Objektsystemspezifikation vorhanden sein könnte.

```
<child id="EF.Version" objectType="LinearFixedElementaryFile">
  <attributes>
    <attribute id="fileIdentifier">2f10</attribute>
    <attribute id="shortFileIdentifier">10</attribute>
    <attribute id="lifeCycleStatus">ACTIVATED</attribute>
    <attribute id="shareable">TRUE</attribute>
    <attribute id="accessRules">
    CB{                                 <!-- begin CB AR-->
        [                               <!-- begin LCS_SE_AR -->
            ACTIVATED,                      <!-- LCS -->
            ??,                         <!-- all SE -->
            {                           <!-- begin List AR -->
                (                       <!-- begin 1st elem. AR -->
                    {                   <!-- begin accessMode -->
```

```
                    00||b2||??||??,    <!-- Read Record -->
                    00||a2||??||??,    <!-- Search Record -->
                },                  <!-- end accessMode -->
                ALW                  <!-- ALWAYS -->
            ),                      <!-- end 1st elem. AR -->
            (                      <!-- begin 2nd elem. AR -->
                {00||dc||??||??},    <!-- Update Record -->
                AND{                  <!-- begin accessCon. -->
                    OR{              <!-- begin OR -->
                        AUT(13),     <!-- SK.CMS.AES128 -->
                        AUT(18)     <!-- SK.CMS.AES256 -->
                    },               <!-- end OR -->
                    SMCMDENC,     <!-- SmCmdEnc -->
                    SMRSPENC        <!-- SmRspEnc -->
                }                  <!-- end accessCon. -->
            )                      <!-- end 2nd elem. AR -->
        }                          <!-- end List AR -->
    ]                              <!-- end LCS_SE_AR -->
},                                  <!-- end CB AR-->
CL{                                  <!-- begin CL AR-->
    [                              <!-- begin LCS_SE_AR -->
        ACTIVATED,                      <!-- LCS -->
        ??,                          <!-- all SE -->
        {                          <!-- begin List AR -->
            (                      <!-- begin 1st elem. AR -->
                {                  <!-- begin accessMode -->
                    00||b2||??||??,    <!-- Read Record -->
                    00||a2||??||??,    <!-- Search Record -->
                },                  <!-- end accessMode -->
                ALW                  <!-- ALWAYS -->
            ),                      <!-- end 1st elem. AR -->
            (                      <!-- begin 2nd elem. AR -->
                {00||dc||??||??},    <!-- Update Record -->
                AND{                  <!-- begin accessCon. -->
                    OR{              <!-- begin OR -->
                        AUT(13),     <!-- SK.CMS.AES128 -->
                        AUT(18)     <!-- SK.CMS.AES256 -->
                    },               <!-- end OR -->
                    SMCMDENC,     <!-- SmCmdEnc -->
                    SMRSPENC        <!-- SmRspEnc -->
                }                  <!-- end accessCon. -->
            )                      <!-- end 2nd elem. AR -->
        }                          <!-- end List AR -->
    ]                              <!-- end LCS_SE_AR -->
}                                  <!-- end CL AR-->
```

```
    </attribute>
    <attribute id="flagTransactionMode">TRUE</attribute>
    <attribute id="flagChecksum">TRUE</attribute>
    <attribute id="maximumNumberOfRecords">04</attribute>
    <attribute id="maximumRecordLength">05</attribute>
    <attribute id="flagRecordLifeCycleStatus>FALSE</attribute>
  </attributes>
  <records>
    <record   id="Explanation of Record 1: COS version information">
        <attribute id="recordValue">0010020003</attribute>    <!-- V1.2.3 -->
        <attribute id="recordLifeCycleStatus">ACTIVATED</attribute>
    </record>
    <record   id="Explanation of Record 2: ObjSys version information">
        <attribute id="recordValue">0100110012</attribute>    <!-- V10.11.12 -->
        <attribute id="recordLifeCycleStatus">ACTIVATED</attribute>
    </record>
    <record   id="Explanation of Record 3: Content version information">
        <attribute id="recordValue">9999989997</attribute>    <!-- V999.998.9997 -->
        <attribute id="recordLifeCycleStatus">ACTIVATED</attribute>
    </record>
    <record   id="Explanation of Record 4: reserved for future use">
        <attribute id="recordValue">0000000000</attribute>    <!-- no information -->
        <attribute id="recordLifeCycleStatus">ACTIVATED</attribute>
    </record>
  </records>
</child>
```

6.4.4 Privates RSA-Schlüsselobjekt

In diesem Kapitel werden drei Beispiele für die XML-Darstellung von privaten RSA-Schlüsselobjekten dargestellt.

6.4.4.1 Privates RSA-Schlüsselobjekt, Objektsystemspezifikation, keine Schlüsseldaten

Hinweis (11): Für dieses Schlüsselobjekt sind keine Schlüsseldaten vorhanden. Nach der "Personalisierung mittels Generate Asymmetric Key Pair Kommando lässt sich der Schlüssel aktivieren und nutzen.

```
<child id="PrK.RSA2048" objectType="PrivateRsaKey">
  <attributes>
    <attribute id="keyIdentifier">02</attribute>
    <attribute id="lifeCycleStatus">DEACTIVATED</attribute>
    <attribute id="accessRules">
    CB{                                <!-- begin CB AR-->
        [                             <!-- begin LCS_SE_AR -->
            ACTIVATED,                     <!-- LCS -->
            ??,                        <!-- all SE -->
```

```
            {                       <!-- begin List AR -->
                ({00||2a||80||86}, ALW)    <!-- PSO Decipher ALWAYS -->
            }                       <!-- end List AR -->
        ],                           <!-- end LCS_SE_AR -->
        [                           <!-- begin LCS_SE_AR -->
            DEACTIVATED,               <!-- LCS -->
            ??,                       <!-- all SE -->
            {                       <!-- begin List AR -->
                ({00||46||??||??}, ALW),    <!-- GenAsymKeyPair ALWAYS -->
                ({00||44||??||??}, ALW),    <!-- Activate ALWAYS -->
            }                       <!-- end List AR -->
        ]                           <!-- end LCS_SE_AR -->
    }                               <!-- end CB AR-->
    </attribute>
    <attribute id="modulusLength">0800</attribute> <!-- 0x0800 = 2048 -->
    <attribute id="n">AttributeNotSet</attribute>
    <attribute id="e">AttributeNotSet</attribute>
    <attribute id="d">AttributeNotSet</attribute>
    <attribute id="p">AttributeNotSet</attribute>
    <attribute id="q">AttributeNotSet</attribute>
    <attribute id="dP">AttributeNotSet</attribute>
    <attribute id="dQ">AttributeNotSet</attribute>
    <attribute id="c">AttributeNotSet</attribute>
    <attribute id="listAlgorithmIdentifier">{[2, (rsaDecipherOaep)]}</attribute>
    <attribute id="keyAvailable">FALSE</attribute>
  </attributes>
</child>
```

6.4.4.2 Privates RSA-Schlüsselobjekt, Objektsystemspezifikation, mit Schlüsseldaten

Hinweis (12): Im Gegensatz zu Festlegungen in [gemSpec_COS#(N002.100)a] wird hier eine sehr kurze Moduluslänge gewählt, damit die Darstellung übersichtlich bleibt.

```
<child id="PrK.RSA2048" objectType="PrivateRsaKey">
  <attributes>
    <attribute id="keyIdentifier">02</attribute>
    <attribute id="lifeCycleStatus">DEACTIVATED</attribute>
    <attribute id="accessRules">
    CB{                             <!-- begin CB AR-->
        [                           <!-- begin LCS_SE_AR -->
            ACTIVATED,                 <!-- LCS -->
            ??,                       <!-- all SE -->
            {                       <!-- begin List AR -->
                ({00||2a||80||86}, ALW)    <!-- PSO Decipher ALWAYS -->
            }                       <!-- end List AR -->
        ],                           <!-- end LCS_SE_AR -->
```

```
        [                           <!-- begin LCS_SE_AR -->
            DEACTIVATED,               <!-- LCS -->
            ??,                      <!-- all SE -->
            {                       <!-- begin List AR -->
                ({00||46||??||??}, ALW),    <!-- GenAsymKeyPair ALWAYS -->
                ({00||44||??||??}, ALW),    <!-- Activate ALWAYS -->
            }                       <!-- end List AR -->
        ]                           <!-- end LCS_SE_AR -->
    }                               <!-- end CB AR-->
    </attribute>
    <attribute id="modulusLength">40</attribute> <!—0x40 = 64 -->
    <attribute id="n">8629a6fa5c3aab47</attribute>
    <attribute id="e">0101</attribute>
    <attribute id="d">4d42d1b61adb5121</attribute>
    <attribute id="p">0122ae079b</attribute>
    <attribute id="q">762803c5</attribute>
    <attribute id="dP">4199d3e5</attribute>
    <attribute id="dQ">64af8bad</attribute>
    <attribute id="c">00aa3bb8b2</attribute>
    <attribute id="listAlgorithmIdentifier">{[2, (rsaDecipherOaep)]}</attribute>
    <attribute id="keyAvailable">FALSE</attribute>
  </attributes>
</child>
```

6.4.4.3 Privates RSA-Schlüsselobjekt, Schnittstelle des Wrappers

```
<child objectType="PrivateRsaKey">
  <attributes>
    <attribute id="keyIdentifier">02</attribute>
    <attribute id="lifeCycleStatus">ACTIVATED</attribute>
    <attribute id="accessRules">
    CB{[ACTIVATED, ??, {({00||2a||80||86, 00||2a||86||B8}, ALW)}]}
    </attribute>
    <attribute id="modulusLength">0c00</attribute>
    <attribute id="listAlgorithmIdentifier">{[2, (rsaDecipherOaep)]}</attribute>
    <attribute id="keyAvailable">TRUE</attribute>
  </attributes>
</child>
```

6.4.5 Objektsystem

Die XML-Darstellung eines Objektsystems wird in einen `card` Knoten eingebettet, der ein Attribut `version` enthält. Dieses Attribut `version` kennzeichnet die Version der verwendeten XML-Darstellung. Für diese Version der XML-Spezifikation gilt:

```
    version="2"
```

In diesem Unterkapitel werden zwei Beispiele für die XML-Darstellung eines Objektsystems beschrieben.

6.4.5.1 Objektsystem in einer Objektsystemspezifikation

Dargestellt ist hier ein Beispiel, wie es in Testlaborkarten Verwendung finden könnte mit einer vollständigen Angabe aller Attribute. Im Rahmen von Objektsystemspezifikationen für etwa eGK oder HBA werden demgegenüber diverse Attribute nicht vorgegeben sondern erst im Rahmen einer Personalisierung belegt. Zudem ist hier das Masterfile MF nur auszugsweise dargestellt.

```
<card   version="2">
<objectSystem>
  <attribute id="root">
    <child   id="This is the MF"   objectType="Application">
      <attribute id="accessRules">...</attribute>
      <attribute id="applicationIdentifier">{d2760001448000}</attribute>
      <attribute id="lifeCycleStatus">ACTIVATED</attribute>
      <attribute id="shareable">TRUE</attribute>
      <children>   <!-- begin children MF -->
        <child   id="EF.ATR"   objectType="TransparentElementaryFile">
            ...
        </child>   <!-- end EF.ATR -->
        ...
        <child   id=DF.HCA"   objectType="Application">
          <attribute id="accessRules">...</attribute>
          <attribute id="applicationIdentifier">{d27600000102}</attribute>
          <attribute id="lifeCycleStatus">ACTIVATED</attribute>
          <attribute id="shareable">TRUE</attribute>
          <children>   <!-- begin children DF.HCA -->
            <child   id="EF.GVD"   objectType="TransparentElementaryFile">
                ...
            </child>   <!-- end EF.GVD -->
            ...
            <child   id=DF.NFD"   objectType="Application">
                ...
            </child>   <!-- end DF.NFD -->
          </children>   <!-- end children DF.HCA -->
        </child>   <!-- end child DF.HCA -->
      </children>   <!-- end children MF -->
    </child>   <!-- end MF -->
  </attribute>
  <attribute id="coldAnswerToReset">
    3bdd97ff81b1fe451f03006404050803739621d0009000c8
  </attribute>
  <attribute id="warmAnswerToReset">
    3bdd96ff81b1fe451f03006404050803739621d0009000c9
```

```
  </attribute>
  <attribute id="iccsn8">1122334455667788</attribute>
  <attribute id="lifeCycleStatus">ACTIVATED</attribute>
  <attribute id="pointInTime">010400050203</attribute>
</objectSystem>
</card>
```

6.4.5.2 Objektsystem an der Schnittstelle des Wrappers

```
<card   version="2">
  <objectSystem>
    <attribute id="root">d2760001448000</attribute>
    <attribute id="coldAnswerToReset">
        3bdd97ff81b1fe451f03006404050803739621d0009000c8
    </attribute>
    <attribute id="warmAnswerToReset">
        3bdd96ff81b1fe451f03006404050803739621d0009000c9
    </attribute>
    <attribute id="iccsn8">1122334455667788</attribute>
    <attribute id="persistentPublicKeyList">
        <objectLocator>
            e0174f06f123456789abb60d9501808308112233445566778 8
        </objectLocator>
        <objectLocator>
            e0174f06f123456789abb60d950180830888776655443322 11
        </objectLocator>
    </attribute>
    <attribute id="lifeCycleStatus">ACTIVATED</attribute>
    <attribute id="pointInTime">010400050203</attribute>
    <listOfApplication>
        <applicationIdentifier>d2760001448000</applicationIdentifier>
        <applicationIdentifier>d27600014407</applicationIdentifier>
    </listOfApplication>
  </objectSystem>
</card>
```

7 - Verzeichnisse

7.1 Abkürzungen

Kürzel	Erläuterung
APDU	Application Protocol Data Unit

7.2 Glossar

Begriff	Erläuterung
Funktionsmerkmal	Der Begriff beschreibt eine Funktion oder auch einzelne, eine logische Einheit bildende Teilfunktionen der TI im Rahmen der funktionalen Zerlegung des Systems.

Das Glossar wird als eigenständiges Dokument, vgl. [gemGlossar] zur Verfügung gestellt.

7.3 Abbildungsverzeichnis

7.4 Tabellenverzeichnis

7.5 Referenzierte Dokumente

7.5.1 Dokumente der gematik

Die nachfolgende Tabelle enthält die Bezeichnung der in dem vorliegenden Dokument referenzierten Dokumente der gematik zur Telematikinfrastruktur. Der mit der vorliegenden Version korrelierende Entwicklungsstand dieser Konzepte und Spezifikationen wird pro Release in einer Dokumentenlandkarte definiert, Version und Stand der referenzierten Dokumente sind daher in der nachfolgenden Tabelle nicht aufgeführt. Deren zu diesem Dokument passende jeweils gültige Versionsnummer sind in der aktuellsten, von der gematik veröffentlichten Dokumentenlandkarte enthalten, in der die vorliegende Version aufgeführt wird.

[Quelle]	Herausgeber: Titel
[gemGlossar]	gematik: Glossar der Telematikinfrastruktur
[gemSpec_COS]	gematik: Spezifikation des Card Operating System (COS), Elektrische Schnittstelle

7.5.2 Weitere Dokumente

[Quelle]	Herausgeber (Erscheinungsdatum): Titel
[ISO/IEC 7816-3]	Identification cards — Integrated circuit cards — Part 3: Cards with contacts Electrical interface and transmission protocols, third edition, 2006-11-01
[ISO/IEC 7816–4]	Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange, third edition, 2013
[ISO/IEC 8825–1]	Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf
[Zertifizierungskonzept]	Zertifizierungskonzept für Karten der Generation G2, Version v1.0, 26.

	Juni 2013, Bundesamt für Sicherheit in der Informationstechnik (BSI)

8 - Sourcecode

Der in diesem Anhang gezeigten Sourcecode ist informativ. Normativ ist das von der gematik gemäß Anforderung „externe Abhängigkeit“ gelieferte JAR-Archiv (siehe 3.7).

8.1 ApduLayerException

```
package de.gematik.smartcard.g2.wrapper;



/**
* This class provides an exception handling for interface {@link IApduLayer}.
* <p>
* <b>History:</b>
* <ul>
*   <li>2013-07-18: first edition
* </ul>
*
* @version 0.1
* @author <a href="mailto:alfred.fiedler@gematik.de">Alfred Fiedler</a>
*/
public class ApduLayerException extends Exception {
  /** automatic generated number */
  private static final long serialVersionUID = 770749330558211284L;



  /**
   * This class provides a collection of reasons why there is no corresponding response
   * APDU for a command APDU sent via {@link IApduLayer#sendAPDU(byte[])}.
   * <p>
   * <b>History:</b>
   * <ul>
   *   <li>2013-07-18: first edition
   * </ul>
   *
   * @version 0.1
   * @author <a href="mailto:alfred.fiedler@gematik.de">Alfred Fiedler</a>
   */
  public enum EnumApduLayerException {
    /**
     * The connection to the ICC is broken, e.g. the card was removed from IFD.
     */
```

```
    BrokenConnection,

    /**
     * This reason is intended for all other error situations. Instead of using
     * this reason a new enum SHALL be added to the list of reasons on request.
     */
    Other;
  } // end EnumApduLayerException

  /** contains the reason why the exception is thrown */
  final private EnumApduLayerException m_Reason;

  /** additional explanation in case {@link EnumApduLayerException#Other} is used */
  final private String m_Explanation;

  /**
   * @param reason causing the exception
   */
  public ApduLayerException(final EnumApduLayerException reason) {
    m_Reason      = reason;
    m_Explanation = "";
  } // end constructor */

  /**
   * @param explanation describing why the exception is thrown
   */
  public ApduLayerException(final String explanation) {
    m_Reason      = EnumApduLayerException.Other;
    m_Explanation = explanation;
  } // end constructor */

  /* (non-Javadoc)
   * @see java.lang.Throwable#getMessage()
   */
  @Override
  public String getMessage() {
    String result = m_Reason.toString();

    if(EnumApduLayerException.Other == m_Reason) {
      result += ", " + m_Explanation;
    } // end if

    return result;
  } // end method */
} // end class
```

8.2 IApduLayer

```
package de.gematik.smartcard.g2.wrapper;


/**
* This interface specifies an ISO/IEC 7816-4 APDU layer.
* <p>
* <b>History:</b>
* <ul>
*   <li>2013-07-18: {@link ApduLayerException} added
*   <li>2013-04-10: first edition
* </ul>
*
* @version 0.2
* @author <a href="mailto:alfred.fiedler@gematik.de">Alfred Fiedler</a>
*/
public interface IApduLayer {
  /**
   * This function retrieves a response APDU corresponding to the given command APDU
   * by sending the command APDU to a communication channel.
   * <p>
   * This method SHALL either return a response APDU corresponding to ISO/IEC 7816-4
   * or throw an {@link ApduLayerException}.
   *
   * @param commandAPDU
   *        octet string representation of an ISO/IEC 7816-4 command APDU
   *
   * @return octet string representation of an ISO/IEC 7816-4 response APDU
   *         corresponding to the given command APDU
   *
   * @throws ApduLayerException in case a response APDU is not available
   */
  public byte[] sendAPDU(final byte[] commandAPDU) throws ApduLayerException;
} // end interface
```

8.3 IWrapper

```
package de.gematik.smartcard.g2.wrapper;
```

```
import org.w3c.dom.Node;


/**
* This interface specifies functions to be implemented by a wrapper.
* <p>
* <b>History:</b>
* <ul>
*   <li>2014-05-21: version 0.2
*     <ol>
*       <li>name of method changed
*       <li>method returns an array rather than a single {@link Node}
*     </ol>
*   <li>2014-03-07: first edition
* </ul>
*
* @version 0.2
* @author <a href="mailto:alfred.fiedler@gematik.de">Alfred Fiedler</a>
*/
public interface IWrapper {
  /**
   * This function retrieves information about the attributes of an object.
   * <p>
   *
   * @param apduLayer
   *        used for smart card communication. The caller of this
   *        method shall ensure that the connection is usable.
   *
   * @param objectLocator
   *        DER encoded octet string with an identifier referencing objects
   *        for which information is requested.
   *
   * @return {@link Node}[] with information about objects referenced by
   *         objectLocator. The XML structure of each element SHALL be in
   *         accordance to [gemSpec_COS-Wrapper].
   *
   * @throws ApduLayerException if this exception was thrown when
   *         using the method {@link IApduLayer#sendAPDU(byte[])}
   *
   * @throws WrapperException in case no proper information could
   *         be provided in the return value.
   */
  public Node[] getInformation(
      final IApduLayer apduLayer,
      final byte[] objectLocator
```

```
  ) throws ApduLayerException, WrapperException;


  /**
   * This function prepares the device under test such that
   * the command FINGERPRINT could be successfully performed.
   * <p>
   *
   * @param apduLayer
   *        used for smart card communication. The caller of this
   *        method shall ensure that the connection is usable.
   *
   * @return true if device under test is prepared for FINGERPRINT command,
   *         false otherwise
   *
   * @throws ApduLayerException if this exception was thrown when
   *         using the method {@link IApduLayer#sendAPDU(byte[])}
   */
  public boolean prepareFingerprint(
      final IApduLayer apduLayer
  ) throws ApduLayerException;
} // end interface
```

8.4 WrapperException

```
package de.gematik.smartcard.g2.wrapper;


import de.gematik.smartcard.g2.wrapper.ApduLayerException.EnumApduLayerException;




/**
* This class provides an exception handling for implementations of {@link IWrapper}.
* <p>
* <b>History:</b>
* <ul>
*   <li>2013-07-18: first edition
* </ul>
*
* @version 0.1
* @author <a href="mailto:alfred.fiedler@gematik.de">Alfred Fiedler</a>
*/
```

```
public class WrapperException extends Exception {
  /** automatic generated number */
  private static final long serialVersionUID = 7306540806842016482L;



  /**
   * This class provides a collection of reasons why there is no corresponding response
   * APDU for a command APDU sent via {@link IApduLayer#sendAPDU(byte[])}.
   * <p>
   * <b>History:</b>
   * <ul>
   *   <li>2013-10-10: replace item <code>NoInstanceAvailable</code> by
   *                   <code>{@link #NoWrapperInstanceAvailable}</code>
   *   <li>2013-10-10: item <code>{@link #ObjectNotFound}</code> added
   *   <li>2013-07-18: first edition
   * </ul>
   *
   * @version 0.2
   * @author <a href="mailto:alfred.fiedler@gematik.de">Alfred Fiedler</a>
   */
  public enum EnumWrapperException {
    /**
     * Object locator given in {@link IWrapper#getInformation(IApduLayer, byte[])}
     * is invalid due to
     * <ul>
     *   <li>byte array is not a valid TLV structure
     *   <li>TLV structure is in contradiction to specification
     *       (e.g. wrong tags, wrong tag sequence, etc.)
     * </ul>
     */
    InvalidObjectLocator,

    /**
     * An instance of a subclass of {@link IWrapper} could not be created.
     */
    NoWrapperInstanceAvailable,

    /**
     * An object pointed to by object locator could not be found in 'device under test'
     */
    ObjectNotFound,

    /**
     * This reason is intended for all other error situations. Instead of using
     * this reason a new enum SHALL be added to the list of reasons on request.
```

```
     */
    Other;
  } // end EnumWrapperException

  /** contains the reason why the exception is thrown */
  final private EnumWrapperException m_Reason;

  /** additional explanation in case {@link EnumApduLayerException#Other} is used */
  final private String m_Explanation;

  /**
   * @param reason causing the exception
   */
  public WrapperException(final EnumWrapperException reason) {
    m_Reason      = reason;
    m_Explanation = "";
  } // end constructor */

  /**
   * @param explanation describing why the exception is thrown
   */
  public WrapperException(final String explanation) {
    m_Reason      = EnumWrapperException.Other;
    m_Explanation = explanation;
  } // end constructor */

  /* (non-Javadoc)
   * @see java.lang.Throwable#getMessage()
   */
  @Override
  public String getMessage() {
    String result = m_Reason.toString();

    if(EnumWrapperException.Other == m_Reason) {
      result += ", " + m_Explanation;
    } // end if

    return result;
  } // end method */
} // end class
```