gematik
Gesellschaft für Telematikanwendungen der Gesundheitskarte mbH

Einführung der Gesundheitskarte

Übergreifende Spezifikation

Card Proxy

Version:	1.0.0
Revision:	\main\rel_ors2\15
Stand:	02.08.2017
Status:	freigegeben
Klassifizierung:	öffentlich
Referenzierung:	gemSpec_CardProxy

Dokumentinformationen

Änderungen zur Vorversion

Es handelt sich um die Erstversion des Dokumentes.

Dokumentenhistorie

Version	Stand	Kap./ Seite	Grund der Änderung, besondere Hinweise	Bearbeitung
1.0.0	02.08.17		freigegeben	gematik

Inhaltsverzeichnis

1 Einordnung des Dokuments

1.1 Zielsetzung

Die vorliegende übergreifende Spezifikation definiert Anforderungen für den Themenbereich „Ansteuerung von Smartcards in der Telematikinfrastruktur“, die bei der Realisierung (bzw. dem Betrieb) von Produkttypen der TI zu beachten sind. Diese Anforderungen sind als übergreifende Regelungen relevant für Interoperabilität und Verfahrenssicherheit.

Die Telematikinfrastruktur im deutschen Gesundheitswesen setzt Smartcards an diversen Stellen ein. Als prominente Vertreter seien an dieser Stelle die elektronische Gesundheitskarte eGK und der Heilberufeausweis HBA genannt. Die gematik spezifiziert im Auftrag ihrer Gesellschafter sowohl die Smartcards, als auch die Fachanwendungen, welche diese Smartcards nutzen.

Im Rahmen der Spezifikation von Fachanwendungen werden Abläufe beschrieben, in die möglicherweise auch Smartcards eingebunden sind. Die in diesem Dokument spezifizierte Komponente „Card Proxy“ beschreibt die Nutzung der Smartcards so generisch, dass beispielsweise ein Wechsel der Kartengeneration typischerweise keine Auswirkung auf die Spezifikation von Fachanwendungen hat. Zusätzlich ist die Schnittstelle, die Card Proxy dabei bereitstellt, so einfach, dass ein detailliertes Wissen über Smartcards nicht erforderlich ist, um Fachanwendungen hinreichend genau zu spezifizieren.

Dieses Dokument impliziert nicht, dass es eine dedizierte Komponente „Card Proxy“ als eigenständigen Produkttypen gibt, oder dass so eine Komponente zwingend Bestandteil eines kartennutzenden Produkttypen wird (etwa des Konnektors). Es ist eher so, dass über einen solchen kartennutzenden Produkttypen Interaktionen mit Smartcards ausgelöst werden. Beispielsweise „Ändere PIN“ an einem AdV-Terminal. Dann liefert die Beschreibung der fachlichen Abläufe in Verbindung mit den dortigen Aufrufen von Card Proxy die Interaktionen, die an der Schnittstelle zur jeweiligen Smartcard sichtbar werden. In diesem Sinne hilft das vorliegende Dokument zu Card Proxy

(1) bei der Beschreibung fachlicher Abläufe, da dort auch komplexe Kartenoperationen als einfache Funktionsaufrufe beschreibbar sind,

(2) bei der Implementierung kartennaher Aktionen, da die Kapitel 6 und 9 so fein granular die Interaktionen mit einer Smartcard beschreiben, dass damit sofort Code erzeugbar ist und

(3) bei der Beschreibung Testerwartungshaltung an der Kartenschnittstelle, da die detaillierte Beschreibung insbesondere in Kapitel 9 direkt zur Testerwartungshaltung führt.

1.2 Zielgruppe

Das Dokument ist hilfreich für die Spezifikation von Komponenten, welche Smartcards der Telematikinfrastruktur nutzen, weil der Zugriff und die Nutzung von Smartcards sich mit der hier beschriebenen Schnittstelle zum Card Proxy einfacher beschreiben lässt.

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 (zum Beispiel Dokumentenlandkarte, Produkttypsteckbrief, Leistungsbeschreibung) festgelegt und bekannt gegeben.

Die Festlegungen des Dokuments gelten derzeit ausschließlich im Zusammenhang mit den Anwendungen des Versichterten (AdV) – konkret für den Server–Anteil der Kostenträger-AdV.

Wichtiger 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

Es ist keine Abgrenzung gegenüber anderen Spezifikationen/Konzepten oder im Kontext derzeit nicht relevanten Themen erforderlich.

1.5 Methodik

Anforderungen als Ausdruck normativer Festlegungen werden durch eine eindeutige ID in eckigen Klammern sowie die dem [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.

Anforderungen werden im Dokument wie folgt dargestellt:

⌦ **gemxxxxxx_AFO_0000 <Titel der Afo>**

Text / Beschreibung⌫

Dabei umfasst die Anforderung sämtliche innerhalb der Textmarken angeführten Inhalte.

1.6 Hinweis auf offene Punkte

Das Kapitel wird in einer späteren Version des Dokumentes ergänzt.

2 Konzept

2.1 Konzept der Komponente Card Proxy

Das Konzept zur Komponente Card Proxy basiert auf den folgenden Annahme: Fachanwendungen sind rein für die fachlichen Aspekte einer Anwendung zuständig. Fachanwendungen kennen zumindest den Namen und den Typ der Artefakte in denen ihre Daten auf einer Karte gespeichert sind. Fachanwendungen wissen, wie Daten in den ihnen zugeordneter Dateien gespeichert sind (XML, ASN.1, TLV, gezippt, …).

Für Fachanwendungen ist es irrelevant

(1) wo die Artefakte auf einer Karte liegen,

(2) welche Zugriffsrechte zum Ausführen von Aktionen erforderlich sind,

(3) wie erforderliche Zugriffsrechte erworben werden können, oder

(4) wie auf Daten zugegriffen oder gewisse Aktionen ausgeführt werden,

weil diese Informationen von Card Proxy gekapselt werden.

Card Proxy enthält pro unterstütztem Produkttyp (beispielsweise eGK G1, eGK G1+, eGK G2, …, SMC-B, …) eine Datenbank mit folgenden Informationen:

(1) Eineindeutiger Identifikator eines Artefaktes: Dieser wird an der Schnittstelle zum Card Proxy verwendet um Artefakte zu adressieren.

(2) Ort eines Artefaktes: Diese Information wird von Card Proxy verwendet um ein Artefakt innerhalb einer Karte zu adressieren.

(3) Aktionen, die mit einem Artefakt ausführbar sind: Je nach Typ sind Aktionen aus der Menge {lesen, schreiben, …} oder aber {verschlüsseln, entschlüsseln, signieren, …} oder aber {verifizieren, entsperren, …} relevant.

(4) Pro Aktion eine Zugriffsbedingung, die für eine erfolgreiche Aktion erfüllt sein muss, beispielsweise:

   a. lesen: MRPIN.XYZ

   b. schreiben: PIN.abc AND AuthentisierungMitFlag.ijk

Diese Datenbank ist im Wesentlichen ein spezifischer Auszug aus der zugehörigen Objektsystemspezifikation. Die Datenbank enthält maximal die Aktionen, die gemäß Objektsystemspezifikation erlaubt sind. Jede darüber hinausgehende Aktion würde von der Karte abgewiesen. Es ist denkbar, dass die Datenbank weniger Aktionen enthält, als per Objektsystemspezifikation möglich. Dann würde Card Proxy eine Aktion, die über das von der Datenbank erlaubte hinausgeht, unterbinden, auch wenn so eine Aktion laut Objektsystemspezifikation möglich ist.

Hinweis (1): In diesem Dokument wird lediglich Card Proxy vor dem Hintergrund KTR-AdV betrachtet. Allerdings ist Card Proxy konzeptionell so angelegt, dass Card

Proxy auch in anderen Umgebungen einsetzbar wäre, etwa in LE-AdV. Im Allgemeinen wird die Datenbank eGK G2 je nach Umgebung einen anderen Inhalt haben. Das bedeutet beispielsweise, dass die Datenbank einer eGK für ein AdV Terminal etwa die Aktion „Zugriffsprotokol lesen“ enthält, während so ein Eintrag in der Datenbank einer eGK in einer Arztumgebung nicht vorhanden ist.

2.2 Konzept der Komponente Kartenterminal Proxy

Das Konzept zu Card Proxy setzt voraus, dass Card Proxy eine Schnittstelle zu einem Kartenterminal Proxy besitzt, der wiederum Zugriff auf eine Smartcard hat. Es wird davon ausgegangen, dass die Komponente Kartenterminal Proxy alle herstellerspezifischen Aspekte eines Kartenterminals kapselt.

Falls beispielsweise ein Sicherheitsklasse-1-Kartenleser zur Verfügung steht und Card Proxy stößt eine Benutzerverifikation an, dann sorgt der Kartenterminal Proxy in diesem Fall dafür, dass der Benutzer an der „ganz normalen“ Ausgabe (Bildschirm oder ähnliches) zur Eingabe der PIN aufgefordert wird. Der Kartenterminal Proxy wird in diesem Fall die PIN von der „ganz normalen“ Eingabe (Tastatur oder ähnliches) entgegennehmen.

Falls zu einem anderen Zeitpunkt ein Sicherheitsklasse 3 Kartenleser zur Verfügung steht und Card Proxy stößt wieder eine Benutzerverifikation an, dann wird der Kartenterminal Proxy den Benutzer idealerweise am Display des Kartenlesers zur PIN-Eingabe auffordern und den Sicherheitsklasse-3-Kartenleser so ansteuern, dass die PIN am PIN-Pad eingegeben wird und direkt zur Karte gelangt. Die PIN verlässt in diesem Fall die Hardware des Sicherheitsklasse-3-Kartenlesers nicht.

Falls eine kontaktlose Verbindung zu einer Karte aufgebaut wird, dann kapselt der Kartenterminal Proxy auch die PACE Authentisierung und den dabei etablierten sicheren Kanal, ohne dass Card Proxy oder übergeordnete Komponenten davon berührt wären.

Da dieses Dokument auf Card Proxy fokussiert, werden im Folgenden nur solche Aktionen für die Komponente Kartenterminal Proxy betrachtet, die für Card Proxy relevant sind.

Falls zu einem späteren Zeitpunkt die Komponente „Kartenterminal Proxy“ ausführlich in einem eigenen Dokument behandelt wird, dann wären die folgenden Unterkapitel durch Referenzen in so ein Dokument zu ersetzen.

2.2.1 Aktion „Karte verfügbar“

Falls das Ereignis „Karte verfügbar“ am Kartenterminal Proxy eintritt sind für Card Proxy folgende Aspekte relevant:

Typischerweise enthalten kontaktbehaftete Kartenleser einen Sensor, der erkennt, wenn eine Smartcard gesteckt wird. Der Kartenterminal Proxy aktiviert die Smartcard und prüft, ob ein ATR empfangen wird. Falls kein ATR empfangen wird, so kann dies verschiedene Ursachen haben (Karte falsch gesteckt, Karte defekt, …), die hier nicht weiter betrachtet werden.

Auch kontaktlose Kartenleser erkennen typischerweise anhand von Sensoren, dass eine Smartcard im Feld des Lesers verfügbar ist und bauen einen Kommunikationskanal zu diesen auf.

Der Kartenterminal Proxy sendet an den Card Proxy das Signal „Karte verfügbar“, wenn der Kommunikationskanal zur Smartcard so geöffnet ist, dass APDU zwischen Smartcard und Kartenterminal ausgetauscht werden können.

2.2.2 Aktion „Karte entfernt“

Typischerweise erkennen Kartenleser, dass eine Smartcard aus dem Leser entfernt wurde. In diesem Fall informiert der Kartenterminal Proxy den Card Proxy über das Signal „Karte entfernt“ über dieses Ereignis.

2.2.3 Ändern eines Benutzergeheimnisses

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Ändern eines Benutzergeheimnisses, welche vom Card Proxy genutzt wird, siehe 9.4.2 Schritt 3)e) in der Variante *mode*=replace. Der Fall *mode*=set wird in 2.2.7 behandelt.

Die Umgebung erfragt vom Benutzer den alten und den neuen PIN-Wert. Es wird empfohlen, dass die vom Benutzer eingegebene Ziffernfolge für den neuen PIN-Wert anhand der von Card Proxy übergebenen Parameter *minimumLength* und *maximumLength* auf Plausibilität geprüft wird.

Die Ziffernfolgen für alten und neuen PIN-Wert werden von der Umgebung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierten Ziffernfolgen werden an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 1 zeigt einige Beispiele, die im Folgenden näher erläutert werden:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*=‘01‘=1, das durch die Änderung auch von vier auf fünf Stellen verlängert wird.

2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘03‘=3, das durch die Änderung auch von neun auf sechs Stellen verkürzt wird.

3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘0C‘=12, das durch die Änderung auch von zwölf auf vier Stellen verkürzt wird.

Tabelle 1: Beispiele für die Erzeugung einer Kommando APDU, PIN ändern

commandApduPart	alter PIN Wert neuer PIN Wert	CHANGE REFERENCE DATA Kommando APDU	Bemerkung
0024000110	1234 54321	´0024 0001 10 2412 34FF FFFF FFFF 2554 321F FFFF FFFF´	Beispiel 1
0024008310	987654321 321654	´0024 0083 10 2998 7654 321F FFFF 2632 1654 FFFF FFFF´	Beispiel 2
0024008C10	012345678912 3241	´0024 008C 10 2C01 2345 6789 12FF 2432 41FF FFFF FFFF´	Beispiel 3

2.2.4 Ausschalten Notwendigkeit Benutzerverifikation

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Ausschalten der Notwendigkeit eine Benutzerverifikation durchzuführen, welche vom Card Proxy genutzt wird, siehe 9.4.5 Schritt 3)e). Im Rahmen der Aktion erfragt die Umgebung vom Benutzer typischerweise eine PIN. Es wird empfohlen, dass die vom Benutzer eingegebene Ziffernfolge anhand der von Card Proxy übergebenen Parameter *minimumLength* und *maximumLength* auf Plausibilität geprüft wird. Die Ziffernfolge wird von der Umge-

bung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierte Ziffernfolge wird an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 2 zeigt einige Beispiele, die im Folgenden näher erläutert werden:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*='01'=1, wobei zum Ausschalten das vierstellige Benutzergeheimnis mitgesendet wird.

2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*='03'=3, wobei zum Ausschalten das neunstellige Benutzergeheimnis mitgesendet wird.

3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*='0C'=12, wobei zum Ausschalten kein Benutzergeheimnis mitgesendet wird.

Tabelle 2: Beispiele für die Erzeugung einer Kommando APDU, disable

commandApduPart	PIN	DISABLE VERIFICATION REQUIREMENT APDU	Bemerkung
0026000108	1234	´0026 0001 08 2412 34FF FFFF FFFF´	Beispiel 1
0026008308	987654321	´0026 0083 08 2998 7654 321F FFFF´	Beispiel 2
0026018C	-	´0026 018C´	Beispiel 3

2.2.5 Benutzerverifikation

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zur Benutzerverifikation, welche vom Card Proxy genutzt wird, siehe 9.4.10 Schritt 3)e). Im Rahmen der Benutzerverifikation erfragt die Umgebung vom Benutzer eine PIN. Es wird empfohlen, dass die vom Benutzer eingegebene Ziffernfolge anhand der von Card Proxy übergebenen Parameter *minimumLength* und *maximumLength* auf Plausibilität geprüft wird. Die Ziffernfolge wird von der Umgebung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierte Ziffernfolge wird an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 3 zeigt einige Beispiele, die im Folgenden näher erläutert werden:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*='01'=1 und der Verifikation eines vierstelligen Benutzergeheimnisses.

2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*='03'=3 und der Verifikation eines neunstelligen Benutzergeheimnisses.

3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*='0C'=12 und der Verifikation eines zwölfstelligen Benutzergeheimnisses.

Tabelle 3: Beispiele für die Erzeugung einer Kommando APDU, verify

commandApduPart	PIN	VERIFY Kommando APDU	Bemerkung
0020000108	1234	´0020 0001 08 2412 34FF FFFF FFFF´	Beispiel 1
0020008308	987654321	´0020 0083 08 2998 7654 321F FFFF´	Beispiel 2
0020008C08	012345678912	´0020 008C 08 2C01 2345 6789 12FF´	Beispiel 3

Hinweis (2): Der Card Proxy erhebt keine Anforderungen, wie die Umgebung die Funktion zur Benutzerauthentisierung implementiert. Folgende Fälle sind denkbar, die sinngemäß auch auf die übrigen Beispiele mit Passwortobjekten in 2.2 übertragbar sind:

a. Die Umgebung erfragt vom Benutzer das Geheimnis (PIN) und sendet ein passendes VERIFY Kommando zur Karte. Die Antwortnachricht dieses VERIFY Kommandos wird zurückgemeldet.

b. *Es ist denkbar, dass die Umgebung anhand der Antwortnachricht auf das erste VERIFY Kommando*
   - *einen Transportschutz oder eine blockierte PIN erkennt und dieses Problem selbständig beseitigt, bevor ein erneutes VERIFY Kommando gesendet wird.*
   - *eine fehlerhafte PIN-Eingabe erkennt und den Benutzer selbständig ein weiteres Mal zur Eingabe des Geheimnisses auffordert, bevor ein erneutes VERIFY Kommando gesendet wird.*

c. *Falls der Benutzer das Geheimnis zu oft hintereinander falsch eingibt und dabei das Passwortobjekt blockiert, könnte die Umgebung mit einer passenden Fehlermeldung abbrechen oder selbständig versuchen die Blockade wieder zu lösen.*
d. *Weitere Fälle sind denkbar.*

2.2.6 Einschalten Notwendigkeit Benutzerverifikation

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Einschalten der Notwendigkeit eine Benutzerverifikation durchzuführen, welche vom Card Proxy genutzt wird, siehe 9.4.6 Schritt 3)e). Im Rahmen der Aktion erfragt die Umgebung vom Benutzer typischerweise eine PIN. Es wird empfohlen, dass die vom Benutzer eingegebene Ziffernfolge anhand der von Card Proxy übergebenen Parameter *minimumLength* und *maximumLength* auf Plausibilität geprüft wird. Die Ziffernfolge wird von der Umgebung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierte Ziffernfolge wird an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 4 zeigt einige Beispiele, die im Folgenden näher erläutert werden:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*=‘01‘=1, wobei zum Einschalten das vierstellige Benutzergeheimnis mitgesendet wird.

2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘03‘=3, wobei zum Einschalten das neunstellige Benutzergeheimnis mitgesendet wird.

3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘0C‘=12, wobei zum Einschalten kein Benutzergeheimnis mitgesendet wird.

Tabelle 4: Beispiele für die Erzeugung einer Kommando APDU, enable

commandApduPart	PIN	ENABLE VERIFICATION REQUIREMENT APDU	Bemerkung
0028000108	1234	´0028 0001 08 2412 34FF FFFF FFFF´	Beispiel 1
0028008308	987654321	´0028 0083 08 2998 7654 321F FFFF´	Beispiel 2
0028018C	-	´0028 018C´	Beispiel 3

2.2.7 Setzen eines Benutzergeheimnisses

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Setzen eines Benutzergeheimnisses, welche vom Card Proxy genutzt wird, siehe 9.4.2 Schritt 3)e) in der Variante *mode*=set. Der Fall *mode*=replace wird in 2.2.3 behandelt.

Die Umgebung erfragt vom Benutzer den neuen PIN-Wert. Es wird empfohlen, dass die vom Benutzer eingegebene Ziffernfolge für den neuen PIN-Wert anhand der von Card Proxy übergebenen Parameter *minimumLength* und *maximumLength* auf Plausibilität geprüft wird.

Die Ziffernfolge für den neuen PIN-Wert wird von der Umgebung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierten Ziffernfolgen werden an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 5 zeigt einige Beispiele, die im Folgenden näher erläutert werden:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*=‘01‘=1 und dem Setzen eines vierstelligen Benutzergeheimnisses.

2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘03‘=3 und dem Setzen eines sechsstelligen Benutzergeheimnisses.

3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘0C‘=12 und dem Setzen eines zwölfstelligen Benutzergeheimnisses.

Tabelle 5: Beispiele für die Erzeugung einer Kommando APDU, set PIN

commandApduPart	neuer PIN Wert	CHANGE REFERENCE DATA Kommando APDU	Bemerkung
0024010108	4321	´0024 0101 08 2443 21FF FFFF FFFF´	Beispiel 1
0024018308	321654	´0024 0183 08 2632 1654 FFFF FFFF´	Beispiel 2
0024018C08	012345678912	´0024 018C 08 2C01 2345 6789 12FF´	Beispiel 3

2.2.8 UnblockWithPukAndSet

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Entsperren eines Benutzergeheimnisses, welche vom Card Proxy genutzt wird, siehe 9.4.9 Schritt 3)e) in der Variante *mode*=UnblockWithPuKAndSet.

Die Umgebung erfragt vom Benutzer die PUK und den neuen PIN-Wert. Es wird empfohlen, dass die vom Benutzer eingegebene Ziffernfolge für den neuen PIN-Wert anhand der von Card Proxy übergebenen Parameter *minimumLength* und *maximumLength* auf Plausibilität geprüft wird. Die PUK besteht stets aus acht Ziffern.

Die Ziffernfolgen PUK und neuer PIN-Wert werden von der Umgebung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierten Ziffernfolgen werden an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 6 zeigt einige Beispiele, die im Folgenden näher erläutert werden, wobei die PUK, wie in der Telematikinfrastruktur üblich, stets acht Stellen lang ist:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*=‘01‘=1 und dem Setzen eines fünfstelligen Benutzergeheimnisses.

2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘03‘=3 und dem Setzen eines sechsstelligen Benutzergeheimnisses.

3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘0C‘=12 und dem Setzen eines zwölfstelligen Benutzergeheimnisses.

Tabelle 6: Beispiele für die Erzeugung einer Kommando APDU, unblock PUK set

commandApduPart	PUK neuer PIN Wert	RESET RETRY COUNTER Kommando APDU	Bemerkung
002C000110	12345678 54321	´002C 0001 10 2812 3456 78FF FFFF 2554 321F FFFF FFFF´	Beispiel 1
002C008310	98765432 321654	´002C 0083 10 2898 7654 32FF FFFF 2632 1654 FFFF FFFF´	Beispiel 2
002C008C10	01234567 012345678912	´002C 008C 10 2801 2345 67FF FFFF 2C01 2345 6789 12FF´	Beispiel 3

2.2.9 UnblockWithPuk

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Entsperren eines Benutzergeheimnisses, welche vom Card Proxy genutzt wird, siehe 9.4.9 Schritt 3)e) in der Variante *mode*=UnblockWithPuK.

Die Umgebung erfragt vom Benutzer die PUK. Die PUK besteht stets aus acht Ziffern.

Die Ziffernfolge PUK wird von der Umgebung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierte Ziffernfolge wird an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 7 zeigt einige Beispiele, die im Folgenden näher erläutert werden, wobei die PUK, wie in der Telematikinfrastruktur üblich, stets acht Stellen lang ist:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*=‘01‘=1.
2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘03‘=3.
3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘0C‘=12.

Tabelle 7: Beispiele für die Erzeugung einer Kommando APDU, unblock PUK

commandApduPart	PUK	RESET RETRY COUNTER Kommando APDU	Bemerkung
002C010108	12345678	´002C 0101 08 2812 3456 78FF FFFF´	Beispiel 1
002C018308	98765432	´002C 0183 08 2898 7654 32FF FFFF´	Beispiel 2
002C018C08	01234567	´002C 018C 08 2801 2345 67FF FFFF´	Beispiel 3

2.2.10 UnblockAndSet

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Entsperren eines Benutzergeheimnisses, welche vom Card Proxy genutzt wird, siehe 9.4.9 Schritt 3)e) in der Variante *mode*=UnblockAndSet.

Die Umgebung erfragt vom Benutzer den neuen PIN-Wert. Es wird empfohlen, dass die vom Benutzer eingegebene Ziffernfolge für den neuen PIN-Wert anhand der von Card Proxy übergebenen Parameter *minimumLength* und *maximumLength* auf Plausibilität geprüft wird.

Die Ziffernfolge des neuen PIN-Wertes wird von der Umgebung gemäß Format-2-PIN-Block codiert, siehe 12.3. Die so codierte Ziffernfolge wird an den Parameter commandApduPart angehängt und dann zur Karte gesendet.

Tabelle 8 zeigt einige Beispiele, die im Folgenden näher erläutert werden:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*=‘01‘=1 und dem Setzen eines fünfstelligen Benutzergeheimnisses.
2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘03‘=3 und dem Setzen eines sechsstelligen Benutzergeheimnisses.
3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘0C‘=12 und dem Setzen eines zwölfstelligen Benutzergeheimnisses.

Tabelle 8: Beispiele für die Erzeugung einer Kommando APDU, unblock set

commandApduPart	neuer PIN Wert	RESET RETRY COUNTER Kommando APDU	Bemerkung
002C020108	54321	´002C 0201 08 2554 321F FFFF FFFF´	Beispiel 1
002C028308	321654	´002C 0283 08 2632 1654 FFFF FFFF´	Beispiel 2
002C028C08	012345678912	´002C 028C 08 2C01 2345 6789 12FF ´	Beispiel 3

2.2.11 Unblock

Der Kartenterminal Proxy als Teil der Umgebung implementiert eine Funktion zum Entsperren eines Benutzergeheimnisses, welche vom Card Proxy genutzt wird, siehe 9.4.9 Schritt 3)e) in der Variante *mode*=Unblock. Da in dieser Variante vom Benutzer keine Daten zu erfassen sind, wird in dieser Variante der Parameter commandApduPart unverändert als Kommando APDU verwendet.

Tabelle 9 zeigt einige Beispiele, die im Folgenden näher erläutert werden:

1) Beispiel 1: Globales Passwortobjekt mit *pwdIdentifier*=‘01‘=1.

2) Beispiel 2: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘03‘=3.

3) Beispiel 3: DF-spezifisches Passwortobjekt *pwdIdentifier*=‘0C‘=12.

Tabelle 9: Beispiele für die Erzeugung einer Kommando APDU, unblock

commandApduPart	neuer PIN Wert	RESET RETRY COUNTER Kommando APDU	Bemerkung
002C0301	-	´002C 0301´	Beispiel 1
002C0383	-	´002C 0383´	Beispiel 2
002C038C	-	´002C 038C´	Beispiel 3

3 Bausteine innerhalb von Card Proxy

(1) *channelContext* analog zu [gemSpec_COS#(N029.900)] mit den Punkten:

a. (N029.900)a *currentFolder* plus dessen logischem Wert von *lifeCycleStatus*, woraus sich die jeweils gültige Zugriffsregel ergibt.

b. (N029.900)c keyReferenceList

c. (N029.900)e globalSecurityList

d. (N029.900)f dfSpecificSecurityList

e. (N029.900)h bitSecurityList

f. (N029.900)i globalPasswordList

g. (N029.900)j dfSpecificPasswordList

h. (N029.900)m *currentEF* plus dessen logischem Wert von *lifeCycleStatus*, woraus sich die jeweils gültige Zugriffsregel ergibt.

(2) SupportedActions: Pro unterstütztem Kartentyp eine Datenbank mit adressierbaren Objekten, Aktionen und Zugriffsbedingungen. Beispiele dazu finden sich in Kapitel 8.

(3) CV-CertificateStore: Speicher mit Root und Sub-CA-CV-Zertifikaten sowie dem End-Entity-CV-Zertifikat der SM-B im AdV-Server.

(4) StatusTransparentChannel: open oder closed.

(5) Buffersize, enthält Informationen über die maximale Größe einer Nachricht, die zu einer Smartcard geschickt werden, oder von dort abgeholt werden kann.

(6) Sicherheitsstatus: Hier wird gespeichert, für welche Passwortobjekte ein Sicherheitszustand in der Smartcard gesetzt wurde und Rollen bzw. Flaglisten per Card-2-Card in der Smartcard authentisiert wurden. Diese Informationen werden in Kapitel 10 benötigt. Alternativ ist es möglich den Sicherheitszustand

a. eines Passwortobjektes von der Smartcard mittels cardOperation(IdentifikatorDesPasswortObjektes, getStatus) zu erfragen und

b. einer Flaglist cardOperation(„Wildcard“, getSecurityStatusFlagList, *oid*, *flagList*) und

c. einer Rolle mittels cardOperation(„Wildcard“, getSecurityStatusRole, *role*).

4 Ablauf einer Kartensession

In diesem Kapitel wird davon ausgegangen, dass Card Proxy vom Kartenterminal Proxy über die Ereignisse „Karte verfügbar“ (siehe 2.2.1) und „Karte entfernt“ (siehe 2.2.2) informiert wird. Dieses Kapitel betrachtet die Aktionen, die Card Proxy in so einem Fall durchführt.

4.1 Signal „Karte verfügbar“ empfangen

Sobald Card Proxy das Signal „Karte verfügbar“ empfängt ist eine Kommunikation zwischen Card Proxy und der Smartcard möglich. Der Card Proxy ermittelt dann zunächst die Stammdaten der Smartcard und meldet diese an die Umgebung. Sobald die Umgebung die Stammdaten der Smartcard empfangen hat, ist es für die Umgebung möglich über den Card Proxy mit der Smartcard zu arbeiten.

Die Ermittlung der Stammdaten startet mit der Abfrage in 4.1.1.

4.1.1 Kartentyp ermitteln

Card Proxy sendet eine SELECT Kommando gemäß [gemSpec_COS#(N041.300)] an die Smartcard: Selektiere das Wurzelverzeichnis, Antwortdaten mit File Control Parametern FCP, Kommando APDU = ´00 A4 04 04 00 0000´.

Falls die Antwortnachricht einen Trailer enthält, der nicht Element der Menge {

1) ´62 83´ = FileDeactivated,
2) ´62 85´ = FileTerminated,
3) ´90 00´ = NoError

} ist, dann meldet Card Proxy der Umgebung in den Stammdaten den Kartentyp „ungültig“.

Andernfalls wird in den FCP nach einem *applicationIdentifier* gesucht. Falls die FCP einen *applicationIdentifier* mit dem Wert

4)	´D276 0001 44 80 00´	enthalten → eGK,	weiter mit 4.1.2
5)	´D276 0001 46 01´	enthalten → HBA,	weiter mit 4.1.3
6)	´D276 0001 46 06´	enthalten → SMC-B,	weiter mit 4.1.4
7)	´D276 0001 44 80 01´	enthalten → gSMC-K,	weiter mit 4.1.5
8)	´D276 0001 44 80 01´	enthalten → gSMC-KT,	weiter mit 4.1.6

Falls keiner der vorgenannten *applicationIdentifier* vorhanden ist, dann meldet Card Proxy der Umgebung in den Stammdaten den Kartentyp „ungültig“.

4.1.2 eGK, Kartengeneration ermitteln

1) Kartengeneration ermitteln:
   a) Es wird eine cardOperation(EF.Version, read) ausgeführt. Falls der Inhalt des ersten Rekords

i) ´003 000 0000´ ist, dann liegt eine Generation 1 eGK vor. Dem Inhalt des zweiten Rekords wird die Version des Objektsystems entnommen, weiter mit 4.1.2.1.
ii) ´004 000 0000´ ist, dann liegt eine Generation 2 eGK vor. Weiter mit 4.1.2.2.

4.1.2.1 Generation 1 eGK, weitere Stammdaten ermitteln

1) Puffergrößen ermitteln für G1 eGK: Es wird eine cardOperation(EF.ATR, read) ausgeführt. Der Inhalt des Datenobjektes mit Tag ´E0´ wird gemäß [gemSpec_Karten_Fach_TIP#Card-G2-A_2386] ausgewertet und die dort angegebenen Puffergrößen werden in den Baustein Buffersize übernommen.
2) Eine G1 eGK enthält keine Produkttypversion für das COS. Implizit ergibt sich die Produkttypversion aber bereits aus der Information „G1 eGK“.
3) Eine G1 eGK enthält keine Produkttypversion für das Objektsystem. Implizit ergibt sich die Produkttypversion aber dem zweiten Rekord von EF.Version, der bereits in 4.1.2 Punkt 1)a) ausgelesen wurde. Aus dem Inhalt des zweiten Rekords lassen sich die Typen G1 und G1+ unterscheiden.
4) End-Entity-CV-Zertifikat auslesen: Es wird aus EF.C.eGK.AUT_CVC ausgelesen.
5) Konditional: Falls das Sub-CA-CV-Zertifikat zu C.eGK.AUT_CVC nicht im Baustein CV-CertificateStore gespeichert ist, wird diese aus EF.C.CA_eGK.CS ausgelesen und im Baustein CV-CertificateStore gespeichert.
6) Falls beim Import der CV-Zertifikate in den Baustein CV-CertificateStore festgestellt wird, dass
   a) mindestens eines der CV-Zertifikate nicht gültig ist, dann meldet Card Proxy der Umgebung in den Stammdaten den Kartentyp „ungültig“.
   b) alle CV-Zertifikate gültig sind, wird dem End-Entity-CV-Zertifikat die ICCSN entnommen sowie der öffentliche Schlüssel PuK.C.eGK.AUT_CVC.
7) Die Echtheit (nicht die Gültigkeit) der Smartcard wird geprüft. Dazu werden folgende Schritte ausgeführt:
   a) Es wird eine cardOperation(PrK.eGK.AUT_CVC, rsaRoleAuthentication) durchgeführt.
   b) Die Signatur aus dem vorherigen INTERNAL AUTHENTICATE Kommando wird mittels PuK.eGK_AUT_CVC geprüft. Falls die Signaturprüfung
      i) nicht erfolgreich verläuft, dann wird als Kartentyp „ungültig“ in die Stammdaten eingetragen.
      ii) erfolgreich verläuft, dann wird in den Stammdaten die Smartcard als „echt“ gekennzeichnet.
   c) Die Stammdaten werden an die Umgebung gemeldet und bestehen dabei entweder nur aus „Kartentyp ungültig“, oder aus den folgenden Artefakten:
      i) Kartentyp,
      ii) Produkttypversion COS,
      iii) Produkttypversion Objektsystem,
      iv) ICCSN und
      v) Smartcard „echt“.

4.1.2.2 Generation 2 eGK, weitere Stammdaten ermitteln

1) Puffergrößen ermitteln für G2 eGK: Es wird eine cardOperation(EF.ATR, read) ausgeführt. Der Inhalt des Datenobjektes mit Tag ´E0´ wird gemäß [gemSpec_Karten_Fach_TIP#Card-G2-A_2386] ausgewertet und die dort angegebenen Puffergrößen werden in den Baustein Buffersize übernommen.

2) Die Produkttypversion des COS wird dem Inhalt des EF.ATR entnommen, siehe [gemSpec_Karten_Fach_TIP# Card-G2-A_3488].
3) Die Produkttypversion des vorhandenen Objektsystems wird EF.Version2 entnommen, siehe [gemSpec_Karten_Fach_TIP# Tab_Karten_Fach_TIP_002] und dort das Datenobjekt mit dem „Pfad“ ‘EF‘ → ‘C1‘.
4) End-Entity-CV-Zertifikat auslesen: Es wird aus EF.C.eGK.AUT_CVC.E256 ausgelesen.
5) Konditional: Falls das Sub-CA-CV-Zertifikat zu C.eGK.AUT_CVC nicht im Baustein CV-CertificateStore gespeichert ist, wird diese aus EF.C.CA_eGK.CS.E256 ausgelesen und im Baustein CV-CertificateStore gespeichert.
6) Falls beim Import der CV-Zertifikate in den Baustein CV-CertificateStore festgestellt wird, dass
   a) mindestens eines der CV-Zertifikate nicht gültig ist, dann meldet Card Proxy der Umgebung in den Stammdaten den Kartentyp „ungültig“.
   b) alle CV-Zertifikate gültig sind, wird dem End-Entity-CV-Zertifikat die ICCSN entnommen sowie der öffentliche Schlüssel PuK.C.eGK.AUT_CVC.
7) Die Echtheit (nicht die Gültigkeit) der Smartcard wird geprüft. Dazu werden folgende Schritte ausgeführt:
   a) Es wird eine cardOperation(PrK.eGK.AUTR_CVC.E256, elcRoleAuthentication) durchgeführt.
   b) Die Signatur aus dem vorherigen INTERNAL AUTHENTICATE Kommando wird mittels PuK.eGK_AUT_CVC geprüft. Falls die Signaturprüfung
      i) nicht erfolgreich verläuft, dann wird als Kartentyp „ungültig“ in die Stammdaten eingetragen.
      ii) erfolgreich verläuft, dann wird in den Stammdaten die Smartcard als „echt“ gekennzeichnet.
   c) Die Stammdaten werden an die Umgebung gemeldet und bestehen dabei entweder nur aus „Kartentyp ungültig“, oder aus den folgenden Artefakten:
      i) Kartentyp,
      ii) Produkttypversion COS,
      iii) Produkttypversion Objektsystem,
      iv) ICCSN und
      v) Smartcard „echt“.

4.1.3 HBA, weitere Stammdaten

Dieser Kartentyp ist für die KTR-AdV irrelevant. Falls dieses Dokument auch für andere Einsatzzwecke verwendet wird, dann ist es möglich dieses Kapitel in einer späteren Version inhaltlich zu füllen.

4.1.4 SMC-B, weitere Stammdaten

1) Puffergrößen ermitteln für SMC-B: Es wird eine cardOperation(EF.ATR, read) ausgeführt. Der Inhalt des Datenobjektes mit Tag ´E0´ wird gemäß [gemSpec_Karten_Fach_TIP#Card-G2-A_2386] ausgewertet und die dort angegebenen Puffergrößen werden in den Baustein Buffersize übernommen.
2) Die Produkttypversion des COS wird dem Inhalt des EF.ATR entnommen, siehe [gemSpec_Karten_Fach_TIP# Card-G2-A_3488].
3) Die Produkttypversion des vorhandenen Objektsystems wird EF.Version2 entnommen, siehe [gemSpec_Karten_Fach_TIP# Tab_Karten_Fach_TIP_002] und dort das Datenobjekt mit dem „Pfad“ ‘EF‘ → ‘C1‘.
4) End-Entity-CV-Zertifikat auslesen: Sie werden aus

a) EF.C.SMC.AUTR_CVC.E256 und
b) EF.C.CA_SMC.CS.R2048 ausgelesen.

5) Konditional: Falls das Sub-CA-CV-Zertifikat zu
   a) C.SMC.AUT_CVC.E256 nicht im Baustein CV-CertificateStore gespeichert ist, wird diese aus EF.C.CA_SMC.CS.E256 ausgelesen und im Baustein CV-CertificateStore gespeichert.
   b) C.SMC.AUT_CVC.R2048 nicht im Baustein CV-CertificateStore gespeichert ist, wird diese aus EF.C.CA_SMC.CS.R2048 ausgelesen und im Baustein CV-CertificateStore gespeichert.
6) Falls beim Import der CV-Zertifikate in den Baustein CV-CertificateStore festgestellt wird, dass
   a) mindestens eines der CV-Zertifikate nicht gültig ist, dann meldet Card Proxy der Umgebung in den Stammdaten den Kartentyp „ungültig“.
   b) alle CV-Zertifikate gültig sind, wird dem E256 End-Entity-CV-Zertifikat die ICCSN entnommen sowie der öffentliche Schlüssel PuK.C.SMC.AUT_CVC.E256.
7) Die Echtheit (nicht die Gültigkeit) der Smartcard wird geprüft. Dazu werden folgende Schritte ausgeführt:
   a) Es wird eine cardOperation(PrK.SMC.AUTR_CVC.E256, elcRoleAuthentication) durchgeführt.
   b) Die Signatur aus dem vorherigen INTERNAL AUTHENTICATE Kommando wird mittels PuK.eGK_AUT_CVC.E256 geprüft. Falls die Signaturprüfung
      i) nicht erfolgreich verläuft, dann wird als Kartentyp „ungültig“ in die Stammdaten eingetragen.
      ii) erfolgreich verläuft, dann wird in den Stammdaten die Smartcard als „echt“ gekennzeichnet.
   c) Die Stammdaten werden an die Umgebung gemeldet und bestehen dabei entweder nur aus „Kartentyp ungültig“, oder aus den folgenden Artefakten:
      i) Kartentyp,
      ii) Produkttypversion COS,
      iii) Produkttypversion Objektsystem,
      iv) ICCSN und
      v) Smartcard „echt“.

4.1.5 gSMC-K, weitere Stammdaten

Dieser Kartentyp ist für die KTR-AdV irrelevant. Falls dieses Dokument auch für andere Einsatzzwecke verwendet wird, dann ist es möglich dieses Kapitel in einer späteren Version inhaltlich zu füllen.

4.1.6 gSMC-KT, weitere Stammdaten

Dieser Kartentyp ist für die KTR-AdV irrelevant. Falls dieses Dokument auch für andere Einsatzzwecke verwendet wird, dann ist es möglich dieses Kapitel in einer späteren Version inhaltlich zu füllen.

4.2 Signal „Karte entfernt“ empfangen

Falls Card Proxy das Signal „Karte entfernen“ empfängt, dann werden alle kartenspezifischen Informationen aus den Bausteinen des Card Proxys entfernt.

5 Anforderungserhebung

5.1 Zielsetzung

Die kurzen, knappen Ausführungen in 1.1 werden an dieser Stelle etwas ausführlicher am Beispiel eines Kostenträger AdV-Terminals behandelt. Die folgenden Unterkapitel gehen dabei näher auf die Bereiche aus 1.1 ein.

5.1.1 Card Proxy unterstützt die Beschreibung fachlicher Abläufe

Dazu werden die Use Cases „Versichertendaten anzeigen“ und „Zugriffsprotokoll anzeigen“ betrachtet. Der folgenden Beschreibung liegt dabei die Annahme zu Grunde, dass es an der Benutzerschnittstelle Buttons oder ähnliches gibt, mit denen sich unter anderem diese Use Cases auslösen lassen.

Beispielsweise ist es denkbar, dass eine Beschreibung des Use Cases „Versichertendaten anzeigen“ unter anderem folgendes enthält: Falls der Benutzer die Aktion „Versichertendaten anzeigen“ auslöst werden folgende Schritte ausgeführt:

1) Falls noch nicht geschehen, wird in der eGK das Verzeichnis DF.HCA selektiert.
2) Falls noch nicht geschehen, wird PIN.CH vom Benutzer abgefragt und deren Sicherheitszustand in der eGK gesetzt.
3) Falls noch nicht geschehen, wird eine Freischaltung der eGK wie folgt durchgeführt:
   a) Import von CV-Zertifikaten, wie folgt …
   b) Authentisierungsprotokoll, wie folgt …
4) Die Versichertendaten werden wie folgt ausgelesen:
   a) Daten aus EF.StatusVD werden mittels READ BINARY ausgelesen.
   b) Daten aus EF.GVD werden mittels READ BINARY ausgelesen.
   c) Daten aus EF.PD werden mittels READ BINARY ausgelesen.
   d) Daten aus EV.VD werden mittels READ BINARY ausgelesen.
5) Die ausgelesenen Daten werden wie folgt angezeigt: …

Beispielsweise ist es denkbar, dass eine Beschreibung des Use Cases „Zugriffsprotokoll anzeigen“ folgendes enthält:

1) Falls noch nicht geschehen, wird in der eGK das Verzeichnis DF.HCA selektiert.
2) Falls noch nicht geschehen, wird PIN.CH vom Benutzer abgefragt und deren Sicherheitszustand in der eGK gesetzt.
3) Falls noch nicht geschehen, wird eine Freischaltung der eGK wie folgt durchgeführt:
   a) Import von CV-Zertifikaten, wie folgt …

b) Authentisierungsprotokoll, wie folgt …

4) Alle Rekords werden aus EF.Logging mittels READ RECORD ausgelesen.

5) Die ausgelesenen Daten werden wie folgt angezeigt: …

Die Beispiele zeigen, dass jeweils die ersten drei Schritte aus fachlicher Sicht eher uninteressant sind und sehr viel mit der konkreten Kartenimplementierung zu tun haben. Bei dieser Art der Darstellung entsteht viel Beschreibungsaufwand im kartennahen Bereich, der sich vielfach wiederholt.

Unter Zuhilfenahme einer (möglicherweise fiktiven) Komponente „Card Proxy“ ließen sich die oben dargestellten Use Cases etwa wie folgt formulieren:

Use Case „Versichtertendaten anzeigen“:

1) Die Versichertendaten werden wie folgt ausgelesen:

   a) cardOperation(EF.StatusVD, read)

   b) cardOperation(EF.GVD, read)

   c) cardOperation(EF.PD, read)

   d) cardOperation(EF.VD, read)

2) Die ausgelesenen Daten werden wie folgt angezeigt: …

Use Case „Zugriffsrprotokoll anzeigen“:

1) cardOperation(EF.Logging, read)

2) Die ausgelesenen Daten werden wie folgt angezeigt: …

Die kartennahen Implementierungsdetails, etwa wo die Daten liegen, welche Kommandos zum Auslesen erforderlich sind, wie wird der Sicherheitszustand passend gesetzt, all das wird von Card Proxy gekapselt.

Dieses Dokument erhebt keine Anforderungen wie fachliche Abläufe darzustellen sind. Falls bei der Beschreibung fachlicher Abläufe kartennahe Operationen gekapselt werden, dann unterstützt Card Proxy das.

5.1.2 Card Proxy unterstützt die Implementierung

Ähnlich wie in 5.1.1 bei der Beschreibung fachlicher Use Cases ist es auch bei der Implementierung eines KTR-AdV-Terminals denkbar, dass jeder Use Case für sich separat implementiert wird, was voraussichtlich zu vielen Dopplungen im Code führte.

Andererseits ist es denkbar, dass so ein KTR-AdV-Terminal intern Subkomponenten verwendet, und eine der Subkomponenten die Implementierung eines Card Proxy gemäß diesem Dokument ist.

Dieses Dokument erhebt keine Anforderungen an eine Implementierung. Aus Sicht dieses Dokumentes ist eine Vielzahl von Implementierungsvarianten zulässig, von einem monolithischen Block bis hin zu einer hochgradig strukturierten Software.

Card Proxy, so wie er in diesem Dokument beschrieben ist, ist nur eine mögliche Realisierungsvariante um kartennahe Operationen zu kapseln.

5.1.3 Card Proxy unterstützt die Beschreibung der Testerwartungshaltung

Im Rahmen von funktionalen Zulassungstests (hier wieder am Beispiel eines KTR-AdV-Terminals) spielen unter anderem zwei Schnittstellen eine Rolle: Benutzerschnittstelle und Kartenschnittstelle.

Fachliche Abläufe lassen sich, wie in 5.1.1 gezeigt, auf verschiedene Arten darstellen, oder, wie in 5.1.2 gezeigt auf verschiedene Arten implementieren. Im Folgenden wird davon ausgegangen, dass bei der Beschreibung fachlicher Abläufe Card Proxy verwendet wird. Für die Kartenschnittstelle bedeutet das, dass innerhalb der fachlichen Abläufe Card Proxy möglicherweise zum Einsatz kommt (oder auch nicht) und die Beschreibung von Card Proxy in diesem Dokument beschreibt dann, welche Kommandos an der Kartenschnittstelle erwartet werden. Daraus ergibt sich dann die Testerwartungshaltung.

Im Folgenden soll exemplarisch am Beispiel der Use Case „Versichertendaten anzeigen“ (siehe 5.1.1) dargestellt werden, wie sich die Testerwartungshaltung ableiten lässt. Für die Testerwartungshaltung an der Kartenschnittstelle ist für den Use Case „Versichertendaten anzeigen“ folgendes relevant, wobei die gezeigte Abfolge als Beispiel zu verstehen ist. Es ist denkbar, dass aus fachlicher Sicht eine andere Reihenfolge zwingend vorgeschrieben ist. Es ist auch denkbar, dass aus fachlicher Sicht explizit keine Reihenfolge vorgeschrieben wird (das heißt ein Auslesen in einer beliebigen Reihenfolge wäre erlaubt). In diesen Fällen änderte sich die Testerwartungshaltung entsprechend.

1) Die Versichertendaten werden wie folgt ausgelesen:
   a) Schritt 1: cardOperation(EF.StatusVD, read)
   b) Schritt 2: cardOperation(EF.GVD, read)
   c) Schritt 3: cardOperation(EF.PD, read)
   d) Schritt 4: cardOperation(EF.VD, read)

Für das weitere Verständnis ist es wichtig zu erwähnen, dass Card Proxy gemäß Kapitel 3 in *channelContext* den inneren Zustand er Karte mitführt (so gut es geht). Zur Abschätzung des inneren Zustandes (soweit es Card Proxy und die Testerwartungshaltung betrifft) sind die Anforderungen aus [gemSpec_COS] hinreichend. Weitergehende oder herstellerspezifische Kenntnisse sind nicht erforderlich. Im Folgenden werden zwei, aus Sicht dieses Dokumentes zulässige, Implementierungsvarianten unterschieden:

1) Implementierung 1, aufwändiger *channelContext*:
Diese Implementierung sei dadurch gekennzeichnet, dass sie den inneren Kartenzustand sehr genau abbildet, wozu ein hoher Aufwand erforderlich ist. Weil diese Implementierung den inneren Kartenzustand sehr genau kennt, sind für die Implementierung eines Use Cases mitunter weniger Kartenkommandos erforderlich, was die Performanz verbessert.

2) Implementierung 2, rudimentärer channelContext:
Diese Implementierung sei dadurch gekennzeichnet, dass sie den inneren Kartenzustand nur rudimentär abbildet, was wenig aufwändig ist. Dafür sind zusätzliche Kartenkommandos erforderlich, was Zeit erfordert.

Annahmen:

1) Es werde der Use Case „Versichertendaten anzeigen“ ausgeführt.
2) Der Sicherheitszustand zum Auslesen von EF.StatusVD, EF.GVD, EV.PD und EF.VD ist wegen vorangegangener Aktionen in der Karte bereits passend gesetzt, das heißt:
   a) Sicherheitszustand von PIN.CH ist gesetzt und
   b) flagTI.29 ist in *bitSecurityList* eingetragen.
3) In der Karte ist der Ordner DF.ESIGN selektiert.

Tabelle 10: Kartenschnittstelle „Versichertendaten anzeigen“

cardOperation	Implementierung 1	Implementierung 2	Bemerkung
EF.StatusVD read	SELECT DF.HCA NoError	SELECT DF.HCA NoError	*currentFolder* wird passend gesetzt
	SELECT EF.StatusVD NoError	SELECT EF.StatusVD NoError	*currentEF* wird passend gesetzt
	READ BINARY xx…yy, NoError	READ BINARY xx…yy, NoError	Daten werden ausgelesen
EF.GVD read	-	SELECT DF.HCA NoError	nur Implementierung 1 weiß, dass *currentFolder* passend gesetzt ist
	SELECT EF.GVD NoError	SELECT EF.GVD NoError	*currentEF* wird passend gesetzt
	-	GET PIN STATUS NoError	nur Implementierung 1 weiß, dass der Sicherheitsstatus PIN.CH passend gesetzt ist
	-	GET SEC. STATUS KEY NoError	nur Implementierung 1 weiß, dass *bitSecurityList* passend gesetzt ist
	READ BINARY xx…yy, NoError	READ BINARY xx…yy, NoError	Daten werden ausgelesen
EF.PD read	-	SELECT DF.HCA NoError	nur Implementierung 1 weiß, dass *currentFolder* passend gesetzt ist
	SELECT EF.PD NoError	SELECT EF.PD NoError	*currentEF* wird passend gesetzt
	READ BINARY xx…yy, NoError	READ BINARY xx…yy, NoError	Daten werden ausgelesen
EF.VD read	-	SELECT DF.HCA NoError	nur Implementierung 1 weiß, dass *currentFolder* passend gesetzt ist
	SELECT EF.VD NoError	SELECT EF.VD NoError	*currentEF* wird passend gesetzt
	READ BINARY xx…yy, NoError	READ BINARY xx…yy, NoError	Daten werden ausgelesen

Die Anforderungen in diesem Dokument betreffen die Testerwartungshalt und sind so gewählt, dass beide Implementierungen diesbezüglich zulassungsfähig sind. Dem liegt folgendes Konzept zu Grunde:

1) Eine Testerwartungshaltung für die Kartenschnittstelle lässt sich nur auf Basis eines Funktionsaufrufes mit konkreten Parametern ermitteln, wobei hier als Einstieg nur die Funktionen aus Kapitel 6 relevant sind.
2) Kartenoperationen sind in der Reihenfolge auszuführen, die in diesem Dokument beschrieben ist.
3) Falls Kartenoperationen ausgeführt werden, die im Rahmen einer Aktion nicht beschrieben sind, dann wird das als zulassungsverhindernd gewertet.

4) Falls die Selektion eines Ordners, einer Datei oder eines Schlüsselobjektes gemäß diesem Dokument

   a) zwingend erforderlich ist, dann wird dies verpflichtend in die Testerwartungshaltung aufgenommen.

   b) deshalb nicht erforderlich ist, weil sie bereits zuvor passend erfolgte, dann wird eine nochmalige Selektion nicht als zulassungsverhindernd eingestuft.

5) Falls das Setzen eines Sicherheitszustandes gemäß diesem Dokument

   a) zwingend erforderlich ist, dann wird dies verpflichtend in die Testerwartungshaltung aufgenommen.

   b) nicht erforderlich ist, dann wird ein nochmaliges Setzen dieses Sicherheitszustandes als zulassungsverhindernd eingestuft (damit Benutzer nicht unnötig oft zur PIN-Eingabe aufgefordert werden).

6) Im Rahmen der Anpassung des Sicherheitsstatus ist die Abfrage eines Sicherheitsstatus zulässig und wird nicht als zulassungsverhindernd eingestuft.

5.2 Anforderungen

Der Einstieg in die Testerwartungshaltung ist der Aufruf einer Schnittstellenfunktion aus Kapitel 6.

⌦ **GS-A_5534 Testerwartungshaltung aus Funktionsaufruf**

Der Tester MUSS bei der Ermittlung einer Testerwartungshaltung für die Abfolge von Kommandonachrichten an der Schnittstelle zu einer Smartcard davon ausgehen, dass eine der folgenden Funktionen aufgerufen wurde:

(1) Funktion cardOperation gemäß Tabelle 11

(2) Funktion transparentChannel gemäß Tabelle 19

⌫

Die genaue Testerwartungshaltung ergibt sich daraus, dass nach dem Aufruf der Funktion diese so abgearbeitet wird, wie in diesem Dokument dargestellt. Das bedeutet, dass

1) die Reihenfolge einzuhalten ist,

2) notwendige Aktionen ausgeführt werden,

3) konditionale Aktionen, die aufgrund der Kondition überflüssig sind, toleriert werden,

4) keine zusätzlichen Aktionen ausgeführt werden.

⌦ **GS-A_5535 Testerwartungshaltung durch Abarbeiten der Funktion**

Der Tester MUSS bei der Ermittlung einer Testerwartungshaltung davon ausgehen, dass eine Funktion im Prüfling so abgearbeitet wird wie in den Kapiteln 6 und 9 dargestellt.⌫

GS-A_5536 Reihenfolge einhalten

Der Tester MUSS bei der Ermittlung einer Testerwartungshaltung davon ausgehen, dass Aktionen vom Prüfling in der Reihenfolge ausgeführt werden, die sich aus den Kapiteln 6 und 9 ergibt.

GS-A_5537 Notwendige Aktionen ausführen

Der Tester MUSS bei der Ermittlung einer Testerwartungshaltung davon ausgehen, dass Aktionen, die gemäß der Kapiteln 6 und 9 zwingend notwendig sind, vom Prüfling ausgeführt werden.

GS-A_5538 Konditionale Aktionen tolerieren

Der Tester MUSS bei der Ermittlung einer Testerwartungshaltung davon ausgehen, dass konditionale Aktionen, die ein Objekt selektieren, toleriert werden, wenn ein Prüfling diese ausführt, obwohl die Konditionen so gewählt sind, dass eine Ausführung gemäß der Kapitel 6 und 9 nicht vorgesehen ist.

Hinweis (3): Die Anforderung „Konditionale Aktionen tolerieren“ wird im Folgenden anhand eines Beispiels beschrieben. Angenommen eine transparente Datei EF.a sei in einer Smartcard als currentEF markiert und im Prüfling werde die Aktion cardOperation(EF.a, read) aufgerufen, siehe 9.2.6. Dann wird gemäß 9.2.6 Schritt 1 zunächst eine Aktion select ausgeführt. Dort ist in 9.2.7 Schritt 1 eine Kondition enthalten derzufolge der Prüfling nur dann ein oder mehrere SELECT Kommandos an die Smartcard schickt, falls currentEF im Prüfling (ja, currentEF im Prüfling, nicht currentEF in der Smartcard) nicht passend gesetzt ist. Falls der Prüfling currentEF nicht implementiert, wird er (überflüssigerweise) ein oder mehrere SELECT Kommandos an die Smartcard schicken, was gemäß der vorstehenden Anforderung in der Testerwartungshaltung zu tolerieren ist.

GS-A_5539 Sicherheitszustände nicht unnötig erneut setzen

Falls im Rahmen einer Aktion gemäß Kapitel 9 beschrieben ist, dass der Sicherheitszustand passend zu setzen ist, dann MUSS der Tester bei der Ermittlung einer Testerwartungshaltung davon ausgehen, dass ein bereits gesetzter Sicherheitszustand nicht erneut gesetzt wird.

GS-A_5540 Sicherheitszustände abfragen zulässig

Falls im Rahmen einer Aktion gemäß Kapitel 9 beschrieben ist, dass der Sicherheitszustand passend zu setzen ist, dann MUSS der Tester bei der Ermittlung einer Testerwartungshaltung davon ausgehen, dass die Abfrage eines Sicherheitszustandes entweder aus dem Baustein „Sicherheitsstatus“ (siehe Kapitel 3 Punkt (6) im Prüfling erfolgt, oder durch ein passendes Kartenkommando

(1) Aktion getStatus gemäß 9.4.7 für den Sicherheitsstatus von Passwortobjekten,

(2) Aktion getSecurityStatusFlagList gemäß 9.7.2 für den Sicherheitsstatus eines Flags, oder

(3) Aktion getSecurityStatusRole gemäß 9.7.3 für den Sicherheitsstatus einer Rolle.

6 Schnittstelle Card Proxy zu Anwendungen

Dieses Kapitel beschreibt die Schnittstelle zwischen der Komponente Card Proxy und Anwendungen, die Card Proxy nutzen um darüber auf Smartcards zuzugreifen.

Die hier behandelte Schnittstelle steht etwa Fachanwendungen zur Verfügung und hilft diesen mit Daten auf einer Smartcard zu arbeiten, die mit einem Kartenterminal verbunden ist. Die Beschreibung in diesem Kapitel geht von der Sichtweise eines Funktionsaufrufes aus. Die Fachanwendung ruft demzufolge eine Funktion der Schnittstelle auf und übergibt dabei abhängig von der gewünschten Aktion eine Reihe von Parametern. Die Komponente Card Proxy errechnet basierend auf den Inputparametern einen Rückgabewert, der typischerweise Daten und einen Status- oder Fehlercode enthält.

6.1 Funktion cardOperation, Überblick

Mit einem Objekt der Karte im Kartenterminal soll eine Aktion ausgeführt werden. Die Unterkapitel gehen spezifischer auf die erforderlichen Parameter und Rückgabewerte ein. Dabei gibt es pro Objekttyp (beispielsweise Ordner, transparente Datei, strukturierte (rekordorientierte) Datei, Passwortobjekt, Schlüssel, …) ein Unterkapitel.

Tabelle 11: Genereller Ablauf der Funktion cardOperation

Element	Beschreibung
Inputparameter	1) Identifikator des Objektes auf dem die angeforderte Aktion ausgeführt werden soll 2) Aktion, die auf dem adressierten Objekt ausgeführt werden soll 3) … (optional und je nach adressiertem Objekt und angeforderter Aktion kein, ein oder auch mehrere Zusatzparameter)
Vorbedingungen	1) Keine
Standardablauf	1) Falls StatusTransparentChannel = offen ist, bricht die Funktion mit der Fehlermeldung TransparentChannelOpen ab. 2) Es wird geprüft, ob das adressierte Objekt in der Tabelle SupportedAction vorhanden ist. Falls nicht, bricht die Funktion mit der Fehlermeldung UnknownObject ab. 3) Es wird geprüft, ob das adressierte Objekt die angeforderte Aktion laut Datenbank SupportedAction unterstützt. Falls nicht, bricht die Funktion mit der Fehlermeldung UnknownAction ab. 4) Je nach Typ des adressierten Objektes wird die weitere Bearbeitung beschrieben in a) Ordner gemäß [gemSpec_COS#8.3.1] Tabelle 12: Funktion cardOperation für Ordner b) Transparentes Elementary File gemäß [gemSpec_COS#8.3.2.1] Tabelle 13: Funktion cardOperation für transparente c) Strukturiertes Elementary File gemäß [gemSpec_COS#8.3.2.2] Tabelle 14: Funktion cardOperation für strukturiertes Elementary File d) Passwortobjekt gemäß [gemSpec_COS#8.4, 8.5] Tabelle 15: Funktion cardOperation für Passwortobjekte e) Symmetrisches Authentisierungsobjekt gemäß [gemSpec_COS#8.6.1] Dieser Typ wird in dieser Version des Dokumentes absichtlich nicht behandelt, weil es für ihn keinen Anwendungsfall gibt, der von Fachanwendungen der AdV-Umgebung direkt nutzbar ist. f) Symmetrisches Kartenverbindungsobjekt gemäß [gemSpec_COS#8.6.2] Dieser Typ wird in dieser Version des Dokumentes absichtlich nicht behandelt, weil es für ihn keinen Anwendungsfall gibt, der von Fachanwendungen der AdV-Umgebung direkt nutzbar ist. g) Privates Schlüsselobjekt gemäß [gemSpec_COS#8.6.3] Tabelle 16: Funktion cardOperation für private Schlüsselobjekte h) Öffentliches Schlüsselobjekt gemäß [gemSpec_COS#8.6.4]

Element	Beschreibung
	Tabelle 17: Funktion cardOperation für öffentliche Schlüsselobjekte i) „Wildcard“, dies ist kein Objekt im Sinne der zuvor behandelten Typen, sondern ein Platzhalter für die Aktionen, die keinem Objekt des Objektsystems zugeordnet werden können. Tabelle 18: Funktion cardOperation für Aktionen ohne zugeordnetes Objekt
Rückgabewert	1) TransparentChannelOpen: Die Funktion ist nicht ausführbar, solange der „transparente Kanal“ offen ist. 2) UnknownAction: Die angeforderte Aktion für das adressierte Objekt ist unbekannt. 3) UnknownObject: Das adressierte Objekt ist unbekannt. 4) Weitere Rückgabewerte sind abhängig vom adressierten Objekt und der angeforderten Aktion.

6.1.1 cardOperation für Ordner

Als Objekt wird ein Ordner gemäß [gemSpec_COS#8.3.1] adressiert. Die folgende Tabelle enthält lediglich einen Überblick über Aktionen für Ordner. Für die Nutzung der Schnittstelle zwischen Fachanwendungen und Card Proxy ist das hinreichend. Details, die für die Implementierung von Card Proxy relevant sind, werden in den referenzierten Unterkapiteln behandelt.

Hinweis (4): Absichtlich enthält Tabelle 12 keine Aktionen, die zu den folgenden Kommandos aus [gemSpec_COS] gehören:

a. FINGERPRINT, weil dies ein Kommando ist, welches im Rahmen der Kartenzulassung verwendet wird, für reguläre Use Cases aber keine Rolle spielt.

b. LOAD APPLICATION, weil dies eine administrative Aufgabe ist, die typischerweise über einen „transparenten Kanal“ abgewickelt wird.

Tabelle 12: Funktion cardOperation für Ordner

Element	Beschreibung	
Inputparameter	1) Identifikator des Ordners 2) Aktion aus der Menge: a) activate, keine weiteren Inputparameter, siehe 9.1.1 b) deactivate, keine weiteren Inputparameter, siehe 9.1.2 c) delete, keine weiteren Inputparameter, siehe 9.1.3 d) getSecureRandom, siehe 9.1.4 i) *length*, Anzahl zufälliger Oktette, die benötigt werden e) select, keine weiteren Inputparameter, siehe 9.1.5 f) terminate, keine weiteren Inputparameter, siehe 9.1.6	
Vorbedingungen	1) Der Standardablauf gemäß Tabelle 11: Genereller Ablauf der Funktion cardOperation hat die Punkte 1), 2) und 3) erfolgreich durchlaufen.	
Standardablauf	1) Falls erforderlich wird der Ordner selektiert. 2) Falls erforderlich, wird der Sicherheitszustand passend gesetzt. 3) Es wird eine passende Kommandosequenz erzeugt und an den Kartenleser geschickt. Die korrespondierende Antwortsequenz wird von dort empfangen.	
Rückgabewerte activate deactivate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	adressiertes Objekt nicht (de)aktivierbar, da terminiert
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11

Element	Beschreibung	
Rückgabewerte delete terminate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.9
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewert getSecureRand-om	CardTerminated	siehe 12.2.2
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.9
Rückgabewerte select	CardTerminated	siehe 12.2.2
	FileDeactivated	Ordner erfolgreich selektiert, Ordner ist deaktiviert
	OK	Ordner erfolgreich selektiert, Ordner ist aktiviert
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	Ordner erfolgreich selektiert, Ordner ist terminiert

Hinweis (5): Zusätzlich zu den vorgenannten Fehlermeldungen sind auch solche aus Tabelle 69 möglich, die im Rahmen der Anpassung des Sicherheitszustandes auftraten.

6.1.2 cardOperation für transparente Elementary Files

Als Objekt wird ein transparentes Elementary File gemäß [gemSpec_COS#8.3.2.1] adressiert. Die folgende Tabelle enthält lediglich einen Überblick über Aktionen für transparente Elementary Files. Für die Nutzung der Schnittstelle zwischen Fachanwendungen und Card Proxy ist das hinreichend. Details, die für die Implementierung von Card Proxy relevant sind, werden in den referenzierten Unterkapiteln behandelt.

Tabelle 13: Funktion cardOperation für transparentes Elementary File

Element	Beschreibung
Inputparameter	1) Identifikator des transparenten Elementary Files 2) Aktion aus der Menge: a) activate, keine weiteren Inputparameter, siehe 9.2.1 b) append, siehe 9.2.2 i) *newData*, Daten, die an den vorhandenen Dateiinhalt angehängt werden c) deactivate, keine weiteren Inputparameter, siehe 9.2.3 d) delete, keine weiteren Inputparameter, siehe 9.2.4 e) erase, siehe 9.2.5 i) *offset*, ab dem der Dateiinhalt gelöscht werden soll, optional, falls nicht vorhanden, wird *offset*=0 angenommen. f) read, siehe 9.2.6 i) *offset*. ab dem der Dateiinhalt gelesen werden soll, optional, falls nicht vorhanden, wird *offset*=0 angenommen. ii) *length*, Anzahl der Oktette, die gelesen werden sollen, optional, falls nicht vorhanden, wird der Dateiinhalt bis zum Datenende gelesen. g) select, keine weiteren Inputparameter, siehe 9.2.7 h) setLogicalEndOfFile, siehe 9.2.8 i) *offset*, auf den das Ende des Dateninhalts gesetzt werden soll, optional, falls nicht vorhanden, wird *offset*=0 angenommen. i) terminate, keine weiteren Inputparameter, siehe 9.2.9 j) update, siehe 9.2.10 i) *offset*. ab dem der Dateiinhalt ersetzt werden soll, optional, falls nicht vorhanden, wird *offset*=0 angenommen. ii) *newData*, Daten, die den vorhandenen Dateiinhalt ersetzen sollen

Element	Beschreibung	
Vorbedingungen	1) Der Standardablauf gemäß Tabelle 11: Genereller Ablauf der Funktion cardOperation hat die Punkte 1), 2) und 3) erfolgreich durchlaufen.	
Standardablauf	1) Falls erforderlich wird der Ordner selektiert, der das von der Aktion betroffene Elementary File enthält. 2) Falls erforderlich, wird das von der Aktion betroffene Elementary File selektiert. 3) Falls erforderlich, wird der Sicherheitszustand passend gesetzt. 4) Es wird eine passende Kommandosequenz erzeugt und an den Kartenleser geschickt. Die korrespondierende Antwortsequenz wird von dort empfangen.	
Rückgabewerte activate deactivate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	adressiertes Objekt nicht (de)aktivierbar, da terminiert
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte delete terminate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte erase setLogical-EndOfFile	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	OffsetTooBig	Inputparameter *offset* ist zu groß.
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte read	CardTerminated	siehe 12.2.2
	CorruptDataWarning plus angeforderte Daten	siehe 12.2.3
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	OffsetTooBig	Inputparameter *offset* ist zu groß.
	SecurityStatusNotSatisfied	siehe 12.2.10
Rückgabewerte select	CardTerminated	siehe 12.2.2
	FileDeactivated	Datei erfolgreich selektiert, Datei ist deaktiviert
	OK	Datei erfolgreich selektiert, Datei ist aktiviert
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	Datei erfolgreich selektiert, Datei ist terminiert
Rückgabewerte update	CardTerminated	siehe 12.2.2
	DataTooBig	Inputparameter *newData* enthält zu viele Oktette.
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	OffsetTooBig	Inputparameter *offset* ist zu groß.

Element	Beschreibung	
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte write	CardTerminated	siehe 12.2.2
	DataTooBig	Inputparameter *newData* enthält zu viele Oktette.
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11

Hinweis (6): Zusätzlich zu den vorgenannten Fehlermeldungen sind auch solche aus Tabelle 69 möglich, die im Rahmen der Anpassung des Sicherheitszustandes auftraten.

6.1.3 cardOperation für strukturierte Elementary Files

Als Objekt wird ein strukturiertes Elementary File gemäß [gemSpec_COS#8.3.2.2] adressiert. Die folgende Tabelle enthält lediglich einen Überblick über Aktionen für strukturierte Elementary Files. Für die Nutzung der Schnittstelle zwischen Fachanwendungen und Card Proxy ist das hinreichend. Details, die für die Implementierung von Card Proxy relevant sind, werden in den referenzierten Unterkapiteln behandelt.

Tabelle 14: Funktion cardOperation für strukturiertes Elementary File

Element	Beschreibung
Inputparameter	1) Identifikator des strukturierten Elementary Files 2) Aktion aus der Menge: a) activate, keine weiteren Inputparameter, siehe 9.3.1 b) activateRecord, siehe 9.3.2 i) *recordNumber*, Nummer des zu aktivierenden Listenelementes, optional, falls nicht vorhanden, werden alle Rekords aktiviert. c) append, siehe 9.3.3 i) *recordData* mit den Daten des neuen Rekords d) deactivate, keine weiteren Inputparameter, siehe 9.3.4 e) deactivateRecord, siehe 9.3.5 i) *recordNumber*, Nummer des zu deaktivierenden Listenelementes, optional, falls nicht vorhanden, werden alle Rekords deaktiviert. f) delete, keine weiteren Inputparameter, siehe 9.3.6 g) deleteRecord, siehe 9.3.7 i) *recordNumber*, Nummer des zu löschenden Listenelementes, optional, falls nicht vorhanden, werden alle Rekords gelöscht. h) erase, siehe 9.3.8 i) *recordNumber*, Nummer des Listenelementes, dessen Inhalt zu löschen ist, optional, falls nicht vorhanden, wird der Inhalt aller Rekords gelöscht. i) read, siehe 9.3.9 i) *recordNumber*, Liste mit Rekordnummern, optional, falls nicht vorhanden, werden alle Rekords gelesen. j) search, siehe 9.3.10 i) *recordNumber*, Nummer des Rekords, der als erstes durchsucht wird, optional, falls nicht vorhanden, werden alle Rekords durchsucht ii) *searchString*, Muster, nach dem in den Rekords gesucht wird k) select, keine weiteren Inputparameter, siehe 9.3.11 l) terminate, keine weiteren Inputparameter, siehe 9.3.12 m) update, siehe 9.3.13 i) *recordNumber*. Nummer des Rekords, der überschrieben werden soll ii) *newData*, Daten, die den vorhandenen Rekordinhalt ersetzen sollen

Element	Beschreibung	
Vorbedingungen	1) Der Standardablauf gemäß Tabelle 11: Genereller Ablauf der Funktion cardOperation hat die Punkte 1), 2) und 3) erfolgreich durchlaufen.	
Standardablauf	1) Falls erforderlich wird der Ordner selektiert, der das von der Aktion betroffene Elementary File enthält. 2) Falls erforderlich, wird das von der Aktion betroffene Elementary File selektiert. 3) Falls erforderlich, wird der Sicherheitszustand passend gesetzt. 4) Es wird eine passende Kommandosequenz erzeugt und an den Kartenleser geschickt. Die korrespondierende Antwortsequenz wird von dort empfangen.	
Rückgabewerte activate deactivate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	adressiertes Objekt nicht (de)aktivierbar, da terminiert
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte activateRecord deactivateRecord	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	RecordNotFound	adressierter Rekord existiert nicht
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte append	BufferTooSmall	siehe 12.2.1
	CardTerminated	siehe 12.2.2
	FullRecordList	Rekordliste lässt keine weiteren Elemente zu
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	OutOfMemoryError	zu viele Oktette in *recordData*
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
	WrongRecordLength	*recordData* hat nicht die richtige Länge
Rückgabewerte delete	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte deleteRecord erase	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	RecordDeactivated	Rekord(inhalt) nicht löschbar, da Rekord deaktiviert
	RecordNotFound	adressierter Rekord existiert nicht
	SecurityStatusNotSatisfied	siehe 12.2.10

Element	Beschreibung	
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte read List mit folgenden Elementen	CardTerminated	siehe 12.2.2
	CorruptDataWarning plus angeforderte Daten	siehe 12.2.3
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	RecordDeactivated	Rekordinhalt nicht lesbar, da Rekord deaktiviert
	RecordNotFound	adressierter Rekord existiert nicht
	SecurityStatusNotSatisfied	siehe 12.2.10
Rückgabewerte search	BufferTooSmall	siehe 12.2.1
	CardTerminated	siehe 12.2.2
	CorruptDataWarning plus angeforderte Daten	siehe 12.2.3
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	RecordNotFound	adressierter Rekord existiert nicht
	SecurityStatusNotSatisfied	siehe 12.2.10
Rückgabewerte select	CardTerminated	siehe 12.2.2
	FileDeactivated	Datei erfolgreich selektiert, Datei ist deaktiviert
	OK	Datei erfolgreich selektiert, Datei ist aktiviert
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	Datei erfolgreich selektiert, Datei ist terminiert
Rückgabewerte terminate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte update	BufferTooSmall	siehe 12.2.1
	CardTerminated	siehe 12.2.2
	ObjectNotFound	siehe 12.2.9
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	OutOfMemoryError	zu viele Oktette in *newData*
	RecordDeactivated	Rekordinhalt nicht änderbar, da Rekord deaktiviert
	RecordNotFound	adressierter Rekord existiert nicht
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
	WrongRecordLength	*newData* hat nicht die richtige Länge

Hinweis (7): Zusätzlich zu den vorgenannten Fehlermeldungen sind auch solche aus Tabelle 69 möglich, die im Rahmen der Anpassung des Sicherheitszustandes auftraten.

6.1.4 cardOperation für Passwortobjekte

Als Objekt wird ein reguläres-Passwortobjekt gemäß [gemSpec_COS#8.4] oder ein Multireferenz-Passwortobjekt gemäß [gemSpec_COS#8.5] adressiert. Die folgende Tabelle enthält lediglich einen Überblick über Aktionen für Passwortobjekte. Für die Nutzung der Schnittstelle zwischen Fachanwendungen und Card Proxy ist das hinreichend. Details, die für die Implementierung von Card Proxy relevant sind, werden in den referenzierten Unterkapiteln behandelt.

Tabelle 15: Funktion cardOperation für Passwortobjekte

Element	Beschreibung	
Inputparameter	1) Identifikator des Passwortobjektes 2) Aktion aus der Menge: a) activate, keine weiteren Inputparameter, siehe 9.4.1 b) change, siehe 9.4.2 i) *mode*, Modus zur PIN-Änderung, optional, falls nicht vorhanden, wird der Default-Mode aus der Datenbank verwendet, mögliche Werte sind: (1) replace: Der Benutzer gibt den aktuellen PIN-Wert und den neuen PIN-Wert ein. (2) set: Der Benutzer gibt lediglich den neuen PIN-Wert ein, der aktuelle PIN-Wert (sofern gesetzt) wird nicht benötigt. c) deactivate, keine weiteren Inputparameter, siehe 9.4.3 d) delete, keine weiteren Inputparameter, siehe 9.4.4 e) disable, keine weiteren Inputparameter, siehe 9.4.5 f) enable, keine weiteren Inputparameter, siehe 9.4.6 g) getStatus, keine weiteren Inputparameter, siehe 9.4.7 h) terminate, keine weiteren Inputparameter, siehe 9.4.8 i) unblock, siehe 9.4.9 i) *mode*, Modus zur Aufhebung der PIN-Blockade, optional, falls nicht vorhanden, wird der Default-Modus aus der Datenbank verwendet, mögliche Werte sind: (1) UnblockWithPukAndSet: Die Blockade wird mittels PUK aufgehoben und es wird ein neuer Wert für die PIN gesetzt. (2) UnblockWithPuk: Die Blockade wird mittels PUK aufgehoben. Es wird kein neuer Wert für die PIN gesetzt. (3) UnblockAndSet: Die Blockade wird ohne PUK aufgehoben. Es wird ein neuer Wert für die PIN gesetzt. (4) Unblock: Die Blockade wird ohne PUK aufgehoben. Es wird kein neuer Wert für die PIN gesetzt. j) verify, keine weiteren Inputparameter, siehe 9.4.10	
Vorbedingungen	1) Der Standardablauf gemäß Tabelle 11: Genereller Ablauf der Funktion cardOperation hat die Punkte 1), 2) und 3) erfolgreich durchlaufen.	
Standardablauf	1) Falls erforderlich wird der Ordner selektiert, der das von der Aktion betroffene Passwortobjekt enthält. 2) Falls erforderlich, wird der Sicherheitszustand passend gesetzt. 3) Es wird eine passende Kommandosequenz erzeugt und an den Kartenleser geschickt. Die korrespondierende Antwortsequenz wird von dort empfangen.	
Rückgabewerte activate deactivate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	adressiertes Objekt nicht (de)aktivierbar, da terminiert
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte change	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8

Element	Beschreibung	
	OK	Erfolgreiches Setzen der PIN auf neuen Wert
	ObjectNotFound	siehe 12.2.9
	PasswordBlocked	Passwortobjekt blockiert
	SecurityStatusNotSatisfied	siehe 12.2.10
	WrongLength	PIN nicht geändert, da neuer Wert falsche Länge besitzt
	WrongSecretWarning.X	keine Änderung, alter Wert falsch, noch X Versuche
Rückgabewerte delete terminate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte disable enable verify	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	PasswordBlocked	Passwortobjekt blockiert
	PasswordProtected	Passwortobjekt mit aktivem Transportschutz
	SecurityStatusNotSatisfied	siehe 12.2.10
	WrongSecretWarning.X	Benutzerverifikation nicht erfolgreich, noch X Versuche
Rückgabewerte getStatus	CardTerminated	siehe 12.2.2
	OK	kein Transportschutz Passwortschutz eingeschaltet Sicherheitszustand aktuell gesetzt
	ObjectNotFound	siehe 12.2.9
	PasswordDisabled	Passwortschutz ausgeschaltet Sicherheitszustand stets gesetzt
	PasswordProtected	aktiver Transportschutz Passwortschutz eingeschaltet Sicherheitszustand nicht gesetzt
	RetryCounter.X	kein Transportschutz Passwortschutz eingeschaltet, Sicherheitszustand nicht gesetzt noch X Versuche für eine Benutzerverifikation
	SecurityStatusNotSatisfied	siehe 12.2.10
Rückgabewerte unblock	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	PasswordBlocked	Bedienzähler der PUK abgelaufen
	PasswordProtected	Passwortobjekt mit aktivem Transportschutz
	SecurityStatusNotSatisfied	siehe 12.2.10
	WrongLength	Blockade besteht fort, neuer Wert besitzt falsche Länge
	WrongSecretWarning.X	Blockade besteht fort, PUK falsch, noch X Versuche

Hinweis (8): Zusätzlich zu den vorgenannten Fehlermeldungen sind auch solche aus Tabelle 69 möglich, die im Rahmen der Anpassung des Sicherheitszustandes auftraten.

6.1.5 cardOperation für private Schlüsselobjekte

Als Objekt wird ein privates Schlüsselobjekt gemäß [gemSpec_COS#8.6.3] adressiert. Die folgende Tabelle enthält lediglich einen Überblick über Aktionen für private Schlüsselobjekte. Für die Nutzung der Schnittstelle zwischen Fachanwendungen und Card Proxy ist das hinreichend. Details, die für die Implementierung von Card Proxy relevant sind, werden in den referenzierten Unterkapiteln behandelt.

Hinweis (9): Aktionen, die mit dem Schlüsselmaterial arbeiten, verwenden dazu den algorithmIdentifer *gemäß [gemSpec_COS#Tab.268]. Folgende* algorithmIdentifier*, die in [gemSpec_COS] für private Schlüsselobjekte definiert sind, fehlen absichtlich in der unten stehenden Tabelle:*

a. Fachanwendungen der AdV-Umgebung führen administrative Aufgaben nicht direkt durch:
- *elcAsynchronAdmin*

b. Eine AdV-Umgebung baut niemals selbst einen Trusted Channel auf.
- *elcSessionkey4SM*
- *elcSessionkey4TC*
- *rsaSessionkey4SM*
- *rsaSessionkey4TC*

Hinweis (10): Im Rahmen von Signaturoperationen geht diese Version des Dokumentes davon aus, dass die gesamte zu signierende Nachricht dem Card Proxy übergeben wird und als Hashverfahren stets SHA-256 verwendet wird. Die Signaturoperationen geben dann im Erfolgsfall lediglich die Signatur als Ergebnis der Berechnung mit dem privaten Schlüssel zurück.

Tabelle 16: Funktion cardOperation für private Schlüsselobjekte

Element	Beschreibung
Inputparame**ter**	1) Identifikator des privaten Schlüsselobjektes 2) Aktion aus der Menge: a) activate, keine weiteren Inputparameter, siehe 9.5.1 b) deactivate, keine weiteren Inputparameter, siehe 9.5.2 c) delete, keine weiteren Inputparameter, siehe 9.5.3 d) elcRoleAuthentication, siehe 9.5.4 i) *token*, zu authentisierende Daten, stets 24 Oktett lang e) elcSharedSecretCalculation, siehe 9.5.5 i) *cryptogram*, zu entschlüsselnde Daten ii) *publicKeyInformation,* Informationen zur Verschlüsselung, optional, falls nicht vorhanden wird das Kryptogramm lediglich entschlüsselt, falls vorhanden, wird die Klartextnachricht im Kryptogramm umgeschlüsselt f) generate, siehe 9.5.6 i) *mode*, Modus der Schlüsselgenerierung, optional, falls nicht vorhanden, wird der Default-Mode aus der Datenbank verwendet mögliche Werte sind: (1) create+read => P1='80': Schlüsselgenerierung, falls kein Schlüssel vorhanden, Ausgabe (2) create => P1='84': Schlüsselgenerierung, falls kein Schlüssel vorhanden, keine Ausgabe (3) replace+read => P1='C0': Schlüsselgenerierung, ggf. Über-

Element	Beschreibung	
	schreiben, Ausgabe (4) replace => P1='C4':Schlüsselgenerierung, ggf. Überschreiben, keine Ausgabe g) readPublicPart, keine weiteren Inputparameter, siehe 9.5.7 h) rsaClientAuthentication, siehe 9.5.8 i) *token*, eine bis zu 64 Oktette lange Zeichenkette, die signiert wird i) rsaDecipherOaep, siehe 9.5.9 i) *cryptogram*, zu entschlüsselnde Daten ii) *publicKeyInformation,* Informationen zur Verschlüsselung, optional, falls nicht vorhanden wird das Kryptogramm lediglich entschlüsselt, falls vorhanden, wird die Klartextnachricht im Kryptogramm umgeschlüsselt j) rsaDecipherPKCS1_V1_5, siehe 9.5.10 i) *cryptogram*, zu entschlüsselnde Daten ii) *publicKeyInformation,* Informationen zur Verschlüsselung, optional, falls nicht vorhanden wird das Kryptogramm lediglich entschlüsselt, falls vorhanden, wird die Klartextnachricht im Kryptogramm umgeschlüsselt k) rsaRoleAuthentication, siehe 9.5.11 i) *token*, zu authentisierende Daten, stets 16 Oktett lang l) sign9796_2_DS2, siehe 9.5.12 i) *message*, zu signierende Nachricht m) signECDSA, siehe 9.5.13 i) *message*, zu signierende Nachricht n) signPKCS1_V1_5, siehe 9.5.14 i) *message*, zu signierende Nachricht o) signPSS, siehe 9.5.15 i) *message*, zu signierende Nachricht p) terminate, keine weiteren Inputparameter, siehe 9.5.16	
Vorbedingungen	1) Der Standardablauf gemäß Tabelle 11: Genereller Ablauf der Funktion cardOperation hat die Punkte 1), 2) und 3) erfolgreich durchlaufen.	
Standardablauf	1) Falls erforderlich wird der Ordner selektiert, der das von der Aktion betroffene private Schlüsselobjekt enthält. 2) Falls erforderlich, wird der Sicherheitszustand passend gesetzt. 3) Es wird eine passende Kommandosequenz erzeugt und an den Kartenleser geschickt. Die korrespondierende Antwortsequenz wird von dort empfangen.	
Rückgabewerte activate deactivate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	adressiertes Objekt nicht (de)aktivierbar, da terminiert
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte delete terminate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11

Element	Beschreibung	
Rückgabewerte elcRoleAuth. rsaClientAuth. rsaRoleAuth.	CardTerminated	siehe 12.2.2
	KeyInvalid	siehe 12.2.7
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	WrongToken	fehlerhaftes *token*
Rückgabewerte elcShared- SecretCalculation rsaDecipherOaep rsaDecipher PKCS1_V1_5	CardTerminated	siehe 12.2.2
	KeyInvalid	siehe 12.2.7
	OK plus entschlüsselte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	WrongCiphertext	Das Kryptogramm konnte nicht entschlüsselt werden.
	WrongCryptogram	Der verschlüsselte Text ist fehlerhaft.
Rückgabewerte generate	CardTerminated	siehe 12.2.2
	KeyAlreadyPresent	Der gewählte Modus gestattet das Überschreiben nicht.
	MemoryFailure	siehe 12.2.8
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte readPublicPart	CardTerminated	siehe 12.2.2
	KeyInvalid	siehe 12.2.7
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
Rückgabewerte signECDSA signPKCS1_V1.5 signPSS	CardTerminated	siehe 12.2.2
	KeyInvalid	siehe 12.2.7
	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10

Hinweis (11): Zusätzlich zu den vorgenannten Fehlermeldungen sind auch solche aus Tabelle 69 möglich, die im Rahmen der Anpassung des Sicherheitszustandes auftraten.

6.1.6 cardOperation für öffentliche Schlüsselobjekte

Als Objekt wird ein öffentliches Schlüsselobjekt gemäß [gemSpec_COS#8.6.4] adressiert. Die folgende Tabelle enthält lediglich einen Überblick über Aktionen für private Schlüsselobjekte. Für die Nutzung der Schnittstelle zwischen Fachanwendungen und Card Proxy ist das hinreichend. Details, die für die Implementierung von Card Proxy relevant sind, werden in den referenzierten Unterkapiteln behandelt.

Hinweis (12): Aktionen, die mit dem Schlüsselmaterial arbeiten, verwenden dazu den algorithmIdentifer *gemäß [gemSpec_COS#Tab.268]. Folgende* algorithmIdentifier*, die in [gemSpec_COS] für öffentliche Schlüsselobjekte definiert sind, fehlen absichtlich in der unten stehenden Tabelle:*

a. *Signaturprüfungen im Rahmen des Importes von CV-Zertifikaten werden an der äußeren Schnittstelle von Card Proxy nicht benötigt. Der Import von CV-Zertifikaten wird bei Bedarf von Card Proxy veranlasst:*
   - *sigS_ISO9796-2Withrsa_sha256*
   - *ecdsa-with-SHA256*
   - *ecdsa-with-SHA384*
   - *ecdsa-with-SHA512*

b. *Externe Authentisierungen werden an der äußeren Schnittstelle von Card Proxy nicht benötigt, sondern bei Bedarf intern veranlasst.*
   - *autS_ISO9796-2Withrsa_sha256_mutual*
   - *authS_gemSpec-COS-G2_ecc-with-sha256*
   - *authS_gemSpec-COS-G2_ecc-with-sha384*
   - *authS_gemSpec-COS-G2_ecc-with-sha512*

Tabelle 17: Funktion cardOperation für öffentliche Schlüsselobjekte

Element	Beschreibung	
Inputparameter	1) Identifikator des privaten Schlüsselobjektes 2) Aktion aus der Menge: a) activate, keine weiteren Inputparameter, siehe 9.6.1 b) deactivate, keine weiteren Inputparameter, siehe 9.6.2 c) delete, keine weiteren Inputparameter, siehe 9.6.3 d) elcSharedSecretCalculation, siehe 9.6.4 i) *plainText*, zu verschlüsselnde Klartextnachricht e) rsaEncipherOaep, siehe 9.6.5 i) *plainText*, zu verschlüsselnde Klartextnachricht f) rsaEncipherPKCS1_V1_5, siehe 9.6.6 i) *plainText*, zu verschlüsselnde Klartextnachricht g) terminate, keine weiteren Inputparameter, siehe 9.6.7	
Vorbedingungen	1) Der Standardablauf gemäß Tabelle 11: Genereller Ablauf der Funktion cardOperation hat die Punkte 1), 2) und 3) erfolgreich durchlaufen.	
Standardablauf	1) Falls erforderlich wird der Ordner selektiert, der das von der Aktion betroffene private Schlüsselobjekt enthält. 2) Falls erforderlich, wird der Sicherheitszustand passend gesetzt. 3) Es wird eine passende Kommandosequenz erzeugt und an den Kartenleser geschickt. Die korrespondierende Antwortsequenz wird von dort empfangen.	
Rückgabewerte activate deactivate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt
	ObjectNotFound	siehe 12.2.9
	ObjectTerminated	adressiertes Objekt nicht (de)aktivierbar, da terminiert
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11
Rückgabewerte delete terminate	CardTerminated	siehe 12.2.2
	MemoryFailure	siehe 12.2.8
	OK	Aktion erfolgreich durchgeführt

Element	Beschreibung	
	ObjectNotFound	siehe 12.2.9
	SecurityStatusNotSatisfied	siehe 12.2.10
	UpdateRetryWarning	Aktion erfolgreich durchgeführt, siehe 12.2.11

Hinweis (13): Zusätzlich zu den vorgenannten Fehlermeldungen sind auch solche aus Tabelle 69 möglich, die im Rahmen der Anpassung des Sicherheitszustandes auftraten.

6.1.7 cardOperation ohne zugeordnetes Objekt

In diesem Unterkapitel werden Aktionen behandelt, die keinem Objekt des Objektsystems zugeordnet werden können und deshalb in den vorangegangenen Unterkapiteln unberücksichtigt blieben. Allen hier behandelten Aktionen ist gemeinsam, dass als Identifikator in den Inputparametern der Wert „Wildcard“ verwendet wird.

Hinweis (14): Absichtlich enthält Tabelle 18 keine Aktionen, die zu den folgenden Kommandos aus [gemSpec_COS] gehören:

a. LIST PUBLIC KEY, weil dies ein Kommando ist, welches im Rahmen der Kartenzulassung verwendet wird, für reguläre Use Cases aber keine Rolle spielt.

b. MANAGE SECURITY ENVIRONMENT, weil dieses Kommando im Rahmen einer Smartcard Nutzung zur Schlüsselauswahl verwendet wir, welche Teil der übrigen Aktionen ist.

Tabelle 18: Funktion cardOperation für Aktionen ohne zugeordnetes Objekt

Element	Beschreibung	
Inputparame**ter**	1) Identifikator = „Wildcard“ 2) Aktion aus der Menge: a) getRandom, siehe 9.7.1 i) *length*, Anzahl zufälliger Oktette, die benötigt werden b) getSecurityFlagList, siehe 9.7.2 i) *oid*, der Flagliste, für welche der Sicherheitsstatus erfragt wird, ii) *flagList*, Flagliste, für welche der Sicherheitsstatus erfragt wird. c) getSecurityFlagRole, siehe 9.7.3 i) *role*, Rolle, für welche der Sicherheitsstatus erfragt wird. d) resetChannel, keine weiteren Inputparameter, siehe 9.7.4	
Vorbedingungen	1) Der Standardablauf gemäß Tabelle 11: Genereller Ablauf der Funktion cardOperation hat die Punkte 1), 2) und 3) erfolgreich durchlaufen.	
Standardablauf	*Hinweis (15): Eine Ordnerselektion findet niemals statt.* *Hinweis (16): Eine Anpassung des Sicherheitszustands ist niemals erforderlich.* 1) Es wird eine passende Kommandosequenz erzeugt und an den Kartenleser geschickt. Die korrespondierende Antwortsequenz wird von dort empfangen.	
Rückgabewert getRandom	OK plus angeforderte Daten	Aktion erfolgreich durchgeführt
Rückgabewert getSecurityStatus-FlagList	NoAuthentication	Erfragter Authentisierungsstatus ist nicht gesetzt

Element	Beschreibung	
getSecurityStauts-Role	OK	Erfragter Authentisierungsstatus ist gesetzt
Rückgabewert resetChannel	OK	Aktion erfolgreich durchgeführt

Hinweis (17): Da der Sicherheitszustand für die hier behandelten Aktionen irrelevant ist, treten die Fehlermeldungen aus Tabelle 69 hier nicht auf.

6.2 Funktion transparentChannel

Diese Funktion ermöglicht die Nutzung eines transparenten Kanals zur Smartcard. Ein transparenter Kanal lässt sich öffnen, nutzen und schließen. Solange ein transparenter Kanal offen ist, ist die Funktion cardOperation nicht nutzbar.

Tabelle 19: Funktion transparentChannel

Element	Beschreibung
Inputparameter	1) Aktion aus der Menge a) open, öffnet einen „transparenten Kanal“, falls er geschlossen ist b) sendAPDU, sendet die in Parameter 2 angegebene Kommando APDU an die Karte im Kartenterminal c) close, schließt einen geöffneten „transparenten Kanal“ 2) commandAPDU, konditional, fehlt für die Aktionen open und close, ist bei der Aktion sendAPDU zwingend vorhanden
Vorbedingungen	1) Keine
Standardablauf	1) Falls der „transparente Kanal“ geöffnet ist, bricht die Funktion für die Aktion open mit der Fehlermeldung TransparentChannelAlreadyOpen ab. 2) Falls der „transparente Kanal“ geschlossen ist, bricht die Funktion für die Aktionen sendAPDU und close mit der Fehlermeldung TransparentChannelNotOpen ab. 3) Falls für die Aktion sendAPDU kein Parameter commandAPDU angegeben ist, bricht die Funktion mit der Fehlermeldung MissingAPDU ab. 4) Falls die Aktion a) open ist, wird ein Kanalreset gemäß Aktion resetChannel (siehe Tabelle 18) ausgeführt und StatusTransparentChannel auf open gesetzt. b) sendAPDU ist, dann wird die Kommando APDU an die Karte im Kartenterminal gesendet und die Antwort APDU von dort empfangen. c) close ist, wird ein Kanalreset gemäß resetChannel (siehe Tabelle 18) ausgeführt und StatusTransparentChannel auf closed gesetzt. 5) Die Funktion wird beendet und der Rückgabewert OK wird zurückgemeldet.
Rückgabewert	1) TransparentChannelAlreadyOpen: Der „transparente Kanal“ ist bereits offen. 2) TransparentChannelNotOpen: Der „transparente Kanal“ ist nicht offen. 3) MissingAPDU: Es fehlt eine Kommando APDU. 4) OK: Die Aktion wurde erfolgreich ausgeführt plus Antwort APDU im Falle der Aktion sendAPDU

7 Schnittstelle Card Proxy und Kartenleser

Diese Schnittstelle wird in [gemSpec_KTR-AdV#6.2.2] beschrieben.

8 Konfigurationstabelle Card Proxy

Dieses Kapitel beschreibt beispielhaft die Tabellen innerhalb von Card Proxy, welche die Objekte und Aktionen enthalten, die produkttypspezifisch ausführbar sind. Die Motivation für diese Art Tabelle ergibt sich aus 2.1 Punkt (1) und folgende.

8.1 Konfigurationstabelle Card Proxy eGK G1

Dieses Kapitel wird bei Bedarf in einer späteren Version des Dokumentes ergänzt.

8.2 Konfigurationstabelle Card Proxy eGK G2 aus AdV-Sicht

Die hier dargestellten Tabellen basieren auf [gemSpec_eGK_ObjSys]. Der hier dargestellte Auszug aus [gemSpec_eGK_ObjSys] wurde unter dem Blickwinkel KTR-AdV verfasst. In anderen Umgebungen (etwa @home, oder in der Leistungserbringerumgebung eines Arztes) gäbe es inhaltlich abweichende Konfigurationstabellen.

Hinweis (18): Derzeit wurden die Informationen dieses Unterkapitels manuell erstellt. Es ist geplant derartige Informationen demnächst automatisiert aus den Objektsystemspezifikationen zu extrahieren. Dann wird es auch einfach möglich sein diese Informationen für andere Umgebungen schnell und zuverlässig zu erstellen.

Hinweis (19): [gemSpec_eGK_ObjSys] spezifiziert den Produkttyp eGK in der Produkttypversion 4.3.2, auch bekannt als G2 eGK, oder G2.0 eGK. Davon zu unterscheiden ist die G2.1 eGK, deren Konfigurationstabellen abweichende Inhalte aufweisen.

Die folgenden Tabellen sind so angeordnet, dass der Pfad alphabetisch aufsteigend sortiert ist. Dabei sind nur die Objekte aufgelistet, für die fachliche Use Cases existieren.

8.2.1 Konfigurationstabellen für Objekte im MF

Pfad Referenz	/ MF [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_006]	
Typ und Identifikator	Ordner	MF
Ort	*applicationIdentifier* = 'D276 0001 44 8000'	
Aktion select	ALWAYS	

Hinweis (20): / MF / EF.ATR fehlt.
Hinweis (21): / MF / EF.CardAccess fehlt.
Hinweis (22): / MF / EF.C.CA_eGK.CS.E256 fehlt.
Hinweis (23): / MF / EF.C.eGK_AUT_CVC_E256 fehlt.
Hinweis (24): / MF / EF.DIR fehlt.
Hinweis (25): / MF / EF.GDO fehlt.
Hinweis (26): / MF / EF.Version fehlt.
Hinweis (27): / MF / EF.Version2 fehlt.
Hinweis (28): / MF / MRPIN.home fehlt.

Pfad Referenz	/ MF / PIN.CH [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_017]	
Typ und Identifikator	Passwortobjekt	PIN.CH
Ort	*passwordReference* = '01'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS	
Aktion verify	ALWAYS	

Hinweis (29): / MF / PrK.eGK.AUT_CVC.E256 fehlt.
Hinweis (30): / MF / PuK.RCA.ADMINCMS.CS.E256 fehlt.
Hinweis (31): / MF / PuK.RCA.CS.E256 fehlt.
Hinweis (32): / MF / SK.CMS.AES128 fehlt.
Hinweis (33): / MF / SK.CMS.AES256 fehlt.
Hinweis (34): / MF / SK.CMS.VSD128 fehlt.
Hinweis (35): / MF / SK.CMS.VSD256 fehlt.
Hinweis (36): / MF / SK.CAN fehlt.

8.2.2 Konfigurationstabellen für Objekte im DF.HCA

Pfad Referenz	/ MF / DF.HCA [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_033]	
Typ und Identifikator	Ordner	DF.HCA
Ort	*applicationIdentifier* ='D276 0000 01 02'	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.Einwilligung [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_034]	
Typ und Identifikator	strukturiertes Elementary File	EF.Einwilligung
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D005'	
Aktion activateRecord	PWD(PIN.CH) AND flagTI.24	
Aktion deactivateRecord	PWD(PIN.CH) AND flagTI.24	
Aktion deleteRecord	PWD(PIN.CH) AND flagTI.25	
Aktion erase	PWD(PIN.CH) AND flagTI.25	
Aktion read	PWD(PIN.CH) AND flagTI.25	
Aktion search	PWD(PIN.CH) AND flagTI.25	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.GVD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_035]	
Typ und Identifikator	transparentes Elementary File	EF.GVD
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D003'	
Aktion read	PWD(PIN.CH) AND flagTI.29	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.Logging [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_036]	
Typ und Identifikator	strukturiertes Elementary File	EF.Logging
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D006'	
Aktion append	PWD(PIN.CH) AND flagTI.31	
Aktion read	PWD(PIN.CH) AND flagTI.33	
Aktion search	PWD(PIN.CH) AND flagTI.33	

Aktion select	ALWAYS

Pfad Referenz	/ MF / DF.HCA / EF.PD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_037]	
Typ und Identifikator	transparentes Elementary File	EF.PD
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D001'	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.Prüfungsnachweis [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_038]	
Typ und Identifikator	transparentes Elementary File	EF.Prüfungsnachweis
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D01C'	
Aktion erase	ALWAYS	
Aktion read	ALWAYS	
Aktion setLogicalEndOfFile	ALWAYS	
Aktion select	ALWAYS	
Aktion update	ALWAYS	
Aktion write	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.Standalone [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_039]	
Typ und Identifikator	transparentes Elementary File	EF.Standalone
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D00A'	
Aktion erase	ALWAYS	
Aktion read	ALWAYS	
Aktion setLogicalEndOfFile	ALWAYS	
Aktion select	ALWAYS	
Aktion update	ALWAYS	
Aktion write	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.StatusVD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_040]	
Typ und Identifikator	transparentes Elementary File	EF.StatusVD
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D00C'	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.TTN [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_041]	
Typ und Identifikator	strukturiertes Elementary File	EF.TTN
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D00F'	
Aktion read	PWD(PIN.CH) AND flagTI.35	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.VD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_042]	
Typ und Identifikator	transparentes Elementary File	EF.VD
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D002'	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / EF.Verweis [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_043]	
Typ und Identifikator	strukturiertes Elementary File	EF.Verweis
Ort	*applicationIdentifier* = 'D276 0000 01 02' → *fileIdentifier* = 'D009'	

Aktion activateRecord	PWD(PIN.CH) AND flagTI.24
Aktion deactivateRecord	PWD(PIN.CH) AND flagTI.24
Aktion read	PWD(PIN.CH) AND flagTI.28
Aktion search	PWD(PIN.CH) AND flagTI.28
Aktion select	ALWAYS
Aktion update	PWD(PIN.CH) AND flagTI.28

8.2.3 Konfigurationstabellen für Objekte in DF.NFD

Pfad Referenz	/ MF / DF.HCA / DF.NFD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_044]	
Typ und Identifikator	Ordner	DF.NFD
Ort	*applicationIdentifier* ='D276 0001 44 07'	
Aktion activate	PWD(MRPIN.NFD) AND flagTI.14	
Aktion deactivate	PWD(MRPIN.NFD) AND flagTI.14	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.NFD / EF.NFD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_045]	
Typ und Identifikator	transparentes Elementary File	EF.NFD
Ort	*applicationIdentifier* = 'D276 0001 44 07' → *fileIdentifier* = 'D010'	
Aktion erase	PWD(MRPIN.NFD) AND flagTI.15	
Aktion read	PWD(MRPIN.NFD_READ) AND flagTI.17	
Aktion setLogicalEndOfFile	PWD(MRPIN.NFD) AND flagTI.15	
Aktion select	ALWAYS	
Aktion update	PWD(MRPIN.NFD) AND flagTI.15	
Aktion write	PWD(MRPIN.NFD) AND flagTI.15	

Pfad Referenz	/ MF / DF.HCA / DF.NFD / EF.StatusNFD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_046]	
Typ und Identifikator	transparentes Elementary File	EF.StatusNFD
Ort	*applicationIdentifier* = 'D276 0001 44 07' → *fileIdentifier* = 'D00E'	
Aktion erase	PWD(MRPIN.NFD) AND flagTI.15	
Aktion read	PWD(MRPIN.NFD_READ) AND flagTI.17	
Aktion setLogicalEndOfFile	PWD(MRPIN.NFD) AND flagTI.15	
Aktion select	ALWAYS	
Aktion update	PWD(MRPIN.NFD) AND flagTI.15	
Aktion write	PWD(MRPIN.NFD) AND flagTI.15	

Pfad Referenz	/ MF / DF.HCA / DF.NFD / MRPIN.NFD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_047]	
Typ und Identifikator	Passwortobjekt	MRPIN.NFD
Ort	*applicationIdentifier* = 'D276 0001 44 07' → *passwordReference* = '83'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion disable	ALWAYS	
Aktion enable	ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS	
Aktion verify	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.NFD / MRPIN.NFD_READ [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_092]	
Typ und Identifikator	Passwortobjekt	MRPIN.NFD_READ

Ort	*applicationIdentifier* = ‘D276 0001 44 07‘ → *passwordReference* = ‘87‘
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS
Aktion getStatus	ALWAYS
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS
Aktion verify	ALWAYS

8.2.4 Konfigurationstabelle für Objekte in DF.DPE

Pfad Referenz	/ MF / DF.HCA / DF.DPE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_049]	
Typ und Identifikator	Ordner	DF.DPE
Ort	*applicationIdentifier* =‘D276 0001 44 08‘	
Aktion activate	PWD(MRPIN.DPE) AND flagTI.19	
Aktion deactivate	PWD(MRPIN.DPE) AND flagTI.19	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.DPE / EF.DPE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_050]	
Typ und Identifikator	transparentes Elementary File	EF.DPE
Ort	*applicationIdentifier* = ‘D276 0001 44 08‘ → *fileIdentifier* = ‘D01B‘	
Aktion erase	PWD(MRPIN.DPE) AND flagTI.20	
Aktion read	PWD(MRPIN.DPE_READ) AND flagTI.22	
Aktion setLogicalEndOfFile	PWD(MRPIN.DPE) AND flagTI.20, mit P1P2=’9B00’	
Aktion select	ALWAYS	
Aktion update	PWD(MRPIN.DPE) AND flagTI.20	
Aktion write	PWD(MRPIN.DPE) AND flagTI.20	

Pfad Referenz	/ MF / DF.HCA / DF.DPE / EF.StatusDPE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_051]	
Typ und Identifikator	transparentes Elementary File	EF.StatusDPE
Ort	*applicationIdentifier* = ‘D276 0001 44 07‘ → *fileIdentifier* = ‘D018‘	
Aktion erase	PWD(MRPIN.DPE) AND flagTI.20	
Aktion read	PWD(MRPIN.DPE_READ) AND flagTI.22	
Aktion setLogicalEndOfFile	PWD(MRPIN.DPE) AND flagTI.20, mit P1P2=’9800’	
Aktion select	ALWAYS	
Aktion update	PWD(MRPIN.DPE) AND flagTI.20	
Aktion write	PWD(MRPIN.DPE) AND flagTI.20	

Pfad Referenz	/ MF / DF.HCA / DF.DPE / MRPIN.DPE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_052]	
Typ und Identifikator	Passwortobjekt	MRPIN.DPE
Ort	*applicationIdentifier* = ‘D276 0001 44 07‘ → *passwordReference* = ‘84‘	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion disable	ALWAYS	
Aktion enable	ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS	
Aktion verify	ALWAYS	

Pfad	/ MF / DF.HCA / DF.DPE / MRPIN.DPE_READ

Referenz	[gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_180]	
Typ und Identifikator	Passwortobjekt	MRPIN.DPE_READ
Ort	*applicationIdentifier* = 'D276 0001 44 07' → *passwordReference* = '88'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS	
Aktion verify	ALWAYS	

8.2.5 Konfigurationstabelle für Objekte in DF.GDD

Pfad Referenz	/ MF / DF.HCA / DF.GDD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_054]	
Typ und Identifikator	Ordner	DF.GDD
Ort	*applicationIdentifier* ='D276 0001 44 0A'	
Aktion activate	PWD(MRPIN.GDD) AND flagTI.39	
Aktion deactivate	PWD(MRPIN.GDD) AND flagTI.39	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.GDD / EF.EinwilligungGDD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_055]	
Typ und Identifikator	strukturiertes Elementary File	EF. EinwilligungGDD
Ort	*applicationIdentifier* = 'D276 0001 44 0A' → *fileIdentifier* = 'D013'	
Aktion append	PWD(MRPIN.GDD) AND flagTI.40	
Aktion deleteRecord	PWD(MRPIN.GDD) AND flagTI.40	
Aktion erase	PWD(MRPIN.GDD) AND flagTI.40	
Aktion read	PWD(MRPIN.GDD) AND flagTI.40	
Aktion search	PWD(MRPIN.GDD) AND flagTI.40	
Aktion update	PWD(MRPIN.GDD) AND flagTI.40	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.GDD / EF.VerweiseGDD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_057]	
Typ und Identifikator	strukturiertes Elementary File	EF. VerweiseGDD
Ort	*applicationIdentifier* = 'D276 0001 44 0A' → *fileIdentifier* = 'D01A'	
Aktion append	PWD(MRPIN.GDD) AND flagTI.40	
Aktion deleteRecord	PWD(MRPIN.GDD) AND flagTI.40	
Aktion erase	PWD(MRPIN.GDD) AND flagTI.40	
Aktion read	PWD(MRPIN.GDD) AND flagTI.40	
Aktion search	PWD(MRPIN.GDD) AND flagTI.40	
Aktion update	PWD(MRPIN.GDD) AND flagTI.40	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.GDD / MRPIN.GDD [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_056]	
Typ und Identifikator	Passwortobjekt	MRPIN.GDD
Ort	*applicationIdentifier* = 'D276 0001 44 07' → *passwordReference* = '85'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion disable	ALWAYS	
Aktion enable	ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	

	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS
Aktion verify	ALWAYS

8.2.6 Konfigurationstabelle für Objekte in DF.OSE

Pfad Referenz	/ MF / DF.HCA / DF.OSE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_184]	
Typ und Identifikator	Ordner	DF.OSE
Ort	*applicationIdentifier* ='D276 0001 44 0B'	
Aktion activate	PWD(MRPIN.OSE) AND flagTI.44	
Aktion deactivate	PWD(MRPIN.OSE) AND flagTI.44	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.OSE / EF.OSE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_185]	
Typ und Identifikator	transparentes Elementary File	EF.OSE
Ort	*applicationIdentifier* = 'D276 0001 44 0B' → *fileIdentifier* = 'E001'	
Aktion erase	PWD(MRPIN.OSE) AND flagTI.43	
Aktion read	PWD(MRPIN.OSE) AND flagTI.41	
Aktion setLogicalEndOfFile	PWD(MRPIN.OSE) AND flagTI.43, mit P1P2='8100'	
Aktion select	ALWAYS	
Aktion update	PWD(MRPIN.OSE) AND flagTI.43	
Aktion write	PWD(MRPIN.OSE) AND flagTI.43	

Pfad Referenz	/ MF / DF.HCA / DF.OSE / EF.StatusOSE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_051]	
Typ und Identifikator	transparentes Elementary File	EF.StatusOSE
Ort	*applicationIdentifier* = 'D276 0001 44 0B' → *fileIdentifier* = 'E002'	
Aktion erase	PWD(MRPIN.OSE) AND flagTI.43	
Aktion read	PWD(MRPIN.OSE) AND flagTI.41	
Aktion setLogicalEndOfFile	PWD(MRPIN.OSE) AND flagTI.43, mit P1P2='8100'	
Aktion select	ALWAYS	
Aktion update	PWD(MRPIN.OSE) AND flagTI.43	
Aktion write	PWD(MRPIN.OSE) AND flagTI.43	

Pfad Referenz	/ MF / DF.HCA / DF.OSE / MRPIN.OSE [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_187]	
Typ und Identifikator	Passwortobjekt	MRPIN.OSE
Ort	*applicationIdentifier* = 'D276 0001 44 07' → *passwordReference* = '89'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS	
Aktion verify	ALWAYS	

8.2.7 Konfigurationstabellen für DF.AMTS

Pfad Referenz	/ MF / DF.HCA / DF.AMTS [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_189]	
Typ und Identifikator	Ordner	DF.AMTS
Ort	*applicationIdentifier* ='D276 0001 44 0C'	

Aktion activate	PWD(MRPIN.AMTS) AND flagTI.45
Aktion deactivate	PWD(MRPIN.AMTS) AND flagTI.45
Aktion select	ALWAYS

Pfad Referenz	/ MF / DF.HCA / DF.AMTS / EF.EinwilligungAMTS [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_055]	
Typ und Identifikator	strukturiertes Elementary File	EF. EinwilligungAMTS
Ort	*applicationIdentifier* = 'D276 0001 44 0C' → *fileIdentifier* = 'E004'	
Aktion append	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion deleteRecord	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion erase	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion read	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	
Aktion search	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	
Aktion update	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.AMTS / EF.AMTS [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_185]	
Typ und Identifikator	transparentes Elementary File	EF.AMTS
Ort	*applicationIdentifier* = 'D276 0001 44 0C' → *fileIdentifier* = 'E005'	
Aktion erase	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.47	
Aktion read	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	
Aktion select	ALWAYS	
Aktion update	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.47	

Pfad Referenz	/ MF / DF.HCA / DF.AMTS / EF.VerweiseAMTS [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_192]	
Typ und Identifikator	strukturiertes Elementary File	EF. VerweiseAMTS
Ort	*applicationIdentifier* = 'D276 0001 44 0C' → *fileIdentifier* = 'E006'	
Aktion append	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion deleteRecord	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion erase	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion read	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	
Aktion search	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	
Aktion update	PWD(MRPIN.AMTS) AND flagTI.47	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.AMTS / EF.StatusAMTS [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_193]	
Typ und Identifikator	transparentes Elementary File	EF.StatusAMTS
Ort	*applicationIdentifier* = 'D276 0001 44 0C' → *fileIdentifier* = 'E007'	
Aktion read	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	
Aktion select	ALWAYS	
Aktion update	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.47	

Pfad Referenz	/ MF / DF.HCA / DF.AMTS / MRPIN.AMTS [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_194]	
Typ und Identifikator	Passwortobjekt	MRPIN.AMTS
Ort	*applicationIdentifier* = 'D276 0001 44 0C' → *passwordReference* = '8C'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS	
Aktion verify	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.AMTS / PIN.AMTS_REP [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_195]	
Typ und Identifikator	Passwortobjekt	PIN.AMTS_REP
Ort	*applicationIdentifier* = 'D276 0001 44 0C' → *passwordReference* = '8D'	
Aktion change	*mode*=set=1 => newSecret, PWD(MRPIN.AMTS)	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockAndSet=2 => *newSecret*, PWD(MRPIN.AMTS)	
Aktion verify	ALWAYS	

Pfad Referenz	/ MF / DF.HCA / DF.AMTS / PrK.AMTS.ENC.E256 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_197]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.AMTS.ENC.E256
Ort	*applicationIdentifier* = 'D276 0001 44 0C' → *keyReference* = '88'	
Aktion elcSharedSecretCalculation	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	
Aktion generate	*mode*=replace+read='C0' PWD(MRPIN.AMTS AND flagTI.47	
Aktion readPublicPart	(PWD(MRPIN.AMTS) OR (PWD(PIN.AMTS_REP))) AND flagTI.46	

8.2.8 Konfigurationstabellen für DF.ESIGN

Pfad Referenz	/ MF / DF.ESIGN [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_059]	
Typ und Identifikator	Ordner	DF.ESIGN
Ort	*applicationIdentifier* ='A000 0001 67 4553 4947 4E'	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / EF.C.CH.AUT.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_060]	
Typ und Identifikator	transparentes Elementary File	EF.C.CH.AUT.R2048
Ort	*applicationIdentifier* = 'A000 0001 67 4553 4947 4E ' → *fileIdentifier* = 'C500'	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / EF.C.CH.AUTN.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_061]	
Typ und Identifikator	transparentes Elementary File	EF.C.CH.AUTN.R2048
Ort	*applicationIdentifier* = 'A000 0001 67 4553 4947 4E ' → *fileIdentifier* = 'C509'	
Aktion read	PWD(PIN.CH) AND flagTI.8	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / EF.C.CH.ENC.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_062]	
Typ und Identifikator	transparentes Elementary File	EF.C.CH.ENC.R2048
Ort	*applicationIdentifier* = 'A000 0001 67 4553 4947 4E ' → *fileIdentifier* = 'C200'	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / EF.C.CH.ENCV.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_063]	
Typ und Identifikator	transparentes Elementary File	EF.C.CH.ENCV.R2048

Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *fileIdentifier* = ‘C50A‘
Aktion read	PWD(PIN.CH) AND flagTI.10
Aktion select	ALWAYS

Pfad Referenz	/ MF / DF.ESIGN / PrK.CH.AUT.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_064]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.CH.AUT.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *keyReference* = ‘82‘	
Aktion readPublicPart	ALWAYS	
Aktion sign9796_2_DS2	PWD(PIN.CH) AND flagTI.12	
Aktion signPKCS1_V1_5	PWD(PIN.CH) AND flagTI.12	
Aktion signPSS	PWD(PIN.CH) AND flagTI.12	

Pfad Referenz	/ MF / DF.ESIGN / PrK.CH.AUTN.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_067]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.CH.AUTN.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *keyReference* = ‘86‘	
Aktion sign9796_2_DS2	PWD(PIN.CH) AND flagTI.8	
Aktion signPSS	PWD(PIN.CH) AND flagTI.8	

Pfad Referenz	/ MF / DF.ESIGN / PrK.CH.ENC.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_070]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.CH.ENC.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *keyReference* = ‘83‘	
Aktion rsaDecipherOaep	PWD(PIN.CH) AND flagTI.13	
Aktion rsaDecipherPKCS1_V1_5	PWD(PIN.CH) AND flagTI.13	
Aktion readPublicPart	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / PrK.CH.ENCV.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_076]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.CH.ENCV.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *keyReference* = ‘87‘	
Aktion rsaDecipherOaep	PWD(PIN.CH) AND flagTI.10	
Aktion rsaDecipherPKCS1_V1_5	PWD(PIN.CH) AND flagTI.10	
Aktion readPublicPart	ALWAYS	

8.2.9 Konfigurationstabellen für DF.CIA_ESIGN

Für das Verzeichnis DF.CIA_ESIGN und die darin enthaltenen Objekte gibt es keine fachlichen Use Cases. Deshalb entfallen diesbezügliche Konfigurationstabellen.

8.2.10 Konfigurationstabellen für DF.QES

Pfad Referenz	/ MF / DF.QES [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_086]	
Typ und Identifikator	Ordner	DF.QES
Ort	*applicationIdentifier* = ‘D276 0000 66 01’	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.QES / EF.C.CH.QES.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_087]	
Typ und Identifikator	transparentes Elementary File	EF.C.CH.QES.R2048
Ort	*applicationIdentifier* = 'D276 0000 66 01 ' → *fileIdentifier* = 'C000'	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.QES / PIN.QES [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_088]	
Typ und Identifikator	Passwortobjekt	PIN.QES
Ort	*applicationIdentifier* = 'D276 0000 66 01 ' → *passwordReference* = '81'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
Aktion verify	ALWAYS, *startSsec* < undendlich	

Pfad Referenz	/ MF / DF.QES / PrK.CH.QES.R2048 [gemSpec_eGK_ObjSys#Tab_eGK_ObjSys_089]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.CH.QES.R2048
Ort	*applicationIdentifier* = 'D276 0000 66 01 ' → *keyReference* = '84'	
Aktion readPublicPart	ALWAYS	
Aktion sign9796_2_DS2	PWD(PIN.QES)	
Aktion signPSS	PWD(PIN.QES)	

8.3 Konfigurationstabelle Card Proxy HBA

Dieses Kapitel wird bei Bedarf in einer späteren Version des Dokumentes ergänzt.

8.4 Konfigurationstabelle Card Proxy SMC-B

Die hier dargestellten Tabellen basieren auf [gemSpec_SMC-B_ObjSys]. Der hier dargestellte Auszug aus [gemSpec_SMC-B_ObjSys] wurde unter dem Blickwinkel AdV-Server verfasst. In anderen Umgebungen (etwa in der Leistungserbringerumgebung eines Arztes) gäbe es inhaltlich abweichende Konfigurationstabellen.

Hinweis (37): Derzeit wurden die Informationen dieses Unterkapitels manuell erstellt. Es ist geplant derartige Informationen demnächst automatisiert aus den Objektsystemspezifikationen zu extrahieren. Dann wird es auch einfach möglich sein diese Informationen für andere Umgebungen schnell und zuverlässig zu erstellen.

Die folgenden Tabellen sind so angeordnet, dass der Pfad alphabetisch aufsteigend sortiert ist. Dabei sind nur die Objekte aufgelistet, für die fachliche Use Cases existieren.

8.4.1 Konfigurationstabellen für Objekte im MF

Pfad Referenz	/ MF [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_002]	
Typ und Identifikator	Ordner	MF
Ort	*applicationIdentifier* = 'D276 0001 46 06'	
Aktion getSecureRandom	ALWAYS	
Aktion select	ALWAYS	

Hinweis (38): / MF / EF.ATR fehlt.
Hinweis (39): / MF / EF.C.CA_SMC.CS.E256 fehlt.
Hinweis (40): / MF / EF.C.CA_SMC.CS.R2048 fehlt.
Hinweis (41): / MF / EF.C.SMC.AUTD_RPE_CVC_E256 fehlt.
Hinweis (42): / MF / EF.C.SMC.AUTR_CVC_E256 fehlt.
Hinweis (43): / MF / EF.C.SMC.AUTR_CVC_R2048 fehlt.
Hinweis (44): / MF / EF.DIR fehlt.
Hinweis (45): / MF / EF.GDO fehlt.
Hinweis (46): / MF / EF.Version2 fehlt.
Hinweis (47): / MF / PrK.SMC.AUTD_RPE.E256 fehlt.

Pfad Referenz	/ MF / PIN.SMC [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_020]	
Typ und Identifikator	Passwortobjekt	PIN.CH
Ort	*passwordReference* = '01'	
Aktion change	*mode*=replace=0 => oldSecret || newSecret, ALWAYS	
Aktion getStatus	ALWAYS	
Aktion unblock	*mode*=UnblockWithPuk=1 => *PUK*, ALWAYS	
	mode=UnblockWithPukAndSet=0 => *PUK* || *newSecret*, Default-Modus ALWAYS	
Aktion verify	ALWAYS	

Pfad Referenz	/ MF / PrK.SMC.AUTR_CVC.R2048 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_021]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.SMC.AUTR_CVC.R2048
Ort	*keyReference* = '10'	
Aktion readPublicPart	ALWAYS	
Aktion rsaRoleAuthentication	PWD(PIN.SMC)	

Pfad Referenz	/ MF / PrK.SMC.AUTR_CVC.E256 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_022]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.SMC.AUTR_CVC.R2048
Ort	*keyReference* = '06'	
Aktion readPublicPart	ALWAYS	
Aktion elcRoleAuthentication	PWD(PIN.SMC)	

Hinweis (48): / MF / PuK.RCA.ADMINCMS.CS.E256 fehlt.
Hinweis (49): / MF / PuK.RCA.CS.E256 fehlt.
Hinweis (50): / MF / PuK.RCA.CS.R2048 fehlt.
Hinweis (51): / MF / SK.CMS.AES128 fehlt.
Hinweis (52): / MF / SK.CMS.AES256 fehlt.
Hinweis (53): / MF / SK.CUP.VSD128 fehlt.
Hinweis (54): / MF / SK.CUP.VSD256 fehlt.

8.4.2 Konfigurationstabellen für DF.SMA

Bei Bedarf wird dieses Kapitel inhaltlich ergänzt.

8.4.3 Konfigurationstabellen für DF.ESIGN

Pfad Referenz	/ MF / DF.ESIGN [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_040]	
Typ und Identifikator	Ordner	DF.ESIGN
Ort	*applicationIdentifier* =‘A000 0001 67 4553 4947 4E‘	
Aktion getSecureRandom	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / EF.C.HCI.AUT.R2048 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_042]	
Typ und Identifikator	transparentes Elementary File	EF.C.HCI.AUT.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *fileIdentifier* = ‘C500‘	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / EF.C.HCI.ENC.R2048 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_062]	
Typ und Identifikator	transparentes Elementary File	EF.C.HCI.ENC.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *fileIdentifier* = ‘C200‘	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / EF.C.HCI.OSIG.R2048 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_041]	
Typ und Identifikator	transparentes Elementary File	EF.C.HCI.OSIG.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *fileIdentifier* = ‘C000‘	
Aktion read	ALWAYS	
Aktion select	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / PrK.HCI.AUT.R2048 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_047]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.HCI.AUT.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *keyReference* = ‘82‘	
Aktion readPublicPart	ALWAYS	
Aktion sign9796_2_DS2	PWD(PIN.SMC)	
Aktion signPKCS1_V1_5	PWD(PIN.SMC)	
Aktion signPSS	PWD(PIN.SMC)	

Pfad Referenz	/ MF / DF.ESIGN / PrK.HCI.ENC.R2048 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_050]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.HCI.ENC.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ → *keyReference* = ‘83‘	
Aktion rsaDecipherOaep	PWD(PIN.SMC)	
Aktion rsaDecipherPKCS1_V1_5	PWD(PIN.SMC)	
Aktion readPublicPart	ALWAYS	

Pfad Referenz	/ MF / DF.ESIGN / PrK.HCI.OSIG.R2048 [gemSpec_SMC-B_ObjSys#Tab_SMC-B_ObjSys_044]	
Typ und Identifikator	privates Schlüsselobjekt	PrK.HCI.OSIG.R2048
Ort	*applicationIdentifier* = ‘A000 0001 67 4553 4947 4E ‘ →	

	keyReference = ‘84‘
Aktion sign9796_2_DS2	PWD(PIN.SMC)
Aktion signPSS	PWD(PIN.SMC)
Aktion readPublicPart	ALWAYS

8.5 Konfigurationstabelle Card Proxy gSMC-K

Dieses Kapitel wird bei Bedarf in einer späteren Version des Dokumentes ergänzt.

8.6 Konfigurationstabelle Card Proxy gSMC-KT

Dieses Kapitel wird bei Bedarf in einer späteren Version des Dokumentes ergänzt.

9 Details zur Implementierung von Aktionen

9.1 cardOperation für Ordner

Dieses Kapitel beschreibt Aktionen für den Objekttyp Ordner.

9.1.1 Aktion activate für Ordner

Diese Aktion hat das Ziel den adressierten Ordner in den Zustand „Operational state (active)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.1.1]. Dazu werden die unten beschriebenen Schritte ausgeführt.

Hinweis (55): Gemäß [gemSpec_COS#(N048.400), (N048.500)] liefert die Aktion select den logischen Wert von lifeCycleStatus und gemäß [gemSpec_COS#(N020.600)] zeigt der Rückgabewert „OK“ der Aktion select an, dass der physikalische Wert von lifeCycleStatus des Ordner bereits „Operational state (active)“ ist. Deshalb ist es hier in Schritt 1.e zulässig die Aktion frühzeitig abzubrechen.

1) Schritt 1: Der Ordner wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 12 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion activate bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion activate fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion activate bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion activate bricht mit dem Rückgabewert ObjectTerminated ab.
   e) OK: Die Aktion activate bricht mit dem Rückgabewert OK ab.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein ACTIVATE Kommando gemäß [gemSpec_COS#(N034.800)] zur Karte geschickt. Die Trailer des ACTIVATE Kommandos gemäß [gemSpec_COS#Tab.28, Tab.29] werden gemäß Tabelle 20 auf die Rückgabewerte der Aktion activate abgebildet. Die Aktion activate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion activate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 20: Rückgabewerte ACTIVATE, Typ Ordner

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Ordners besitzt den Wert „Operational state (active)“, siehe auch 12.2.11. In CardProxy → *channelContext* → *currentFolder* wird der Wert von *lifeCycleStatus* auf „Operational state (active)“ gesetzt.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Ordners besitzt den Wert „Operational state (active)“. In CardProxy → *channelContext* → *currentFolder* wird der Wert von *lifeCycleStatus* auf „Operational state (active)“ gesetzt.

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activate mit Erläuterung
´64 00´	ObjectTerminated	Dieser Trailer ist hier irrelevant, da dieser Fall bereits in Schritt 1 behandelt wird.
´65 81´	MemoryFailure	MemoryFailure Der Ordner wird nochmals mittels der Aktion select ausgewählt, damit in CardProxy → *channelContext* → *currentFolder* der Wert von *lifeCycleStatus* korrekt gesetzt ist, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.

9.1.2 Aktion deactivate für Ordner

Diese Aktion hat das Ziel den adressierten Ordner in den Zustand „Operational state (deactived)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.3.1]. Dazu werden die unten beschriebenen Schritte ausgeführt:

Hinweis (56): Gemäß [gemSpec_COS#(N048.400), (N048.500)] liefert die Aktion select den logischen Wert von lifeCycleStatus und gemäß [gemSpec_COS#(N020.600)] zeigt der Rückgabewert „FileDeactivated“ der Aktion select nicht in jedem Fall an, dass der physikalische Wert von lifeCycleStatus des Ordner bereits „Operational state (deactivated)“ ist. Deshalb ist es hier in Schritt 1.b NICHT zulässig die Aktion frühzeitig abzubrechen.

1) Schritt 1: Der Ordner wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 12 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion deactivate bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion deactivate fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion deactivate bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion deactivate bricht mit dem Rückgabewert ObjectTerminated ab.
   e) OK: Die Aktion deactivate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DEACTIVATE Kommando gemäß [gemSpec_COS#(N036.000)] zur Karte geschickt. Die Trailer des DEACTIVATE Kommandos gemäß [gemSpec_COS#Tab.34, Tab.35] werden gemäß Tabelle 21 auf die Rückgabewerte der Aktion deactivate abgebildet. Die Aktion deactivate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion deactivate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 21: Rückgabewerte DEACTIVATE, Typ Ordner

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deactivate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Ordners besitzt den Wert „Operational state (deactivated)“, siehe auch 12.2.11. In CardProxy → *channelContext* → *currentFolder* wird der Wert von

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deactivate mit Erläuterung
		lifeCycleStatus auf „Operational state (deactivated)“ gesetzt.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Ordners besitzt den Wert „Operational state (deactivated)“. In CardProxy → *channelContext* → *currentFolder* wird der Wert von *lifeCycleStatus* auf „Operational state (deactivated)“ gesetzt.
´64 00´	ObjectTerminated	Dieser Trailer ist hier irrelevant, da dieser Fall bereits in Schritt 1 behandelt wird.
´65 81´	MemoryFailure	MemoryFailure Der Ordner wird nochmals mittels der Aktion select ausgewählt, damit in CardProxy → *channelContext* → *currentFolder* der Wert von *lifeCycleStatus* korrekt gesetzt ist, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.

9.1.3 Aktion delete für Ordner

Diese Aktion hat das Ziel den adressierten Ordner inklusive aller darin enthaltenen Objekte zu löschen, siehe [gemSpec_COS#14.2.4.1]. Dazu werden folgende Schritte ausgeführt:

1) Schritt 1: Der Ordner wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 12 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion delete bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion delete fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion delete bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion delete fährt mit Schritt 2 fort.
   e) OK: Die Aktion delete fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DELETE Kommando gemäß [gemSpec_COS#(N037.100)] zur Karte geschickt. Die Trailer des DELETE Kommandos gemäß [gemSpec_COS#Tab.40, Tab.41] werden gemäß Tabelle 22 auf die Rückgabewerte der Aktion delete abgebildet. Die Aktion delete fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion delete gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 22: Rückgabewerte DELETE, Typ Ordner

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Der Ordner wurde gelöscht, siehe auch 12.2.11. In CardProxy wird *channelContext* aktualisiert gemäß [gemSpec_COS#(N037.700)a].
´90 00´	NoError	OK Der Ordner wurde gelöscht. In CardProxy wird *channelContext* aktualisiert gemäß [gemSpec_COS#(N037.700)a].

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
´65 81´	MemoryFailure	MemoryFailure Card Proxy führt selbständig cardOperation(parent, select) aus mit „parent“ als dem Verzeichnis des in dieser Aktion adressierten Ordners, damit *channelContext* mit der Smartcard synchronisiert wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.

9.1.4 Aktion getSecureRandom für Ordner

Diese Aktion hat das Ziel eine kryptographisch sichere Zufallszahl zurückzuliefern, siehe [gemSpec_COS#14.9.5].

Hinweis (57): Gemäß [gemSpec_COS#(N099.344)] ist ein COS nur dann verpflichtet das GET RANDOM Kommando zu unterstützen, wenn es die Option_logische_Kanäle anbietet. Die Option_logische_Kanäle ist für eine eGK gemäß [gemSpec_eGK_ObjSys#4.1.3] nicht verpflichtend. Das GET RANDOM Kommando wird an dieser Stelle behandelt, obwohl es für eine eGK keine Relevanz hat, damit die Beschreibung von Card Proxy auf andere Kartentypen leichter übertragbar ist.

Hinweis (58): Falls eine Zufallszahl benötigt wird, an die keine besonderen kryptographischen Anforderungen bestehen, ist es möglich die Aktion „getRandom“ (siehe 9.7.1) zu verwenden, die von allen Kartentypen unterstützt wird.

1) Schritt 1: Falls der von der betroffenen Aktion adressierte Ordner als *currentFolder* im *channelContext* eingetragen ist, wird mit Schritt 2 fortgefahren. Andernfalls wird der Ordner mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 12 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion getSecureRandom bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion getSecureRandom fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion getSecureRandom bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion getSecureRandom fährt mit Schritt 2 fort.
   e) OK: Die Aktion getSecureRandom fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Unter Beachtung der maximalen Puffergröße (siehe Kapitel 3 Punkt (5) wird mittels einem oder mehrerer GET RANDOM Kommandos gemäß [gemSpec_COS#(N099.322)] so viele Zufallsdaten erzeugt, wie mit dem Inputparameter *length* angefordert. Die Trailer der GET RANDOM Kommandos gemäß [gemSpec_COS#Tab.241, Tab.242] werden gemäß Tabelle 23 auf die Rückgabewerte der Aktion getSecureRandom abgebildet. Die Aktion getSecureRandom fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion getSecureRandom gibt den unter Schritt 3 ermittelten Rückgabewert zusammen den ausgelesenen Daten zurück.

Tabelle 23: Rückgabewerte GET RANDOM, Typ Ordner

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion getSecureRandom mit Erläuterung
´90 00´	NoError	OK plus angeforderte Daten Erfolgreiche Operation, dieser Rückgabewert wird zusammen mit

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion getSecureRandom mit Erläuterung
		den ausgelesenen Daten zurückgegeben.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.

9.1.5 Aktion select für Ordner

Diese Aktion hat das Ziel den adressierten Ordner zu selektieren und damit *currentFolder* (siehe Kapitel 3 Punkt (1)a) zu ändern, siehe [gemSpec_COS#14.2.6.5]. Dazu werden die unten beschriebenen Schritte ausgeführt.

Hinweis (59): Gemäß [gemSpec_COS#(N048.200)b] ändert sich innerhalb der Smartcard nichts, falls dort currentFolder bereits passend gesetzt ist und currentEF unbestimmt ist. Deshalb ist es zulässig die Aktion select in diesem Fall in Schritt 2 abzubrechen ohne mit der Smartcard kommuniziert zu haben.

1) Schritt 1: Falls der von der betroffenen Aktion adressierte Ordner nicht als *currentFolder* im *channelContext* eingetragen ist oder *currentEF* im *channelContext* nicht den Wert „unbestimmt“ hat, dann fährt die Aktion select mit Schritt 3 fort. Andernfalls (also *currentFolder* passend gesetzt und *currentEF* „unbestimmt“) fährt die Aktion select mit Schritt 2 fort.
2) Schritt 2: Die Aktion select bricht mit dem Rückgabewert
   a) FileDeactivated ab, falls in *channelContext* → *currentFolder* der Wert „Operational state (deactivated)“ dem *lifeCycleStatus* zugeordnet ist.
   b) ObjectTerminated ab, falls in *channelContext* → *currentFolder* der Wert „Termination state“ dem *lifeCycleStatus* zugeordnet ist.
   c) OK ab, falls in *channelContext* → *currentFolder* der Wert „Operational state (active)“ dem *lifeCycleStatus* zugeordnet ist.
3) Schritt 3: Es wird ein SELECT Kommando gemäß [gemSpec_COS#(N042.700)] zur Karte geschickt. Die Trailer des SELECT Kommandos gemäß [gemSpec_COS#Tab.61, Tab.62] werden gemäß Tabelle 24 auf die Rückgabewerte der Aktion select abgebildet:
   a) FileDeactivated: Die Aktion select fährt mit Schritt 4 fort.
   b) ObjectTerminated: Die Aktion select fährt mit Schritt 4 fort.
   c) OK: Die Aktion select fährt mit Schritt 4 fort.
   d) ObjectNotFound: Die Aktion select bricht mit dem Rückgabewert ObjectNotFound ab.
   e) CardTerminated: Die Aktion select bricht mit dem Rückgabewert CardTerminated ab.
4) Schritt 4: *channelContext* (siehe Kapitel 3 Punkt (1) wird gemäß [gemSpec_COS#(N048.200)b] aktualisiert.
5) Schritt 5: Die Aktion select gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 24: Rückgabewerte SELECT, Typ Ordner und den Typ Datei

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion select mit Erläuterung
´62 83´	FileDeactivated	FileDeactivated Erfolgreiche Selektion, das Attribute *lifeCycleStatus* (siehe [gemSpec_COS#(N009.800)]) des Ordners besitzt den Wert „Operational state (deactivated)“.
´62 85´	FileTerminated	ObjectTerminated Erfolgreiche Selektion, das Attribute *lifeCycleStatus* (siehe [gemSpec_COS#(N009.800)]) des Ordners besitzt den Wert „Termination state“.
´90 00´	NoError	OK

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion select mit Erläuterung
		Erfolgreiche Selektion, das Attribute *lifeCycleStatus* (siehe [gemSpec_COS#(N009.800)]) des Ordners besitzt den Wert „Operational state (active)“.
´6A 82´	FileNotFound	ObjectNotFound Zu selektierendes Artefakt wurde nicht gefunden.
´6D 00´	InstructionNotSupported	CardTerminated Die Karte befindet sich im Zustand „Termination state“.

9.1.6 Aktion terminate für Ordner

Diese Aktion hat das Ziel den adressierten Ordner in den Zustand „Termination state“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.8.1]. Dazu werden folgende Schritte ausgeführt:

1) Schritt 1: Der Ordner wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 12 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion terminate bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion terminate fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion terminate bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion terminate bricht mit dem Rückgabewert OK ab.
   e) OK: Die Aktion terminate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein TERMINATE DF Kommando gemäß [gemSpec_COS#(N048.800)] zur Karte geschickt. Die Trailer des TERMINATE DF Kommandos gemäß [gemSpec_COS#Tab.67, Tab.68] werden gemäß Tabelle 25 auf die Rückgabewerte der Aktion terminate abgebildet. Die Aktion terminate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion terminate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 25: Rückgabewerte TERMINATE DF**, Typ Ordner**

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion terminate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Ordners besitzt den Wert „Termination state“, siehe auch 12.2.11. In CardProxy → *channelContext* → *currentFolder* wird der Wert von *lifeCycleStatus* auf „Termination state“ gesetzt.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Ordners besitzt den Wert „Termination state“. In CardProxy → *channelContext* → *currentFolder* wird der Wert von *lifeCycleStatus* auf „Termination state“ gesetzt.
´65 81´	MemoryFailure	MemoryFailure Der Ordner wird nochmals mittels der Aktion select ausgewählt, damit in CardProxy → *channelContext* → *currentFolder* der Wert von *lifeCycleStatus* korrekt gesetzt ist, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.

9.2 cardOperation für transparente Elementary Files

Dieses Kapitel beschreibt Aktionen für den Objekttyp transparentes EF.

9.2.1 Aktion activate für transparente Elementary Files

Hinweis (60): Die hier beschriebene Aktion bezieht sich sowohl auf transparente Elementary Files, als auch auf strukturierte Elementary Files.

Diese Aktion hat das Ziel die adressierte Datei in den Zustand „Operational state (active)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.1.1]. Dazu werden die unten beschriebenen Schritte ausgeführt.

Hinweis (61): Gemäß [gemSpec_COS#(N048.400), (N048.500)] liefert die Aktion select den logischen Wert von lifeCycleStatus und gemäß [gemSpec_COS#(N020.600)] zeigt der Rückgabewert „OK“ der Aktion select an, dass der physikalische Wert von lifeCycleStatus des transparenten Elementary Files bereits „Operational state (active)“ ist. Deshalb ist es hier in Schritt 1.e zulässig die Aktion frühzeitig abzubrechen.

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion activate bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion activate fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion activate bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion activate bricht mit dem Rückgabewert ObjectTerminated ab.
   e) OK: Die Aktion activate bricht mit dem Rückgabewert OK ab.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein ACTIVATE Kommando gemäß [gemSpec_COS#(N034.800)] zur Karte geschickt. Die Trailer des ACTIVATE Kommandos gemäß [gemSpec_COS#Tab.28, Tab.29] werden gemäß Tabelle 26 auf die Rückgabewerte der Aktion activate abgebildet. Die Aktion activate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion activate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 26: Rückgabewerte ACTIVATE, Typ Datei

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* der Datei besitzt den Wert „Operational state (active)“, siehe auch 12.2.11. In CardProxy → *channelContext* → *currentEF* wird der Wert von *lifeCycleStatus* auf „Operational state (active)“ gesetzt.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* der Datei besitzt den Wert „Operational state (active)“. In CardProxy → *channelContext* → *currentEF* wird der Wert von *lifeCycleStatus* auf „Operational state (active)“ gesetzt.
´64 00´	ObjectTerminated	Dieser Trailer ist hier irrelevant, da dieser Fall bereits in Schritt 1 behandelt wird.
´65 81´	MemoryFailure	MemoryFailure Die Datei wird nochmals mittels der Aktion select ausgewählt, damit in CardProxy → *channelContext* → *currentEF* der Wert von *lifeCyc-*

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activate mit Erläuterung
		leStatus korrekt gesetzt ist, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.

9.2.2 Aktion append für transparent Elementary Files

Diese Aktion hat das Ziel, zusätzliche Daten an den Inhalt der adressierten Datei anzuhängen, siehe [gemSpec_COS#14.3.6].

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion append bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion append fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion append bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion append fährt mit Schritt 2 fort.
   e) OK: Die Aktion append fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst.
3) Schritt 3: Falls der Parameter *offset* größer als 32.767 ist, bricht die Aktion append mit dem Rückgabewert OffsetTooBig ab. Andernfalls fährt die Aktion append mit Schritt 4 fort.
4) Schritt 4: Falls der Parameter *newData* mehr als 65.535 Oktette enthält, bricht die Aktion append mit dem Rückgabewert DataTooBig ab. Andernfalls fährt die Aktion mit Schritt 5 fort.
5) Schritt 5: Unter Beachtung der maximalen Puffergröße (siehe Kapitel 3 Punkt (5) wird mittels einem oder mehreren WRITE BINARY Kommandos gemäß [gemSpec_COS#(N055.205)] *newData* an den vorhandenen Dateiinhalt angehängt. Die Trailer der WRITE BINARY Kommandos gemäß [gemSpec_COS#Tab.93, Tab.94] werden gemäß Tabelle 27 auf die Rückgabewerte der Aktion append abgebildet. Die Aktion append fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion append gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 27: Rückgabewerte WRITE BINARY, Typ transparentes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion append mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Dateiinhalt erfolgreich geändert, siehe auch 12.2.11.
´90 00´	NoError	OK Der Dateiinhalt wurde erfolgreich geändert.
´65 81´	MemoryFailure	MemoryFailure siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion append mit Erläuterung
		ausgegangen wird.
´6A 84´	DataTooBig	DataTooBig Inputparameter *newData* enthält zu viele Oktette.

9.2.3 Aktion deactivate für transparente Elementary Files

Hinweis (62): Die hier beschriebene Aktion bezieht sich sowohl auf transparente Elementar Files, als auch auf strukturierte Elementary Files.

Diese Aktion hat das Ziel die adressierte Datei in den Zustand „Operational state (deactived)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.3.1]. Dazu werden die unten beschriebenen Schritte ausgeführt.

Hinweis (63): Gemäß [gemSpec_COS#(N048.400), (N048.500)] liefert die Aktion select den logischen Wert von lifeCycleStatus und gemäß [gemSpec_COS#(N020.600)] zeigt der Rückgabewert „FileDeactivated“ der Aktion select nicht in jedem Fall an, dass der physikalische Wert von lifeCycleStatus des Ordner bereits „Operational state (deactivated)“ ist. Deshalb ist es hier in Schritt 1.b NICHT zulässig die Aktion frühzeitig abzubrechen.

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion deactivate bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion deactivate fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion deactivate bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion deactivate bricht mit dem Rückgabewert ObjectTerminated ab.
   e) OK: Die Aktion deactivate bricht mit dem Rückgabewert OK ab.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DEACTIVATE Kommando gemäß [gemSpec_COS#(N036.000)] zur Karte geschickt. Die Trailer des DEACTIVATE Kommandos gemäß [gemSpec_COS#Tab.34, Tab.35] werden gemäß Tabelle 28 auf die Rückgabewerte der Aktion deactivate abgebildet. Die Aktion deactivate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion deactivate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 28: Rückgabewerte DEACTIVATE, Typ Datei

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deactivate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* der Datei besitzt den Wert „Operational state (deactivated)“, siehe auch 12.2.11. In CardProxy → *channelContext* → *currentEF* wird der Wert von *lifeCycleStatus* auf „Operational state (deactivated)“ gesetzt.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* der Datei besitzt den Wert „Operational state (deactivated)“. In CardProxy → *channelContext* → *currentEF* wird der Wert von *lifeCycleStatus* auf „Operational state (deactivated)“ gesetzt.
´64 00´	ObjectTerminated	Dieser Trailer ist hier irrelevant, da dieser Fall bereits in Schritt 1 behandelt wird.
´65 81´	MemoryFailure	MemoryFailure

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deactivate mit Erläuterung
		Die Datei wird nochmals mittels der Aktion select ausgewählt, damit in CardProxy → *channelContext* → *currentEF* der Wert von *lifeCycleStatus* korrekt gesetzt ist, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.

9.2.4 Aktion delete für transparente Elementary Files

Hinweis (64): Die hier beschriebene Aktion bezieht sich sowohl auf transparente Elementar Files, als auch auf strukturierte Elementary Files.

Diese Aktion hat das Ziel die adressierte Datei zu löschen, siehe [gemSpec_COS#14.2.4.1]. Dazu werden folgende Schritte ausgeführt:

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion delete bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion delete fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion delete bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion delete fährt mit Schritt 2 fort.
   e) OK: Die Aktion delete fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DELETE Kommando gemäß [gemSpec_COS#(N037.100)] zur Karte geschickt. Die Trailer des DELETE Kommandos gemäß [gemSpec_COS#Tab.40, Tab.41] werden gemäß Tabelle 29 auf die Rückgabewerte der Aktion delete abgebildet. Die Aktion delete fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion delete gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 29: Rückgabewerte DELETE, Typ Datei

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Die Datei wurde gelöscht, siehe auch 12.2.11. In CardProxy wird *channelContext* aktualisiert gemäß [gemSpec_COS#(N037.700)b].
´90 00´	NoError	OK Die Datei wurde gelöscht. In CardProxy wird *channelContext* aktualisiert gemäß [gemSpec_COS#(N037.700)b].
´65 81´	MemoryFailure	MemoryFailure Card Proxy führt selbständig cardOperation(parent, select) aus mit „parent“ als dem Verzeichnis der in dieser Aktion adressierten Datei, damit *channelContext* mit der Smartcard synchronisiert wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner behandelt wird.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Ordner

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
		behandelt wird.

9.2.5 Aktion erase für transparent Elementary Files

Diese Aktion hat das Ziel, den Inhalt der adressierten Datei ganz oder teilweise auf den Wert ´00´ zu setzen, siehe [gemSpec_COS#14.3.1]. Der Dateiinhalt wird dabei gelöscht. Im Gegensatz zur Aktion delete bleibt die Datei selbst aber erhalten. Dazu werden die unten beschriebenen Schritte ausgeführt.

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion erase bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion erase fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion erase bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion erase fährt mit Schritt 2 fort.
   e) OK: Die Aktion erase fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls der Parameter *offset* größer als 32.767 ist, bricht die Aktion erase mit dem Rückgabewert OffsetTooBig ab. Andernfalls fährt die Aktion erase mit Schritt 4 fort.
4) Schritt 4: Es wird ein ERASE BINARY Kommando gemäß [gemSpec_COS#(N049.100)] zur Smartcard geschickt. Die Trailer des ERASE BINARY Kommandos gemäß [gemSpec_COS#Tab.77, Tab.78] werden gemäß Tabelle 30 auf die Rückgabewerte der Aktion erase abgebildet. Die Aktion erase fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion erase gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 30: Rückgabewerte ERASE BINARY, Typ transparentes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion erase mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Dateiinhalt erfolgreich gelöscht, siehe auch 12.2.11.
´90 00´	NoError	OK Der Dateiinhalt wurde erfolgreich gelöscht.
´65 81´	MemoryFailure	MemoryFailure siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6B 00´	OffsetTooBig	OffsetTooBig Inputparameter *offset* ist zu groß.

9.2.6 Aktion read für transparent Elementary Files

Diese Aktion hat das Ziel, den Inhalt der adressierten Datei ganz oder teilweise auszulesen, siehe [gemSpec_COS#14.3.2].

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion read bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion read fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion read bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion read fährt mit Schritt 2 fort.
   e) OK: Die Aktion read fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls der Parameter *offset* größer als 32.767 ist, bricht die Aktion read mit dem Rückgabewert OffsetTooBig ab. Andernfalls fährt die Aktion read mit Schritt 4 fort.
4) Schritt 4: Falls der Parameter *length* größer als 65.536 ist, wird der Parameter *length* auf 65.536 gesetzt.
5) Schritt 5: Unter Beachtung der maximalen Puffergröße (siehe Kapitel 3 Punkt (5) wird mittels einem oder mehreren READ BINARY Kommandos gemäß [gemSpec_COS#(N051.100)] so viele Daten aus der adressierten Datei ausgelesen, bis entweder das Dateiende erreicht ist, *length* Oktette gelesen wurden. Die Trailer der READ BINARY Kommandos gemäß [gemSpec_COS#Tab.81, Tab.82] werden gemäß Tabelle 31 auf die Rückgabewerte der Aktion read abgebildet. Die Aktion read fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion read gibt den unter Schritt 5 ermittelten Rückgabewert zusammen den ausgelesenen Daten zurück.

Tabelle 31: Rückgabewerte READ BINARY, Typ transparentes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion read mit Erläuterung
´62 81´	CorruptDataWarning	CorruptDataWarning plus angeforderte Daten Dieser Rückgabewert wird zusammen mit den ausgelesenen Daten zurückgegeben, falls mindestens ein READ BINARY Kommandos diesen Trailer lieferte, siehe 12.2.3.
´62 82´	EndOfFileWarning	OK plus angeforderte Daten Möglicherweise weniger Daten gelesen, als angefordert, dieser Rückgabewert wird zusammen mit den ausgelesenen Daten zurückgegeben.
´90 00´	NoError	OK plus angeforderte Daten Erfolgreiche Leseoperation, dieser Rückgabewert wird zusammen mit den ausgelesenen Daten zurückgegeben.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6B 00´	OffsetTooBig	OffsetTooBig Inputparameter *offset* ist zu groß.

9.2.7 Aktion select für transparente Elementary Files

Hinweis (65): Die hier beschriebene Aktion bezieht sich sowohl auf transparente Elementar Files, als auch auf strukturierte Elementary Files.

Diese Aktion hat das Ziel die adressierte Datei zu selektieren und damit *currentEF* (siehe Kapitel 3 Punkt (1)h zu ändern, siehe [gemSpec_COS#14.2.6.13]. Dazu werden die unten beschriebenen Schritte ausgeführt.

Hinweis (66): Gemäß [gemSpec_COS#(N048.200)a] ändert sich innerhalb der Smartcard nichts, falls dort currentEF bereits passend gesetzt ist. Deshalb ist es zulässig die Aktion select in diesem Fall in Schritt 2 abzubrechen.

1) Schritt 1: Falls die von der betroffenen Aktion adressierte Datei nicht als *currentEF* im *channelContext* eingetragen ist, dann fährt die Aktion select mit Schritt 3 fort. Andernfalls (also *currentEF* passend gesetzt) fährt die Aktion select mit Schritt 2 fort.
2) Schritt 2: Die Aktion select bricht mit dem Rückgabewert
   a) FileDeactivated ab, falls in *channelContext* → *currentEF* der Wert „Operational state (deactivated)“ dem *lifeCycleStatus* zugeordnet ist.
   b) ObjectTerminated ab, falls in *channelContext* → *currentEF* der Wert „Termination state“ dem *lifeCycleStatus* zugeordnet ist.
   c) OK ab, falls in *channelContext* → *currentEF* der Wert „Operational state (active)“ dem *lifeCycleStatus* zugeordnet ist.
3) Schritt 3: Für den Parent-Ordner der adressierten Datei wird die Aktion select ausgeführt. Dabei sind gemäß Tabelle 12 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion select bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion select fährt mit Schritt 4 fort.
   c) ObjectNotFound: Die Aktion select bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion select fährt mit Schritt 4 fort.
   e) OK: Die Aktion select fährt mit Schritt 4 fort.
4) Schritt 4: Es wird ein SELECT Kommando gemäß [gemSpec_COS#(N046.700)] zur Karte geschickt. Die Trailer des SELECT Kommandos gemäß [gemSpec_COS#Tab.61, Tab.62] werden gemäß Tabelle 24 auf die Rückgabewerte der Aktion select abgebildet:
   a) FileDeactivated: Die Aktion select fährt mit Schritt 5 fort.
   b) ObjectTerminated: Die Aktion select fährt mit Schritt 5 fort.
   c) OK: Die Aktion select fährt mit Schritt 5 fort.
   d) ObjectNotFound: Die Aktion select bricht mit dem Rückgabewert ObjectNotFound ab.
   e) CardTerminated: Die Aktion select bricht mit dem Rückgabewert CardTerminated ab.
5) Schritt 5: *channelContext* (siehe Kapitel 3 Punkt (1) wird gemäß [gemSpec_COS#(N048.200)a] aktualisiert.
6) Schritt 6: Die Aktion select gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

9.2.8 Aktion setLogicalEndOfFile für transparent Elementary Files

Diese Aktion hat das Ziel, das Dateiendezeichen der adressierten Datei Richtung Dateianfang zu verschieden, ohne die Dateigröße zu ändern, siehe [gemSpec_COS#14.3.4].

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion setLogicalEndOfFile bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion setLogicalEndOfFile fährt mit Schritt 2 fort.

c) ObjectNotFound: Die Aktion setLogicalEndOfFile bricht mit dem Rückgabewert ObjectNotFound ab.
d) ObjectTerminated: Die Aktion setLogicalEndOfFile fährt mit Schritt 2 fort.
e) OK: Die Aktion setLogicalEndOfFile fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls der Parameter *offset* größer als 32.767 ist, bricht die Aktion setLogicalEndOfFile mit dem Rückgabewert OffsetTooBig ab. Andernfalls fährt die Aktion erase mit Schritt 4 fort.
4) Schritt 4: Es wird ein SET LOGICAL EOF Kommando gemäß [gemSpec_COS#(N052.932)] zur Smartcard geschickt. Die Trailer des SET LOGICAL EOF Kommandos gemäß [gemSpec_COS#Tab.85, Tab.86] werden gemäß Tabelle 32 auf die Rückgabewerte der Aktion setLogicalEndOfFile abgebildet. Die Aktion setLogicalEndOfFile fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion erase gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 32: Rückgabewerte SET LOGICAL EOF, Typ transparentes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion setLogicalEndOfFile mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Dateiendezeichen erfolgreich geändert, siehe 12.2.11.
´90 00´	NoError	OK Das Dateiendezeichen wurde erfolgreich geändert.
´65 81´	MemoryFailure	MemoryFailure siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6B 00´	OffsetTooBig	OffsetTooBig Inputparameter *offset* ist zu groß.

9.2.9 Aktion terminate für transparente Elementary Files

Hinweis (67): Die hier beschriebene Aktion bezieht sich sowohl auf transparente Elementar Files, als auch auf strukturierte Elementary Files.

Diese Aktion hat das Ziel, die adressierte Datei in den Zustand „Termination state“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.9.1]. Dazu werden folgende Schritte ausgeführt:

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion terminate bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion terminate fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion terminate bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion terminate bricht mit dem Rückgabewert OK ab.
   e) OK: Die Aktion terminate fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein TERMINATE Kommando gemäß [gemSpec_COS#(N048.903)] zur Karte geschickt. Die Trailer des TERMINATE Kommandos gemäß [gemSpec_COS#Tab.37, Tab.74] werden gemäß Tabelle 33 auf die Rückgabewerte der Aktion terminate abgebildet. Die Aktion terminate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion terminate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 33: Rückgabewerte TERMINATE, TYP Datei

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion terminate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribut *lifeCycleStatus* des Ordners besitzt den Wert „Termination state“, siehe auch 12.2.11. In CardProxy → *channelContext* → *currentFolder* wird der Wert von *lifeCycleStatus* auf „Termination state“ gesetzt.
´90 00´	NoError	OK Das Attribut *lifeCycleStatus* des Ordners besitzt den Wert „Termination state“. In CardProxy → *channelContext* → *currentFolder* wird der Wert von *lifeCycleStatus* auf „Termination state“ gesetzt.
´65 81´	MemoryFailure	MemoryFailure Der Ordner wird nochmals mittels der Aktion select ausgewählt, damit in CardProxy → *channelContext* → *currentFolder* der Wert von *lifeCycleStatus* korrekt gesetzt ist, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Objekttyp Datei behandelt wird.

9.2.10 Aktion update für transparent Elementary Files

Diese Aktion hat das Ziel, den Inhalt der adressierten Datei ganz oder teilweise zu überschreiben, siehe [gemSpec_COS#14.3.5].

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion update bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion update fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion update bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion update fährt mit Schritt 2 fort.
   e) OK: Die Aktion update fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls der Parameter *offset* gößer als 32.767 ist, bricht die Aktion update mit dem Rückgabewert OffsetTooBig ab. Andernfalls fährt die Aktion update mit Schritt 4 fort.

4) Schritt 4: Falls der Parameter *newData* mehr als 65.535 Oktette enthält, bricht die Aktion update mit dem Rückgabewert DataTooBig ab. Andernfalls fährt die Aktion mit Schritt 5 fort.
5) Schritt 5: Unter Beachtung der maximalen Puffergröße (siehe Kapitel 3 Punkt (5) wird mittels einem oder mehreren UPDATE BINARY Kommandos gemäß [gemSpec_COS#(N053.200)] *newData* in den Dateiinhalt geschrieben. Die Trailer der UPDATE BINARY Kommandos gemäß [gemSpec_COS#Tab.89, Tab.90] werden gemäß Tabelle 34 auf die Rückgabewerte der Aktion update abgebildet. Die Aktion update fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion update gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 34: Rückgabewerte UPDATE BINARY, Typ transparentes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion update mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Dateiinhalt erfolgreich geändert, siehe auch 12.2.11.
´90 00´	NoError	OK Der Dateiinhalt wurde erfolgreich geändert.
´65 81´	MemoryFailure	MemoryFailure siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 84´	DataTooBig	DataTooBig Inputparameter *newData* enthält zu viele Oktette.
´6B 00´	OffsetTooBig	OffsetTooBig Inputparameter *offset* ist zu groß.

9.3 cardOperation für strukturierte Elementary Files

Dieses Kapitel beschreibt Aktionen für den Objekttyp strukturiertes EF.

9.3.1 Aktion activate für strukturierte Elementary Files

Die Aktion activate läuft für strukturierte und transparente Elementary Files gleichartig ab. Deshalb wird hier auf die Ausführungen in 9.2.1 verwiesen.

9.3.2 Aktion activateRecord für strukturierte Elementary Files

Diese Aktion hat das Ziel einen oder alle Rekords eines strukturierten Elementary Files in den Zustand „Operational state (active)“ zu überführen, siehe [gemSpec_COS#(N007.800), 14.4.1].

1) Schritt 1: Das strukturierte Elementary File wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion activateRecord bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion activateRecord fährt mit Schritt 2 fort.

c) ObjectNotFound: Die Aktion activateRecord bricht mit dem Rückgabewert Object-NotFound ab.
d) ObjectTerminated: Die Aktion activateRecord fährt mit Schritt 2 fort.
e) OK: Die Aktion activateRecord fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls *recordNumber* kleiner als 1 oder größer als 254 ist, bricht die Aktion activateRecord mit dem Rückgabewert RecordNotFound ab. Andernfalls fährt die Aktion activateRecord mit Schritt 4 fort.
4) Schritt 4: Falls der optionale Inputparameter *recordNumber*
   a) fehlt, dann wird ein ACTIVATE RECORD Kommando gemäß [gemSpec_COS#(N056.200)] mit *recordNumber*=1 zur Karte geschickt und mit Schritt 5 fortgefahren.
   b) vorhanden ist, dann ein ACTIVATE RECORD Kommando gemäß [gemSpec_COS#(N055.500)] zur Karte geschickt und mit Schritt 5 fortgefahren.
5) Schritt 5: Die Trailer des ACTIVATE RECORD Kommandos gemäß [gemSpec_COS#Tab.99, Tab.100] werden gemäß Tabelle 35 auf die Rückgabewerte der Aktion activateRecord abgebildet. Die Aktion activateRecord fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion activateRecord gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 35: Rückgabewerte ACTIVATE RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activateRecord + Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Die Attribute *lifeCycleStatus* der adressierten Records besitzen den Wert „Operational state (active)“, siehe auch 12.2.11.
´90 00´	NoError	OK Die Attribute *lifeCycleStatus* der adressierten Records besitzen den Wert „Operational state (active)“.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 85´	NoRecordLifeCycleStatus	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 83´	RecordNotFound	RecordNotFound Das mittels *recordNumber* adressierte Listenelement ist nicht vorhanden.

9.3.3 Aktion append für strukturierte Elementary Files

Die Aktion hat das Ziel der Liste von Rekords einen neuen Rekord hinzuzufügen, siehe [gemSpec_COS#(N012.200), 14.4.2].

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion append bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion append fährt mit Schritt 2 fort.

c) ObjectNotFound: Die Aktion append bricht mit dem Rückgabewert ObjectNotFound ab.
d) ObjectTerminated: Die Aktion append fährt mit Schritt 2 fort.
e) OK: Die Aktion append fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls der Parameter *recordData* mehr als 65.535 Oktette enthält, bricht die Aktion append mit dem Rückgabewert WrongRecordLength ab. Andernfalls fährt die Aktion mit Schritt 4 fort.
4) Schritt 4: Falls der Parameter *recordData* so viele Oktette enthält, dass die maximale Puffergröße (siehe Kapitel 3 Punkt (5) überschritten würde, bricht die Aktion append mit dem Rückgabewert BufferTooSmall ab. Andernfalls fährt die Aktion mit Schritt 5 fort.
5) Schritt 5: Es wird ein APPEND RECORD Kommandos gemäß [gemSpec_COS#(N058.400)] zur Karte geschickt. Die Trailer des APPEND RECORD Kommandos gemäß [gemSpec_COS#Tab.103, Tab.104] werden gemäß Tabelle 36 auf die Rückgabewerte der Aktion append abgebildet. Die Aktion append fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion append gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 36: Rückgabewerte APPEND RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion append mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Liste der Rekords erfolgreich ergänzt, siehe auch 12.2.11.
´90 00´	NoError	OK Liste der Rekords erfolgreich ergänzt.
´65 81´	MemoryFailure	MemoryFailure siehe auch 12.2.8.
´67 00´	WrongRecordLength	WrongRecordLength Dieser Rückgabewert weist darauf hin, dass *recordData* eine Länge besitzt, die von der adressierten Datei grundsätzlich nicht unterstützt wird.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 84´	FullRecordList	FullRecordList Dieser Rückgabewert weist darauf hin, dass die Liste der Rekords ihre maximale Größe erreicht hat. Ein Hinzufügen weiterer Rekords ist daher nicht möglich.
´6A 84´	OutOfMemory	OutOfMemory Dieser Rückgabewert weist darauf hin, dass alle Elemente der Liste der Rekords plus *recordData* zusammengenommen zu viele Oktette enthalten.

9.3.4 Aktion deactivate für strukturierte Elementary Files

Die Aktion deactivate läuft für strukturierte und transparente Elementary Files gleichartig ab. Deshalb wird hier auf die Ausführungen in 9.2.3 verwiesen.

9.3.5 Aktion deactivateRecord für strukturierte Elementary Files

Diese Aktion hat das Ziel einen oder alle Rekords eines strukturierten Elementary Files in den Zustand Operational state (deactivated)“ zu überführen, siehe [gemSpec_COS#(N007.800), 14.4.3].

1) Schritt 1: Das strukturierte Elementary File wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion deactivateRecord bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion deactivateRecord fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion deactivateRecord bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion deactivateRecord fährt mit Schritt 2 fort.
   e) OK: Die Aktion deactivateRecord fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls *recordNumber* kleiner als 1 oder größer als 254 ist, bricht die Aktion deactivateRecord mit dem Rückgabewert RecordNotFound ab. Andernfalls fährt die Aktion deactivateRecord mit Schritt 4 fort.
4) Schritt 4: Falls der optionale Inputparameter *recordNumber*
   a) fehlt, dann wird ein DEACTIVATE RECORD Kommando gemäß [gemSpec_COS#(N061.400)] mit *recordNumber*=1 zur Karte geschickt und mit Schritt 5 fortgefahren.
   b) vorhanden ist, dann wird ein DEACTIVATE RECORD Kommando gemäß [gemSpec_COS#(N060.700)] zur Karte geschickt und mit Schritt 5 fortgefahren.
5) Schritt 5: Die Trailer des DEACTIVATE RECORD Kommandos gemäß [gemSpec_COS#Tab.109, Tab.110] werden gemäß Tabelle 37 auf die Rückgabewerte der Aktion deactivateRecord abgebildet. Die Aktion deactivateRecord fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion deactivateRecord gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 37: Rückgabewerte DEACTIVATE RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert Aktion deactivateRecord + Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Die Attribute *lifeCycleStatus* der adressierten Records besitzen den Wert „Operational state (deactivated)“, siehe auch 12.2.11.
´90 00´	NoError	OK Die Attribute *lifeCycleStatus* der adressierten Records besitzen den Wert „Operational state (deactivated)“.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 85´	NoRecordLifeCycleStatus	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 83´	RecordNotFound	RecordNotFound Das mittels *recordNumber* adressierte Listenelement ist nicht vor-

Trailer gemäß [gemSpec_COS]		Rückgabewert Aktion deactivateRecord + Erläuterung
		handen.

9.3.6 Aktion delete für strukturierte Elementary Files

Die Aktion delete läuft für strukturierte und transparente Elementary Files gleichartig ab. Deshalb wird hier auf die Ausführungen in 9.2.4 verwiesen.

9.3.7 Aktion deleteRecord

Diese Aktion hat das Ziel einen oder alle Rekords aus der Liste der Rekords eines strukturierten Elementary Files zu entfernen, siehe [gemSpec_COS#(N012.200), 14.4.4].

1) Schritt 1: Das strukturierte Elementary File wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion deleteRecord bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion deleteRecord fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion deleteRecord bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion deleteRecord fährt mit Schritt 2 fort.
   e) OK: Die Aktion deleteRecord fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls *recordNumber* kleiner als 1 oder größer als 254 ist, bricht die Aktion deleteRecord mit dem Rückgabewert RecordNotFound ab. Andernfalls fährt die Aktion deleteRecord mit Schritt 4 fort.
4) Schritt 4: Falls der optionale Inputparameter *recordNumber*
   a) fehlt, dann wird folgender Algorithmus ausgeführt:
      i) Der potentielle Rückgabewert wird auf OK gesetzt.
      ii) Es wird ein DELETE RECORD Kommando gemäß [gemSpec_COS#(N063.422)] mit *recordNumber*=1 zur Karte geschickt. Dabei sind gemäß Tabelle 38 folgende Rückgabewerte möglich:
         (1) UpdateRetryWarning: Falls der potentielle Rückgabewert auf OK steht, wird er auf UpdateRetryWarning geändert. In jedem Fall wird Schritt 4)a)ii) wiederholt.
         (2) NoError: Der Schritt 4)a)ii) wird wiederholt.
         (3) RecordDeactivated: Falls der potentielle Rückgabewert auf OK oder UpdateRetryWarning steht, wird er auf RecordDeactivated geändert. In jedem Fall wird Schritt 4)a)ii) wiederholt.
         (4) MemoryFailure: Die Aktion deleteRecord bricht mit dem Rückgabewert MemoryFailure ab.
         (5) SecurityStatusNotSatisfied: Die Aktion deleteRecord bricht mit dem Rückgabewert SecurityStatusNotSatisfied ab.
         (6) RecordNotFound: Die Aktion deleteRecord bricht mit dem potentiellen Rückgabewert ab.
   b) vorhanden ist, dann wird ein DELETE RECORD Kommando gemäß [gemSpec_COS#(N063.422)] zur Karte geschickt, und mit Schritt 5 fortgefahren.
5) Schritt 5: Die Trailer des DELETE RECORD Kommandos gemäß [gemSpec_COS#Tab.113, Tab.114] werden gemäß Tabelle 38 auf die Rückgabewerte der Aktion deleteRecord abgebildet. Die Aktion deleteRecord fährt mit Schritt 6 fort.

6) Schritt 6: Die Aktion deleteRecord gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 38: Rückgabewerte DELETE RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deleteRecord + Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Adressierter Rekord erfolgreich gelöscht, siehe auch 12.2.11.
´90 00´	NoError	OK Adressierter Rekord erfolgreich gelöscht.
´62 87´	RecordDeactivated	RecordDeactivated Rekord nicht löschbar, da Rekord deaktiviert.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 83´	RecordNotFound	RecordNotFound Das mittels *recordNumber* adressierte Listenelement ist nicht vorhanden.

9.3.8 Aktion erase

Diese Aktion hat das Ziel jedes Oktett des Inhaltes eines oder aller Rekords auf den Wert ´00´ zu setzen, siehe [gemSpec_COS14.4.5].

1) Schritt 1: Das strukturierte Elementary File wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion erase bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion erase fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion erase bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion erase fährt mit Schritt 2 fort.
   e) OK: Die Aktion erase fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls *recordNumber* kleiner als 1 oder größer als 254 ist, bricht die Aktion erase mit dem Rückgabewert RecordNotFound ab. Andernfalls fährt die Aktion erase mit Schritt 4 fort.
4) Schritt 4: Falls der optionale Inputparameter *recordNumber*
   a) fehlt, dann wird folgender Algorithmus ausgeführt:
      i) Der potentielle Rückgabewert wird auf OK und *recordNumber*=0 gesetzt.
      ii) Die Variable *recordNumber* wird um eins inkrementiert. Es wird ein ERASE RECORD Kommando gemäß [gemSpec_COS#(N063.600)] zur Karte geschickt. Dabei sind gemäß Tabelle 39 folgende Rückgabewerte möglich:
         (1) UpdateRetryWarning: Falls der potentielle Rückgabewert auf OK steht, wird er auf UpdateRetryWarning geändert. In jedem Fall wird Schritt 4)a)ii) wiederholt.
         (2) NoError: Der Schritt 4)a)ii) wird wiederholt.

(3) RecordDeactivated: Falls der potentielle Rückgabewert auf OK oder UpdateRetryWarning steht, wird er auf RecordDeactivated geändert. In jedem Fall wird Schritt 4)a)ii) wiederholt.
(4) MemoryFailure: Die Aktion eraseRecord bricht mit dem Rückgabewert MemoryFailure ab.
(5) SecurityStatusNotSatisfied: Die Aktion eraseRecord bricht mit dem Rückgabewert SecurityStatusNotSatisfied ab.
(6) RecordNotFound: Die Aktion eraseRecord bricht mit dem potentiellen Rückgabewert ab.

b) vorhanden ist, dann wird ein ERASE RECORD Kommando gemäß [gemSpec_COS#(N063.600)] zur Karte geschickt, und mit Schritt 5 fortgefahren.

5) Schritt 5: Die Trailer des ERASE RECORD Kommandos gemäß [gemSpec_COS#Tab.117, Tab.118] werden gemäß Tabelle 39 auf die Rückgabewerte der Aktion erase abgebildet. Die Aktion erase fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion erase gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 39: Rückgabewerte ERASE RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion erase mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Inhalt des adressierten Rekords erfolgreich gelöscht, siehe auch 12.2.11.
´90 00´	NoError	OK Inhalt des adressierten Rekords erfolgreich gelöscht.
´62 87´	RecordDeactivated	RecordDeactivated Rekordinhalt nicht löschbar, da Rekord deaktiviert.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 83´	RecordNotFound	RecordNotFound Das mittels *recordNumber* adressierte Listenelement ist nicht vorhanden.

9.3.9 Aktion read für strukturierte Elementary Files

Diese Aktion hat das Ziel einen oder alle Rekords eines strukturierten Elementary Files auszulesen, siehe [gemSpec_COS#14.4.6].

1) Schritt 1: Das strukturierte Elementary File wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion read bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion read fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion read bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion read fährt mit Schritt 2 fort.
   e) OK: Die Aktion read fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls *recordNumber* kleiner als 1 oder größer als 254 ist, bricht die Aktion read mit dem Rückgabewert RecordNotFound ab. Andernfalls fährt die Aktion read mit Schritt 4 fort.
4) Schritt 4: Falls der optionale Inputparameter *recordNumber*
   a) fehlt, dann wird folgender Algorithmus ausgeführt:
      i) Es werden einige Initialisierungen vorgenommen:
         (1) Der potentielle Rückgabewert wird auf OK gesetzt.
         (2) Die Variable *recordNumbe*r wird auf den Wert 0 gesetzt.
         (3) Als Rückgabeliste wird eine leere Liste mit Rückgabewerten erzeugt.
      ii) Die Variable *recordNumber* wird um eins inkrementiert. Es wird ein READ RECORD Kommando gemäß [gemSpec_COS#(N065.700)] und *length*=WildCardShort zur Karte geschickt. Dabei sind gemäß Tabelle 40 folgende Rückgabewerte möglich:
         (1) CorruptDataWarning: Die Rückgabeliste wird um einen Eintrag gemäß Tabelle 40 erweitert. In jedem Fall wird Schritt 4)a)ii) wiederholt.
         (2) NoError: Die Rückgabeliste wird um einen Eintrag gemäß Tabelle 40 erweitert. In jedem Fall wird Schritt 4)a)ii) wiederholt.
         (3) RecordDeactivated: Die Rückgabeliste wird um einen Eintrag gemäß Tabelle 40 erweitert. In jedem Fall wird Schritt 4)a)ii) wiederholt.
         (4) SecurityStatusNotSatisfied: Die Aktion read bricht mit dem Rückgabewert SecurityStatusNotSatisfied ab.
         (5) RecordNotFound: Die Aktion read meldet die vorbereitete Rückgabeliste zurück.
   b) vorhanden ist, dann ein READ RECORD Kommando gemäß [gemSpec_COS#(N065.700)] und *length*=WildCardShort zur Karte geschickt und mit Schritt 5 fortgefahren.
5) Schritt 5: Die Trailer der READ RECORD Kommandos gemäß [gemSpec_COS#Tab.121, Tab.122] werden gemäß Tabelle 40 auf die Rückgabewerte der Aktion read abgebildet. Die Aktion read fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion read gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 40: Rückgabewerte READ RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion read mit Erläuterung
´62 81´	CorruptDataWarning	CorruptDataWarning plus angeforderte Daten Dieser Rückgabewert wird zusammen mit den ausgelesenen Daten zurückgegeben, siehe 12.2.3.
´62 82´	EndOfRecordWarning	Dieser Trailer ist hier irrelevant, da für das Le-Feld WildCard verwendet wird.
´90 00´	NoError	OK plus angeforderte Daten Erfolgreiche Leseoperation, dieser Rückgabewert wird zusammen mit den ausgelesenen Daten zurückgegeben.
´62 87´	RecordDeactivated	RecordDeactivated Rekordinhalt nicht lesbar, da Rekord deaktiviert.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 83´	RecordNotFound	RecordNotFound

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion read mit Erläuterung
		Das mittels *recordNumber* adressierte Listenelement ist nicht vorhanden.

9.3.10 Aktion search für strukturierte Elementary Files

Die Aktion hat das Ziel die Liste der Rekords nach einem Muster zu durchsuchen und die Nummern der Listenelemente zurückzumelden, in denen das Muster gefunden wurde, siehe [gemSpec_COS#14.4.7].

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion update bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion search fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion search bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion search fährt mit Schritt 2 fort.
   e) OK: Die Aktion search fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls *recordNumber* kleiner als 1 oder größer als 254 ist, bricht die Aktion search mit dem Rückgabewert RecordNotFound ab. Andernfalls fährt die Aktion search mit Schritt 4 fort.
4) Schritt 4: Falls der Parameter *searchString* mehr als 65.535 Oktette enthält, bricht die Aktion search mit dem Rückgabewert BufferTooSmall ab. Andernfalls fährt die Aktion mit Schritt 5 fort.
5) Schritt 5: Falls der Parameter *searchString* so viele Oktette enthält, dass die maximale Puffergröße (siehe Kapitel 3 Punkt (5) überschritten würde, bricht die Aktion search mit dem Rückgabewert BufferTooSmall ab. Andernfalls fährt die Aktion mit Schritt 6 fort.
6) Schritt 6: Es wird ein SEARCH RECORD Kommandos gemäß [gemSpec_COS#(N067.900)] zur Karte geschickt, wobei *length*=WildCardShort gesetzt wird. Die Trailer des SEARCH RECORD Kommandos gemäß [gemSpec_COS#Tab.125, Tab.126] werden gemäß Tabelle 41 auf die Rückgabewerte der Aktion search abgebildet. Die Aktion search fährt mit Schritt 7 fort.
7) Schritt 7: Die Aktion search gibt den unter Schritt 6 ermittelten Rückgabewert zurück.

Tabelle 41: Rückgabewerte SEARCH RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion search mit Erläuterung
´62 81´	CorruptDataWarning	CorruptDataWarning plus angeforderte Daten Dieser Rückgabewert wird zusammen mit den Nummern der Listenelemente zurückgegeben, siehe 12.2.3.
´62 82´	UnsuccessfulSearch	UnsuccessfulSearch Das Muster wurde in keinem Rekord gefunden.
´90 00´	NoError	OK plus angeforderte Daten Dieser Rückgabewert wird zusammen mit den Nummern der Listenelemente zurückgegeben.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion search mit Erläuterung
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 83´	RecordNotFound	RecordNotFound Das mittels *recordNumber* adressierte Listenelement ist nicht vorhanden.

9.3.11 Aktion select für strukturierte Elementary Files

Die Aktion select läuft für strukturierte und transparente Elementary Files gleichartig ab. Deshalb wird hier auf die Ausführungen in 9.2.7 verwiesen.

9.3.12 Aktion terminate für strukturierte Elementary Files

Die Aktion terminate läuft für strukturierte und transparente Elementary Files gleichartig ab. Deshalb wird hier auf die Ausführungen in 9.2.9 verwiesen.

9.3.13 Aktion update für strukturierte Elementary Files

Die Aktion hat das Ziel den Inhalt eines Rekords zu ersetzen, siehe [gemSpec_COS#14.4.8].

1) Schritt 1: Die Datei wird mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   a) CardTerminated: Die Aktion update bricht mit dem Rückgabewert CardTerminated ab.
   b) FileDeactivated: Die Aktion update fährt mit Schritt 2 fort.
   c) ObjectNotFound: Die Aktion update bricht mit dem Rückgabewert ObjectNotFound ab.
   d) ObjectTerminated: Die Aktion update fährt mit Schritt 2 fort.
   e) OK: Die Aktion update fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Falls *recordNumber* kleiner als 1 oder größer als 254 ist, bricht die Aktion update mit dem Rückgabewert RecordNotFound ab. Andernfalls fährt die Aktion update mit Schritt 4 fort.
4) Schritt 4: Falls der Parameter *newData* mehr als 65.535 Oktette enthält, bricht die Aktion update mit dem Rückgabewert WrongRecordLength ab. Andernfalls fährt die Aktion mit Schritt 5 fort.
5) Schritt 5: Falls der Parameter *newData* so viele Oktette enthält, dass die maximale Puffergröße (siehe Kapitel 3 Punkt (5) überschritten würde, bricht die Aktion update mit dem Rückgabewert BufferTooSmall ab. Andernfalls fährt die Aktion mit Schritt 6 fort.
6) Schritt 6: Es wird ein UPDATE RECORD Kommandos gemäß [gemSpec_COS#(N070.300)] zur Karte geschickt. Die Trailer des UPDATE RECORD Kommandos gemäß [gemSpec_COS#Tab.129, Tab.130] werden gemäß Tabelle 42 auf die Rückgabewerte der Aktion update abgebildet. Die Aktion update fährt mit Schritt 7 fort.
7) Schritt 7: Die Aktion update gibt den unter Schritt 6 ermittelten Rückgabewert zurück.

Tabelle 42: Rückgabewerte UPDATE RECORD, Typ strukturiertes Elementary File

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion update mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Rekordinhalt erfolgreich ersetzt, siehe auch 12.2.11.
´90 00´	NoError	OK Rekordinhalt erfolgreich ersetzt.
´62 87´	RecordDeactivated	RecordDeactivated Rekordinhalt nicht änderbar, da Rekord deaktiviert.
´65 81´	MemoryFailure	MemoryFailure siehe auch 12.2.8.
´67 00´	WrongRecordLength	WrongRecordLength Dieser Rückgabewert weist darauf hin, dass *newData* eine Länge besitzt, die von der adressierten Datei grundsätzlich nicht unterstützt wird.
´69 81´	WrongFileType	Dieser Trailer ist hier irrelevant, da von einer konsistenten Datenbank ausgegangen wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 82´	FileNotFound	Dieser Trailer ist hier irrelevant, da von einem erfolgreichen Schritt 1 ausgegangen wird.
´6A 83´	RecordNotFound	RecordNotFound Das mittels *recordNumber* adressierte Listenelement ist nicht vorhanden.
´6A 84´	OutOfMemory	OutOfMemory Dieser Rückgabewert weist darauf hin, dass alle Elemente der Liste der Rekords plus *newData* zusammengenommen zu viele Oktette enthalten.

9.4 cardOperation für Passwortobjekte

Dieses Kapitel beschreibt Aktionen für den Objekttyp Passwortobjekt.

9.4.1 Aktion activate für Passwortobjekte

Diese Aktion hat das Ziel das adressierte Passwortobjekt in den Zustand „Operational state (active)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.1.4].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion activate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion activate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion activate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion activate fährt mit Schritt 2 fort.
      v) OK: Die Aktion activate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.

3) Schritt 3: Es wird ein ACTIVATE Kommando gemäß [gemSpec_COS#(N034.834)] zur Karte geschickt. Die Trailer des ACTIVATE Kommandos gemäß [gemSpec_COS#Tab.28, Tab.29] werden gemäß Tabelle 43 auf die Rückgabewerte der Aktion activate abgebildet. Die Aktion activate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion activate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 43: Rückgabewerte ACTIVATE, Typ Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Passwortobjektes besitzt den Wert „Operational state (active)“, siehe 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Passwortobjektes besitzt den Wert „Operational state (active)“.
´64 00´	ObjectTerminated	ObjectTerminated Das Objekt ist nicht aktivierbar, da es terminiert ist
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.2 Aktion change für Passwortobjekte

Diese Aktion hat das Ziel das Attribut *secret* des Passwortobjektes zu ändern oder zu setzen, siehe [gemSpec_COS#(N015.200), 14.6.1.1, 14.6.1.2].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion change bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion change fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion change bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion change fährt mit Schritt 2 fort.
      v) OK: Die Aktion change fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird eine Funktion der Schnittstelle „Umgebung“ mit folgenden Parametern aufgerufen:
   a) Identifikator: Dieser Parameter ist identisch zum gleichnamigen Aufrufparameter in cardOperation. Er informiert die Umgebung darüber, für welches Passwortobjekt die Aktion durchzuführen ist. Die Umgebung ist somit in der Lage für dieses Passwortobjekt spezifische Texte an der Benutzeroberfläche anzuzeigen.
   b) Aktion=mode: Dieser Parameter informiert die Umgebung darüber, ob das Attribut secret zu ändern ist (*mode*=replace, siehe 2.2.3), oder ob das Attribut secret zu

setzen ist (*mode*=set, siehe 2.2.7). Die Umgebung ist somit in der Lage der Aktion angemessene Texte an der Benutzeroberfläche anzuzeigen.

c) minimumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.

d) maximumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.

e) commandApduPart: Teil der Kommando APDU, die Umgebung ergänzt diesen Teil um das Benutzergeheimnis PIN, siehe 2.2.3 und 2.2.7.

4) Schritt 4: Die Funktion gibt die Antwortnachricht des CHANGE REFERENCE DATA Kommandos zurück. Die Trailer des CHANGE REFERENCE DATA Kommandos gemäß [gemSpec_COS#Tab.133, Tab.134] werden gemäß Tabelle 44 auf die Rückgabewerte der Aktion change abgebildet. Die Aktion change fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion change gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 44: Rückgabewerte CHANGE REFERENCE DATA, Typ Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion change mit Erläuterung
´63 Cx´ ´63 CF´ … ´63 C1´ ´63 C0´	WrongSecretWarning = 15 oder mehr Versuche … ein weiterer Versuch kein weiterer Versuch	WrongSecretWarning Keine Änderung des PIN-Wertes, da der alte Wert falsch war. Der Wert X gibt an, wie viele Versuche dem Benutzer zum Ändern des PIN-Wertes noch zur Verfügung stehen.
´90 00´	NoError	OK Erfolgreiches Setzen des Benutzergeheimnisses auf neuen Wert
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 83´	PasswordBlocked	PasswordBlocked Abgelaufener Fehlbedienungszähler
´69 85´	LongPassword ShortPassword	WrongLength PIN nicht geändert, da neuer Wert falsche Länge besitzt
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.3 Aktion deactivate für Passwortobjekte

Diese Aktion hat das Ziel das adressierte Passwortobjekt in den Zustand „Operational state (deactivated)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.3.4].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion deactivate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion deactivate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion deactivate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion deactivate fährt mit Schritt 2 fort.
      v) OK: Die Aktion deactivate fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DEACTIVATE Kommando gemäß [gemSpec_COS#(N036.034)] zur Karte geschickt. Die Trailer des DEACTIVATE Kommandos gemäß [gemSpec_COS#Tab.34, Tab.35] werden gemäß Tabelle 45 auf die Rückgabewerte der Aktion deactivate abgebildet. Die Aktion deactivate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion deactivate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 45: Rückgabewerte DEACTIVATE, Typ Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deactivate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Passwortobjektes besitzt den Wert „Operational state (deactivated)“, siehe auch 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Passwortobjektes besitzt den Wert „Operational state (deactivated)“.
´64 00´	ObjectTerminated	ObjectTerminated Das Objekt ist nicht deaktivierbar, da es terminiert ist
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.4 Aktion delete für Passwortobjekte

Diese Aktion hat das Ziel das adressierte Passwortobjekt zu löschen, siehe [gemSpec_COS14.2.4.4].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion delete bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion delete fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion delete bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion delete fährt mit Schritt 2 fort.
      v) OK: Die Aktion delete fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DELETE Kommando gemäß [gemSpec_COS#(N037.134)] zur Karte geschickt. Die Trailer des DELETE Kommandos gemäß [gemSpec_COS#Tab.40, Tab.41] werden gemäß Tabelle 46 auf die Rückgabewerte der Aktion delete abgebildet. Die Aktion delete fährt mit Schritt 4 fort.

4) Schritt 4: Die Aktion delete gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 46: Rückgabewerte DELETE, Typ Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Passwortobjekt wurde gelöscht, siehe auch 12.2.11.
´90 00´	NoError	OK Das Passwortobjekt wurde gelöscht.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.5 Aktion disable für Passwortobjekte

Diese Aktion hat das Ziel die Notwendigkeit einer Benutzerverifikation auszuschalten, siehe [gemSpec_COS#(N015.700), (N022.200)a.2, 14.6.2.1]. Nachdem die Aktion erfolgreich abgeschlossen ist, verhält sich das adressierte Passwortobjekt so, als sei eine erfolgreiche Benutzerverifikation erfolgt.

Hinweis (68): Gemäß [gemSpec_COS#14.6.2.2] unterstützt das COS auch eine Variante ohne Angabe einer PIN in der Kommandonachricht. Diese Variante wird in der hier beschriebenen Version von Card Proxy absichtlich nicht unterstützt.

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion disable bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion disable fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion disable bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion disable fährt mit Schritt 2 fort.
      v) OK: Die Aktion disable fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird eine Funktion der Schnittstelle „Umgebung“ mit folgenden Parametern aufgerufen:
   a) Identifikator: Dieser Parameter ist identisch zum gleichnamigen Aufrufparameter in cardOperation. Er informiert die Umgebung darüber, für welches Passwortobjekt die Aktion durchzuführen ist. Die Umgebung ist somit in der Lage für dieses Passwortobjekt spezifische Texte an der Benutzeroberfläche anzuzeigen.
   b) Aktion=disable: Dieser Parameter informiert die Umgebung darüber, dass das adressierte Passwortobjekt ausgeschaltet werden soll. Die Umgebung ist somit in der Lage der Aktion angemessene Texte an der Benutzeroberfläche anzuzeigen.
   c) minimumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der

Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.

d) maximumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.

e) commandApduPart: Teil der Kommando APDU, die Umgebung ergänzt diesen Teil um das Benutzergeheimnis PIN, siehe 2.2.4.

4) Schritt 4: Die Funktion gibt die Antwortnachricht des DISABLE VERIFICATION REQUIREMENT Kommandos zurück. Die Trailer des DISABLE VERIFICATION REQUIREMENT Kommandos gemäß [gemSpec_COS#Tab.137, Tab.138] werden gemäß Tabelle 47 auf die Rückgabewerte der Aktion disable abgebildet. Die Aktion disable fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion disable gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 47: Rückgabewerte DISABLE VERIFICATION REQUIREMENT**, Typ Passwortobjekt**

Trailer gemäß [gemSpec_COS]		**Rückgabewert der Aktion disable mit Erläuterung**
´63 Cx´ ´63 CF´ … ´63 C1´ ´63 C0´	WrongSecretWarning = 15 oder mehr Versuche … ein weiterer Versuch kein weiterer Versuch	WrongSecretWarning Ausschalten fehlgeschlagen Der Wert X gibt an, wie viele Versuche dem Benutzer zur richtigen PIN-Eingabe noch zur Verfügung stehen.
´90 00´	NoError	OK Notwendigkeit der Benutzerverifikation ausgeschaltet
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 83´	PasswordBlocked	PasswordBlocked Abgelaufener Fehlbedienungszähler
´69 85´	PasswordNotUsable	PasswordProtected Passwort mit Transportschutz versehen
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.6 Aktion enable für Passwortobjekte

Diese Aktion hat das Ziel die Notwendigkeit einer Benutzerverifikation einzuschalten, siehe [gemSpec_COS#(N015.700), (N022.200)a.2, 14.6.3.1]. Nachdem die Aktion erfolgreich abgeschlossen ist, ist ein Setzen des Sicherheitszustandes durch Benutzerverifikation wieder erforderlich.

Hinweis (69): Gemäß [gemSpec_COS#14.6.3.2] unterstützt das COS auch eine Variante ohne Angabe einer PIN in der Kommandonachricht. Diese Variante wird in der hier beschriebenen Version von Card Proxy absichtlich nicht unterstützt.

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion enable bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion enable fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion enable bricht mit dem Rückgabewert ObjectNotFound ab.

iv) ObjectTerminated: Die Aktion enable fährt mit Schritt 2 fort.
v) OK: Die Aktion enable fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird eine Funktion der Schnittstelle „Umgebung“ mit folgenden Parametern aufgerufen:
   a) Identifikator: Dieser Parameter ist identisch zum gleichnamigen Aufrufparameter in cardOperation. Er informiert die Umgebung darüber, für welches Passwortobjekt die Aktion durchzuführen ist. Die Umgebung ist somit in der Lage für dieses Passwortobjekt spezifische Texte an der Benutzeroberfläche anzuzeigen.
   b) Aktion=enable: Dieser Parameter informiert die Umgebung darüber, dass das adressierte Passwortobjekt eingeschaltet werden soll. Die Umgebung ist somit in der Lage der Aktion angemessene Texte an der Benutzeroberfläche anzuzeigen.
   c) minimumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.
   d) maximumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.
   e) commandApduPart: Teil der Kommando APDU, die Umgebung ergänzt diesen Teil um das Benutzergeheimnis PIN, siehe 2.2.6.
4) Schritt 4: Die Funktion gibt die Antwortnachricht des ENABLE VERIFICATION REQUIREMENT Kommandos zurück. Die Trailer des ENABLE VERIFICATION REQUIREMENT Kommandos gemäß [gemSpec_COS#Tab.141, Tab.142] werden gemäß Tabelle 48 auf die Rückgabewerte der Aktion enable abgebildet. Die Aktion enable fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion enable gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 48: Rückgabewerte ENABLE VERIFICATION REQUIREMENT, Typ Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion enable mit Erläuterung
´63 Cx´ ´63 CF´ … ´63 C1´ ´63 C0´	WrongSecretWarning = 15 oder mehr Versuche … ein weiterer Versuch kein weiterer Versuch	WrongSecretWarning Einschalten fehlgeschlagen Der Wert X gibt an, wie viele Versuche dem Benutzer zur richtigen PIN-Eingabe noch zur Verfügung stehen.
´90 00´	NoError	OK Notwendigkeit der Benutzerverifikation eingeschaltet
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 83´	PasswordBlocked	PasswordBlocked Abgelaufener Fehlbedienungszähler
´69 85´	PasswordNotUsable	PasswordProtected Passwort mit Transportschutz versehen
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.7 Aktion getStatus für Passwortobjekte

Diese Aktion hat das Ziel den Status des adressierten Passwortobjektes abzufragen, siehe [gemSpec_COS#(N015.600), 15.700), 14.6.4].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion getStatus bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion getStatus fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion getStatus bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion getStatus fährt mit Schritt 2 fort.
      v) OK: Die Aktion getStatus fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein GET PIN STATUS Kommando gemäß [gemSpec_COS#(N077.900)] zur Karte geschickt. Die Trailer des GET PIN STATUS Kommandos gemäß [gemSpec_COS#Tab.144, Tab.145] werden gemäß Tabelle 49 auf die Rückgabewerte der Aktion getStatus abgebildet. Die Aktion getStatus fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion getStatus gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 49: Rückgabewerte GET PIN STATUS, TYP **Passwortobjekt**

Trailer gemäß [gemSpec_COS]		**Rückgabewert der Aktion getStatus mit Erläuterung**
´62 C1´	TransportStatus = Transport-PIN	PasswordProtected Das Attribut *transportStatus* des Passwortobjektes besitzt den Wert Transport-PIN. Das Attribut *flagEnabled* besitzt den Wert true. Sicherheitszustand des Passwortobjektes nicht gesetzt.
´62 C7´	TransportStatus = Leer-PIN	PasswordProtected Das Attribut *transportStatus* des Passwortobjektes besitzt den Wert Leer-PIN. Das Attribut *flagEnabled* besitzt den Wert true. Sicherheitszustand des Passwortobjektes nicht gesetzt.
´62 D0´	PasswordDisabled	PasswordDisabled Das Attribut *flagEnabled* besitzt den Wert false. Eine Verifikation ist nicht erforderlich.
´63 Cx´ ´63 CF´ … ´63 C1´	RetryCounter = 15 oder mehr … eins	RetryCounter.X Das Attribut *transportStatus* des Passwortobjektes besitzt den Wert regularPassword. Das Attribut *flagEnabled* besitzt den Wert true. Sicherheitszustand des Passwortobjektes nicht gesetzt. Der Wert X gibt an, wie viele Versuche der Benutzer zur richtigen PIN-Eingabe noch hat.
´90 00´	NoError	OK Das Attribut *transportStatus* des Passwortobjektes besitzt den Wert regularPassword. Das Attribut *flagEnabled* besitzt den Wert true. Der Sicherheitszustand des Passwortobjektes ist gesetzt.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.8 Aktion terminate für Passwortobjekte

Diese Aktion hat das Ziel das adressierte Passwortobjekt in den Zustand „Termination state“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.9.4].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion terminate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion terminate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion terminate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion terminate fährt mit Schritt 2 fort.
      v) OK: Die Aktion terminate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein TERMINATE Kommando gemäß [gemSpec_COS#(N048.934)] zur Karte geschickt. Die Trailer des TERMINATE Kommandos gemäß [gemSpec_COS#Tab.73, Tab.74] werden gemäß Tabelle 50 auf die Rückgabewerte der Aktion terminate abgebildet. Die Aktion terminate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion terminate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 50: Rückgabewerte TERMINATE, TYP Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion terminate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribut *lifeCycleStatus* des Passwortobjektes besitzt den Wert „Termination state“, siehe 12.2.11.
´90 00´	NoError	OK Das Attribut *lifeCycleStatus* des Passwortobjektes besitzt den Wert „Termination state“.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´6A 88´	KeyNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Passwortobjekt behandelt wird.
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.9 Aktion unblock für Passwortobjekte

Diese Aktion hat das Ziel die Blockade des adressierten Passwortobjektes aufzuheben, siehe [gemSpec_COS#14.6.5].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.

b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   i) CardTerminated: Die Aktion unblock bricht mit dem Rückgabewert CardTerminated ab.
   ii) FileDeactivated: Die Aktion unblock fährt mit Schritt 2 fort.
   iii) ObjectNotFound: Die Aktion unblock bricht mit dem Rückgabewert ObjectNotFound ab.
   iv) ObjectTerminated: Die Aktion unblock fährt mit Schritt 2 fort.
   v) OK: Die Aktion unblock fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird eine Funktion der Schnittstelle „Umgebung“ mit folgenden Parametern aufgerufen:
   a) Identifikator: Dieser Parameter ist identisch zum gleichnamigen Aufrufparameter in cardOperation. Er informiert die Umgebung darüber, für welches Passwortobjekt die Aktion durchzuführen ist. Die Umgebung ist somit in der Lage für dieses Passwortobjekt spezifische Texte an der Benutzeroberfläche anzuzeigen.
   b) Aktion=mode: Dieser Parameter informiert die Umgebung darüber, welche Benutzergeheimnisse in den Kommandodaten erwartet werden. Die Umgebung ist somit in der Lage der Aktion angemessene Texte an der Benutzeroberfläche anzuzeigen.
      i) *mode*=UnblockWithPukAndSet: Die Kommandodaten enthalten einen PUK und den neuen Wert der PIN, siehe 2.2.8.
      ii) *mode*=UnblockWithPuk: Die Kommandodaten enthalten nur eine PUK, aber keinen neuen Wert für die PIN, siehe 2.2.9.
      iii) *mode*=UnblockAndSet: Die Kommandodaten enthalten nur einen neuen PIN-Wert, aber keine PUK, siehe 2.2.10.
      iv) *mode*=Unblock: Die Kommandodaten sind leer, siehe 2.2.11.
   c) minimumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.
   d) maximumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.
   e) commandApduPart: Teil der Kommando APDU, die Umgebung ergänzt diesen Teil je nach *mode* um PUK und/oder PIN.
      i) *mode*=UnblockWithPukAndSet: siehe 2.2.8
      ii) *mode*=UnblockWithPuk: siehe 2.2.9
      iii) *mode*=UnblockAndSet: siehe 2.2.10
      iv) *mode*=Unblock: siehe 2.2.11
4) Schritt 4: Die Funktion gibt die Antwortnachricht des RESET RETRY COUNTER Kommandos zurück. Die Trailer des RESET RETRY COUNTER Kommandos gemäß [gemSpec_COS#Tab.150, Tab.151] werden gemäß Tabelle 51 auf die Rückgabewerte der Aktion unblock abgebildet. Die Aktion unblock fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion unblock gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 51: Rückgabewerte RESET RETRY COUNTER, Typ Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion unblock mit Erläuterung
´63 Cx´ ´63 CF´ … ´63 C1´ ´63 C0´	WrongSecretWarning = 15 oder mehr Versuche … ein weiterer Versuch kein weiterer Versuch	WrongSecretWarning Blockade besteht fort, da PUK falsch Der Wert X gibt an, wie viele Versuche dem Benutzer zum entsperren der PIN noch zur Verfügung stehen.
´90 00´	NoError	OK Blockade aufgehoben
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 83´	CommandBlocked	PasswordBlocked Abgelaufener Bedienzähler der PUK
´69 85´	LongPassword ShortPassword	WrongLength Blockade nicht aufgehoben, da neuer Wert falsche Länge besitzt
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.4.10 Aktion verify für Passwortobjekte

Diese Aktion hat das Ziel den Sicherheitszustand des adressierten Passwortobjektes zu setzen, siehe [gemSpec_COS#12.5, 14.6.6.1].

1) Schritt 1: Falls das adressierte Passwortobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Passwortobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion verify bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion verify fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion verify bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion verify fährt mit Schritt 2 fort.
      v) OK: Die Aktion verify fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird eine Funktion der Schnittstelle „Umgebung“ mit folgenden Parametern aufgerufen:
   a) Identifikator: Dieser Parameter ist identisch zum gleichnamigen Aufrufparameter in cardOperation. Er informiert die Umgebung darüber, für welches Passwortobjekt die Aktion durchzuführen ist. Die Umgebung ist somit in der Lage für dieses Passwortobjekt spezifische Texte an der Benutzeroberfläche anzuzeigen.
   b) Aktion=verify: Dieser Parameter informiert die Umgebung darüber, dass eine Benutzerverifikation erfolgt. Die Umgebung ist somit in der Lage der Aktion angemessene Texte an der Benutzeroberfläche anzuzeigen.
   c) minimumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.
   d) maximumLength: Dieser Parameter informiert die Umgebung über die Mindestlänge des Benutzergeheimnisses (PIN). Card Proxy entnimmt diesen Wert aus der

Datenbank. Es wird empfohlen, dass die Umgebung diesen Parameter zu Plausibilitätsprüfungen nutzt.

e) commandApduPart: Teil der Kommando APDU, die Umgebung ergänzt diesen Teil um das Benutzergeheimnis PIN, siehe 2.2.5.

4) Schritt 4: Die Funktion gibt die Antwortnachricht des VERIFY Kommandos zurück. Die Trailer des VERIFY Kommandos gemäß [gemSpec_COS#Tab.153, Tab.154] werden gemäß Tabelle 52 auf die Rückgabewerte der Aktion verify abgebildet. Die Aktion verify fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion verify gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 52: Rückgabewerte VERIFY, Typ Passwortobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion verify mit Erläuterung
´63 Cx´ ´63 CF´ … ´63 C1´ ´63 C0´	WrongSecretWarning = 15 oder mehr Versuche … ein weiterer Versuch kein weiterer Versuch	WrongSecretWarning Benutzerverifikation nicht erfolgreich Der Wert X gibt an, wie viele Versuche dem Benutzer zur richtigen PIN-Eingabe noch zur Verfügung stehen.
´90 00´	NoError	OK Benutzerverifikation erfolgreich durchgeführt
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 83´	PasswordBlocked	PasswordBlocked Abgelaufener Fehlbedienungszähler
´69 85´	PasswordNotUsable	PasswordProtected Passwort mit Transportschutz versehen
´6A 88´	PasswordNotFound	ObjectNotFound Adressiertes Passwortobjekt wurde nicht gefunden.

9.5 cardOperation für private Schlüsselobjekte

Dieses Kapitel beschreibt Aktionen für den Objekttyp privates Schlüsselobjekt.

9.5.1 Aktion activate für private Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt in den Zustand „Operational state (active)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.1.2].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion activate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion activate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion activate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion activate fährt mit Schritt 2 fort.
      v) OK: Die Aktion activate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.

3) Schritt 3: Es wird ein ACTIVATE Kommando gemäß [gemSpec_COS#(N034.814)] zur Karte geschickt. Die Trailer des ACTIVATE Kommandos gemäß [gemSpec_COS#Tab.28, Tab.29] werden gemäß Tabelle 53 auf die Rückgabewerte der Aktion activate abgebildet. Die Aktion activate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion activate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 53: Rückgabewerte ACTIVATE, Typ privates Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (active)“, siehe 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (active)“.
´64 00´	ObjectTerminated	ObjectTerminated Das Objekt ist nicht aktivierbar, da es terminiert ist
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Typ privates Schlüsselobjekt behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.5.2 Aktion deactivate für private Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt in den Zustand „Operational state (deactivated)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.3.2].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion deactivate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion deactivate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion deactivate bricht mit dem Rückgabewert Object-NotFound ab.
      iv) ObjectTerminated: Die Aktion deactivate fährt mit Schritt 2 fort.
      v) OK: Die Aktion deactivate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DEACTIVATE Kommando gemäß [gemSpec_COS#(N036.014)] zur Karte geschickt. Die Trailer des DEACTIVATE Kommandos gemäß [gemSpec_COS#Tab.34, Tab.35] werden gemäß Tabelle 54 auf die Rückgabewerte der Aktion deactivate abgebildet. Die Aktion deactivate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion deactivate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 54: Rückgabewerte DEACTIVATE, Typ privates Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deactivate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (deactivated)“, siehe auch 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (deactivated)“.
´64 00´	ObjectTerminated	ObjectTerminated Das Objekt ist nicht deaktivierbar, da es terminiert ist
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Typ privates Schlüsselobjekt behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.5.3 Aktion delete für private Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt zu löschen, siehe [gemSpec_COS14.2.4.2].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion delete bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion delete fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion delete bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion delete fährt mit Schritt 2 fort.
      v) OK: Die Aktion delete fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DELETE Kommando gemäß [gemSpec_COS#(N037.114)] zur Karte geschickt. Die Trailer des DELETE Kommandos gemäß [gemSpec_COS#Tab.40, Tab.41] werden gemäß Tabelle 46 auf die Rückgabewerte der Aktion delete abgebildet. Die Aktion delete fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion delete gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 55: Rückgabewerte DELETE, Typ privates Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Schlüsselobjekt wurde gelöscht, siehe auch 12.2.11.
´90 00´	NoError	OK Das Schlüsselobjekt wurde gelöscht.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	ObjectNotFound

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
		Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.5.4 Aktion elcRoleAuthentication für private Schlüsselobjekte

Diese Aktion hat das Ziel das als Parameter übergebene *token* mit dem adressierten Schlüsselobjekt zu signieren, siehe [gemSpec_COS#(N086.200)a, (N086.900)a].

1) Schritt 1: Die Aktion elcRoleAuthentication bricht mit dem Rückgabewert WrongToken ab, falls
   a) *token* eine Länge hat, die laut Datenbank nicht erlaubt ist, oder
   b) die acht LSByte von *token* identisch sind zu den acht LSByte der ICCSN von der Smartcard, mit der Card Proxy in Verbindung steht.
2) Schritt 2: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 3 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion elcRoleAuthentication bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion elcRoleAuthentication fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion elcRoleAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion elcRoleAuthentication fährt mit Schritt 3 fort.
      v) OK: Die Aktion elcRoleAuthentication fährt mit Schritt 3 fort.
3) Schritt 3: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus elcRoleAuthentication
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N100.900)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im channelContext von Card Proxy passend eingetragen. Die Aktion elcRoleAuthentication fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion elcRoleAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
      iii) KeyNotFound: Die Aktion elcRoleAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
4) Schritt 4: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
5) Schritt 5: Es wird ein INTERNAL AUTHENTICATE Kommando gemäß [gemSpec_COS#(N086.400)] zur Karte geschickt. Die Trailer des INTERNAL AUTHENTICATE Kommandos gemäß [gemSpec_COS#Tab.181, Tab.182] werden gemäß Tabelle 56 auf die Rückgabewerte der Aktion elcRoleAuthentication abgebildet. Die Aktion elcRoleAuthentication fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion elcRoleAuthentication gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 56: Rückgabewerte INTERNAL AUTHENTICATE**, Typ privates Schlüsselobjekt**

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion elcRoleAuthentication mit Erläuterung
´90 00´	NoError	OK plus Signatur über das *token* Das *token* wurde erfolgreich signiert.
´64 00´	KeyInvalid	KeyInvalid Schlüsselobjekt enthält keine Schlüsseldaten
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Dieser Trailer sollte nach einem erfolgreichen Schritt 4 nicht mehr vorkommen.
´69 85´	NoKeyReference	Dieser Trailer ist wegen Schritt 3 irrelevant.
´69 85´	NoPukReference	Dieser Trailer ist für elcRoleAuthentication irrelevant.
`6A 80´	WrongToken	Dieser Trailer ist wegen Schritt 1 irrelevant.
´6A 81´	UnsupportedFunction	ObjectNotFound Schlüssel unterstützt den angegeben Algorithmus nicht
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PukNotFound	Dieser Trailer ist für elcRoleAuthentication irrelevant.

9.5.5 Aktion elcSharedSecretCalculation für private Schlüsselobjekte

Diese Aktion hat das Ziel das als Parameter übergebene Kryptogramm *cipher* mit dem adressierten Schlüsselobjekt zu entschlüsseln, siehe [gemSpec_COS#(N089.800), (N090.300)c]. Das Kryptogramm *cipher* ist mit dem in [gemSpec_COS#14.8.4.2] beschriebenen Verfahren verschlüsselt.

1) Schritt 1: Die Aktion elcSharedSecretCalculation bricht mit dem Rückgabewert WrongCryptogram ab, falls *cipher* nicht gemäß [gemSpec_COS#14.8.3.2] kodiert ist.
2) Schritt 2: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 3 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion elcSharedSecretCalculation bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion elcSharedSecretCalculation fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion elcSharedSecretCalculation bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion elcSharedSecretCalculation fährt mit Schritt 3 fort.
      v) OK: Die Aktion elcSharedSecretCalculation fährt mit Schritt 3 fort.
3) Schritt 3: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus elcSharedSecretCalculation
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N103.800)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im *channelContext* von Card Proxy passend eingetragen. Die Aktion elcSharedSecretCalculation fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion elcSharedSecretCalculation bricht mit dem Rückgabewert ObjectNotFound ab.
      iii) KeyNotFound: Die Aktion elcSharedSecretCalculation bricht mit dem Rückgabewert ObjectNotFound ab.

4) Schritt 4: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
5) Schritt 5: Es wird ein PSO Decipher Kommando gemäß [gemSpec_COS#(N089.800)] zur Karte geschickt. Die Trailer des PSO Decipher Kommandos gemäß [gemSpec_COS#Tab.189, Tab.190] werden gemäß Tabelle 57 auf die Rückgabewerte der Aktion rsaDecipherOaep abgebildet. Die Aktion rsaDecipherOaep fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion rsaDecipherOaep gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 57: Rückgabewerte PSO Decipher, Typ privates ELC Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion rsaClientAuthentication + Erläuterung
´90 00´	NoError	OK plus Klartext aus dem Kryptogramm *cipher* Das Kryptogramm *cipher* wurde erfolgreich entschlüsselt.
´64 00´	KeyInvalid	KeyInvalid Schlüsselobjekt enthält keine Schlüsseldaten
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Dieser Trailer sollte nach einem erfolgreichen Schritt 4 nicht mehr vorkommen.
´69 85´	NoKeyReference	Dieser Trailer ist wegen Schritt 3 irrelevant.
´6A 80´	WrongCiphertext	WrongCiphertext Das Kryptogramm konnte nicht entschlüsselt werden.
´6A 81´	UnsupportedFunction	ObjectNotFound Schlüssel unterstützt den angegeben Algorithmus nicht
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.

9.5.6 Aktion generate für private Schlüsselobjekte

Diese Aktion hat das Ziel ein neues Schlüsselpaar zu erzeugen, siehe [gemSpec_COS#14.9.3].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion generate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion generate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion generate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion generate fährt mit Schritt 2 fort.
      v) OK: Die Aktion generate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein GENERATE ASYMMETRIC KEY PAIR Kommando gemäß dem übergebenen Modus zur Karte geschickt:
   a) *mode* = create+read => [gemSpec_COS#(N097.246)]
   b) *mode* = create => [gemSpec_COS#(N096.644)]
   c) *mode* = replace+read => [gemSpec_COS#(N097.266)]
   d) *mode* = replace => [gemSpec_COS#(N096.664)]

4) Schritt 4: Die Trailer des GENERATE ASYMMETRIC KEY PAIR Kommandos gemäß [gemSpec_COS#Tab.234, Tab.235] werden gemäß Tabelle 58 auf die Rückgabewerte der Aktion generate abgebildet. Die Aktion generate fährt mit Schritt 5 fort.
5) Schritt 5: Die Aktion generate gibt den unter Schritt 4 ermittelten Rückgabewert zurück.

Tabelle 58: Rückgabewerte GAKP create/replace, Typ privates Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion generate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Schlüsselpaar erfolgreich erzeugt, siehe auch 12.2.11.
´90 00´	NoError	OK plus (optional) öffentlicher Schlüssel Der öffentliche Schlüssel wurde erfolgreich erzeugt.
´64 00´	KeyInvalid	Dieser Trailer ist für Schlüsselgenerierung irrelevant.
´65 81´	MemoryFailure	MemoryFailure siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 85´	KeyAlreadyPresent	KeyAlreadyPresent Der gewählte Modus gestattet das Überschreiben nicht.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.

9.5.7 Aktion readPublicPart für private Schlüsselobjekte

Diese Aktion hat das Ziel den öffentlichen Teil des adressierten, privaten Schlüsselobjektes auszulesen, siehe [gemSpec_COS#(N096.946)].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion readPublicPart bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion readPublicPart fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion readPublicPart bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion readPublicPart fährt mit Schritt 2 fort.
      v) OK: Die Aktion readPublicPart fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein GENERATE ASYMMETRIC KEY PAIR Kommando gemäß [gemSpec_COS#(N096.946)] zur Karte geschickt. Die Trailer des GENERATE ASYMMETRIC KEY PAIR Kommandos gemäß [gemSpec_COS#Tab.234, Tab.235] werden gemäß Tabelle 59 auf die Rückgabewerte der Aktion readPublicPart abgebildet. Die Aktion readPublicPart fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion readPublicPart gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 59: Rückgabewerte GAKP Read, Typ privates Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion readPublicPart mit Erläuterung
´63 Cx´	UpdateRetryWarning	Dieser Trailer ist für Leseoperationen irrelevant.
´90 00´	NoError	OK plus öffentlicher Schlüssel Der öffentliche Schlüssel wurde erfolgreich ausgelesen.

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion readPublicPart mit Erläuterung
´64 00´	KeyInvalid	KeyInvalid Schlüsselobjekt enthält keine Schlüsseldaten
´65 81´	MemoryFailure	Dieser Trailer ist für Leseoperationen irrelevant.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 85´	KeyAlreadyPresent	Dieser Trailer ist für Leseoperationen irrelevant.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.

9.5.8 Aktion rsaClientAuthentication für private Schlüsselobjekte

Diese Aktion hat das Ziel das als Parameter übergebene *token* mit dem adressierten Schlüsselobjekt zu signieren, siehe [gemSpec_COS#(N087.300)a, (N088.600)a].

Hinweis (70): Die Umsetzung dieser Aktion verwendet absichtlich lediglich das PSO Compute Digital Signature Kommando und nicht das funktionsgleiche INTERNAL AUTHENTICATE Kommando.

1) Schritt 1: Die Aktion rsaClientAuthentication bricht mit dem Rückgabewert WrongToken ab, falls *token* eine Länge hat, die laut Datenbank nicht erlaubt ist.
2) Schritt 2: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 3 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion rsaClientAuthentication bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion rsaClientAuthentication fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion rsaClientAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion rsaClientAuthentication fährt mit Schritt 3 fort.
      v) OK: Die Aktion rsaClientAuthentication fährt mit Schritt 3 fort.
3) Schritt 3: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus rsaClientAuthentication
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N102.900)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im channelContext von Card Proxy passend eingetragen. Die Aktion rsaClientAuthentication fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion rsaClientAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
      iii) KeyNotFound: Die Aktion rsaClientAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
4) Schritt 4: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
5) Schritt 5: Es wird ein PSO Compute Digital Signature Kommando gemäß [gemSpec_COS#(N087.500)] zur Karte geschickt. Die Trailer des PSO Compute Digital Signature Kommandos gemäß [gemSpec_COS#Tab.189, Tab.190] werden gemäß Tabelle 60 auf die Rückgabewerte der Aktion rsaClientAuthentication abgebildet. Die Aktion rsaClientAuthentication fährt mit Schritt 6 fort.

6) Schritt 6: Die Aktion rsaClientAuthentication gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 60: Rückgabewerte PSO Compute Digital Sign., Typ privates Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion rsaClientAuthentication + Erläuterung
´90 00´	NoError	OK plus Signatur über das *token* Das *token* wurde erfolgreich signiert.
´64 00´	KeyInvalid	KeyInvalid Schlüsselobjekt enthält keine Schlüsseldaten
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Dieser Trailer sollte nach einem erfolgreichen Schritt 4 nicht mehr vorkommen.
´69 85´	NoKeyReference	Dieser Trailer ist wegen Schritt 3 irrelevant.
´6A 81´	UnsupportedFunction	ObjectNotFound Schlüssel unterstützt den angegeben Algorithmus nicht
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.

9.5.9 Aktion rsaDecipherOaep für private Schlüsselobjekte

Diese Aktion hat das Ziel das als Parameter übergebene Kryptogramm *C* mit dem adressierten Schlüsselobjekt zu entschlüsseln, siehe [gemSpec_COS#(N089.200), (N090.300)b]. Das Kryptogramm *C* ist mit dem Verfahren RSAES-OAEP gemäß [PKCS #1] Kapitel 7.1 verschlüsselt.

1) Schritt 1: Die Aktion rsaDecipherOaep bricht mit dem Rückgabewert WrongCryptogram ab, falls *C* eine Länge hat, die laut Datenbank nicht erlaubt ist.
2) Schritt 2: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 3 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion rsaDecipherOaep bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion rsaDecipherOaep fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion rsaDecipherOaep bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion rsaDecipherOaep fährt mit Schritt 3 fort.
      v) OK: Die Aktion rsaDecipherOaep fährt mit Schritt 3 fort.
3) Schritt 3: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus rsaDecipherOaep
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N103.800)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im *channelContext* von Card Proxy passend eingetragen. Die Aktion rsaDecipherOaep fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion rsaDecipherOaep bricht mit dem Rückgabewert ObjectNotFound ab.
      iii) KeyNotFound: Die Aktion rsaDecipherOaep bricht mit dem Rückgabewert ObjectNotFound ab.

4) Schritt 4: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
5) Schritt 5: Es wird ein PSO Decipher Kommando gemäß [gemSpec_COS#(N089.200)] zur Karte geschickt. Die Trailer des PSO Decipher Kommandos gemäß [gemSpec_COS#Tab.189, Tab.190] werden gemäß Tabelle 61 auf die Rückgabewerte der Aktion rsaDecipherOaep abgebildet. Die Aktion rsaDecipherOaep fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion rsaDecipherOaep gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

Tabelle 61: Rückgabewerte PSO Decipher, Typ privates RSA Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion rsaClientAuthentication + Erläuterung
´90 00´	NoError	OK plus Klartext aus dem Kryptogramm *C* Das Kryptogramm *C* wurde erfolgreich entschlüsselt.
´64 00´	KeyInvalid	KeyInvalid Schlüsselobjekt enthält keine Schlüsseldaten
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Dieser Trailer sollte nach einem erfolgreichen Schritt 4 nicht mehr vorkommen.
´69 85´	NoKeyReference	Dieser Trailer ist wegen Schritt 3 irrelevant.
´6A 80´	WrongCiphertext	WrongCiphertext Das Kryptogramm konnte nicht entschlüsselt werden.
´6A 81´	UnsupportedFunction	ObjectNotFound Schlüssel unterstützt den angegeben Algorithmus nicht
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.

9.5.10 Aktion rsaDecipherPKCS1_V1_5 für private Schlüsselobjekte

Diese Aktion hat das Ziel das als Parameter übergebene Kryptogramm *C* mit dem adressierten Schlüsselobjekt zu entschlüsseln, siehe [gemSpec_COS#(N089.200), (N090.300)a]. Das Kryptogramm *C* ist mit dem Verfahren RSAES-PKCS1-v1_5 gemäß [PKCS #1] Kapitel 7.2 verschlüsselt.

Der einzige Unterschied zur Aktion rsaDecipherOaep aus 9.5.9 ist, dass hier ein anderes Verschlüsselungsverfahren verwendet wird und deshalb in Schritt 3)b) im Rahmen der Schlüsselselektion eine andere *algId* zu verwenden ist.

9.5.11 Aktion rsaRoleAuthentication für private Schlüsselobjekte

Diese Aktion hat das Ziel das als Parameter übergebene *token* mit dem adressierten Schlüsselobjekt zu signieren, siehe [gemSpec_COS#(N086.200)c, (N086.900)c].

1) Schritt 1: Die Aktion rsaRoleAuthentication bricht mit dem Rückgabewert WrongToken ab, falls
   a) *token* eine Länge hat, die laut Datenbank nicht erlaubt ist, oder
   b) die acht LSByte von *token* identisch sind zu den acht LSByte der ICCSN von der Smartcard, mit der Card Proxy in Verbindung steht.
2) Schritt 2: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 3 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:

i) CardTerminated: Die Aktion rsaRoleAuthentication bricht mit dem Rückgabewert CardTerminated ab.
ii) FileDeactivated: Die Aktion rsaRoleAuthentication fährt mit Schritt 3 fort.
iii) ObjectNotFound: Die Aktion rsaRoleAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
iv) ObjectTerminated: Die Aktion rsaRoleAuthentication fährt mit Schritt 3 fort.
v) OK: Die Aktion rsaRoleAuthentication fährt mit Schritt 3 fort.

3) Schritt 3: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus rsaRoleAuthentication
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N100.900)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im channelContext von Card Proxy passend eingetragen. Die Aktion rsaRoleAuthentication fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion rsaRoleAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
      iii) KeyNotFound: Die Aktion rsaRoleAuthentication bricht mit dem Rückgabewert ObjectNotFound ab.
4) Schritt 4: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
5) Schritt 5: Es wird ein INTERNAL AUTHENTICATE Kommando gemäß [gemSpec_COS#(N086.400)] zur Karte geschickt. Die Trailer des INTERNAL AUTHENTICATE Kommandos gemäß [gemSpec_COS#Tab.181, Tab.182] werden gemäß Tabelle 56 auf die Rückgabewerte der Aktion rsaRoleAuthentication abgebildet. Die Aktion rsaRoleAuthentication fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion rsaRoleAuthentication gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

9.5.12 Aktion sign9796_2_DS2 für private Schlüsselobjekte

Diese Aktion hat das Ziel die als Parameter übergebene *message* mit dem adressierten Schlüsselobjekt zu signieren, siehe [gemSpec_COS#(N088.000), (N088.600)b].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion sign9796_2_DS2 bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion sign9796_2_DS2 fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion sign9796_2_DS2 bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion sign9796_2_DS2 fährt mit Schritt 3 fort.
      v) OK: Die Aktion sign9796_2_DS2 fährt mit Schritt 3 fort.
2) Schritt 2: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus sign9796_2_DS2
   a) eingetragen ist, dann wird mit Schritt 3 fortgefahren.

b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N102.900)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
   i) NoError: Das adressierte Schlüsselobjekt wird im channelContext von Card Proxy passend eingetragen. Die Aktion sign9796_2_DS2 fährt mit Schritt 4 fort.
   ii) UnsupportedFunction: Die Aktion sign9796_2_DS2 bricht mit dem Rückgabewert ObjectNotFound ab.
   iii) KeyNotFound: Die Aktion sign9796_2_DS2 bricht mit dem Rückgabewert ObjectNotFound ab.

3) Schritt 3: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
4) Schritt 4: Aus *message* werden gemäß [ISO/IEC 9796-2#9] und mit SHA-256 die Kommandoparameter *M1* und *hashM2* berechnet und die Aktion sign9796_2_DS2 fährt mit Schritt 5 fort.
5) Schritt 5: Es wird ein PSO Compute Digital Signature Kommando gemäß [gemSpec_COS#(N088.000)] zur Karte geschickt. Die Trailer des PSO Compute Digital Signature Kommandos gemäß [gemSpec_COS#Tab.189, Tab.190] werden gemäß Tabelle 60 auf die Rückgabewerte der Aktion sign9796_2_DS2 abgebildet. Die Aktion sign9796_2_DS2 fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion sign9796_2_DS2 gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

9.5.13 Aktion signECDSA für private Schlüsselobjekte

Diese Aktion hat das Ziel die als Parameter übergebene *message* mit dem adressierten Schlüsselobjekt zu signieren, siehe [gemSpec_COS#(N087.300)b, (N088.600)c].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion signECDSA bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion signECDSA fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion signECDSA bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion signECDSA fährt mit Schritt 3 fort.
      v) OK: Die Aktion signECDSA fährt mit Schritt 3 fort.
2) Schritt 2: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus signECDSA
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N102.900)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im channelContext von Card Proxy passend eingetragen. Die Aktion signECDSA fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion signECDSA bricht mit dem Rückgabewert ObjectNotFound ab.

iii) KeyNotFound: Die Aktion signECDSA bricht mit dem Rückgabewert Object-NotFound ab.

3) Schritt 3: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
4) Schritt 4: Die Nachricht *message* gehasht und der Hashwert als Kommandoparameter *dataToBeSigned* verwendet und die Aktion signECDSA fährt mit Schritt 5 fort, wobei gilt: Falls der private Schlüssel die Kurvenparameter
   a) ansix9p256r1 verwendet, dann wird SHA-256 zum hashen verwendet.
   b) ansix9p384r1 verwendet, dann wird SHA-384 zum hashen verwendet.
   c) brainpoolP256r1 verwendet, dann wird SHA-256 zum hashen verwendet.
   d) brainpoolP384r1 verwendet, dann wird SHA-384 zum hashen verwendet.
   e) brainpoolP512r1 verwendet, dann wird SHA-512 zum hashen verwendet.
5) Schritt 5: Es wird ein PSO Compute Digital Signature Kommando gemäß [gemSpec_COS#(N087.500)] zur Karte geschickt. Die Trailer des PSO Compute Digital Signature Kommandos gemäß [gemSpec_COS#Tab.189, Tab.190] werden gemäß Tabelle 60 auf die Rückgabewerte der Aktion signECDSA abgebildet. Die Aktion signECDSA fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion signECDSA gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

9.5.14 Aktion signPKCS1_V1_5 für private Schlüsselobjekte

Diese Aktion hat das Ziel die als Parameter übergebene *message* mit dem adressierten Schlüsselobjekt zu signieren, siehe [gemSpec_COS#(N087.300)c, (N088.600)d].

Hinweis (71): Die Umsetzung dieser Aktion verwendet absichtlich lediglich das PSO Compute Digital Signature Kommando und nicht das funktionsgleiche INTERNAL AUTHENTICATE *Kommando.*

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion signPKCS1_V1_5 bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion signPKCS1_V1_5 fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion signPKCS1_V1_5 bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion signPKCS1_V1_5 fährt mit Schritt 3 fort.
      v) OK: Die Aktion signPKCS1_V1_5 fährt mit Schritt 3 fort.
2) Schritt 2: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus signPKCS1_V1_5
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N102.900)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im channelContext von Card Proxy passend eingetragen. Die Aktion signPKCS1_V1_5 fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion signPKCS1_V1_5 bricht mit dem Rückgabewert ObjectNotFound ab.

iii) KeyNotFound: Die Aktion signPKCS1_V1_5 bricht mit dem Rückgabewert ObjectNotFound ab.

3) Schritt 3: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
4) Schritt 4: Aus *message* wird gemäß [PKCS #1] Kapitel 9.2 die DER Codierung *T* unter Verwendung von SHA-256 berechnet und *T* als Kommandoparameter *dataToBeSigned* verwendet und die Aktion signPKCS1_V1_5 fährt mit Schritt 5 fort.
5) Schritt 5: Es wird ein PSO Compute Digital Signature Kommando gemäß [gemSpec_COS#(N087.500)] zur Karte geschickt. Die Trailer des PSO Compute Digital Signature Kommandos gemäß [gemSpec_COS#Tab.189, Tab.190] werden gemäß Tabelle 60 auf die Rückgabewerte der Aktion signPKCS1_V1_5 abgebildet. Die Aktion signPKCS1_V1_5 fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion signPKCS1_V1_5 gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

9.5.15 Aktion signPSS für private Schlüsselobjekte

Diese Aktion hat das Ziel die als Parameter übergebene *message* mit dem adressierten Schlüsselobjekt zu signieren, siehe [gemSpec_COS#(N087.300)d, (N088.600)a].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion signPSS bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion signPSS fährt mit Schritt 3 fort.
      iii) ObjectNotFound: Die Aktion signPSS bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion signPSS fährt mit Schritt 3 fort.
      v) OK: Die Aktion signPSS fährt mit Schritt 3 fort.
2) Schritt 2: Falls das adressierte Schlüsselobjekt in *channelContext.keyReferenceList* mit dem Algorithmus signPSS
   a) eingetragen ist, dann wird mit Schritt 4 fortgefahren.
   b) nicht eingetragen ist, dann wird es mittels MSE Set Kommando gemäß [gemSpec_COS#(N102.900)] ausgewählt. Dabei sind gemäß [gemSpec_COS#Tab.265, Tab. 266] folgende Rückgabewerte möglich:
      i) NoError: Das adressierte Schlüsselobjekt wird im channelContext von Card Proxy passend eingetragen. Die Aktion signPSS fährt mit Schritt 4 fort.
      ii) UnsupportedFunction: Die Aktion signPSS bricht mit dem Rückgabewert ObjectNotFound ab.
      iii) KeyNotFound: Die Aktion signPSS bricht mit dem Rückgabewert ObjectNotFound ab.
3) Schritt 3: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
4) Schritt 4: Aus *message* wird mittels SHA-256 gehasht und der Hashwert wird als Kommandoparameter *dataToBeSigned* verwendet und die Aktion signPSS fährt mit Schritt 5 fort.

5) Schritt 5: Es wird ein PSO Compute Digital Signature Kommando gemäß [gemSpec_COS#(N087.500)] zur Karte geschickt. Die Trailer des PSO Compute Digital Signature Kommandos gemäß [gemSpec_COS#Tab.189, Tab.190] werden gemäß Tabelle 60 auf die Rückgabewerte der Aktion signPSS abgebildet. Die Aktion signPSS fährt mit Schritt 6 fort.
6) Schritt 6: Die Aktion signPSS gibt den unter Schritt 5 ermittelten Rückgabewert zurück.

9.5.16 Aktion terminate für private Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt in den Zustand „Termination state“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.9.2].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion terminate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion terminate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion terminate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion terminate fährt mit Schritt 2 fort.
      v) OK: Die Aktion terminate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein TERMINATE Kommando gemäß [gemSpec_COS#(N048.914)] zur Karte geschickt. Die Trailer des TERMINATE Kommandos gemäß [gemSpec_COS#Tab.73, Tab.74] werden gemäß Tabelle 62 auf die Rückgabewerte der Aktion terminate abgebildet. Die Aktion terminate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion terminate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 62: Rückgabewerte TERMINATE, TYP privates Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion terminate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Termination state“, siehe 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Termination state“.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Typ privates Schlüsselobjekt behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.6 cardOperation für öffentliche Schlüsselobjekte

Dieses Kapitel beschreibt Aktionen für den Objekttyp öffentliches Schlüsselobjekt.

9.6.1 Aktion activate für öffentliche Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt in den Zustand „Operational state (active)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.1.3].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion activate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion activate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion activate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion activate fährt mit Schritt 2 fort.
      v) OK: Die Aktion activate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein ACTIVATE Kommando gemäß [gemSpec_COS#(N034.824)] zur Karte geschickt. Die Trailer des ACTIVATE Kommandos gemäß [gemSpec_COS#Tab.28, Tab.29] werden gemäß Tabelle 63 auf die Rückgabewerte der Aktion activate abgebildet. Die Aktion activate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion activate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 63: Rückgabewerte ACTIVATE, Typ öffentliches Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion activate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (active)“, siehe 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (active)“.
´64 00´	ObjectTerminated	ObjectTerminated Das Objekt ist nicht aktivierbar, da es terminiert ist
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	OK Fehlermeldung der Smartcard wird absichtlich auf OK abgebildet
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.6.2 Aktion deactivate für öffentliche Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt in den Zustand „Operational state (deactivated)“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.3.3].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion deactivate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion deactivate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion deactivate bricht mit dem Rückgabewert Object-NotFound ab.
      iv) ObjectTerminated: Die Aktion deactivate fährt mit Schritt 2 fort.
      v) OK: Die Aktion deactivate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DEACTIVATE Kommando gemäß [gemSpec_COS#(N036.024)] zur Karte geschickt. Die Trailer des DEACTIVATE Kommandos gemäß [gemSpec_COS#Tab.34, Tab.35] werden gemäß Tabelle 64 auf die Rückgabewerte der Aktion deactivate abgebildet. Die Aktion deactivate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion deactivate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 64: Rückgabewerte DEACTIVATE, Typ öffentliches Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion deactivate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (deactivated)“, siehe auch 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Operational state (deactivated)“.
´64 00´	ObjectTerminated	ObjectTerminated Das Objekt ist nicht deaktivierbar, da es terminiert ist
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	OK Fehlermeldung der Smartcard wird absichtlich auf OK abgebildet
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.6.3 Aktion delete für öffentliche Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt zu löschen, siehe [gemSpec_COS14.2.4.3].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.

b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
   i) CardTerminated: Die Aktion delete bricht mit dem Rückgabewert CardTerminated ab.
   ii) FileDeactivated: Die Aktion delete fährt mit Schritt 2 fort.
   iii) ObjectNotFound: Die Aktion delete bricht mit dem Rückgabewert ObjectNotFound ab.
   iv) ObjectTerminated: Die Aktion delete fährt mit Schritt 2 fort.
   v) OK: Die Aktion delete fährt mit Schritt 2 fort.

2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein DELETE Kommando gemäß [gemSpec_COS#(N037.124)] zur Karte geschickt. Die Trailer des DELETE Kommandos gemäß [gemSpec_COS#Tab.40, Tab.41] werden gemäß Tabelle 65 auf die Rückgabewerte der Aktion delete abgebildet. Die Aktion delete fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion delete gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 65: Rückgabewerte DELETE, Typ öffentliches Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion delete mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Schlüsselobjekt wurde gelöscht, siehe auch 12.2.11.
´90 00´	NoError	OK Das Schlüsselobjekt wurde gelöscht.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.6.4 Aktion elcSharedSecretCalculation für öffentliche Schlüsselobjekte

Das Verschlüsseln mit persistent in einer Smartcard gespeicherten Schlüsseln gehört nicht zum verpflichtenden Funktionsumfang einer eGK. Deshalb wird es hier derzeit nicht behandelt. Bei Bedarf ist es möglich dieses Kapitel entsprechend zu ergänzen.

9.6.5 Aktion rsaEncipherOaep für öffentliche Schlüsselobjekte

Das Verschlüsseln mit persistent in einer Smartcard gespeicherten Schlüsseln gehört nicht zum verpflichtenden Funktionsumfang einer eGK. Deshalb wird es hier derzeit nicht behandelt. Bei Bedarf ist es möglich dieses Kapitel entsprechend zu ergänzen.

9.6.6 Aktion rsaEncipherPKCS1_V1_5 für öffentliche Schlüsselobjekte

Das Verschlüsseln mit persistent in einer Smartcard gespeicherten Schlüsseln gehört nicht zum verpflichtenden Funktionsumfang einer eGK. Deshalb wird es hier derzeit nicht behandelt. Bei Bedarf ist es möglich dieses Kapitel entsprechend zu ergänzen.

9.6.7 Aktion terminate für öffentliche Schlüsselobjekte

Diese Aktion hat das Ziel das adressierte Schlüsselobjekt in den Zustand „Termination state“ zu überführen, siehe [gemSpec_COS#(N007.100)a, 14.2.9.3].

1) Schritt 1: Falls das adressierte Schlüsselobjekt einem Ordner zugeordnet ist, welcher
   a) im Pfad zu *currentFolder* liegt, dann wird mit Schritt 2 fortgefahren.
   b) nicht im Pfad zu *currentFolder* liegt, dann wird der Ordner, dem das adressierte Schlüsselobjekt zugeordnet ist mittels der Aktion select ausgewählt. Dabei sind gemäß Tabelle 13 folgende Rückgabewerte möglich:
      i) CardTerminated: Die Aktion terminate bricht mit dem Rückgabewert CardTerminated ab.
      ii) FileDeactivated: Die Aktion terminate fährt mit Schritt 2 fort.
      iii) ObjectNotFound: Die Aktion terminate bricht mit dem Rückgabewert ObjectNotFound ab.
      iv) ObjectTerminated: Die Aktion terminate fährt mit Schritt 2 fort.
      v) OK: Die Aktion terminate fährt mit Schritt 2 fort.
2) Schritt 2: Der Sicherheitszustand wird gemäß Kapitel 10 angepasst. Falls dabei OK zurückgemeldet wird, dann wird mit dem nächsten Schritt fortgefahren, andernfalls wird mit dem zurückgemeldeten Rückgabewert abgebrochen.
3) Schritt 3: Es wird ein TERMINATE Kommando gemäß [gemSpec_COS#(N048.924)] zur Karte geschickt. Die Trailer des TERMINATE Kommandos gemäß [gemSpec_COS#Tab.73, Tab.74] werden gemäß Tabelle 66 auf die Rückgabewerte der Aktion terminate abgebildet. Die Aktion terminate fährt mit Schritt 4 fort.
4) Schritt 4: Die Aktion terminate gibt den unter Schritt 3 ermittelten Rückgabewert zurück.

Tabelle 66: Rückgabewerte TERMINATE, TYP öffentliches Schlüsselobjekt

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion terminate mit Erläuterung
´63 Cx´	UpdateRetryWarning	UpdateRetryWarning Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Termination state“, siehe 12.2.11.
´90 00´	NoError	OK Das Attribute *lifeCycleStatus* des Schlüsselobjektes besitzt den Wert „Termination state“.
´65 81´	MemoryFailure	MemoryFailure Schreibfehler, siehe auch 12.2.8.
´69 81´	VolatileKeyWithoutLCS	Dieser Trailer ist hier irrelevant, da hier nur der Typ privates Schlüsselobjekt behandelt wird.
´69 82´	SecurityStatusNotSatisfied	SecurityStatusNotSatisfied Der erforderliche Sicherheitszustand wurde nicht gesetzt.
´69 86´	NoCurrentEF	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.
´6A 88´	KeyNotFound	ObjectNotFound Adressiertes Schlüsselobjekt wurde nicht gefunden.
´6A 88´	PasswordNotFound	Dieser Trailer ist hier irrelevant, da hier nur der Typ Schlüsselobjekt behandelt wird.

9.7 cardOperation ohne zugeordnetes Objekt

Dieses Kapitel beschreibt Aktionen für den Objekttyp Ordner.

9.7.1 Aktion getRandom

Die Aktion hat das Ziel ein Zufallszahl zurückzuliefern, an die keine besonderen kryptographischen Anforderungen gestellt werden, siehe [gemSpec_COS#14.9.4].

Hinweis (72): Diese Aktion wird von allen Kartentypen unterstützt. Dabei wird ein Kommando verwendet, welches normalerweise im Rahmen von Authentisierungsprotokollen verwendet wird um Replay-Attacken zu vermeiden. Die zurückgelieferten Zufallszahlen sind also mindestens so hochwertig (und „einmalig"), dass Replay-Attacken hinreichend sicher auszuschließen sind.

Hinweis (73): Je nach Kartenimplementierung ist es (aber nicht sicher) möglich, dass diese Aktion Zufallszahlen zurückliefert, die aus kryptographischer Sicht genauso hochwertig sind, wie diejenigen aus der Aktion getSecureRandom (siehe 9.1.4).

1) Schritt 1: Es wird mittels einem oder mehrerer GET CHALLENGE Kommandos gemäß [gemSpec_COS#(N098.600)] (für Generation 1 Karten), oder [gemSpec_COS#(N098.625)] (für Generation 2 Karten) so viele Zufallsdaten erzeugt, wie mit dem Inputparameter *length* angefordert. Die Trailer der GET CHALLENGE Kommandos gemäß [gemSpec_COS#Tab.238, Tab.239] werden gemäß Tabelle 67 auf die Rückgabewerte der Aktion getRandom abgebildet. Die Aktion getRandom fährt mit Schritt 2 fort.
2) Schritt 2: Die Aktion getRandom gibt den unter Schritt 1 ermittelten Rückgabewert zusammen den ausgelesenen Daten zurück.

Tabelle 67: Rückgabewerte GET CHALLENGE Kommando

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion getRandom mit Erläuterung
´90 00´	NoError	OK plus angeforderte Daten Erfolgreiche Operation, dieser Rückgabewert wird zusammen mit den ausgelesenen Daten zurückgegeben.

9.7.2 Aktion getSecurityStatusFlagList

Die Aktion hat das Ziel den Sicherheitszustand der übergebenen Flagliste in der Smartcard abzufragen, siehe [gemSpec_COS#(N085.444)].

1) Schritt 1: Es wird ein GET SECURITY STATUS KEY Kommando gemäß [gemSpec_COS#(N085.444)] zur Smartcard geschickt. Die Trailer der GET SECURITY STATUS KEY Kommandos gemäß [gemSpec_COS#Tab.178, Tab.179] werden gemäß Tabelle 68 auf die Rückgabewerte der Aktion getSecurityStatusFlagList abgebildet. Die Aktion getSecurityStatusFlagList fährt mit Schritt 2 fort.
2) Schritt 2: Die Aktion getSecurityStatusFlagList gibt den unter Schritt 1 ermittelten Rückgabewert zurück.

Tabelle 68: Rückgabewerte GET SECURITY STATUS KEY Kommando

Trailer gemäß [gemSpec_COS]		Rückgabewert der Aktion getSecurityStatus… mit Erläuterung
´63 CF´	NoAuthentication	NoAuthentication Erfragter Authentisierungsstatus ist nicht gesetzt.
´90 00´	NoError	OK Erfragter Authentisierungsstatus ist gesetzt.
´69 82´	SecurityStatusNotSatisfied	Dieser Trailer ist für asymmetrische Sicherheitszustände irrelevant
´6A 88´	KeyNotFound	Dieser Trailer ist für asymmetrische Sicherheitszustände irrelevant

9.7.3 Aktion getSecurityStatusRole

Die Aktion hat das Ziel den Sicherheitszustand der übergebenen Rolle in der Smartcard abzufragen, siehe [gemSpec_COS#(N085.444)].

1) Schritt 1: Es wird ein GET SECURITY STATUS KEY Kommando gemäß [gemSpec_COS#(N085.400)] zur Smartcard geschickt. Die Trailer der GET SECURITY STATUS KEY Kommandos gemäß [gemSpec_COS#Tab.178, Tab.179] werden gemäß Tabelle 68 auf die Rückgabewerte der Aktion getSecurityStatusRole abgebildet. Die Aktion getSecurityStatusRole fährt mit Schritt 2 fort.
2) Schritt 2: Die Aktion getSecurityStatusRole gibt den unter Schritt 1 ermittelten Rückgabewert zurück.

9.7.4 Aktion resetChannel

Die Aktion hat das Ziel, den logischen Kanal, in welchem das Kommando gesendet wird, in den Zustand zu versetzen, in welchem dieser Kanal unmittelbar nach seinem Öffnen gewesen ist.

Hinweis (74): Unmittelbar nach dem Öffnen eine logischen Kanals gilt:
- *a. Es sind keine Sicherheitszustände gesetzt.*
- *b. Das Wurzelverzeichnis ist selektiert.*
- *c. Ein Elementary File ist nicht ausgewählt.*
- *d. Das Security Environment SE#1 ist aktiv.*

1) Schritt 1: Die Aktion wird abhängig von der Kartengeneration ausgeführt. Handelt es sich um eine
   a) Generation 1 Karten, dann werden folgende Kommandos geschickt:
      i) SELECT Kommando gemäß [gemSpec_COS#(N040.800)], ohne AID, first, keine Antwortdaten.
      ii) MSE Restore Kommando gemäß [gemSpec_COS#(N099.900)] mit *seNo*=1.
   b) Generation 2 Karten, dann wird folgendes Kommando geschickt:
      MANAGE CHANNEL Kommando gemäß [gemSpec_COS#(N099.524)].
2) Schritt 2: *channelContext* (siehe Kapitel 3 Punkt (1) wird (5)entsprechend dieser Aktion angepasst.
3) Schritt 3: Die Aktion gibt als Rückgabewert stets OK zurück.

10 Sicherheitszustand

Dieses Kapitel beschreibt, wie Card Proxy innerhalb der Bearbeitung einer Aktion dafür sorgt, dass der Sicherheitszustand passend gesetzt ist.

Card Proxy ermittelt mit dem Identifikator der aktuellen Aktion und der Aktion selbst (activate, read, getStatus, signPSS, …) aus den Konfigurationstabellen welche Bedingungen die Smartcard an das Ausführen der aktuellen Aktion für das betroffene Objekt stellt.

Gemäß dem derzeitigen Konzept von Card Proxy, welches nur vergleichsweise einfache Zugriffsbedingungen abzubilden in der Lage ist, treten dabei die folgenden Fälle auf:

1) ALWAYS: Die Aktion ist stets ausführbar. An dieser Stelle sind keine weiteren Aktionen erforderlich und es wird OK zurückgemeldet.
2) PIN-Schutz: Die Aktion ist genau dann ausführbar, wenn der Sicherheitszustand des in der Konfigurationstabelle genannten Passwortobjektes in der Smartcard gesetzt ist. Falls der Sicherheitszustand des Passwortobjektes laut Baustein „Sicherheitsstatus“ (siehe Kapitel 3 Punkt (6)
   a) gesetzt ist, dann ist hier keine weitere Aktion erforderlich und es wird OK zurückgemeldet.
   b) nicht gesetzt ist, dann wird hier eine passende Benutzerverifikation angestoßen:
      i) Schritt 1: cardOperation(IdentifikatorDesPasswortObjektes, verify), es ist möglich, dass diese Aktion
         (1) erfolgreich verläuft, woraufhin OK zurückgemeldet wird.
         (2) nicht erfolgreich verläuft, woraufhin ErrorUserVerification zurückgemeldet wird.

3) Card-2-Card: Die Aktion ist genau dann ausführbar, wenn der Sicherheitszustand der in der Konfigurationstabelle genannten Rolle oder Flagliste in der Smartcard gesetzt ist. Falls das laut Baustein „Sicherheitsstatus“ (siehe Kapitel 3 Punkt (6)
   a) der Fall ist, dann ist hier keine weitere Aktion erforderlich und es wird OK zurückgemeldet.
   b) nicht der Fall ist, dann wird hier eine passende Card-2-Card Aktion angestoßen:
      i) Schritt 1: Dem Zertifikatsspeicher (siehe Kapitel 3 Punkt (3) wird das End-Entity-CV-Zertifikat der Entität nebst Zertifikatskette entnommen, die für Card-2-Card zuständig ist.
      ii) Schritt 2: Falls das End-Entity-CV-Zertifikat gemäß der darin enthaltenen Rolle oder Flagliste nicht in der Lage ist, die erforderlichen Rechte zu verschaffen, bricht der Algorithmus ab und meldet WrongEndEntityCVC zurück.
      iii) Schritt 3: Das End-Entity-CV-Zertifikat wird von Card Proxy in die mit diesem verbundene Smartcard importiert, siehe Kapitel 11. Falls es dabei zu Fehlern auf Kartenebene kommt, bricht der Algorithmus ab und meldet ErrorImportCVC zurück.
      iv) Schritt 4: Card Proxy steuert das Authentisierungsprotokoll gemäß [gemSpec_COS#15.1]. Dabei werden folgende Nachrichten ausgetauscht:
         (1) Card Proxy holt mittels GET CHALLENGE Kommando von der freizuschaltenden Smartcard eine Zufallszahl ab.
         (2) Card Proxy erstellt aus der Zufallszahl und der Seriennummer der „Freischaltkarte“ aus dem End-Entity-CV-Zertifikat ein passendes *token*.

(3) Card Proxy übergibt das *token* an die Umgebung und die Umgebung sorgt dafür, dass diese Zufallszahl passend signiert wird. Die Umgebung übergibt die Signatur an Card Proxy.
(4) Card Proxy bettet die Signatur in ein EXTERNAL AUTHENTICATE Kommando gemäß [gemSpec_COS#(N083.500)] ein und schickt dieses an die freizuschaltende Smartcard.

v) Falls eine der vorgenannten Aktionen nicht erfolgreich verläuft, bricht der Algorithmus ab und es wird ErrorAuthentication zurückgemeldet.
vi) Falls alle der vorgenannten Aktionen erfolgreich verlaufen, wird OK zurückgemeldet.

4) Einfaches AND: Die Zugriffsbedingung verknüpft genau ein Passwortobjekt mit genau einer Rolle oder genau einer Flagliste. Dies ist eine Kombination aus PIN-Schutz und Card-2-Card. Card Proxy versucht beides auszuführen.
   a) Der Algorithmus aus Punkt 2) „PIN-Schutz“ wird durchlaufen. Falls dabei ein Fehler auftritt, bricht dieser Algorithmus mit der dort erzeugten Fehlermeldung ab.
   b) Der Algorithmus aus Punkt 3) „Card-2-Card“ wird durchlaufen. Falls dabei ein Fehler auftritt, bricht dieser Algorithmus mit der dort erzeugten Fehlermeldung ab.
   c) Falls sowohl „PIN-Schutz“, als auch „Card-2-Card“ erfolgreich verliefen wird OK zurückgemeldet.

5) OR-Passwortobjekte AND Card-2-Card: Die Zugriffsbedingung verknüpft entweder genau eine Rolle oder genau einer Flagliste mit einer Liste von alternativen Passwortobjekten. Aktuell ist dies lediglich im Rahmen von AMTS mit Vertreter-PIN der Fall. Die Aktion ist genau dann ausführbar, wenn mindestens für eines der Passwortobjekte der Sicherheitszustand in der Smartcard gesetzt ist und zusätzlich der Sicherheitszustand der Rolle bzw. Flagliste.
   a) Schritt 1: Falls für keines der ver-ODER-ten Passwortobjekte der Sicherheitszustand gesetzt ist, versucht Card Proxy die Liste dieser ver-ODER-ten Passwortobjekts abzuarbeiten. Falls
      i) mindestens ein Passwortobjekt erfolgreich verifiziert wurde, wird mit Schritt 2, Card-2-Card aus Punkt 5)b) fortgefahren.
      ii) keines der Passwortobjekte erfolgreich verifiziert wurde, bricht der Algorithmus ab und es wird ErrorUserVerification zurückgemeldet.
   b) Schritt 2: Es wird ein Card-2-Card gemäß Punkt 3) ausgeführt.

6) NEVER: Die Aktion ist unter den aktuellen Umständen nicht ausführbar. Entweder befindet sich das Objekt in einem LCS-Zustand, der die Aktion nicht unterstützt oder es wurde ein Security Environment ausgewählt, in welchem die Aktion nicht unterstützt wird. Durch einen Wechsel des LCS-Zustandes und/oder des Security Environments lassen sich die Umstände so ändern, dass die Aktion ausführbar ist. Es wird WrongCircumstances zurückgemeldet.

Tabelle 69: Rückgabewerte im Rahmen der Anpassung des Sicherheitszustandes

Schutz	Rückgabewert	Beschreibung
ALWAYS	OK	Der Sicherheitszustand ist passend gesetzt.
PIN-Schutz	ErrorUserVerification	Die Benutzerverifikation ist fehlgeschlagen.
	OK	Der Sicherheitszustand ist passend gesetzt.
Card-2-Card	ErrorAuthentication	Im Rahmen des Authentisierungsprotokolls trat ein Fehler auf.
	ErrorImportCVC	Der Import eines CV-Zertifikates schlug fehl.
	OK	Sicherheitszustand passend gesetzt
	WrongEndEntityCVC	Das End-Entity-CV-Zertifikat enthält nicht die Rechte, die nötig sind um die Aktion freizuschalten.
NEVER	WrongCircumstances	Das Objekt befindet sich in einem LCS-Zustand und/oder es wurde ein Security Environment gewählt, in welchem die Aktion nicht ausführbar ist.

Hinweis (75): Gemäß Tabelle 52 ist es möglich, dass eine „verify“ Aktion für ein Passwortobjekt detailliert meldet, woran die „verify“ Aktion gescheitert ist. Card Proxy bildet alle erfolglosen „verify“ Aktionen auf einen einzigen Fehler ErrorUserVerification ab, da die „verify“ Aktion von der Umgebung ausgeführt wird und diese deshalb über die genauere Fehlerursache bereits informiert ist.

Hinweis (76): Im obigen Punkt 3)b)i) ist verkürzend nur von „Entität, die für Card-2-Card zuständig ist“ die Rede. Damit ist folgendes gemeint: Card Proxy ist vom Konzept her so angelegt, dass der Baustein CV-CertificateStore im Laufe der Zeit eine Menge End-Entity-CV-Zertifikate von HBA, SMC-B, etc. sammeln könnte. Auf das Signal „Karte verfügbar“ (siehe 4.1) ermittelt Card Proxy die Stammdaten der verbundenen Smartcard und meldet diese an die Umgebung. Die Umgebung hat dann dafür zu sorgen, dass dem Baustein CV-CertificateStore im Card Proxy eine passende CV-Zertifikatskette zur Verfügung steht und (falls dort mehrere passende gespeichert sind) das aktuell zu verwendende markiert wird. So ist es denkbar, dass je nach Kartengeneration (G1, G2, …), Kartenherausgeber (Krankenkasse A, Krankenkasse B, …) oder auch je nach Lastverteilung im AdV-Server dort eine von mehreren SMC-B für Card-2-Card ausgewählt wird. Im Endeffekt sorgt die Umgebung dafür, dass es zu der mit Card Proxy verbundenen Smartcard eine dedizierte „Freischaltkarte“ gibt. Das End-Entity-CV-Zertifikat dieser „Freischaltkarte“ muss in CV-CertificateStore von Card Proxy vorhanden und markiert sein.

Hinweis (77): Das Durchprobieren einer Liste von Passwortobjekten in Punkt 5)a) ist aus Benutzersicht unschön. Besser wäre es die Schnittstelle zur Umgebung so zu ändern, dass der Umgebung eine Liste zur Verifikation übergeben werden könnte. Normalerweise umfasst diese Liste nur ein Element und vom Benutzer ist nichts auszuwählen. Falls diese Liste mehr als einen Eintrag enthält, dann könnte die Umgebung den Benutzer wählen lassen, welches Passwortobjekt er verifiziert haben möchte. TODO, ggf. Schnittstelle in [gemSpec_KTR-AdV] anpassen

11 Import von End-Entity-CV-Zertifikaten

11.1 Annahmen

Ohne Beschränkung der Allgemeingültigkeit wird in diesem Kapitel von folgender Situation ausgegangen:

1) Card Proxy sei mit einer Zielkarte verbunden und es gilt in diese Zielkarte ein End-Entity-CV-Zertifikat (EECVC.C2C) zu importieren.
2) Card Proxy hat aus der Zielkarte die „Stammdaten“ (siehe 4.1) ausgelesen, unter anderem auch dessen End-Entity-CV-Zertifikat (EECVC.ZK) und gegebenen falls das CV-Zertifikat der CVC-Sub-CA welches EECVC.ZK ausstellte.
3) Die Zielkarte enthält mindestens den öffentlichen Schlüssel der Root-CVC-CA, welches das CV-Zertifikat der CVC-Sub-CA ausstellte.
4) Das zu importierende EECVC.C2C liegt im Baustein CV-CertificateStore (siehe Kapitel 3 Punkt (3) vor nebst passender CVC-Kette.

Hinweis (78): Es ist möglich, dass der Cache der Zielkarte weitere öffentliche Schlüssel aus früheren Importen enthält. Da das Auswählen eines Schlüssels erheblich schneller erfolgt, als der Import, geht der im Folgenden beschriebene Algorithmus davon aus, dass der Cache passend gefüllt ist.

Jedes CV-Zertifikat enthält unter anderem ein Feld CAR, welches angibt, von welcher CA es ausgestellt wurde. Mit dieser Information lässt sich eine Kette bilden vom

5) EECVC.C2C über dessen ausstellende
6) CVC-Sub-CA über dessen ausstellende
7) CVC-Root-CA,
8) kein, ein oder mehrere Link-CV-Certificate bis zum
9) öffentlichen Schlüssel der CVC-Root-CA, aus dessen PKI EECVC.ZK stammt und der in der Zielkarte in jedem Fall vorhanden ist (sofern es sich um eine spezifikationskonforme Smartcard handelt).

11.2 Algorithmus zum Import eines End-Entity-CV-Zertifikates

Der folgende Algorithmus geht von der in 11.1 beschriebenen CVC-Kette aus, die so gerichtet sei, dass EECVC.C2C am Ende der Kette liege.

1) Schritt 1, Initialisierung: Setze einen Zeiger auf das letzte CV-Zertifikat der Kette, also EECVC.C2C.
2) Schritt 2, Schlüsselselektion: Versuche den öffentlichen Schlüssel des CV-Zertifikates zu selektieren, auf den der Zeiger aktuell zeigt. Falls der Zeiger auf ein
   a) End-Entity-CV-Zertifikat zeigt, wird gemäß [gemSpec_COS#(N101.900)] ein MSE Set Kommando ausgeführt und dabei *keyRef* auf den Wert CHR aus dem End-Entity-CV-Zertifikat gesetzt und *algId* auf den Wert
      i) elcRoleCheck, falls es sich um ein ELC End-Entity-CV-Zertifikat handelt.
      ii) rsaRoleCheck, falls es sich um ein RSA End-Entity-CV-Zertifikat handelt.

   b) kein End-Entity-CV-Zertifikat zeigt, wird gemäß [gemSpec_COS#(N103.300)] ein MSE Set Kommando ausgeführt und dabei *keyRef* auf den Wert CHR aus dem End-Entity-CV-Zertifikat gesetzt.
3) Schritt 3, Auswertung des Rückgabewertes: Falls die Selektion aus Schritt 2)
   a) KeyNotFound meldet, dann setze den Zeiger auf das vorherige CV-Zertifikat in der Kette und fahre mit Schritt 2) fort.
   b) UnsupportedFunction meldet, dann breche den Algorithmus erfolglos mit der Fehlermeldung ErrorImportCVC ab.
   c) NoError meldet und der Zeiger zeigt
      i) auf das letzte Element der Kette, dann endet der Import erfolgreich mit dem Rückgabewert OK.
      ii) nicht auf das letzte Element der Kette, dann setzte den Zeiger auf das nachfolgende CV-Zertifikat und fahre mit Schritt 4) fort.
4) Schritt 4, CV-Import: Das CV-Zertifikat, auf welches der Zeiger zeigt, wird importiert, PSO Verify Certificate Kommando
   a) RSA: [gemSpec_COS#(N095.100)]
   b) ELC: [gemSpec_COS#(N095.410)]
5) Schritt 5, Auswertung des Rückgabewertes: Falls der Import
   a) nicht erfolgreich verlief, dann breche den Algorithmus erfolglos mit der Fehlermeldung ErrorImportCVC ab.
   b) erfolgreich verlief, dann fahre mit Schritt 2) fort.

12 Verschiedenes

12.1 Trailer einer Smartcard

Tabelle 70: Trailer → Fehlername gemäß [gemSpec_COS#Tab.272]

Wert	Name	Bedeutung
´62 00´	DataTruncated	Antwortdaten unvollständig
´62 81´	CorruptDataWarning	Die Integrität der Antwortdaten ist nicht gewährleistet
´62 82´	EndOfFileWarning	Es wurden mehr Daten angefordert als die Datei enthält
´62 82´	EndOfRecordWarning	Es wurden mehr Daten angefordert als der Rekord enthält
´62 82´	UnsuccessfulSearch	Pattern wurde in keinem der adressierten Rekords gefunden
´62 83´	FileDeactivated	File, auf welches sich die Operation bezieht, ist deaktiviert
´62 85´	FileTerminated	File, auf welches sich die Operation bezieht, ist terminiert
´62 87´	RecordDeactivated	Rekord, auf welchen sich Operation bezieht, ist deaktiviert
´62 Cx´	TransportStatus	Indikation des Transportschutzverfahrens
´62 D0´	PasswordDisabled	Passwortobjekt ausgeschaltet, Verifikation nicht erforderlich
´63 00´	AuthenticationFailure	Authentisierung fehlgeschlagen
´63 CF´	NoAuthentication	Keine Authentisierung mit dem referenzierten Schlüssel
´63 Cx´	RetryCounter	Wert des Fehlbedienungszählers
´63 Cx´	UpdateRetryWarning	Schreibschwierigkeiten
´63 Cx´	WrongSecretWarning	Falsches Passwort in den Kommandodaten
´64 00´	EncipherError	Fehlerhafte Verschlüsselungsoperation
´64 00´	KeyInvalid	Schlüsseldaten fehlen, Generierung erforderlich
´64 00´	ObjectTerminated	Objekt befindet sich im Zustand „Termination state“
´64 00´	ParameterMismatch	Domainparameter passen nicht zusammen
´65 81´	MemoryFailure	Schreibfehler
´67 00´	WrongRecordLength	Falsche Rekordlänge
´68 81´	ChannelClosed	Logischer Kanal nicht geöffnet
´69 81´	NoMoreChannelsAvailable	kein weiterer logischer Kanal verfügbar
´69 81´	VolatileKeyWithoutLCS	volatile Schlüssel werden vom Kommando nicht unterstützt
´69 81´	WrongFileType	Datei unterstützt das aktuelle Kommando nicht
´69 82´	SecurityStatusNotSatisfied	Zugriffsregel nicht erfüllt
´69 83´	CommandBlocked	Rücksetzen des Fehlbedienungszählers nicht mehr möglich
´69 83´	KeyExpired	Der Gültigkeitsbereich des Schlüssels ist abgelaufen
´69 83´	PasswordBlocked	Fehlbedienungszähler abgelaufen
´69 85´	KeyAlreadyPresent	Schlüsseldaten bereits gesetzt, Generierung unmöglich
´69 85´	LongPassword	Neues Passwort zu lang
´69 85´	NoKeyReference	Schlüsselreferenz fehlt, MSE-Set ist notwendig
´69 85´	NoPrkReference	Schlüsselreferenz fehlt, MSE-Set ist notwendig
´69 85´	NoPukReference	Schlüsselreferenz fehlt, MSE-Set ist notwendig
´69 85´	NoRandom	Keine Zufallszahl, GET CHALLENGE ist notwendig
´69 85´	NoRecordLifeCycleStatus	Datei unterstützt das aktuelle Kommando nicht
´69 85´	PasswordNotUsable	Transportschutz aktiv, CHANGE REF. DATA notwendig
´69 85´	WrongRandomLength	Zufallszahl hat falsche Länge, GET CHALLENGE erforderlich
´69 85´	ShortPassword	Neues Passwort zu kurz
´69 86´	NoCurrentEF	Kommandobearbeitung unmöglich, da keine Datei selektiert
´69 88´	IncorrectSmDo	Fehlerhaftes Secure Messaging
´6A 80´	NewFileSizeWrong	*newFileSize* kein Vielfaches der Rekordlänge
´6A 80´	NumberPreconditionWrong	Vorbedingung zum Laden des Scenarios nicht erfüllt
´6A 80´	NumberScenarioWrong	Scenario wurde bereits geladen

Wert	Name	Bedeutung
´6A 80´	VerificationError	Fehlerhaftes CV-Zertifikat
´6A 80´	WrongCiphertext	Fehlerhaftes Chiffrat
`6A 80´	WrongToken	Token ist fehlerhaft
´6A 81´	UnsupportedFunction	Schlüssel unterstützt den angegeben Algorithmus nicht
´6A 82´	FileNotFound	Referenzierte Datei nicht gefunden
´6A 83´	RecordNotFound	Referenzierter Rekord nicht verwendbar
´6A 84´	DataTooBig	Zu viele Daten
´6A 84´	FullRecordList	Rekordliste bereits komplett gefüllt
´6A 84´	MessageTooLong	Klartext zu lang für Verschlüsselung
´6A 84´	OutOfMemory	Zu wenig Speicherplatz
´6A 88´	InconsistentKeyReference	Schlüsselreferenz im CV-Zertifikat fehlerhaft
´6A 88´	KeyNotFound	Referenzierten Schlüssel nicht gefunden
´6A 88´	PasswordNotFound	Referenziertes Passwort nicht gefunden
´6A 88´	PrKNotFound	Referenzierten Schlüssel nicht gefunden
´6A 88´	PukNotFound	Referenzierten Schlüssel nicht gefunden
´6A 89´	DuplicatedObject	Neu anzulegendes Objekt existiert bereits
´6A 8A´	DfNameExists	Neu anzulegende Applikation existiert bereits
´6B 00´	OffsetTooBig	Offset zu groß
´6D 00´	InstructionNotSupported	Der im INS-Byte angezeigte Befehl wird nicht unterstützt
´90 00´	NoError	Normale Kommandoausführung, kein Fehler, keine Warnung

12.2 Besondere Fehlersituationen

12.2.1 BufferTooSmall

Es sollen mehr Daten zur Karte übertragen werden, als in den Übertragungspuffer des Kartenlesers und/oder der Smartcard passen. Dieser Fehler tritt nur dann auf, wenn Card Proxy sehr viele Daten übergeben werden oder sehr viele Daten aus der Smartcard auszulesen sind und Card Proxy keine Möglichkeit hat die Daten auf mehrere Kommando- und oder Antwortnachrichten zu verteilen.

Bei spezifikationskonformer Verwendung der Smartcard ist die Ursache dieses Problems niemals die Smartcard, sondern immer der Kartenleser.

12.2.2 CardTerminated

Der Lebenszyklus der Smartcard ist terminiert, eine reguläre Nutzung ist nicht mehr möglich. Es erscheint empfehlenswert jede weitere Interaktion mit der Smartcard unter zu Hilfenahme von Card Proxy zu vermeiden, weil Card Proxy für diese Umstände nicht ausgelegt ist.

12.2.3 CorruptDataWarning

Diese Warnung ist nur im Zusammenhang mit lesenden Operationen möglich und sie zeigt an, dass die gelesenen Daten nicht zuverlässig sind, weil der Speicherinhalt nicht integer ist. Diese Situation tritt auf, wenn etwa wegen Alterungsprozessen sich der Speicherinhalt ändert. Die Smartcard erkennt dies beispielsweise anhand von Checksummenprüfungen. Je nachdem welche Speicherzelle davon betroffen ist und welche Speicherzellen gelesen wurden, ist es möglich (aber nicht sicher), dass die ausgelesenen Daten trotz

der Warnung korrekt sind. Es erscheint empfehlenswert die betroffene Smartcard baldmöglichst auszutauschen.

12.2.4 ErrorAuthentication

Das Freischalten einer Smartcard durch externe Authentisierung schlug fehl. Als wahrscheinlichste Ursachen kommen in Betracht:

- Es wird eine Entität zur Freischaltung verwendet, die nicht zum importierten End-Entity-CV-Zertifikat EECVC.C2C passt.
- Die Zufallszahl der freizuschaltenden Smartcard oder die Signatur der freischaltenden Smartcard wurden auf dem Transportweg verändert.

12.2.5 ErrorImportCVC

Beim Import von CV-Zertifikaten trat ein Fehler auf. Als wahrscheinlichste Ursachen kommen in Betracht:

- Die Zertifikatskette ist nicht korrekt.
- Eines der zu importierenden CV-Zertifikate ist fehlerhaft.

12.2.6 ErrorUserVerification

Eine Benutzerverifikation im Rahmen der Anpassung des Sicherheitszustandes schlug fehl. Dieser Fehler wird von Card Proxy nur dann verwendet, wenn Card Proxy versuchte selbständig eine Benutzerverifikation durchzuführen.

Wenn die Benutzerverifikation als Aktion „verify“ von außen angestoßen wird, dann meldet Card Proxy gemäß Tabelle 15 aussagekräftigere Fehlermeldungen.

12.2.7 KeyInvalid

Ein privates Schlüsselobjekt enthält keine Schlüsseldaten. Dieser Fehler tritt dann auf, wenn ein privates Schlüsselobjekt erst nach Auslieferung der Smartcard beschlüsselt wird und diese Beschlüsselung noch nicht erfolgte.

12.2.8 MemoryFailure

Der Rückgabewert zeigt an, dass eine Speicherzelle nicht auf den vorgesehen Wert gesetzt wurde. Wegen dieses Speicherfehlers ist es angebracht die Smartcard baldmöglichst auszutauschen.

Über den Erfolg oder Misserfolg sagt dieser Rückgabewert wenig aus, weil typischerweise Schreiboperationen transaktionsgeschützt sind. Es ist also möglich (aber nicht sicher), dass die intendierten Daten wie vorgesehen geschrieben wurden und der Schreibfehler in Bereichen auftrat, die mit der Protokollierung der Transaktion in Verbindung stehen.

12.2.9 ObjectNotFound

Der adressierte Ordner, das adressierte Objekt oder einer der darüber liegenden Ordner wurde nicht gefunden. Dieser Rückgabewert zeigt im Allgemeinen an, dass die Datenbank inkonsistent zum Inhalt der Smartcard ist.

12.2.10 SecurityStatusNotSatisfied

Fehlbedienung, etwa weil versucht wird Aktionen auszuführen, die gesperrt sind, solche Aktionen sollten auf der GUI besser nicht getriggert werden.

12.2.11 UpdateRetryWarning

Die Aktion wurde erfolgreich ausgeführt. Der Rückgabewert zeigt an, dass eine Speicherzelle erst nach mehreren Versuchen auf den vorgesehen Wert gesetzt wurde. Wegen dieses Speicherfehlers ist es angebracht die Smartcard baldmöglichst auszutauschen.

12.2.12 WrongEndEntityCVC

Das End-Entity-CV-Zertifikat enthält nicht die erforderlichen Berechtigungen um die angeforderte Operation freizuschalten.

Im Rahmen eines spezifikationskonformen Betriebes ist dieser Fehler nicht möglich.

12.3 Format-2-PIN-Block

Der Format-2-PIN-Block ist eine Möglichkeit eine Ziffernfolge variabler Länge in acht Oktetten = 16 Nibble zu codieren:

1) Das erste Nibble muss den Wert ´2´ haben.
2) Das zweite Nibble muss hexadezimal die Anzahl der Ziffern codieren.
3) Das (i+2) te Nibbel muss hexadezimal die i-te Ziffer codieren
4) Alle übrigen Nibble müssen den Wert ´F´ haben.

Tabelle 71: Beispiele für die Umwandlung einer PIN in einen Format-2-PIN-Block

PIN (Ziffernfolge)	Format-2-PIN-Block
1234	´2412 34FF FFFF FFFF´
987654321	´2998 7654 321F FFFF´
012345678912	´2C01 2345 6789 12FF´

Anhang A – Verzeichnisse

A1 – Abkürzungen

Kürzel	Erläuterung
-	-

A2 – Glossar

Begriff	Erläuterung
-	-

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

A3 – Tabellenverzeichnis

A4 – Referenzierte Dokumente

A4.1 – 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
[gemSpec_COS]	gematik: Spezifikation des Card Operating System (COS), Elektrische Schnittstelle
[gemSpec_Karten_Fach_TIP]	gematik: Befüllvorschriften für die Plattformanteile der Karten der TI
[gemSpec_KTR-AdV]	gematik: Spezifikation KTR-AdV
[gemSpec_eGK_Fach_TIP]	gematik: Speicherstrukturen der eGK für die TI-Plattform
[gemSpec_eGK_ObjSys]	gematik: Spezifikation der elektronischen Gesundheitskarte, eGK-Objektsystem, Produkttypversion 4.3.2

[Quelle]	Herausgeber: Titel
[gemSpec_eGK_P2]	gematik: Einführung der Gesundheitskarte – Spezifikation elektronische Gesundheitskarte; Teil 2 – Grundlegende Applikationen Release 0.5.3, www.gematik.de
[gemSpec_SMC-B_ObjSys]	gematik: Spezifikation der Security Module Card SMC-B, Objektsystem

A4.2 – Weitere Dokumente

[Quelle]	Herausgeber (Erscheinungsdatum): Titel
[ISO/IEC 9796-2]	Information technology — Security techniques — Digital signature schemes giving message recovery — Part 2: Integer factorization based mechanisms, third edition, 2010-12-15
[PKCS #1]	PKCS #1 v2.1: RSA Cryptography Standard, RSA Laboratories, June 14, 2002 ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf
[RFC2119]	Key words for use in RFCs to Indicate Requirement Levels https://www.ietf.org/rfc/rfc2119.txt