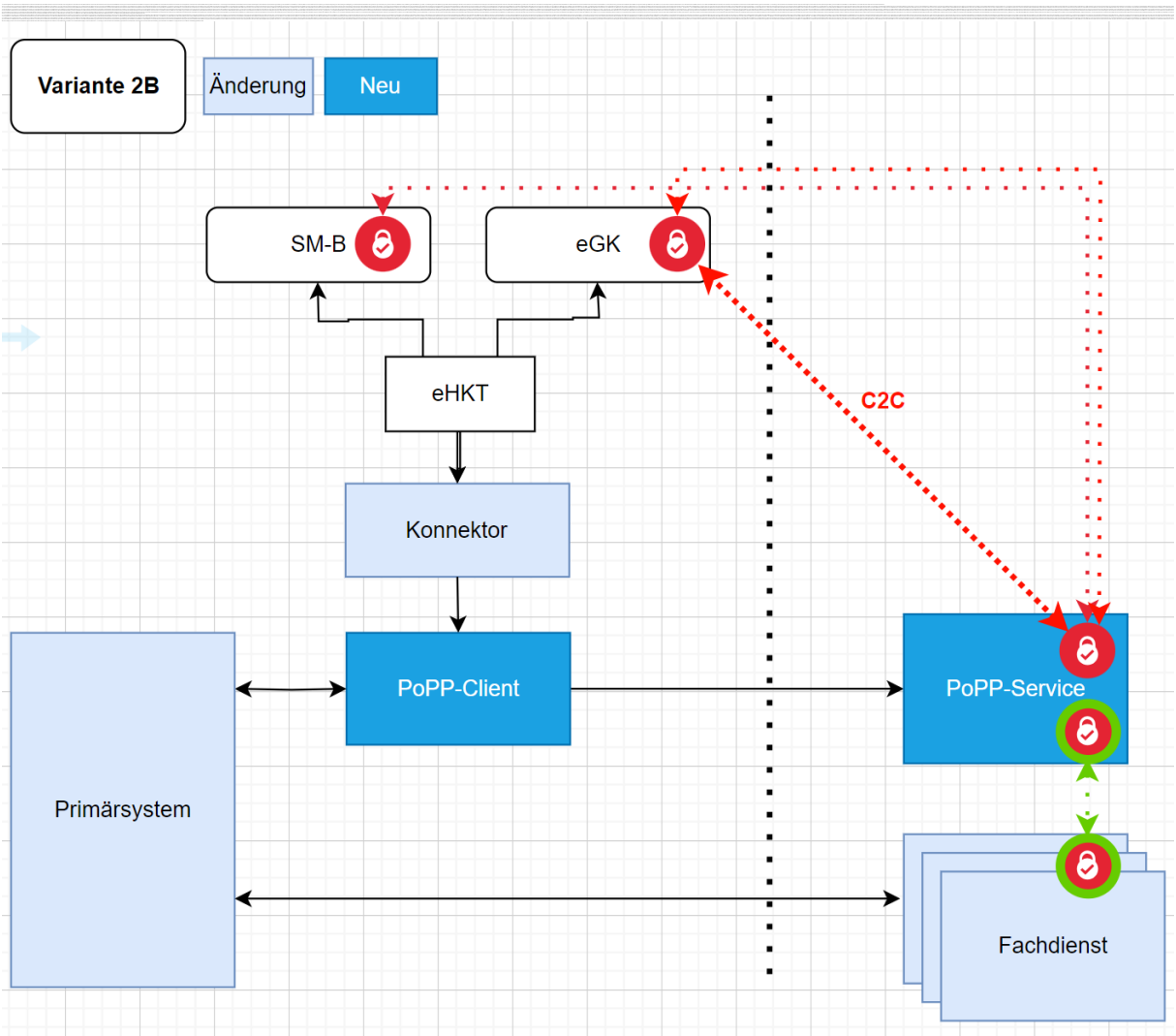
Diese Anlage beschreibt Änderungen am Konnektor, die unten skizzierte PoPP-Lösung unterstützen sollen.

Eine Beschreibung findet sich im Änderungsbedarf des Änderungseintrags.



Inhaltsverzeichnis

1 Änderung in gemSpec_Kon

1.1 Kapitel 4.1.5 "Kartendienst"

In Kapitel 4.1.5.1 wird Anforderung A_25895 am Ende neu aufgenommen:

A_25895 - Exklusive Nutzung des Karten-Kommunikationskanals durch Operation SecureSendAPDU
Wenn eine Karte durch Aufruf der Operation SecureSendAPDU reserviert ist, dann MUSS jeder weitere Aufruf von TUC_KON_023 mit doLock=true, welcher dieselbe Karte zu reservieren versucht, mit Fehlercode 4093 abbrechen.**[**<=, , **]**

In Kapitel 4.1.5.2 "Durch Ereignisse ausgelöste Reaktionen" wirdA_25860 unter A_23702 neu aufgenommen:

A_25860 - Reaktion auf abgelaufenen APDU-Szenario-Timer
Wenn der vorhergehende, zu einer cardSession zugehörige Aufruf der Operation SecureSendAPDU keinen Fehler aufwies und der zu derselben cardSession zugehörige Folgeaufruf von SecureSendAPDU nicht innerhalb des durch den vorhergehenden Aufruf definierten Zeitraums erfolgt, dann müssen vom Konnektor die folgenden Aktionen ausgeführt werden:
1) entferne das Lock von der Karte durch Aufruf von TUC_KON_023 „Karte reservieren“ {
cardSession = $cardSession;
doLock = false }.
2) Ereignis auslösen durch TUC_KON_256 Systemereignis {
topic = „CARD/SecureSendAPDU/TIMEOUT“;
eventType = Op;
severity = Info;
parameters = (Value=True, CardHandle=$CardHandle, CardType=eGK)}
[<=, Konnektor Highspeed, Konnektor PTV6, funkt. Eignung: Test Produkt/FA**]**

Prüfverfahren: Funktionale Eignung

Der Inhalt von Kapitel 4.1.5.4.21 wird auf 4.1.5.4.22 inkrementiert. Die darauf folgenden Kapitel der Dokumentenebene 5.1.5.4.* inkrementieren entsprechend.

Es wird in Kapitel 4.1.5.4.21 eine neue Anforderung A_25822 aufgenommen.

[api-popp] entspricht aktuell dem Pull Request unter dem Link https://github.com/gematik/api-popp/pull/1

A_25822 - Operation SecureSendAPDU
Der Konnektor MUSS an der Außenschnittstelle eine Operation SecureSendAPDU, wie in Tabelle TAB_KON_270 Operation SecureSendAPDU beschrieben, anbieten.

Tabelle 1: TAB_KON_270 Operation SecureSendAPDU

Name	**SecureSendAPDU**	
Beschreibung	Die Operation sendet eine Liste von Kommando-APDUs an eine Karte und liefert die Liste der Rückgabe-APDUs. Die Zuordnung der Kommando-APDUs und der Rückgabe-APDUs ergibt sich aus der Reihenfolge in den Listen. In der Liste der Kommando-APDUs kann vor jedem Kommando-APDU eine Liste mit erwarteten StatusCodes zu dem jeweiligen Kommando-APDU mitgeschickt werden. Die Liste der Rückgabe-APDUs enthält ausschließlich Rückgabe-APDUs.	
Aufrufparameter	Name	Beschreibung
	`CCTX:Context`	MandantId, CsId, WorkplaceId verpflichtend;
	`CONN:CardHandle`	Adressiert die Karte, an die die APDUs geschickt werden sollen. Die Operation MUSS nur eGK unterstützen. Wird die Operation mit einem nicht unterstützten Kartentypen aufgerufen, so MUSS der Konnektor die Bearbeitung mit dem Fehler 4209 abbrechen.
	`Document`	Enthält das zur Signatur gehörende Dokument (siehe [OASIS-DSS] Section 2.4.2 und oben). Das Format von `SIG:Document` entspricht dem Format von `Scenario` wie in [api-popp] beschrieben. Ein `Scenario` ist eine Struktur bestehend aus `Elements`.
	`dss:SignatureObject`	Eine Signatur über das `SIG:Document`. Enthält die zu prüfende Signatur. Hierbei wird sie als `dss:Base64Signature` mit entsprechend gesetztem `Type`-Attribut (siehe `SignatureType`, Operation SignDocument) übergeben, wobei nur CMS-Signatur und der Wert "urn:ietf:rfc:565" unterstützt werden MUSS.

Name	**SecureSendAPDU**	
	X509Certificate	Ein Base64-kodiertes XML-Element, in dem das Zertifikat, das den asymmetrischen Schlüssel enthält (öffentlicher Schlüssel), DER-kodiert übergeben wird.
Rückgabe	Name	Beschreibung
	CONN:Status	Enthält den Ausführungsstatus der Operation
	Document	Enthält die Liste der Rückgabe-APDUs (ResultList). Das Format von ResultList ist in [api-popp] beschrieben.
	SessionID	Optionaler Rückgabeparameter UUID gem. [RFC4122]
Vorbedingung	keine	
Nachbedingung	keine	

Der Ablauf der Operation SecureSendAPDU ist in Tabelle TAB_KON_271 Ablauf SecureSendAPDU beschrieben.

Tabelle 2: TAB_KON_271 Ablauf SecureSendAPDU

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

2.	TUC_KON_000 „Prüfe Zugriffsberechtigung“	Prüfung der Zugriffsberechtigung durch den Aufruf TUC_KON_000 {
		mandantId = Context.mandantId;
		clientSystemId = Context.clientsystemId;
		workplaceId = Context.workplaceId;
		userId = Context.userId;
		CardHandle }
		Tritt bei der Prüfung ein Fehler auf, bricht die Operation mit Fehlercode aus TUC_KON_000 ab.

Nr.	Aufruf Technischer Use Case oder Interne Operation	Beschreibung

3.	TUC_KON_026 „Liefere CardSession“	1. Ermittle $cardSession über TUC_KON_026 {
		mandantId = `Context.mandantId`;
		clientsystemId = `Context.clientsystemId`;
		userId = `Context.userId`;
		`CardHandle` }
		2. Falls im vorhergehenden Aufruf von SendSecureAPDU ein Systemereignis gem. A_25860-* gesendet wurde, dann
		a) sende Ereignis TUC_KON_256 Systemereignis {
		topic = „CARD/SecureSendAPDU/TIMEOUT“;
		eventType = Op;
		severity = Info;
		parameters = (Value=False, CardHandle=$CardHandle, CardType=eGK)}
		b) breche die Operation mit Fehler 4287 ab.

4.	TUC_KON_037 „Zertifikat prüfen“	Die Zertifikatsprüfung erfolgt durch Aufruf von TUC_KON_037. Als Parameter des TUC-Aufrufs gilt: {
		certificate = `Certificate`
		qualifiedCheck =if_QC_present;
		baseTime = SystemTime;
		offlineAllowNoCheck = false;
		policyList = alle zugelassenen Zertifikatstyp-OIDs;
		intendedKeyUsage = empty;
		intendedExtendedKeyUsage = empty;
		gracePeriod = empty;
		validationMode = OCSP;
		ocspResponses = empty}.

5.	TUC_KON_161 „nonQES Dokumentensignatur prüfen“	Die nonQES wird geprüft. Als `signedDocument` wird `SIG:Document` übergeben. Tritt hierbei ein Fehler auf, bricht die Operation ab.

6.	TUC_KON_023 „Karte reservieren“	1. Prüfe, dass der Karte entweder kein Lock zugeordnet ist oder der Aufrufer das Karten-Lock besitzt.
		2. Die an der Operation beteiligte Karte wird für die exklusive Nutzung reserviert. Die Reservierung der Karte erfolgt durch Aufruf von TUC_KON_023 „Karte reservieren“ {
		cardSession = $cardSession;
		doLock = true }.

7.	TUC_KON_200 „SendeAPDU“	1. Setze die Liste der erwarteten StatusCodes ($StatusCodeList) durch Leeren der Liste und Einfügen eines Code-Elements mit Wert 0x9000 zurück. 2. Dekodiere `Document` (siehe Eingabeparameter) und extrahiere als $Scenario eine Liste von $Elements sowie $SequenceCounter und ggf. $SessionID. Das Format von $Scenario und $Element ist in [api-popp] beschrieben. Sind dabei die dekodierten Eingabeparameter nicht nach [api-popp] validierbar, dann bricht die Operation mit Fehler 4286 ab. Falls die laufende Sequenznummer $SequenceCounter nicht die erste und nicht das Inkrement des vorhergehenden Aufrufes ist, dann bricht die Operation mit Fehler 4286 ab. a. Falls eine leere Liste extrahiert wurde: i. und es ist das Erste Szenario einer Sequenz ($SequenceCounter = 0), dann generiere eine UUID gem. [RFC4122], setze `SessionID` mit dieser gleich und persistiere diese im Kontext der $cardSession. ii. andernfalls: A. Falls die aus $Scenario extrahierte $SessionID nicht mit der `SessionID` aus dem zu $cardSession persistierten Kontext (aus vorherigen Aufrufen der Operation) übereinstimmt, dann bricht die Operation mit Fehler 4286 ab. B. andernfalls springe zu Punkt 4. b. andernfalls: i. Falls es das Erste Szenario einer Sequenz ($SequenceCounter = 0) ist, dann bricht die Operation mit Fehler 4286 ab. ii. Falls die aus $Scenario extrahierte $SessionID nicht mit der `SessionID` aus dem zu $cardSession persistierten Kontext (aus vorherigen Aufrufen der Operation) übereinstimmt, dann bricht die Operation mit Fehler 4286 ab. 3. Führe aus für jedes $Element aus $Scenario: • Falls $Element eine Listevon erwarteten StatusCodes (ExpectedStatusWords) ist

Nr.	Aufruf Technischer Use Case oder Interne Operation	Beschreibung

		• $StatusCodeList = "$Element"
		• Falls $Elementeine Kommando-APDU (CommandAPDU) ist
		• Ermittle $responseAPDU aus dem $Elementmittels Aufruf von TUC_KON_200 {
		• cardSession = "$cardSession";
		• ctId nicht übergeben;
		• commandAPDU = "$Element" }
		• Hänge $responseAPDUan das Ende der `ResultList`
		• Falls Status der $responseAPDUnicht in `StatusCodeList`
		• nimmt die Warnung 4284 in die Antwort auf
		• verlasse die Schleife
		• Falls $Element eine Logging-Information (LoggingInformation) ist
		• führe für das $Element keine Aktionen durch
		• In jedem anderen Fall bricht die Operation mit Fehler 4285 ab
		4. Bereite den nächsten Aufruf von SecureSendAPDU vor:
		• Falls $Scenario.TimeSpan = 0 (das Format von `Scenario` ist in [api-popp] beschrieben):
		• entferne das Lock von der Karte durch Aufruf von TUC_KON_023 „Karte reservieren“ {
		cardSession = $cardSession;
		doLock = false}.
		• Entferne die aus $Scenario extrahierte $SessionID aus dem persistierten $cardSession Kontext.
		• Andernfalls
		• setze Timer auf Wert von $Scenario.TimeSpan
		5. Gebe Rückgabeparameter zurück

Tabelle 3: TAB_KON_272 Fehlercodes SecureSendAPDU

Fehlercode	ErrorType	Severity	Fehlertext
Neben den Fehlercodes der aufgerufenen technischen Use Cases können folgende weitere Fehlercodes auftreten:			
4000	Technical	Error	Syntaxfehler
4209	Technical	Error	Kartentyp %CardType% wird durch diese Operation nicht unterstützt.
4284	Technical	Warning	APDU konnte nicht verarbeitet werden.
4285	Technical	Error	Unerwartetes Sequence-Element
4286	Technical	Error	Inhalt von `Document` nicht valide
4287	Technical	Error	Folgeaufruf von SecureSendAPDU zu spät erfolgt (%CardHandle%)

[<=, Konnektor Highspeed, Konnektor PTV6, Sich.techn. Eignung: CC-Evaluierung, funkt. Eignung: Test Produkt/FA, Sich.techn. Eignung: Prüfung durch CC-Prüfstelle**]**

1.2 Anhang F - Übersicht Events

Tabelle 4 TAB_KON_777 Events Interne Mechanismen

Topic Ebene1 /Topic Ebene2 /Topic Ebene3	Typ	Schwere	Prot	An Clients	Parameter	Bedeutung	Auslöser (TUC/Op)
CARD /SecureSendAPDU /TIMEOUT	Op	Info	x	x	Value=true/false; CardType=$; CardHandle=$;	Deadline für Folgeaufruf in cardSession überschritten	SecureSendAPDU

1.3 Kapitel 5.5.1 Dokumente der gematik

Es wird an die Tabelle in dem Kapitel eine neue Zeile angehängt, in der nach der Veröffentlichung der GitHub-Link zum [api-popp] spezifiziert wird.

2 Änderungen in Steckbriefen

2.1 Änderungen in gemProdT_Kon_Highspeed_PTV

Anmerkung: Die Anforderungen der folgenden Tabelle stellen einen Auszug dar und verteilen sich innerhalb der Tabelle des Originaldokuments [gemProdT_...]. Alle Anforderungen der Tabelle des Originaldokuments, die in der folgenden Tabelle nicht ausgewiesen sind, bleiben unverändert bestehenden.

Tabelle 5: Anforderungen zur funktionalen Eignung "Produkttest/Produktübergreifender Test"

Afo-ID	Afo-Bezeichnung	Quelle (Referenz)
A_25822	Operation SecureSendAPDU	gemSpec_Kon
A_25860	Reaktion auf abgelaufenen APDU-Szenario-Timer	gemSpec_Kon

2.2 Änderungen in gemProdT_Kon_PTV6

Anmerkung: Die Anforderungen der folgenden Tabelle stellen einen Auszug dar und verteilen sich innerhalb der Tabelle des Originaldokuments [gemProdT_...]. Alle Anforderungen der Tabelle des Originaldokuments, die in der folgenden Tabelle nicht ausgewiesen sind, bleiben unverändert bestehenden.

Tabelle 6: Anforderungen zur funktionalen Eignung "Produkttest/Produktübergreifender Test"

Afo-ID	Afo-Bezeichnung	Quelle (Referenz)
A_25822	Operation SecureSendAPDU	gemSpec_Kon
A_25860	Reaktion auf abgelaufenen APDU-Szenario-Timer	gemSpec_Kon

2.3 Sonstige Änderungen

Es werden in api-telematik die Dateien `CardService_v8_2_0.xsd` und `CardService_v8_2_0.wsdl` neu aufgenommen.

Aktuell sind sie in Pull Request https://github.com/gematik/api-telematik/pull/21 zu finden.

Die darin enthaltenen Änderungen müssen vom Konnektor umgesetzt werden.