

**Elektronische Gesundheitskarte und Telematikinfrastruktur**

# **Feature: TI-Messenger Automatisierung und Bots**

Version: 1.0.0 CC  
Revision: 1186333  
Stand: 02.04.2025  
Status: zur Abstimmung freigegeben  
Klassifizierung: öffentlich\_Entwurf  
Referenzierung: gemF\_TI-M\_Automatisierung

---

## Dokumentinformationen

---

### Änderungen zur Vorversion

Anpassungen des vorliegenden Dokumentes im Vergleich zur Vorversion können Sie der nachfolgenden Tabelle entnehmen.

### Dokumentenhistorie

Version	Stand	Kap./ Seite	Grund der Änderung, besondere Hinweise	Bearbeitung
0.5.0	03.12.202 4		initiale Erstellung	gematik
1.0.0 CC	02.04.202 5		zur Abstimmung freigegeben	gematik

---

## Inhaltsverzeichnis

---

<b>1 Einordnung des Dokuments.....</b>	<b>5</b>
<b>1.1 Zielsetzung.....</b>	<b>5</b>
<b>1.2 Zielgruppe.....</b>	<b>5</b>
<b>1.3 Abgrenzungen.....</b>	<b>6</b>
<b>1.4 Methodik.....</b>	<b>6</b>
<b>2 Epic und User Stories.....</b>	<b>7</b>
<b>2.1 Epic Automatisierung und Bots.....</b>	<b>7</b>
2.1.1 User Stories eines Leistungserbringers.....	7
2.1.2 User Stories einer Krankenkasse.....	8
2.1.3 User Stories eines Versicherten.....	9
<b>2.2 Ableitungen für Hersteller, Betreiber und Anbieter.....</b>	<b>9</b>
2.2.1 Hersteller.....	9
2.2.2 Anbieter.....	10
2.2.3 Betreiber.....	10
2.2.4 Akzeptanzkriterien und Designziele für technische Komponenten.....	10
<b>3 Einordnung in die Telematikinfrastuktur.....</b>	<b>12</b>
<b>4 Technisches Konzept.....</b>	<b>13</b>
<b>4.1 Architekturüberblick und Struktursicht.....</b>	<b>14</b>
4.1.1 Beschreibung der Komponenten.....	15
4.1.1.1 <i>Bot-Engine (Package)</i> .....	15
4.1.1.1.1 Bot-Engine (Component).....	16
4.1.1.1.2 Bot-Config.....	17
4.1.1.2 <i>Bot (Package)</i> .....	18
4.1.1.2.1 Custom Business Logic.....	18
4.1.1.2.2 Bot-Config (Instanz der Bot-Config).....	18
4.1.1.2.3 Bot-Identität-T.....	18
4.1.1.2.4 Bot-Identität-HC.....	21
4.1.1.3 <i>TI-M HeadlessClient</i> .....	21
4.1.1.3.1 Client-Config.....	22
4.1.1.3.2 Signature-Extension-C.....	22
4.1.1.3.3 Auth-Mapper.....	23
4.1.1.4 <i>TI-M Fachdienst Pro</i> .....	24
4.1.1.4.1 Signature-Extension-MS.....	24
4.1.1.5 <i>Ergänzung zum Testverfahren</i> .....	24
<b>4.2 Laufzeitsicht.....</b>	<b>25</b>
4.2.1 Rollenerweiterung.....	25
4.2.2 Authentication-Flow für den Port Auth1.....	25

4.2.3 zusätzliche Sicherheitsleistung des Messenger-Service.....	28
4.2.4 Authentication-Flow für den Port Auth2.....	30
4.2.4.1 alternativer Authentication-Flow.....	31
4.2.4.2 Verwendung des Auth-Mappers.....	31
4.2.5 Zugriff des Bots auf den Messenger Service.....	32
<b>4.3 Zulassungs-, Deployment- und Betriebssicht.....</b>	<b>32</b>
4.3.1 Szenario 1: Standardfall in einer unvirtualisierten Umgebung.....	34
4.3.2 Szenario 2: Deployment mit Maximalverteilung in einer virtualisierten Umgebung.....	35
4.3.3 Szenario 3: Deployment mit Minimalverteilung virtualisiert (Deployment- Monolith).....	37
4.3.4 Hinweise zu den Deployment-Szenarien.....	38
4.3.5 Betriebliche und organisatorische Auswirkungen.....	38
<b>5 Spezifikation.....</b>	<b>40</b>
<b>6 Anhang A - Verzeichnisse.....</b>	<b>41</b>
<b>6.1 Abkürzungen.....</b>	<b>41</b>
<b>6.2 Abbildungsverzeichnis.....</b>	<b>41</b>
<b>6.3 Tabellenverzeichnis.....</b>	<b>41</b>
<b>6.4 Referenzierte Dokumente.....</b>	<b>42</b>
6.4.1 Dokumente der gematik.....	42
6.4.2 Weitere Dokumente.....	42

---

## 1 Einordnung des Dokuments

---

Das Dokument beschreibt das Feature Automatisierung und Bots für den TI-Messenger. Durch das Feature soll eine effizientere Kommunikation und Automatisierung von Prozessen im Gesundheitswesen ermöglicht werden.

### 1.1 Zielsetzung

Das Feature zielt darauf ab, dass wiederkehrende Aufgaben über den TI-Messenger automatisiert werden können. Die gematik setzt dabei klare Richtlinien und Rahmenbedingungen, um die sichere und rechtmäßige Nutzung von Bots zu gewährleisten. Ein entscheidendes Ziel ist die Steuerung und Regelung von Bots aus Sicht der gematik als Zulassungsorganisation.

### 1.2 Zielgruppe

Die Zielgruppe dieses Features sind Endanwender des TI-Messengers (Leistungserbringer, Krankenkassen, Versicherte). Für diese Nutzer-Gruppen steht die Verbesserung der Kommunikation untereinander im Vordergrund. Der Einsatz von Bots soll die Kommunikation beschleunigen und den Arbeitsaufwand im Gesundheitswesen reduzieren, indem Routineanfragen automatisiert und Informationen schnell bereitgestellt werden. Dies ermöglicht es, administrative Aufgaben zu minimieren und die Fokussierung auf die direkte Patientenversorgung zu verbessern. Ein weiteres Ziel ist die Transparenz und Nachvollziehbarkeit der Kommunikation: Nutzer sollen jederzeit klar erkennen können, ob sie mit einem Bot oder einem menschlichen Ansprechpartner kommunizieren. Dadurch wird das Vertrauen in die Technologie gestärkt und eine klare Verantwortungskette etabliert.

Im Sinne der Entwicklung richtet sich dieses Dokument vorrangig an Hersteller und Anbieter des TI-Messengers. Im Betrieb soll sichergestellt sein, dass Bots zuverlässig, sicher und rund um die Uhr verfügbar sind. Dies erfordert eine robuste technische Infrastruktur, die eine kontinuierliche Überwachung der Systemleistung und der Bot-Aktivitäten ermöglicht. Ein besonderes Augenmerk liegt auf der Messbarkeit und Auswertbarkeit der Bot-Nutzung, um die Effizienz und Effektivität der eingesetzten Lösungen zu bewerten und kontinuierlich zu verbessern. Des Weiteren ist es entscheidend, dass der Betrieb reibungslos abläuft, sodass Bots ihre Aufgaben ohne menschliches Eingreifen erfüllen können. Hierzu gehört auch die schnelle Erkennung und Behebung von Betriebsstörungen sowie eine umfassende Dokumentation aller Aktivitäten.

Die gematik verfolgt spezifische Ziele im Bereich der Test- und Zulassungsprozesse für Bots. Ein zentrales Ziel ist die Schaffung transparenter und effizienter Rahmenbedingungen für Hersteller und Anbieter. Dabei wird die Eigenverantwortung der Hersteller und Anbieter betont. Die gematik setzt damit auch Leitplanken für Bots. Diese unterliegen bereits gesetzlichen Vorgaben, die von den Anbietern unabhängig von zusätzlichen Anforderungen der gematik eingehalten werden müssen. Die technischen Rahmenbedingungen der gematik werden so gestaltet, dass Bots zuverlässig, sicher und flexibel in den TI-Messenger integriert werden können. Ein weiterer Aspekt ist die Vermeidung zusätzlicher bürokratischer Hürden. Es werden keine umfangreichen Zulassungstests und Gutachten gefordert, um Innovationen zu fördern und unnötige

administrative Belastungen zu vermeiden. Stattdessen wird ein risikobasierter Bewertungskatalog eingeführt, der die Komplexität und das Risikoniveau der jeweiligen Bots berücksichtigt und entsprechend differenzierte Anforderungen stellt. Die Eigenverantwortung wird durch einen Selbstdeklarationsprozess gestärkt, bei dem Anbieter die Einhaltung der erforderlichen Standards selbst bestätigen, ergänzt durch stichprobenartige Überprüfungen.

## 1.3 Abgrenzungen

Die gematik schließt explizit folgende Punkte aus dem Feature-Dokument aus ("Out of Scope"):

- Geschäfts-Logiken von Bots (Förderung von Innovationen)
- Kein Zwang zur Nutzung von Bots (Keine Anbieterverpflichtung)
- Keine fachlichen Vorgaben für einzelne Bots (Marktviefalt)
- Keine Zulassungsverfahren für einzelne Bots (Simple Inbetriebnahme)

## 1.4 Methodik

### Betrachtung von Alternativen

Dieses Dokument betrachtet grundsätzlich einen möglichen validen Pfad zur Inbetriebnahme von Bots in einer Organisation. Dennoch ist gelegentlich von Alternativen die Rede, die alternative Handlungsoptionen bezeichnen von denen jeweils nur eine Alternative umzusetzen ist. Diese Alternativen eröffnen einen flexiblen Handlungsspielraum in der Umsetzung der angebotenen Architektur.

### User Stories

Eine User Story ist eine in Alltagssprache formulierte Software-Anforderung. Sie ist bewusst kurz gehalten und umfasst in der Regel nicht mehr als zwei Sätze. User Stories werden im Rahmen der agilen Softwareentwicklung zusammen mit Akzeptanztests zur Spezifikation von Anforderungen eingesetzt.

Aus diesem Grund kann in den User Stories eine abweichende Terminologie genutzt werden, welche für den Leser nachvollziehbar (bspw. Patient = Versicherter) ist.

### Anforderungen

Anforderungen als Ausdruck normativer Festlegungen werden durch eine eindeutige ID sowie die dem RFC 2119 [RFC2119] entsprechenden, in Großbuchstaben geschriebenen deutschen Schlüsselworte MUSS, DARF NICHT, SOLL, SOLL NICHT, KANN gekennzeichnet.

Anforderungen werden im Dokument wie folgt dargestellt:

**<AFO-ID> - <Titel der Afo>**

Text / Beschreibung

[<=]

Dabei umfasst die Anforderung sämtliche zwischen Afo-ID und Textmarke [<=] angeführten Inhalte.

---

## 2 Epic und User Stories

---

In diesem Abschnitt wird das Feature fachlich motiviert und der Mehrwert für Nutzer vorgestellt. Aus diesen Epics und User Stories wird anschließend ein technisches Konzept abgeleitet. Im Hinblick auf eine möglichst weitreichende Entwicklungsfreiheit für Hersteller des TI-Messengers skizzieren die User Stories lediglich beispielhafte Anwendungsfälle. Die User Stories haben keinen normativen Charakter und sollen die Zielsetzung ("effizientere Kommunikation und Automatisierung von Prozessen") verdeutlichen.

### 2.1 Epic Automatisierung und Bots

Die Kommunikation im deutschen Gesundheitswesen findet auf vielen Wegen statt. Der primäre Weg ist eine direkte Kommunikation von Mensch zu Mensch, beispielsweise über folgende Kanäle:

- Persönliche Gespräche
- Telefonate
- E-Mail
- KIM

Mit dem TI-Messenger werden die Kommunikationskanäle im Gesundheitswesen um einen sicheren, verschlüsselten Weg erweitert. Leistungserbringer und Krankenkassen möchten ihre Kommunikation möglichst effizient gestalten. Das Feature Automatisierung und Bots soll sicherstellen, dass Chat-Nachrichten in einer sicheren Umgebung automatisiert erzeugt und beantwortet werden können.

Um den TI-Messenger sinnstiftend nutzen zu können, ist es notwendig, diesen in die bestehende Infrastruktur von Leistungserbringern und Krankenkassen einzubinden. Erst eine Integration ermöglicht es den TI-Messenger als Kanal für einen strukturierten Datenaustausch zu nutzen. Im Rahmen dieser Integration sind Bots ein zentraler Bestandteil um zu ermöglichen, dass Abläufe und Prozesse automatisiert werden können.

#### 2.1.1 User Stories eines Leistungserbringers

Nach heutigem Stand empfinden Leistungserbringer eingehende Kommunikation als zusätzlichen, zeitlichen Aufwand im Versorgungsalltag. Eine Unterstützung des TI-Messengers mit Hilfe von Automatisierung könnte diesen Aufwand reduzieren. Bots sind in der Lage eingehende Anfragen von Patienten zu strukturieren, zu filtern, zu beantworten und damit manuellen Aufwand für die Kommunikation reduzieren.

##### **Als Leistungserbringer möchte ich, dass**

- der Kontakt zu Bestandspatienten erleichtert wird, sodass ich außerhalb eines Behandlungskontextes automatisiert Einladung zu Präventions- und Vorsorgeuntersuchungen senden kann
- Recalls an Patienten ermöglicht werden, sodass ich automatische Erinnerungen von Leistungserbringern an Patienten (Termin-Erinnerungen, Impfungen, Verlaufskontrolle für Chroniker)

- Chroniker engmaschiger betreut werden können, indem ich automatisierte Erinnerungen für Verlaufskontrolle sende
- Patienten vor einer anstehenden Behandlung Unterlagen von mir erhalten (beispielsweise Anamnese-Formulare, Einwilligungserklärungen, Aufklärung im Rahmen einer Untersuchung, bildgebende Verfahren wie CT/MRT, oder Operationen)
- Archivierungen von Chatverläufen automatisiert in mein Primärsysteme angestoßen werden, sodass keine manueller Export und Import mehr stattfinden muss
- nach der Behandlung Informationen über ein Behandlungsergebnis bereitstellen, beispielsweise einen Patientenbrief in leicht-verständlicher Form, um die Gesundheitskompetenz meiner Patienten zu erhöhen
- Nachrichten von Patienten (Medikamenten-Bestellung, Anfragen für Überweisungen, Befundunterlagen, Termine) automatisiert durch einen Bot beantwortet werden, sodass der zeitliche Aufwand für mich und meine Einrichtung reduziert wird

### 2.1.2 User Stories einer Krankenkasse

Auch Krankenkassen möchten ihre Kommunikation möglichst effizient gestalten. Auch hier können Bots unterstützen, dass eingehende Nachrichten für eine adäquate Weiterverarbeitung an den richtigen Ort weitergeleitet werden.

#### Als Krankenkasse möchte ich, dass

- meine Kundenbetreuer keine zusätzliche Benutzeroberfläche eines TI-Messengers bedienen müssen, sodass eingehende TI-M-Nachrichten von Versicherten in die bestehenden Arbeitsabläufe vollständig integriert werden können
- eingehende TI-M-Nachrichten von einem Service-Bot angenommen werden können, damit dieser eine Weiterleitung an das angeschlossene Ticketsystem im Krankenkassensystem anstößt
- eingehende TI-M-Nachrichten einem (oder mehreren) verantwortlichen Sachbearbeitern von einem Service-Bot je nach Inhalt automatisch zugeordnet werden können, damit diese Nachrichten nicht manuell kategorisiert werden müssen
- Nachrichten mit Anhängen automatisiert an zuständige Person (Kundenbetreuer) bzw. Funktion weitergeleitet und zugeordnet werden können
- den Versand (outbound) und die Annahme (inbound) von Anträgen, Genehmigungen und Bescheiden zu Verwaltungsakten nach SGB X über den TI-Messenger erledigen können
- ausgehende papierbasierte Kommunikation reduziert werden kann, sodass Porto-Kosten entfallen
- automatisierte Einladungen an Versicherten zu Präventions- und Vorsorgeuntersuchungen verschickt werden können

#### Als Kundenbetreuer möchte ich

- eingehende Nachrichten von Versicherten aus meiner gewohnten Arbeitsumgebung ("Contact Center") heraus beantworten, um kein weiteres System erlernen, bedienen und überwachen zu müssen

### 2.1.3 User Stories eines Versicherten

#### Als Versicherter möchte ich



- erfahren, ob ein Antrag an meine Krankenkasse bearbeitet wurde bzw. bis wann mit einer Bearbeitung zu rechnen ist
- wie sie an eine bestimmte Leistung kommen. Z.B. „Ich bin Mutter geworden, wie versichere ich jetzt mein Kind?“.
- Versicherte fragen, wie sie etwas beantragen. Z.B. „Mein Kind ist krank. Wie bekomme ich jetzt mein Geld?“
- Versicherte fragVersicherte teilen uns mit: „Ich bin umgezogen. Meine neue Adresse ist xyz.“en, wie sie etwas beantragen (z.B. „Mein Kind ist krank. Wie bekomme ich jetzt mein Geld?“)
- Versicherte schicken uns eine Datei (meistens Foto) eines Belegs – z.B. Kinderpflegekrankengeld-Bescheinigung.
- Versicherte fragen uns z.B. wie viel sie selbst zuzahlen müssen beim Zahnarzt. Wir fragen dann alle Infos ab und berechnen dann die Zuzahlung.
- Nachrichten von meiner Krankenkasse per Chat zugestellt bekommen anstatt per Brief
- meine Nachricht an einen zentrale Kontaktpunkt ("öffentlichen Funktionsaccount") meiner Krankenkasse schicken, sodass ich nicht auswählen muss, an welche Fachabteilung mein Anliegen korrekt adressiert ist.
- 24/7 eine zügige Reaktion auf meine Nachrichten an eine Organisation erhalten, um mir sicher zu sein, dass ich keinen analogen Kanal (Telefon, Brief) als Alternative in Betracht ziehen muss.

## 2.2 Ableitungen für Hersteller, Betreiber und Anbieter

### 2.2.1 Hersteller

Hersteller sollen in die Lage versetzt werden, flexible und anpassbare Bot-Systeme zu entwickeln, die den spezifischen Anforderungen der Nutzer gerecht werden.

#### **Beispielhafte Nutzungsszenarien:**

1. Es soll möglich sein, Systeme zu entwickeln, die Routineaufgaben wie Folgerezepte und Terminvereinbarungen automatisieren, um die Effizienz zu steigern.
2. Es soll möglich sein, personalisierte Nachrichten basierend auf vorheriger Korrespondenz zu versenden.
3. Es soll möglich sein, Systeme zu entwickeln, die mehrere Sprachen unterstützen, um die Zugänglichkeit zu verbessern.
4. Es soll möglich sein, Sicherheitsstandards in die Systemarchitektur zu integrieren, um Daten zu schützen.

### 2.2.2 Anbieter

Anbieter sollen befähigt werden, Bot-Lösungen bereitzustellen, die nahtlos in bestehende Systeme integriert werden können und den spezifischen Kontexten ihrer Kunden gerecht werden.

#### **Beispielhafte Nutzungsszenarien:**

1. Es soll möglich sein, Systeme bereitzustellen, die einfach aktualisiert und an neue Anforderungen angepasst werden können.

2. Es soll möglich sein, Feedback von Nutzern zu sammeln und zur Verbesserung der bereitgestellten Lösungen zu nutzen.
3. Es soll möglich sein, Mechanismen zur Überwachung der Bot-Leistung zu implementieren, um kontinuierliche Optimierungen zu ermöglichen.
4. Es soll möglich sein, Datenschutzrichtlinien klar zu kommunizieren und einzuhalten, um das Vertrauen der Nutzer zu stärken.
5. Es soll möglich sein, die Verwaltung und Wartung der Bots zentral zu steuern, um einen reibungslosen Betrieb sicherzustellen.

### 2.2.3 Betreiber

Betreiber sollen in die Lage versetzt werden, den Betrieb und die Sicherheit von Bot-Systemen effektiv zu überwachen und zu verwalten, um die Integrität und Verfügbarkeit der Dienste sicherzustellen.

#### Beispielhafte Nutzungsszenarien:

1. Es soll möglich sein, Echtzeitüberwachung und Protokollierung der Bot-Aktivitäten durchzuführen, um die Leistung zu sichern.
2. Es soll möglich sein, Alarme für ungewöhnliche oder problematische Antworten zu konfigurieren.
3. Es soll möglich sein, sicherzustellen, dass nur autorisierte Personen Änderungen an den Systemen vornehmen können.
4. Es soll möglich sein, regelmäßige Sicherheitsüberprüfungen und Updates zu unterstützen, um die Sicherheit zu gewährleisten.
5. Es soll möglich sein, klare Kommunikationswege und Eskalationspunkte zu definieren, um die Nutzerfreundlichkeit zu verbessern.
6. Es soll möglich sein, den unterbrechungsfreien Betrieb der Bots sicherzustellen, um die Verfügbarkeit der Dienste zu gewährleisten.
7. Es soll möglich sein, Daten strukturiert zu verwalten und an interne Systeme zu übergeben, um eine effiziente Verarbeitung zu gewährleisten.

### 2.2.4 Akzeptanzkriterien und Designziele für technische Komponenten

#### TI-M HeadlessClient

- Der TI-M HeadlessClient fungiert als Matrix-Client ohne Benutzeroberfläche.
- Er muss in der Lage sein, unterbrechungsfrei und ohne manuelle Eingaben durch Administratoren zu arbeiten.
- Der TI-M HeadlessClient ist voll zulassungspflichtig für Hersteller, was bedeutet, dass er den gematik-Zulassungsprozess durchlaufen muss.

#### Bot-Engine

- Die Bot-Engine ermöglicht es, dass sich Bots am HC anmelden können.
- Sie stellt sicher, dass nur von der Organisation bzw. vom Anbieter explizit autorisierte Bots an Chats teilnehmen dürfen.
- Die Bot-Engine kann auch administrative und fachliche Tätigkeiten von Automation Admins an einzelnen Bots ermöglichen.

- Nur Bots, die mittels der Bot-Engine autorisiert sind, dürfen sich am TI-M HeadlessClient anmelden und an Chats teilnehmen.
- TI-M HeadlessClient & Bot-Engine können theoretisch auch ein System sein.

### **Bot**

- Diese enthalten spezifische Geschäftslogik und interagieren entsprechend mit Benutzern in Chatrooms.
- Bots sind nicht Teil des TI-Messenger-Systems und unterliegen daher nicht der Zulassungspflicht.
- Nur durch die BE aktivierte Bots dürfen an Chats im HC teilnehmen.

### **Bot Laufzeit Umgebung**

- API-basierte Schnittstelle zur Steuerung, Überwachung & Verwaltung des KTR-seitigen Endpunktes der Ende-zu-Ende-Verschlüsselung

### **Kryptographische Signaturen**

- Authentifizierung der Bots über legitime Authentisierungsmittel (z.B. über SM-B Org Signatur des KTR), damit ein hohes Sicherheitsniveau bei der Verwendung von Bots gewährleistet wird.

---

## 3 Einordnung in die Telematikinfrastuktur

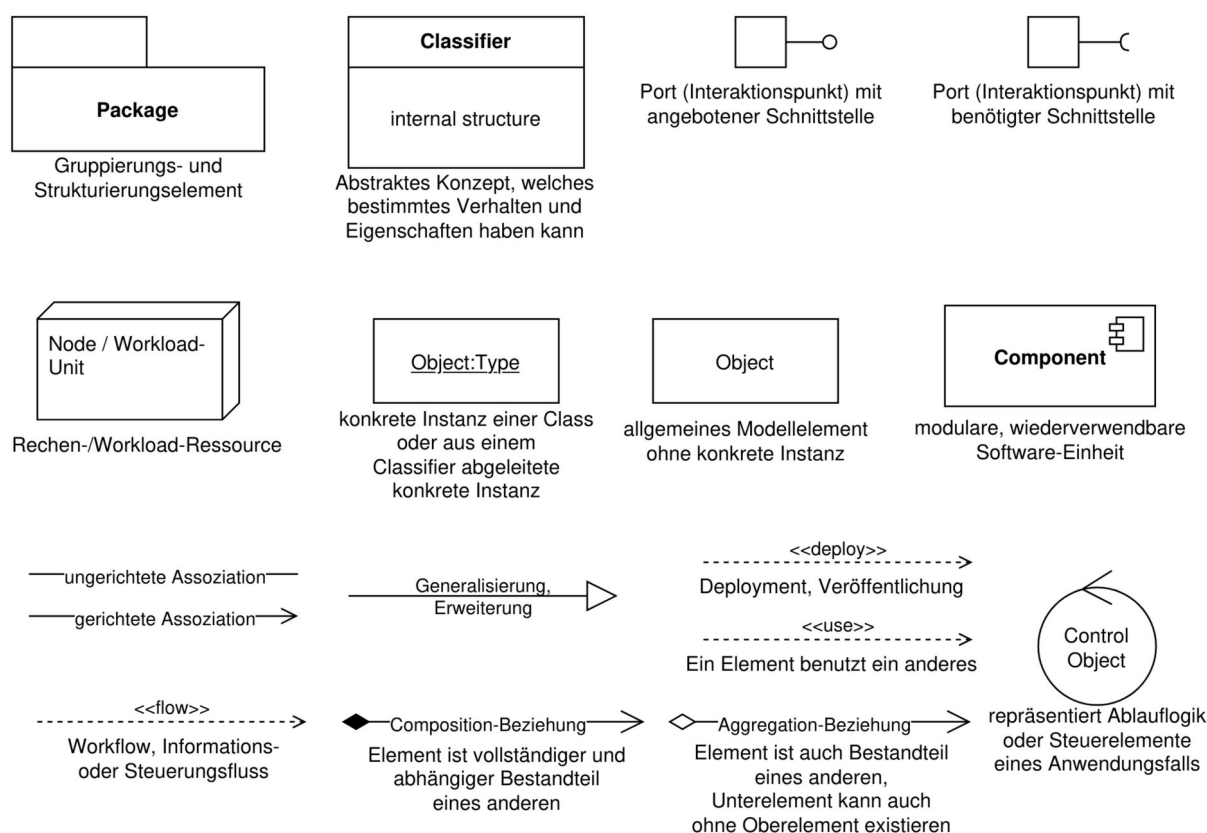
---

**To be continued** (nach Erhalt und Rezipieren des Feedbacks zum technischen Konzept)...

## 4 Technisches Konzept

Mit dem nachstehend beschriebenen technischen Konzept soll es möglich sein, die in Abschnitt 2- Epic und User Stories beschriebenen Bedarfe in der Form zu adressieren, dass im Ergebnis eine Architekturbasis für den Einsatz von Bots vorliegt, mit der sich die beschriebenen User Stories und weitere nach stets demselben Lösungsmuster umsetzen lassen. Die dort genannten User Stories stehen exemplarisch für eine große Spannweite möglicher Nutzungsszenarien. Dabei werden nur diejenigen Bestandteile zulassungsrelevant, die zum Lösungsmuster gehören - es findet eine scharfe Abgrenzung zwischen individueller Funktionalität eines Bots (nicht zulassungsrelevant) und dem Lösungsmuster statt, welches für jeden Bot gleichartig sein soll (zulassungsrelevant). Für den Zulassungsnehmer bedeutet das im Ergebnis, dass die Architekturbasis gemäß dem Lösungsmuster einmal zugelassen wird und damit defacto beliebige Bots implementiert werden können, die aber selbst nicht zugelassen werden müssen (vgl. auch 1.3- Abgrenzungen). Das beschriebene Lösungsmuster selbst lässt außerdem Spielräume für die Umsetzung offen durch die Benennung von Alternativen, Szenarien und Beispielen. Weiterhin eröffnet das Lösungsmuster die Möglichkeit, strukturierte Informationen und Zustandsbeschreibungen zu verarbeiten, auszuwerten und mit diesen Drittsysteme anzubinden.

In diesem Konzept häufig verwendete Modellierungselemente der UML2 werden in folgender Abbildung einführend erläutert:



**Abbildung 1: Legende häufig verwendeter Modellierungselemente**

## 4.1 Architekturüberblick und Struktursicht

Die in diesem Abschnitt beschriebene Architekturperspektive eröffnet einen Blick auf die Struktur der Komponenten, Klassen und Kompositionen. Die beschriebenen Strukturobjekte stellen Umsetzungsanforderungen in dem hier beschriebenen Umfang dar. In der nachfolgenden Architekturüberblicksskizze markieren grüne und grün umschlossene Packages / Komponenten / Klassen Betrachtungsgegenstände im Zulassungsverfahren für jede Organisation bzw. jeden Anbieter, die Bots im TI-Messenger zum Einsatz bringen möchten. Demgegenüber stehen diejenigen Elemente, die in dieser Skizze entweder gar nicht erscheinen oder aber in roter Farbe markiert sind: Sie sind nicht im Zulassungs-Scope und werden selbst dann ignoriert, wenn sie im Zulassungsverfahren als Bestandteil der funktionsfähigen Gesamtlösung vorgelegt werden. Weiß markierte Elemente (nicht von grüner Markierung umschlossen) müssen zwar existent sein, werden aber ansonsten im Zulassungsverfahren inhaltlich ignoriert.

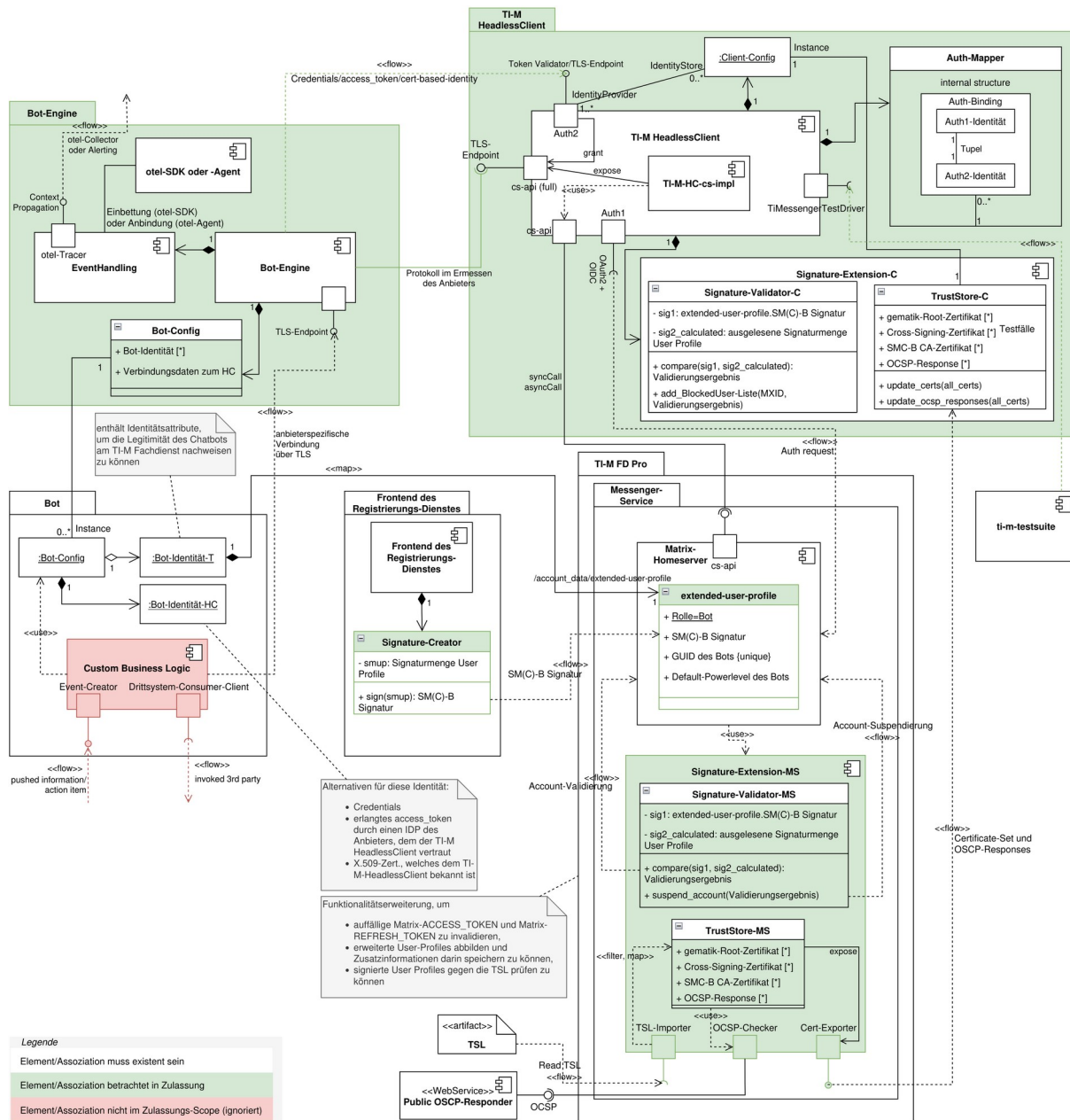


Abbildung 2: Architekturüberblick

### 4.1.1 Beschreibung der Komponenten

In diesem Abschnitt werden die Komponenten der Architekturüberblicksskizze beschrieben, in eine Hierarchie gebracht und begründet. Zudem sind bei jeder der beschriebenen Komponenten weitere Informationen zum avisierten Testverfahren genannt.

#### 4.1.1.1 Bot-Engine (Package)

Die Bot-Engine (Component) wird durch dieses Package als logisches Modul gekapselt, das als Framework den gesamten Ablauf des Zugriffs auf den TI-M HeadlessClient sowie das Processing von ein- und ausgehenden Events regelt. Darüber hinaus definiert es die strukturellen Vorgaben für die Konfiguration eines funktionsfähigen Bots.

Ein Anbieter verwendet dieses logische Modul wiederholt und routinemäßig für verschiedene Bots, wodurch eine effiziente Wiederverwendbarkeit gewährleistet wird. Für die routinemäßige Handhabung des Processings müssen spezifische Vorgaben festgelegt werden, die entweder struktureller oder konfigurativer Natur sind oder durch entsprechende Konventionen definiert werden.

Innerhalb des Moduls erfolgt die zentrale Verwaltung und Vorgabe von Querschnittsaufgaben, zu denen das Logging, das Monitoring sowie das Errorhandling des Event-Processings gehören.

Aus der Perspektive der Zulassung bietet das Modul einen entscheidenden Vorteil: Da das Package eine relative Änderungsstabilität aufweist, entfällt die Notwendigkeit, erneut Bots mit unterschiedlichen Funktionen verschiedener Anbieter einem Zulassungsprozess zu unterziehen.

Die Architektur sieht eine explizite Trennung zwischen dem Package der Bot-Engine und dem Package des TI-M HeadlessClient vor. Diese Trennung erfüllt zwei wichtige Zwecke: Zum einen ermöglicht sie ein separates Deployment beider Komponenten, zum anderen erlaubt sie den Betrieb eines zentralen TI-M HeadlessClient, der zwar als zentrale Instanz fungiert, jedoch nicht eigenständig als Bot agieren kann.

Die Einführung einer zusätzlichen Abstraktionsebene durch diese Modularisierung führt zu einer geringfügigen Erhöhung der Komplexität in der Bot-Implementierung. Falls die Bot-Engine als Bestandteil des Zulassungsprozesses definiert wird, resultiert dies durch die geprüfte Architekturvorgabe in einer Einschränkung der Umsetzungsautonomie.

Als Zulassungsvoraussetzung werden funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet. Es muss möglich sein, das Package zu erzeugen, wobei dies mindestens in Verbindung mit einem konkreten Bot aus dem Bot-Package erfolgen muss. Das Package muss eine eindeutige Identifizierbarkeit aufweisen, die keine Verwechslungen zulässt.

#### 4.1.1.1.1 Bot-Engine (Component)

Die Bot-Engine stellt einen technologieutralen Bestandteil dar, der sich auf die Verarbeitung (Processing) konzentriert. Sie verarbeitet Eingaben der Custom Business Logic, die entweder vom Event-Creator oder dem Drittsystem-Consumer-Client stammen, mittels spezieller EventHandler, die für jeden Verarbeitungstyp individuell vorgehalten werden müssen.

Eingehende Daten aus Chatrooms, wie MessageEvents und StateEvents, werden für die Verarbeitung in der Custom Business Logic entweder vorbereitet oder direkt weitergeleitet, wodurch verschiedene Konnektivitätsmuster und Anbindungsarten ermöglicht werden. Eine Verarbeitung durch die Bot-Engine findet ausschließlich dann statt, wenn eine korrekte Konfiguration (vgl. :Bot-Config im der Abbildung Architekturüberblick und Abschnitt 4.1.1.2.2- Bot-Config (Instanz der Bot-Config)) vorliegt.

Für den Fall, dass die Custom Business Logic dies erfordert, implementiert die Bot-Engine ein Session-Management, das verschiedene Aspekte wie letzte Sync-Actions, Room-Memberships und Profile-Settings des Bots verwaltet.

Die Bot-Engine kapselt den universellen Zugriff auf die vollständig exponierte cs-api (full) des TI-M HeadlessClients, welche selbst die durch die TIM-Spezifikation modifizierte Matrix Client-Server-API darstellt. Darüber hinaus übernimmt die Bot-Engine vollständig die Regelung des Zugriffs auf die Auth2-Schnittstelle des TI-M HeadlessClients.

Aus der Zulassungsperspektive fungiert die Bot-Engine als eine Zwischenschicht, die daraufhin geprüft werden kann, ob sie in der Lage ist, Nachrichten vom TI-M HeadlessClient für die Custom Business Logic sowie Nachrichten von der Custom Business Logic für den TI-M HeadlessClient bereitzustellen. Im Rahmen der Zulassung kann auch die Konfigurationsfähigkeit des Bots in einem noch zu spezifizierenden Umfang



überprüft werden, wobei besonderes Augenmerk auf dem Speichermechanismus für Bot-Identitäten liegt.

Aus softwarearchitektonischer Sicht ermöglicht das Session-Management in der Bot-Engine, dass der TI-M HeadlessClient, mit Ausnahme des Auth-Mappings, zustandslos (stateless) gehalten werden kann. Weitere potenzielle Anforderungen an den TI-M HeadlessClient, die von der bisherigen TIM-Client-Spezifikation abweichen, können funktional in die Bot-Engine ausgelagert werden. Dadurch können die Abweichungen des TI-M HeadlessClient von den regulären TIM-Clients auf ein stabiles Minimum beschränkt werden.

Für die Testbarkeit muss der Zugriff auf die exponierte cs-api (full) mittels einer formalen Schnittstellenspezifikationssprache beschrieben werden, die dann als Grundlage für die Generierung von Testfällen dient. In diesem Zusammenhang muss zum einen die grundsätzliche Erreichbarkeit der Schnittstelle sichergestellt sein. Zum anderen muss die exponierte Menge der cs-api (full) in ihrem inhaltlichen Umfang den Tests entsprechen, die am TiMessengerTestDriver durchgeführt werden.

Als Zulassungsvoraussetzung werden funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet. Zusätzlich plant die gematik die Auswertung der Schnittstellenspezifikation durch Testsuite-Tests.

#### 4.1.1.1.1 EventHandling

Das EventHandling ist als Subkomponente in die Bot-Engine integriert und stellt das entsprechende Event-Processing bereit. Dort können dedizierte Validations- und Routing-Regeln für im Event vorhandene strukturierte Informationen (z.B. FHIR-Ressourcen, Status- und Prozessinformationen) implementiert werden, um dadurch die Kompatibilität eines konkreten Bots zum Senden/Empfangen spezifischer Informationen zu erzwingen. Basierend auf diesen Regeln können gezielt bestimmte Kommunikationsmuster umgesetzt werden.

Über den otel-Tracer-Port ermöglicht das EventHandling den Zugriff auf das Alerting-System des Anbieters oder einen OpenTelemetry (otel) Collector. Die Weiterleitung von Telemetriedaten erfolgt über eine Context Propagation Schnittstelle. Das System ist so konzipiert, dass bestimmte grundlegende Informationen über den otel-Tracer-Port weitergeleitet werden können. Zu diesen Informationen gehören insbesondere diese Kategorien:

1. auftretende Fehler im EventHandling
2. der Status bezüglich Readiness und Aliveness des EventHandlings
3. Informationen über die aktuell verbundenen Bot-Instanzen.

#### 4.1.1.1.2 Bot-Config

Die Bot-Config ist für die Verwaltung der wesentlichen Einstellungen des Bots verantwortlich. Eine besonders wichtige Aufgabe der Bot-Config besteht in der Kapselung der Bot-Identitäten, die entsprechend den Vorgaben des Bot-Engine Packages erfolgen muss.

Während der Testdurchführung wird eine Schemadatei verwendet, die alle Attribute der Bot-Config beinhaltet. Dabei wird geprüft, ob sämtliche geforderten Attribute in dieser Schemadatei vorhanden sind. In der Spezifikationsphase eine Vereinheitlichung des Bot-Config Schemas angestrebt. Es werden als Zulassungsvoraussetzung abermals funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet.

#### 4.1.1.2 Bot (Package)

Das Bot Package dient als logisches Modul zur Kapselung aller Bestandteile des Bots. Innerhalb dieses Moduls wird die proprietäre Custom Business Logic des Anbieters

gekapselt, welche sowohl die Identitätsobjekte als auch die spezifische Konfiguration des Bots enthält. Die Instanziierung des Bots erfolgt durch die Verwendung der Bot-Engine.

Im Zuge der Testdurchführung muss sichergestellt werden, dass der Bot eindeutig identifizierbar ist. Über die Testtreiber-Schnittstelle müssen zusätzliche Informationen über den Bot in Form des Displaynames und des m.room.topic verfügbar sein. Es werden als Zulassungsvoraussetzung funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet.

### 4.1.1.2.1 Custom Business Logic

Die Custom Business Logic kapselt die fachliche Funktionalität, die die eigentlichen Aufgaben des Bots definiert. Für die Anbindung eines Drittsystems muss mindestens einer von zwei Ports implementiert werden: Entweder der Event-Creator oder der Drittsystem-Consumer-Client, die jeweils über ein spezifisches Kommunikationsmuster die Verbindung zum Drittsystem herstellen.

Der Event-Creator-Port ist dabei für den Empfang von Push-Events oder Information Items ausgelegt, beispielsweise nach dem Publish/Subscribe-Verfahren. Die empfangenen Informationen werden durch die Custom Business Logic in Matrix-konforme Events (in der Regel Message Events) umgewandelt und anschließend an das EventHandling der Bot-Engine übergeben.

Der Drittsystem-Consumer-Client-Port hingegen stellt eine oder mehrere Schnittstellen für die Kommunikation mit den Drittsystemen bereit. Die gesamte Custom Business Logic Komponente ist als proprietäre Komponente definiert und bleibt ein Implementierungsdetail des jeweiligen Herstellers bzw. Zulassungsnehmers. Aus der Perspektive der Zulassung ergibt sich eine wichtige Vereinfachung: Die Kommunikation, die verwendeten Protokolle sowie die interne Geschäftslogik müssen im Rahmen des Zulassungsprozesses nicht berücksichtigt werden. Es sind keine Tests vorgesehen, die von der gematik durchgeführt werden oder gegenüber dieser nachgewiesen werden müssen.

### 4.1.1.2.2 Bot-Config (Instanz der Bot-Config)

Hierbei handelt es sich um die konkrete Instanz der in Abschnitt [4.1.1.1.2- Bot-Config](#) festgelegten Bot-Config mit den konkreten Konfigurationsoptionen des individuellen Bots.

### 4.1.1.2.3 Bot-Identität-T

Die Bot-Identität-T ("Bot-Identität-Trusted") stellt eine eindeutige Identität dar, die für Bots einer Organisation verpflichtend ist, um eigenständig im Namen der Organisation (und im Rahmen der Custom Business Logic des Bots) innerhalb der Matrix-Föderation agieren zu können.

Diese Berechtigung ist an einen eindeutigen Verwaltungs- und Inkraftsetzungsakt der Organisation/Institution gebunden:

1. Ein dedizierter User Account muss am Homeserver für jeden Bot verfügbar sein (inkl. eines registrierten Users, der Möglichkeit der Speicherung von User Profile Informationen und kryptographischen Schlüsselmaterial des Users, Möglichkeit der Device Registration usw.). Für die Bereitstellung dieses Accounts kann der Standard-Registrierungs-/Provisionierungsprozess der Organisation nachgenutzt werden.
2. Bestimmte User Account Attribute des User Profiles müssen durch eine föderationsweit prüfbare Signatur dieser Attribute anhand der SM(C)-B erfolgen.
3. Der Bot muss über ein abgesichertes, hohes Schutzniveau bietendes Zugriffsverfahren (wie etwa in den Abschnitten [4.2.2- Authentication-Flow für den Port](#)

Auth1 und 4.2.4- Authentication-Flow für den Port Auth2 beschrieben) auf den Messenger-Service zugreifen können.

Die Signatur des User Profiles ist nötig, weil die Bot-Identität-T ein höchstmögliches Vertrauensniveau aufweisen muss. Erst durch diese Signatur bzw. deren Prüfung würden Manipulationen am User Profile (durch wen auch immer) überhaupt auffallen können. Durch die Erstellung der Signatur bestätigt der Automation-Admin, dass der Bot (mit seiner entsprechenden GUID und der Rollenzuweisung Rolle=Bot) auch tatsächlich so intendiert ist. Der skizzierte Verwaltungs- und Inkraftsetzungsakt verhindert, dass ein Bot mit einem geringeren Schutzniveau in Verkehr gebracht werden kann.

Aus der Test- und Zulassungsperspektive muss zunächst die Uniqueness der Identität nachweislich sichergestellt sein.

Des Weiteren muss im Rahmen der Prüfung die Vollständigkeit und Korrektheit der Vertrauenskette verifizierbar sein unter den in diesem Abschnitt beschriebenen Bedingungen. Außerdem wird getestet, ob die zusätzlichen Attribute in der Identität vorhanden sind, wobei diese rein formal auf ihr Vorhandensein geprüft werden, ohne eine semantische Überprüfung ihrer Funktionalität vorzunehmen. Im Übrigen werden als Zulassungsvoraussetzung auch hier funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet.

### Signierte User Account Attribute

Folgende Attribute sind von der o.g. Signatur umfasst und als Signaturmenge User Profile (SMUP) zusammengefasst:

- MXID (Kernattribut, öffentliche Sichtbarkeit)
- Rolle (Attribut des extended-user-profile, öffentliche Sichtbarkeit)
- Bot-GUID (Attribut des extended-user-profile, öffentliche Sichtbarkeit)
- Default-Powerlevel (Attribut des extended-user-profile, öffentliche Sichtbarkeit)

Die Signatur selbst ist ebenfalls ein öffentlich sichtbares Attribut des User Profiles.

#### 4.1.1.2.3.1 Signatur und Signaturoperationen der User Account Attribute

Für relevante Felder im User Profile wird ein Hash Wert berechnet. Der Automation-Admin oder ein autorisiertes System der Organisation verwendet die SM(C)-B, um die Hash-Werte der ausgewählten Felder zu signieren. Dies geschieht durch die Verwendung eines privaten Schlüssels, der sicher auf der SMC-B-Karte gespeichert ist. Die resultierende digitale Signatur wird inkl. dem Signaturzertifikat anschließend dem User Profile hinzugefügt. Die Signaturen der einzelnen Felder werden direkt im User Profile gespeichert. Um die Signaturen zu überprüfen, wird der Hash des jeweiligen Feldes erneut berechnet und mit der gespeicherten Signatur mittels des beiliegenden Zertifikats verifiziert.

Dementsprechend ergeben sich folgende Routinen:

#### Erstellung

$\text{Signatur}_1 = \text{encrypt}_{\text{PrK.HCI.OSIG.E256}}(\text{Hash}_{\text{CADES-SHA-256}}(\text{SMUP}))$

gemäß TUC\_KON\_160 [vgl. gemSpec\_Kon] oder PL\_TUC\_SIGN\_DOCUMENT\_nonQES (vgl. [gemSpec\_Basis\_Consumer] und [gemSpec\_Systemprozesse\_dezTI]), wobei davon ausgegangen wird, dass in der Organisation Identitäten der G2.1 verfügbar sind.

Signatur<sub>1</sub> wird durch den Automation-Admin oder ein durch ihn autorisiertes System am betreuten Messenger-Service erstellt.

#### Validierung

$\text{Signaturprüfung}_1 = \text{decrypt}_{\text{PuK.HCI.OSIG.E256}}(\text{Signatur aus User Profile})$

$\text{Signaturprüfung}_2 = \text{Hash}_{\text{CADES-SHA-256}}(\text{SMUP-Kopie})$

wobei SMUP-Kopie die vom Signaturprüfer ausgelesene SMUP ist (z.B. durch Abfrage der User Profile Informationen am eigenen Fachdienst oder gespiegelte User Profile Informationen an einem anderen Fachdienst der Föderation).

Signaturprüfung<sub>3</sub> = compare(Signaturprüfung<sub>1</sub>, Signaturprüfung<sub>2</sub>)

Diese Signaturprüfschritte erfolgen gemäß TUC\_PKI\_018 (vgl. [gemSpec\_PKI]), wobei PuK.HCI.OSIG.E256 aus C.HCI.OSIG.E256 stammt. Das Zertifikat C.HCI.OSIG.E256 wird gemäß TUC\_PKI\_002 validiert.

### 4.1.1.2.3.2 Abbildung im TI-M Automation Ökosystem

Die in Abschnitt 4.1.1.2.3.1- Signatur und Signaturoperationen der User Account Attribute beschriebene Funktionalität wird über eigene Classifier (z.B. Komponenten) in der TI-M Automation Umgebung abgebildet.

Folgende Classifier sind dabei relevant:

#### **Signature-Creator**

Dieser wird durch den Automation-Admin genutzt, um für die Signaturmenge User Profile (SMUP), inklusive der Zusatzattribute aus dem extended-user-profile, die Signatur mittels einer zugreifbaren SM(C)-B zu erstellen. Die erstellte Signatur wird ebenfalls anhand des SM(C)-B Signatur-Flows Bestandteil des extended-user-profile. Umgesetzt wird der Signature-Creator als Erweiterung des Frontends des Registrierungs-Dienstes, weil dieser bereits Zugriff auf die SM(C)-B besitzt und somit erleichtert TUC\_KON\_160 bzw. PL\_TUC\_SIGN\_DOCUMENT\_nonQES umsetzen kann. Damit der SM(C)-B Signatur-Flow umgesetzt werden kann, wird die Schnittstelle I\_Admin erweitert.

#### **Signature-Validator**

Den Signatur-Validator existiert in einer clientseitigen (Signature-Validator-C) und einer serverseitigen (Signature-Validator-MS) Ausprägung, die jeweils TUC\_PKI\_018 implementieren. Darüber hinaus gibt es aber Unterschiede in den beiden Ausprägungen:

- **Signature-Validator-C:** Dieses ist clientseitige Funktionalität im TI-M HeadlessClient, die ausgelöst durch den Bot an einem anderen User durchgeführt werden kann. Anhand des Validierungsergebnisses kann der Bot Entscheidungen treffen. Eine bereits vorgesehene Funktion ist es, über die Operation add\_BlockedUser-Liste andere User Accounts auf die eigene Block-Liste des Bots setzen zu können. Diese Funktionalität kann durch andere TI-M-Clients nachgenutzt werden, um eine Peer-to-Peer User Validation durchzuführen. Signature-Validator-C ist Bestandteil der Komponente Signature-Extension-C.
- **Signature-Validator-MS:** Dieses ist serverseitige Funktionalität am Messenger-Service, die an diesem (sowohl auslösbar durch den Automation-Admin als auch als zeitgesteuerter Server-Job) die Signaturprüfung am User Profile (inkl. extended-user-profile) durchführt. Bestandene Signaturprüfungen werden (für andere Föderationsteilnehmer) am User Profile kenntlich gemacht. Bei einer nicht bestandenen Signaturprüfung wird der User Account des Bots am Homeserver über den Flow Account-Suspendierung suspendiert. Signature-Validator-MS ist Bestandteil der Komponente Signature-Extension-MS des Messenger Services (vgl. 4.1.1.4.1- Signature-Extension-MS).

### 4.1.1.2.3.3 Erlaubte Organisationen / Institutionen

Folgende Organisationen bzw. Institutionen sind berechtigt, eine Bot-Identität-T bereitzustellen und zu nutzen, um anhand derer funktionstüchtige Bot-Instanzen in Betrieb zu nehmen:

- Organisationen, deren eigene Institutions-OID mit einer der in "Tab\_PKI\_403-x OID-Festlegung Institutionen im X.509-Zertifikat der SMC-B" der [gemSpec\_OID] gelisteten ProfessionOIDs übereinstimmt.

### 4.1.1.2.4 Bot-Identität-HC

Die Bot-Identität-HC beinhaltet ein spezifisches Identifikationsmerkmal, mit dem sich ein Bot gegenüber dem TI-M HeadlessClient als legitimer Benutzer authentifizieren kann und das vom TI-M HeadlessClient eigenständig validiert werden kann.

Diese Identifikation ermöglicht eine flexible Handhabung verschiedener Identifikationsmittel, die in drei Formen auftreten können:

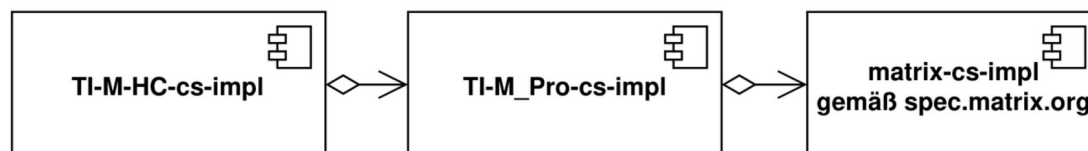
1. Anmeldedaten in Form von Benutzernamen und Passwort bzw. eines API-Keys o.ä. können verwendet werden
2. Die Nutzung eines Access-Tokens ist möglich, das von einem IDP der Organisation ausgestellt wurde und dem der TI-M HeadlessClient durch eigene Validierung vertraut
3. Es kann ein dem TI-M HeadlessClient bekanntes Zertifikat zum Einsatz kommen

Aus der Test- und Zulassungsperspektive muss das von der Organisation gewählte Verfahren durch die gematik geprüft werden.

Dafür wird ein funktionaler Test erfolgen, ob sich mit derbereitgestellten Bot-Identität-HC eine erfolgreiche Anmeldung am TI-M HeadlessClient durchgeführt werden kann. Darüber hinaus werden als Zulassungsvoraussetzung funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet.

### 4.1.1.3 TI-M HeadlessClient

Der TI-M HeadlessClient stellt das vollständige Matrix Client-Server API mit allen TI-Messenger Modifikationen zur Verfügung, die auch ein regulärer TI-M Client Pro besitzt. Dieser bzgl. des Client-Server-API volle TI-M Spezifikations-Funktionsumfang wird in der TI-M-HC-cs-impl Komponente gekapselt und seinen eigenen Clients (den Bot-Instanzen) als API-Bridge über den Port cs-api (full) bereitgestellt.



**Abbildung 3: Zusammensetzung der Kern-Komponente des TI-M HeadlessClient**

Die konkrete technische Umsetzung, also über welches Protokoll oder welche Schnittstellenarchitektur der cs-api (full) Port exponiert wird, liegt in der Verantwortung des Herstellers bzw. Zulassungsnehmers. Wichtig ist dabei nur, dass der Port vollständig kompatibel zu den Aufrufen der eigenen Bot-Engine desselben Herstellers bzw. Zulassungsnehmers sein muss.

Auch die internen Mechanismen, wie beispielsweise die Transformationslogik oder die interne Abstraktion des GUI-Bindings, über die der Port cs-api (full) und dessen Funktionalitäten vom TI-M HeadlessClient bereitgestellt werden, bleiben als Implementierungsdetails beim Hersteller bzw. Zulassungsnehmer.

Die Nutzung des Ports cs-api (full) ist nur nach erfolgreichem Abschluss des Authentication Flows für Credentials/access\_token/cert-based-identity am Auth2-Port möglich. Der TI-M HeadlessClient führt eigenständig die Autorisierung für den Zugriff auf den Port cs-api (full) durch, wobei er dafür den Port Auth2 nutzt. Darüber hinaus verfügt der TI-M HeadlessClient über eine eigene Client-Config, die Identifizierungsinformationen aufnimmt, gegen die sich Bots am Auth2-Port authentisieren können.

Sowohl aus der Architektur- als auch der Zulassungsperspektive ergibt sich der Vorteil der Wiederverwendbarkeit dieser Komponente: Der Funktionsumfang des TI-M HeadlessClient weist eine große Schnittmenge mit dem des TI-M Clients Pro auf. Dies ermöglicht es Organisation, die eine Zulassung des TI-M HeadlessClients anstreben,

ihre bereits vorhandenen TI-M Client Implementierungen entsprechend umzubauen beziehungsweise für den neuen Verwendungszweck zu ertüchtigen.

Der TI-M HeadlessClient wird vollständig über die Testsuite abgetestet und im Rahmen einer Client-Zulassung zugelassen.

### 4.1.1.3.1 Client-Config

Die Client-Config ist für die Verwaltung der wesentlichen Einstellungen des TI-M HeadlessClient verantwortlich. Sie regelt insbesondere die Verwaltung der erlaubten Clients (also der Bots) sowie die Messenger-Services, mit denen sich der Client nach erfolgreichem Durchlaufen des Auth-Request-Flows am Auth1-Port verbindet, wobei diese Verbindung für jeden Messenger-Service einzeln erfolgt. Darüber hinaus steuert die Client-Config die Anbindung und Konfiguration des Auth-Mappers.

Im Zuge der Testdurchführung werden die verwendete Login-Credentials geprüft (vgl. Bot-Identität-HC).

### 4.1.1.3.2 Signature-Extension-C

Anhand dieser Komponente kann eine in den TI-M HeadlessClient integrierte Zertifikatsprüfung von am User Profile hinterlegten Signaturen (und deren Signaturzertifikaten) durchgeführt werden. Diese Komponente kapselt folgende Funktionalitäten:

- 4.1.1.3.2.1- TrustStore-C,
- die in Abschnitt 4.1.1.2.3.2- Abbildung im TI-M Automation Ökosystem beschriebene Validierungsfunktionalität.

#### 4.1.1.3.2.1 TrustStore-C

Der TI-M HeadlessClient implementiert einen eigenen TrustStore, der TI-Zertifikate der TSL enthält. Dazu gehören gematik-Root-Zertifikate, Cross-Signing-Zertifikate, CA-Zertifikate der SM(C)-B Aussteller. Außerdem speichert der TrustStore zeitweise OCSP-Responses für Zertifikatsvalidierungen. Diese Zertifikate und OCSP-Responses werden in vorgegebenen Intervall von der Signature-Extension-MS am Messenger-Service aktualisiert.

Die enthaltenen Zertifikate und OCSP-Responses werden an der Signature-Extension-MS des Messenger-Services über die Operationen `update_certs` und `update_ocsp_responses` in einem festlegbaren Intervall aktualisiert. Eigenständige OCSP-Requests finden durch den TI-M HeadlessClient bzw. TrustStore-C nicht statt. TrustStore-C ist Bestandteil der Komponente Signature-Extension-C.

Sollte ein zertifikatsbasiertes Authentifizierungsverfahren entsprechend Abschnitt 4.2.4.1- alternativer Authentication-Flow für den Port Auth2 zum Einsatz kommen, so wird das entsprechende Bot-Zertifikat ebenfalls im TrustStore-C gespeichert.

Im Rahmen der Testdurchführung durch die gematik müssen die Funktionsfähigkeit der Update-Operationen inkl. der richtigen empfangenen Zertifikatsmenge überprüft werden. Außerdem gehört dazu, den Scheduler, der für die regelmäßige Aktualisierung der TrustStore-C Inhalte zuständig ist, einem Test unterzogen werden. Es werden als Zulassungsvoraussetzung funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet.

### 4.1.1.3.3 Auth-Mapper

Der Auth-Mapper implementiert eine Key-Value-artige Datenstruktur, welche die zugehörigen Autorisierungen an den Schnittstellen Auth1 und Auth2 zusammenführt. Nur wenn beide Autorisierungen - sowohl am Auth1 als auch am Auth2-Port - vorliegen, erhält



ein Bot die Berechtigung, über den HeadlessClient auf einen Messenger-Service zuzugreifen. Der HeadlessClient nutzt diese Struktur, um eingehende Requests am Auth2-Port zur entsprechenden Verbindung an Auth1 aufzulösen. Für eine logische Verbindung eines Bots zu einem Homeserver werden entweder die Schlüssel selbst oder Referenzen auf diese in einem Tupel hinterlegt.

Diese Struktur ist dann erforderlich, wenn der HeadlessClient als zentrale Komponente eingesetzt wird, bei der mehrere Bots gemeinsam einen HeadlessClient nutzen. Im Fall einer statischen Konfiguration, bei der jeder Bot mit seinem eigenen HeadlessClient ausgeliefert und genutzt wird, erfolgt eine statische Belegung der Struktur.

Zusätzlich ermöglicht diese Struktur dem TI-M HeadlessClient ein Context-Mapping zwischen den logischen Verbindungen, wodurch beispielsweise die Zuordnung von Räumen zu den entsprechenden Bots verwaltet werden kann.

Die Implementierung und Verwendung dieser Komponente reduziert die Gesamt-Workload auf dem dem betroffenen Node, da sie von mehreren Bots gleichzeitig genutzt werden kann. Somit steigt der betriebliche Footprint, weil mehrere Instanzen eines TI-M HeadlessClient mit Hilfe dieser Komponente vermieden werden können. Die Implementierung dieser Komponente erfordert einen zusätzlichen Entwicklungsaufwand. Wenn der TI-M HeadlessClient nicht als zentrale Instanz betrieben wird, sondern stattdessen mit jedem Bot als integrierte Lösung ausgeliefert wird, ist die Komponente nicht notwendig, sofern ein mehrinstanzliches Deployment des TI-M HeadlessClient vorgenommen wird. Dieses führt zu einer höheren Arbeitslast im Vergleich zu einer zentralen Instanz, wobei sich dieser Effekt mit steigender Anzahl ausgerollter Bots zunehmend verstärkt.

Als Zulassungsvoraussetzung werden funktionale Tests im Zuge eigenverantwortlichen Testens (EVT) erwartet.

### **Optionalität der Komponente und Modifikation der Zulassung**

Der Auth-Mapper ist nur dann nützlich, wenn der Zulassungsnehmer plant, einen TI-M HeadlessClient zentral zu betreiben, vgl. Alternative B1 aus Abschnitt 4.3- Zulassungs-Deployment- und Betriebssicht. Für die ebenda beschriebene Alternative B2 hingegen ist sie nicht notwendig.

Sollte sich der Zulassungsnehmer entscheiden, seine Bots ausschließlich mit der Alternative B2 betreiben zu wollen, so kann er dieses im Zulassungsantragsverfahren angeben. In diesem Fall findet keine Prüfung des Auth-Mappers statt und die Komponente muss nicht implementiert werden.

#### **4.1.1.4 TI-M Fachdienst Pro**

Der TI-M Fachdienst Pro wird um folgende Elemente bzw. Fähigkeiten erweitert:

- Erweiterung des Messenger-Proxy entsprechend Abschnitt 4.2.3- zusätzliche Sicherheitsleistung des Messenger-Service,
- erweiterte User Profile (extended-user-profile), um dadurch Objekte vom Typ Bot-Identität-T entsprechend Abschnitt 4.1.1.2.3- Bot-Identität-T abbilden zu können,
- eine Signature Extension gemäß Abschnitt 4.1.1.4.1- Signature-Extension-MS, die durch den Matrix-Homeserver benutzt wird.

##### *4.1.1.4.1 Signature-Extension-MS*

Anhand dieser Signatur Extension können Signaturprüfoperationen entsprechend Abschnitt 4.1.1.2.3.1- Signatur und Signaturoperationen der User Account Attribute durchgeführt werden. Die Extension kapselt den in Abschnitt 4.1.1.2.3.2- Abbildung im TI-M Automation Ökosystem beschriebenen Signature-Validator-MS.

Darüber hinaus bietet diese Extension die Ports TSL-Importer, OCSP-Checker und Cert-Exporter, die folgende Eigenschaften und Fähigkeiten aufweisen:

- Über den Port TSL-Importer kann die TSL der TI mit dem Flow Read TSL gemäß TUC\_PKI\_017 und TUC\_PKI\_016 empfangen und gemäß TUC\_PKI\_019 echtheitsgeprüft und authentifiziert werden (vgl. [gemSpec\_PKI]). Außerdem wird durch den Port die Menge der Zertifikate der TSL auf diejenigen gefiltert und reduziert, die für die SM(C)-B basierte Zertifikatsprüfung notwendig sind.
- Über den Port OCSP-Checker OCSP-Requests gemäß TUC\_PKI\_006 gem. [gemSpec\_PKI] durchgeführt werden.
- Über den Flow Certificate-Set und OCSP-Responses am Port Cert-Exporter werden dem TrustStore-C die gespeicherten Zertifikate des TrustStore-MS zur Verfügung gestellt.

#### 4.1.1.4.1.1 TrustStore-MS

Die Signature Extension implementiert einen eigenen TrustStore, der TI-Zertifikate der TSL enthält und über den Port TSL-Importer in diesen importiert werden. Zu diesen TI-Zertifikaten gehören gematik-Root-Zertifikate, Cross-Signing-Zertifikate und CA-Zertifikate der SM(C)-B Aussteller. Durch die Benutzung des OCSP-Checker-Ports führt der TrustStore-MS OCSP-Requests für diese Zertifikate durch, deren Prüfergebnisse er zwischenspeichert. Zertifikate und OCSP-Responses werden über den Port Cert-Exporter und mittels des Flows Certificate-Set und OCSP-Responses an den in Abschnitt 4.1.1.3.2.1- TrustStore-C beschriebenen TrustStore-C propagiert.

### 4.1.1.5 Ergänzung zum Testverfahren

Die Präsentation der Gesamtlösung erfolgt in Form eines Vorführungsworkshops, der im Sinne einer System-Demo die Funktionalität anhand eines konkreten Beispielbots demonstriert.

## 4.2 Laufzeitsicht

Hier werden wichtige Informationsflüsse und Interaktionen von Akteuren und Komponenten detaillierter betrachtet. Dabei werden hier Informationsflüsse betrachtet, die mit Drittsystemen stattfinden und nicht notwendigerweise im o.g. Architekturüberblick enthalten sind.

### 4.2.1 Rollenerweiterung

Mit diesem Konzept wird die Rolle des Automation-Admins eingeführt. Die Rolle Automation-Admin stellt eine Erweiterung der bestehenden Rolle des Org-Admin dar und erbt somit deren Eigenschaften und Fähigkeiten.

Zu den Verantwortlichkeiten des Automation-Admin gehört die Erzeugung der Bot-Identität-T. Der Automation-Admin ist auch dafür zuständig, den Auth1-Flow gegenüber dem Messenger-Service zu starten und zu legitimieren.

Darüber hinaus trägt der Automation-Admin die Build- und Delivery-Verantwortung, indem er über den Zuschnitt der drei Packages entscheidet - basierend auf den Möglichkeiten, die die Architektur gemäß der Deployment-Sicht bietet - sowie deren Paketierung und Deployment durchführt.



## 4.2.2 Authentication-Flow für den Port Auth1

Der Authentication-Flow für den Port Auth1 implementiert einen Device Flow am Authentifizierungs-Dienst der Organisation (in der unten beschriebenen Sequenz handelt es sich beispielhaft um den Matrix Authentication Service (MAS) als OIDC-Broker und spezialisierter Authentifizierungsdienst) unter Nutzung eines Upstream-IDP und ist Teil des Auth request-Flows zwischen dem Port Auth1 und einem Messenger-Service eines Fachdienstes. Zwecks Übersichtlichkeit der Darstellung wurde der Messenger-Proxy in der Darstellung weggelassen, er bleibt jedoch ein aktiver Bestandteil dieser Kommunikation.

Der TI-M HeadlessClient nutzt dabei einen indirekten Weg über ein Device (beispielsweise ein Smartphone) eines Users, der die Rolle Automation-Admin innehat. Dieser Authentication-Flow basiert auf dem OAuth2 Device Authorization Grant in Richtung des OIDC-Brokers, wodurch die Benutzerinteraktion zur Autorisierungsbestätigung auf ein Drittgerät (Device des Admins/Frontchannel zum MAS) ausgelagert wird. Dies eliminiert die Notwendigkeit eines Webviews oder der Handhabung von Konsoleneingaben auf dem Node, auf dem der TI-M HeadlessClient läuft. Gleichzeitig ist durch diesen Flow die Bestätigung der Autorisierung des TI-M HeadlessClient für den User in der Rolle Automation-Admin auf einem bequemen Weg möglich, ohne dass dem TI-M HeadlessClient eine vollautomatisierte Anmeldung zugebilligt werden muss. In Richtung des Upstream-IDPs verwendet dieser Flow exemplarisch einen OAuth2 Authorization Code Flow für die Kommunikation zwischen dem OIDC-Broker und dem Upstream-IDP, weil der Authorization Code Flow momentan der einzige Upstream-Flow ist, der vom Matrix Authentication Service unterstützt wird.

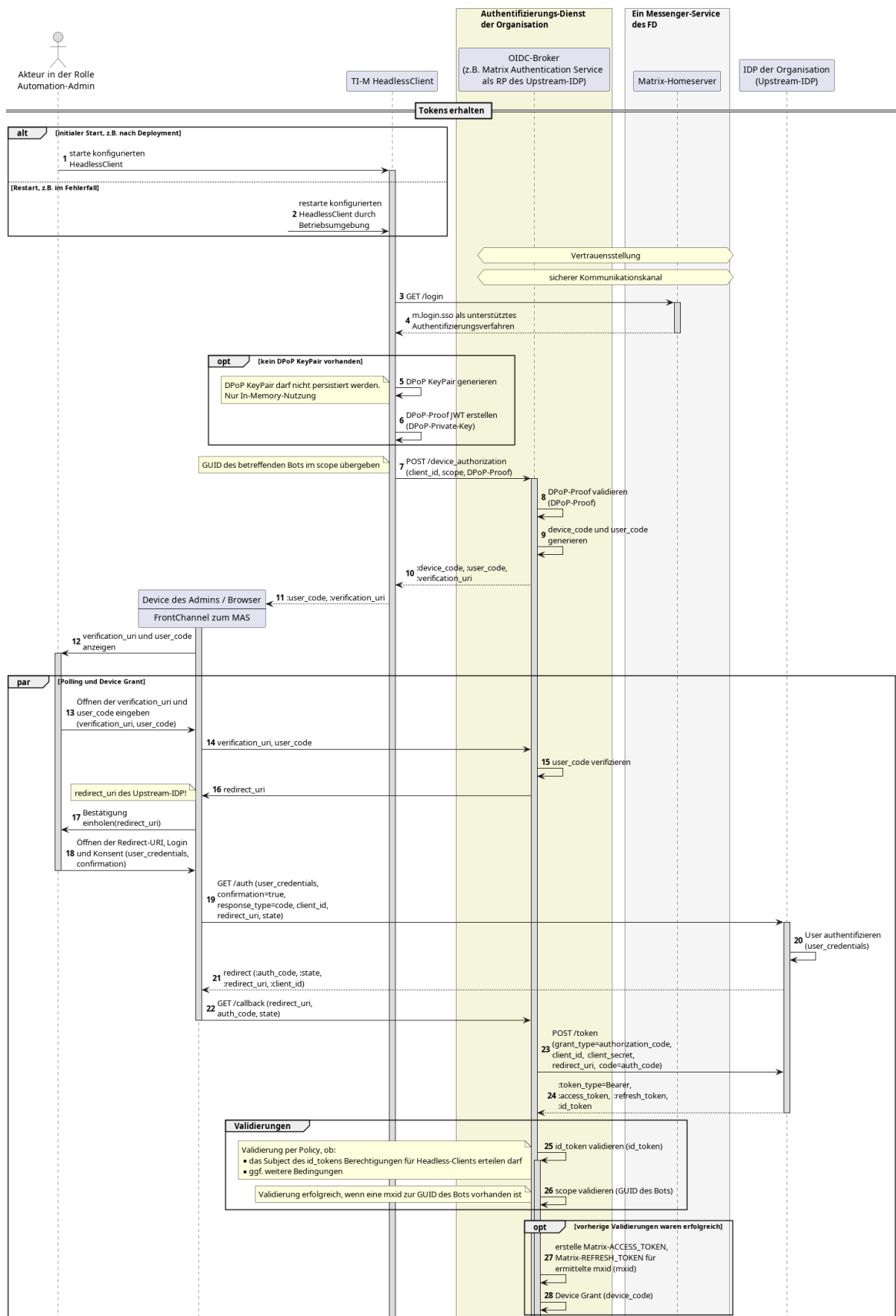
Die grundlegende Struktur des Authentication-Flows basiert auf dem AF\_10057 der TI-M-Spezifikation, wobei zwei wichtige Einschränkungen für den TI-M HeadlessClient gelten:

1. Die vom Matrix-Homeserver zurückgegebenen Tokens, insbesondere der Matrix-REFRESH\_TOKEN, dürfen nicht persistiert werden.
2. Das DPoP KeyPair darf nicht persistiert werden.

Dieser Flow verwendet einen DPoP-Flow gemäß RFC 9449 als zusätzliche Sicherheitsmaßnahme zur Reduzierung von Client-Spoofing und Replay-Attacken. Der DPoP-Flow wurde jedoch leicht modifiziert. Die Modifikation besteht darin, dass der TI-M HeadlessClient seine Echtheit selbst durch Entschlüsselung der Tokens nachweist (nur der echte TI-M HeadlessClient kennt den DPoP-Private-Key), wodurch eine separate Prüfung am Homeserver entfällt und dieser nicht angepasst werden muss.

Grundsätzlich gilt weiterhin, dass die Authentifizierungsverfahren eines TI-M Clients gegenüber dem Messenger-Service durch die Organisation bzw. den Anbieter selbst festgelegt werden. Der hier angebotene Authentication-Flow ist als Vorschlag zum Beleg der Machbarkeit zu verstehen, der dabei - neben der o.g. Eliminierung des Webviews, der Unterstützung der Automation-Admin Rolle und der Absicherung mittels DPoP - folgende weiteren Aspekte berücksichtigt:

- Der Messenger-Service bleibt weitestgehend unmodifiziert, er muss als neue Fähigkeit lediglich ein ID-Token auswerten und ein Role-Mapping für einen Bot-User-Account durchführen können.
- Die Ablauflogik für die Authentifizierung des TI-M HeadlessClient wird durch einen OIDC-Broker abgedeckt, für den mit dem Matrix Authentication Service bereits eine Implementierung am Markt vorhanden ist, die an bestehende IDP-Infrastrukturen mittels Standard-OIDC-Flows angebunden werden kann.
- Weitere Policies für vom Upstream-IDP ausgestellte Token können flexibel in den OIDC-Broker eingebracht werden.



**Abbildung 4: Auth request-Flow zwischen Port Auth1 des TI-M HeadlessClient und dem Messenger-Service**

### **4.2.3 zusätzliche Sicherheitsleistung des Messenger-Service**

Wie in Abschnitt 4.2.2- Authentication-Flow für den Port Auth1 bereits eingeführt, sind zusätzliche Sicherheitsmaßnahmen zur Reduzierung von Client-Spoofing und Replay-Attacken erforderlich. Ergänzend zu der ebenda beschriebenen Sicherheitserweiterung zwischen TI-M HeadlessClient und dem OIDC-Broker wird im Folgenden eine funktionale Erweiterung des Messenger-Proxy beschrieben, anhand derer auffällige Matrix-Tokens invalidiert werden.

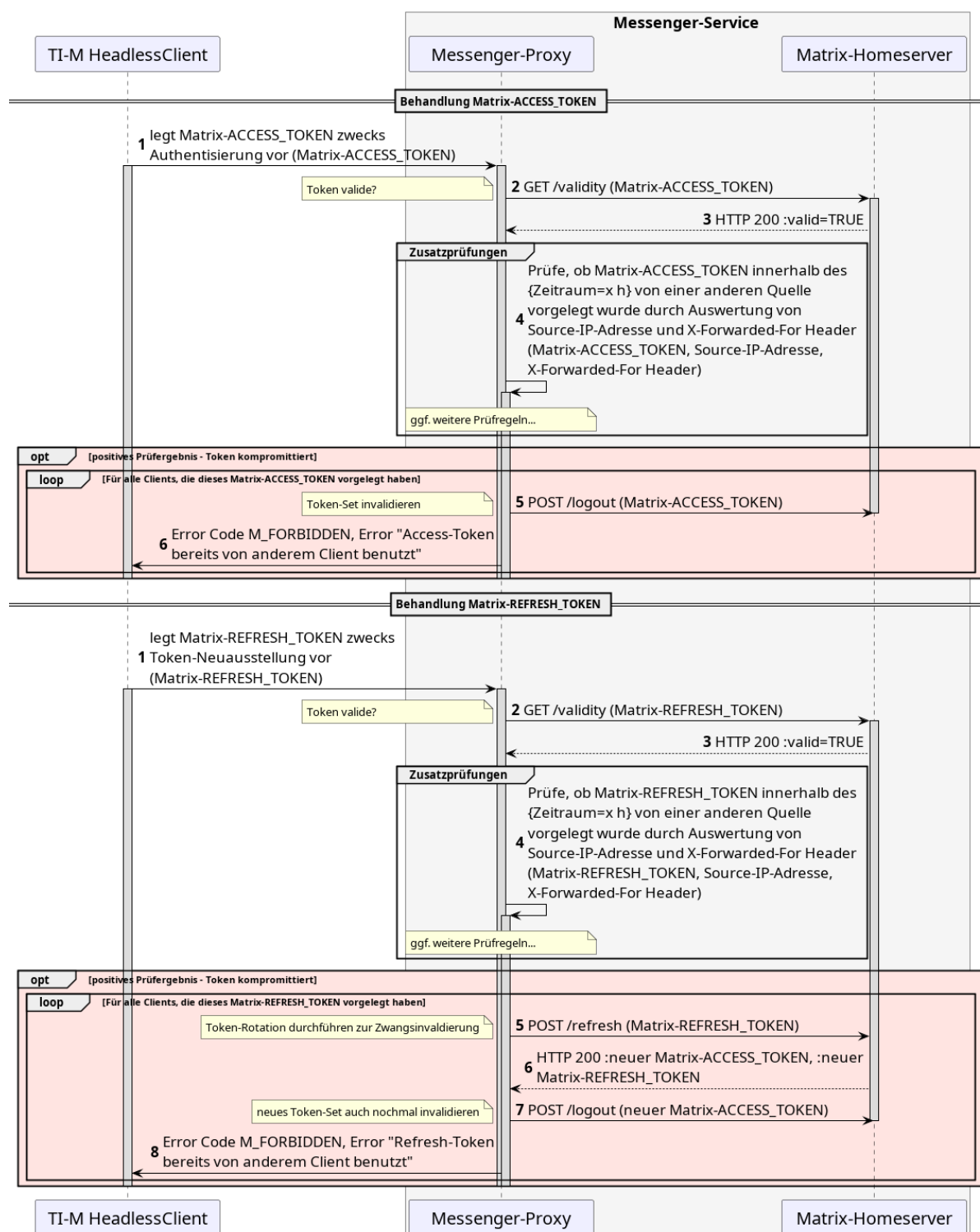


Abbildung 5: zusätzliche Sicherheitsprüfungen am Messenger-Service

Infolgedessen müsste ein TI-M HeadlessClient einen erneuten Login über den Port Auth1 durchführen, nachdem er über eine entsprechende Error Message über Scheitern der Tokenbenutzung informiert wurde.

Es ist zu beachten, dass noch weitere Prüfregele für die Gültigkeitsprüfung der Matrix-Tokens hinzukommen können, die sich dann in dieser Sequenz wiederfinden.

#### 4.2.4 Authentication-Flow für den Port Auth2

Der unten beschriebene Authentication-Flow gehört zum Credentials/access\_token/cert-based-identity-Flow zwischen dem Port Auth2 am TI-M HeadlessClient und der Bot-Engine (Component) aus der Architekturüberblicksskizze und stellt eine der möglichen (leichtgewichtigen) Implementierungen für diese Authentisierung dar. Die Verwendung des Authentication Flows für den Port Auth2 stellt eine Realisierung des Prinzips Know-your-client/Know-your-customer dar: Der TI-M HeadlessClient kann durch einen Bot nicht anonym genutzt werden. In der beschriebenen Implementierung wird ein JWT als Authentisierungsmittel/Access-Token (Bot-Identität-HC) gegenüber dem TI-M HeadlessClient verwendet. Neben den möglichen Implementierungen dieses Flows sind außerdem mehrere alternative Authentisierungsmittel möglich, die für Auth2 grundsätzlich verwendet werden dürfen (vgl. auch 4.2.4.1- alternativer Authentication-Flow).

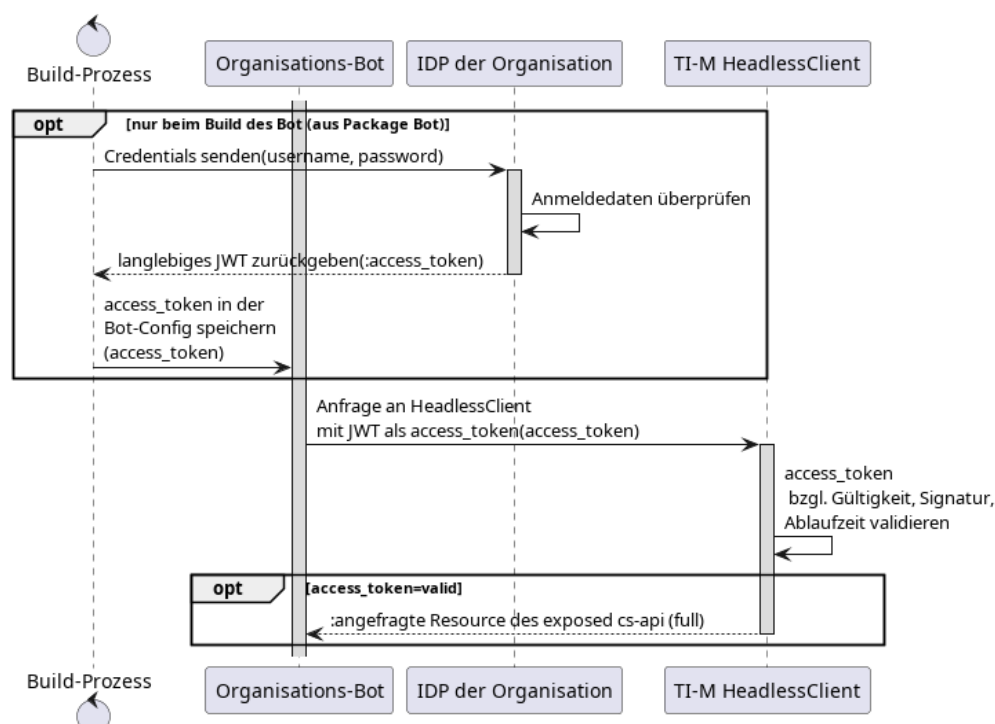
Für alle möglichen Zugriffsverfahren (Credentials/access\_token/cert-based-identity) gilt die Vorgabe, dass die Kommunikationsverbindung des Bots über eine TLS-Schnittstelle erfolgen muss, die von der Bot-Engine verwaltet wird, sobald der Zugriff auf den Auth2-Port durch den TI-M HeadlessClient gewährt wurde. Unverschlüsselte Verbindungen sind dabei nicht zulässig. Das X.509-Server-Zertifikat des TI-M HeadlessClient muss in den Verbindungsdaten der Bot-Config hinterlegt sein und wird bei jedem Verbindungsaufbau routinemäßig validiert.

Folgende alternativen Authentisierungsmittel (und damit einhergehende Zugriffsverfahren) für den Auth2-Port sind grundsätzlich möglich:

- Access-Token, was vom IDP der Organisation ausgestellt wurde und dem der TI-M HeadlessClient durch eigene Validierung vertrauen kann (siehe unten beschriebener Auth2-Flow)
- X.509-Zertifikat
- Sonstige Credentials, die pro Bot einmalig sind (z.B. Username/Password Kombination, ein unique API-Key o.ä.)

Das entsprechende Authentisierungsmittel zur Bot-Identität-HC muss in der Client-Config des TI-M-HeadlessClient hinterlegt oder durch den TI-M HeadlessClient mit eigenen Mitteln validierbar sein.

Eine erfolgreiche Authentifizierung mittels des Authentication Flows am Port Auth2 ermöglicht den Zugriff auf den Port cs-api (full) des TI-M HeadlessClients. Das ist im folgendem Umsetzungsbeispiel beschrieben, in welchem das Authentisierungsmittel statisch am Bot hinterlegt wird:



**Abbildung 6: Authentication-Flow am Port Auth2 des TI-M HeadlessClient**

#### 4.2.4.1 alternativer Authentication-Flow

Wie in Abschnitt 4.2.4- Authentication-Flow für den Port Auth2 bereits erwähnt, können für diesen Flow unterschiedliche Authentisierungsmittel eingesetzt werden. Für das cert-based-identity (zertifikatsbasierte) Zugriffsverfahren wird ergänzend mTLS eingesetzt. Anstelle eines JWT als Bot-Identität-HC kann ein zusätzliches X.509-Client-Zertifikat des Bots verwendet werden, das in der Client-Config des TI-M HeadlessClient (dort in einem eigenen TrustStore, vgl. 4.1.1.3.2.1- TrustStore-C) hinterlegt wird. Dieses Client-Zertifikat muss von einer CA der Organisation (ohne TI-Vertrauensanker, stattdessen mit Vertrauensanker im organisationseigenen Identity Management) ausgestellt werden und übernimmt die Funktion der Bot-Identität-HC.

Der Credentials/access\_token/cert-based-identity-Flow ist dann dementsprechend auszugestalten: Der TI-M HeadlessClient muss in diesem Fall eine zusätzliche CertificateVerify-Anfrage von der Bot-Engine empfangen und den Private Key des Bots zum Client-Zertifikat verifizieren. Dieser private Schlüssel kann entweder in einem sicheren lokalen Schlüsselspeicher der Bot-Config vorgehalten oder aus einem sicheren Remote-Schlüsselspeicher (wie einem HSM) nachgeladen werden.

Bei diesem Ansatz kann der Session Key aus dem erfolgreichen Handshake zwischen TI-M HeadlessClient und Bot direkt für die Kommunikation mit dem Port cs-api (full) über den TLS-Kanal weiterverwendet werden.

#### 4.2.4.2 Verwendung des Auth-Mappers

Die vorstehend genannten Authentisierungsmittel können im Auth-Mapper des TI-M HeadlessClient für die Verbindung eines Bots hinterlegt werden, sofern diese Komponente vom Zulassungsnehmer verwendet wird (vgl. Abschnitt 4.1.1.3.3- Auth-Mapper). Dieser Ansatz bietet einen pragmatischen Weg zur Herstellung einer legitimen Komponentenbindung zwischen Bot und TI-M HeadlessClient, ohne dass diese als Bundle ausgeliefert werden müssen. Das ermöglicht den Betrieb eines zentralen TI-M HeadlessClient, auf den mehrere Anbieter-Bots zugreifen können. Hierfür ist allerdings ein

zusätzliches Role-Mapping zwischen der Bot-Identität am TI-M HeadlessClient und der Identität des TI-M HeadlessClient an einem Messenger-Service erforderlich.

#### 4.2.5 Zugriff des Bots auf den Messenger Service

Die folgenden Entscheidungstabellen beschreiben, unter welchen Bedingungen es einem Bot möglich ist, auf den Homeserver des Messenger-Services zuzugreifen.

**Tabelle 1: Entscheidungstabelle für Zugriff auf den Messenger-Service durch einen Bot (Auth-Mapper vorhanden)**

Autorisierung an Auth1-Port vorhanden	Autorisierung an Auth2-Port vorhanden	Identität von Auth1 und Auth2 als Tupel in Auth-Mapper hinterlegt	Zugriff des Bots auf Homeserver möglich
ja	ja	ja	ja
mindestens 1x nein			nein

**Tabelle 2: Entscheidungstabelle für Zugriff auf den Messenger-Service durch einen Bot (Auth-Mapper nicht vorhanden)**

Autorisierung an Auth1-Port vorhanden	Autorisierung an Auth2-Port vorhanden	Identität von Auth1 und Auth2 als Tupel in Auth-Mapper hinterlegt	Zugriff des Bots auf Homeserver möglich
ja	ja	ignoriert (nicht anwendbar)	ja
mindestens 1x nein		ignoriert (nicht anwendbar)	nein

### 4.3 Zulassungs-, Deployment- und Betriebssicht

Das folgende Diagramm beschreibt die logische Verteilung der wichtigsten Elemente und deren Beziehungen untereinander innerhalb der Anbieterdomäne im Kontext dieses Feature-Dokuments und bildet damit die Grundlage für mögliche Betriebsmodelle.

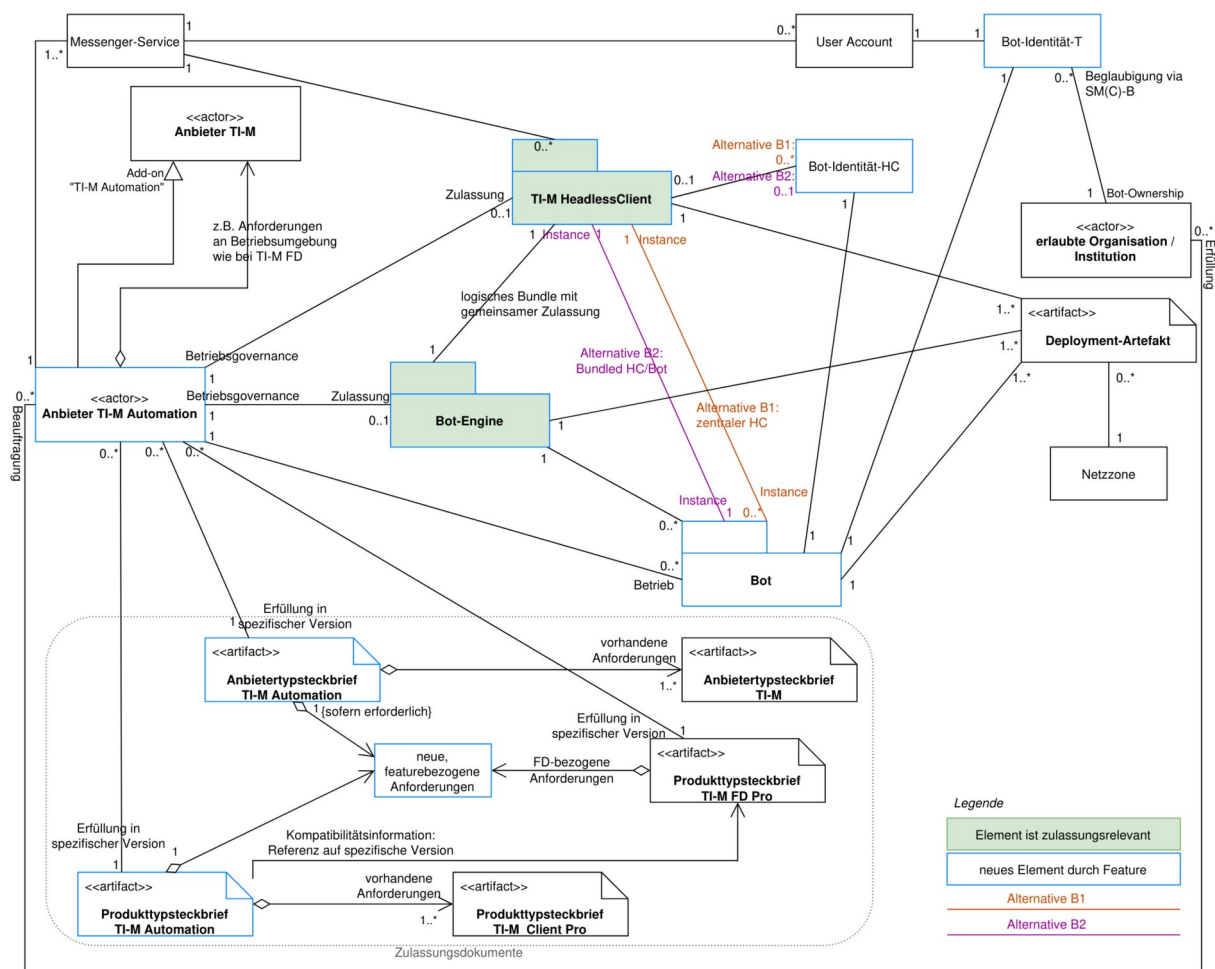
Besonders hervorzuheben sind die Beziehungen zwischen den beiden Packages TI-M HeadlessClient und Bot, die folgende Beziehungen beschreiben:

1. Alternative B1 (zentraler HeadlessClient): Bei dieser Verteilung gibt es einen zentralen TI-M HeadlessClient, den alle Organisations-Bots benutzen.
2. Alternative B2 (Bundled HeadlessClient/Bot): Bei dieser Verteilung sind ein TI-M HeadlessClient und ein Organisations-Bot nur als eineindeutiges Bundle vorhanden, bei dem jeder Bot seinen eigenen TI-M HeadlessClient Instanz verwendet.

Wie diese Beziehungen treffen auch alle anderen Beziehungen Aussagen über die logische Verteilung des Features im Kontext eines Anbieters dieses Features.

Beziehungsbeispiele aus der unten beschriebenen Anbieterdomäne:

1. Die Beziehung "logisches Bundle mit gemeinsamer Zulassung" zwischen den Packages Bot-Engine und TI-M HeadlessClient und versehen mit Kardinalitäten an den Beziehungsenden besagt, dass
  - diese beiden Packages immer gemeinsam als Bundle zugelassen werden müssen,
  - zu einem TI-M HeadlessClient Package genau ein Bot-Engine Package existiert,
  - zu einem Bot-Engine Package genau ein TI-M HeadlessClient Package existiert.
2. Ein Anbieter TI-M Automation ist eine Spezialisierung eines TI-M Anbieters. Alle Anforderungen an einen TI-M Anbieter werden auch im Anbieter TI-M Bots aggregiert.
3. Ein Anbieter TI-M Automation erfüllt genau einen Anbietertypsteckbrief für TI-M Automation in einer spezifischen Version.
  - Ein Anbietertypsteckbrief TI-M Automation kann beliebig viele (oder auch keinen) Anbieter TI-M Automation aufweisen.
  - Ein Anbietertypsteckbrief TI-M Automation aggregiert eine Menge vorhandener Anforderungen des Anbietertypsteckbriefs TI-M, jedoch nicht notwendigerweise alle.



**Abbildung 7: logische Verteilung wichtiger Assets aus der Anbieterdomäne**

Der neue Produkttypsteckbrief TI-M Automation beginnt in seiner ersten Fassung mit einer eigenständigen Versionierung ab 1.0.0. Im diesem Steckbrief werden kompatible Produktversionen des TI-M Fachdienstes Pro gelistet.



Im Folgenden werden nun verschiedene Deployment-Szenarien für die Verteilung der Packages (physische Verteilung von Softwarekomponenten) betrachtet, die typische valide Deployments beschreiben, alternativ eingesetzt werden können und dabei unterschiedliche Vor- und Nachteile aufweisen.

### 4.3.1 Szenario 1: Standardfall in einer unvirtualisierten Umgebung

Bei diesem Szenario handelt es sich um Standardfall für einen On-Premises Betrieb. Es ermöglicht einen TI-M HeadlessClient als Intermediate für alle Bots des Anbieters bzw. der Organisation. TI-M HeadlessClient und die Bot-Implementierung können hier auf unterschiedlichen Nodes betrieben werden, die sich in unterschiedlichen Netzzonen befinden. Die Anbindung der Nodes erfolgt über eine TLS-Verbindung, über welche alle Port- und Schnittstellenaufrufe erfolgen.

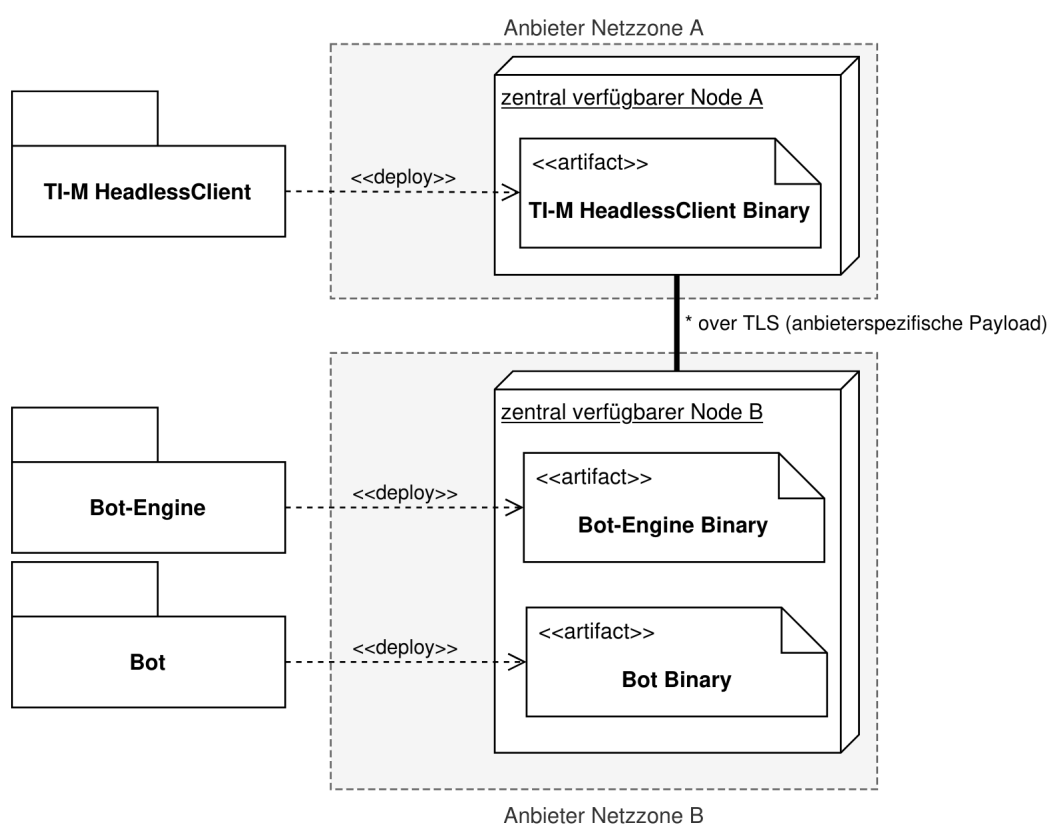


Abbildung 8: Deployment-Diagramm von Szenario 1

### 4.3.2 Szenario 2: Deployment mit Maximalverteilung in einer virtualisierten Umgebung

Das Deployment mit Maximalverteilung in einer virtualisierten Umgebung zeichnet sich durch eine vollständige Verteilung der Packages aus. In dieser Konfiguration wird jedes Package mit einer eigenen Service-Instanz betrieben, wobei jede Service-Instanz auf einem separaten (Remote-)Node läuft.

Diese Architektur ermöglicht den Einsatz eines TI-M HeadlessClient als Vermittlungsinstanz (Intermediate) für sämtliche Bots des Anbieters bzw. der Organisation.

Ein besonderer Vorteil dieses Szenarios ist die Möglichkeit, rollierende Updates eines Bots durchzuführen, ohne dass gleichzeitig Änderungen am TI-M HeadlessClient und der Bot-Engine erforderlich sind.

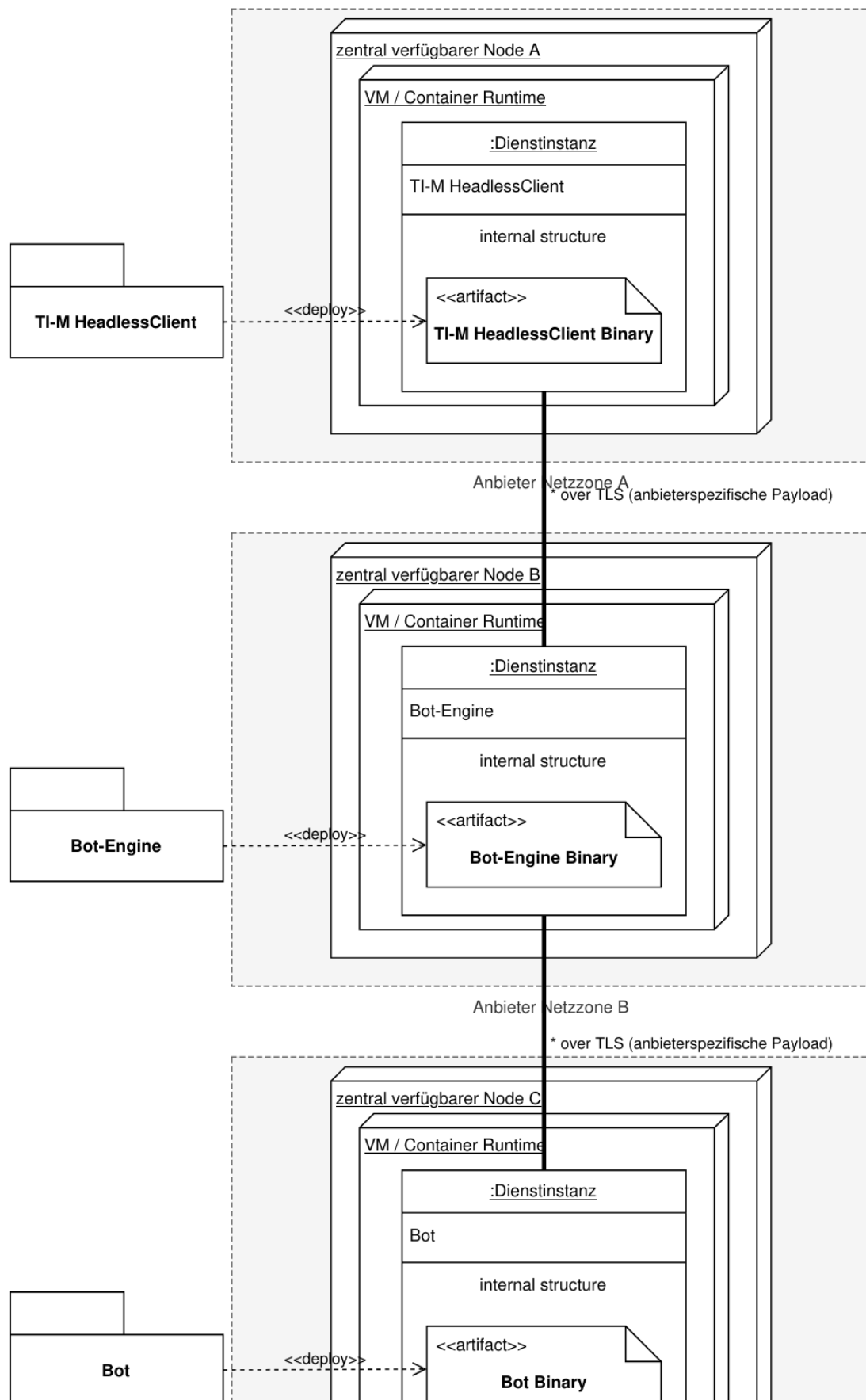


Abbildung 9: Deployment-Diagramm von Szenario 2

### 4.3.3 Szenario 3: Deployment mit Minimalverteilung virtualisiert (Deployment-Monolith)

Das Deployment mit Minimalverteilung in virtualisierter Form wird als Deployment-Monolith umgesetzt und zeichnet sich durch eine minimale Verteilung der Packages aus. In dieser Konfiguration werden alle drei Packages innerhalb einer gemeinsamen Service-Instanz betrieben. Jede Service-Instanz kapselt dabei den TI-M HeadlessClient, die Bot-Engine und den Bot als geschlossene Einheit.

Bei diesem Ansatz erfordert jede Änderung an einem Bot ein erneutes Rollout der gesamten Service-Instanz. Trotz der monolithischen Struktur ist dieser Deployment-Ansatz bereits für einen Einsatz in einer Healthcare Confidential Computing (HCC) Private Cloud-Umgebung vorbereitet.

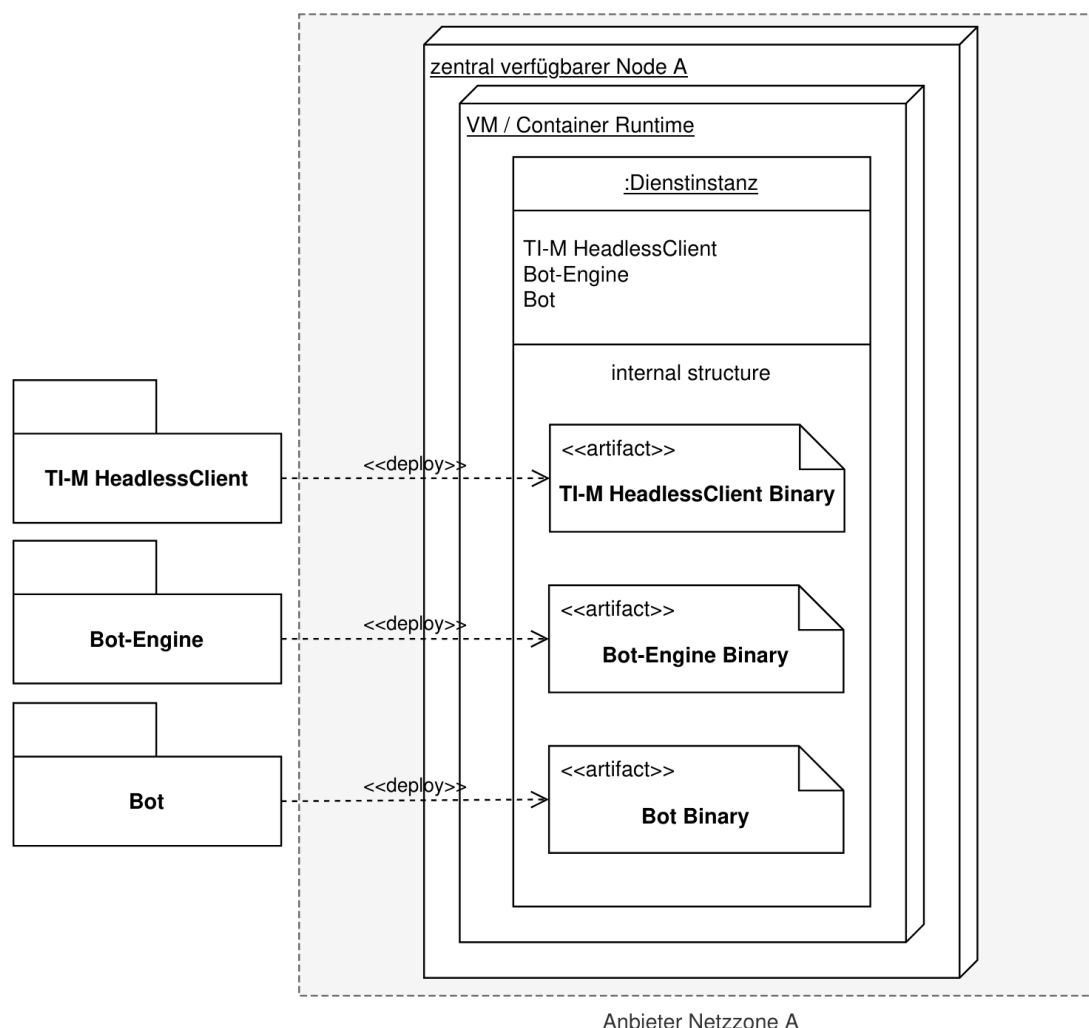


Abbildung 10: Deployment-Diagramm von Szenario 3

## 4.3.4 Hinweise zu den Deployment-Szenarien

### Zulassungstauglichkeit

Die Art des Deployments (Wahl eines der genannten Szenarien oder Kreierung eines neuen) hat keinen Einfluss auf die Zulassung der Packages TI-M HeadlessClient und Bot-Engine.

### Flexibilität

Im Szenario 1 und 3 ist es möglich, die Packages Bot-Engine und Bot im Zuge des Builds zu verschmelzen und als einzelnes Deployment Artifact zu deployen.

### Exkurs: Vorschau auf einen Betrieb in einer HCC-Umgebung

Die beiden zuvor beschriebenen virtualisierten Deployment-Szenarien (Szenario 2 und 3) eignen sich grundsätzlich für den Betrieb als Workloads in Healthcare Confidential Computing (HCC) Cloud-Umgebungen gemäß der gemSpec\_HCC Spezifikation.

Ein wesentlicher Vorteil in dieser Umgebung besteht darin, dass die Dienstinstanzen und ihre Komponenten über einen Attestation Service ihre eigene Service-Identität erhalten, wodurch der separate Identitätsnachweis aus den betroffenen Komponenten heraus nicht mehr erforderlich wäre.

Durch die Nutzung eines Build Service des HCC kann aus einem anbieterspezifischen CI/CD-Prozess, der ein generisches Container-Image für eines der Packages erzeugt, ein attestierter Workload generiert werden. In diesem Kontext wäre es möglich, auf die inhärente Bot-Identität-T zu verzichten, da diese Identitätsfunktion durch den Attestation-Prozess an die attestierte Bot-Dienstinstanz übertragen würde.

## 4.3.5 Betriebliche und organisatorische Auswirkungen

Bedingt durch dieses Feature

- werden keine Auswirkungen auf Anbindung oder Inhalte des VZD-FHIR-Directory erwartet,
- wird eine höhere Last auf durch zusätzliche Anfragen an das VZD-FHIR-Directory erwartet,
- ist die Ausbringung eines separaten Produkttypen (inkl. Produkttypsteckbrief) zu erwarten (TI-M HeadlessClient i.V.m. Bot-Engine),
- sind Änderungen am Anbietertypsteckbrief zu erwarten,
- wird die Spezifikation mindestens eines neuen Anwendungsfalls erwartet, ohne dabei die bestehende TI-M Spezifikation zu modifizieren,
- wird durch Nutzergruppen, die unter die nach 4.1.1.2.3.3- Erlaubte Organisationen / Institutionen erlaubten Organisationen fallen, keine wesentlich höhere Last auf Trust Service Provider der TI (TSP) erwartet (unter Verwendung des hier skizzierten technischen Konzept),
- sind Verfügbarkeitsanforderungen zu definieren, sofern der Anbieter bzw. die Organisation den Betrieb eines zentralen TI-M HeadlessClients plant (Alternative B1). Das ist notwendig, weil beim Ausfall des TI-M HeadlessClients in Alternative B1 alle Bots funktionsuntüchtig wären. Während auf den Messenger-Service durch den Ausfall keine Auswirkung zu erwarten ist, sind Benutzer indirekt betroffen, weil sie bedingt durch den Ausfall keine Dienste von Bots (vgl. Abschnitt 2.1- Epic Automatisierung und Bots) in Anspruch nehmen könnten.



---

## 5 Spezifikation

---

**To be continued** (nach Erhalt und Rezipieren des Feedbacks zum technischen Konzept)...

---

## 6 Anhang A - Verzeichnisse

---

### 6.1 Abkürzungen

Kürzel	Erläuterung

### 6.2 Abbildungsverzeichnis

Abbildung 1: Legende häufig verwendeter Modellierungselemente.....	13
Abbildung 2: Architekturüberblick.....	15
Abbildung 3: Zusammensetzung der Kern-Komponente des TI-M HeadlessClient.....	21
Abbildung 4: Auth request-Flow zwischen Port Auth1 des TI-M HeadlessClient und dem Messenger-Service.....	28
Abbildung 5: zusätzliche Sicherheitsprüfungen am Messenger-Service.....	29
Abbildung 6: Authentication-Flow am Port Auth2 des TI-M HeadlessClient.....	31
Abbildung 7: logische Verteilung wichtiger Assets aus der Anbieterdomäne.....	34
Abbildung 8: Deployment-Diagramm von Szenario 1.....	35
Abbildung 9: Deployment-Diagramm von Szenario 2.....	37
Abbildung 10: Deployment-Diagramm von Szenario 3.....	37

### 6.3 Tabellenverzeichnis

Tabelle 1: Entscheidungstabelle für Zugriff auf den Messenger-Service durch einen Bot (Auth-Mapper vorhanden).....	32
Tabelle 2: Entscheidungstabelle für Zugriff auf den Messenger-Service durch einen Bot (Auth-Mapper nicht vorhanden).....	32



## 6.4 Referenzierte Dokumente

### 6.4.1 Dokumente der gematik

Die nachfolgende Tabelle enthält die Bezeichnung der in dem vorliegenden Dokument referenzierten Dokumente der gematik zur Telematikinfrastruktur.

[Quelle]	Herausgeber: Titel
[gemGlossar]	gematik: Glossar der Telematikinfrastruktur
[gemSpec_Basis_Consumer]	gematik: Spezifikation Basis-Consumer
[gemSpec_Kon]	gematik: Spezifikation Konnektor
[gemSpec_PKI]	gematik: Übergreifende Spezifikation Spezifikation PKI
[gemSpec_Systemprozesse_dezTI]	gematik: Spezifikation Systemprozesse der dezentralen TI

### 6.4.2 Weitere Dokumente

[Quelle]	Herausgeber (Erscheinungsdatum): Titel