
C_12813_Anlage - Einarbeiten von TI Flow im gemKPT_Test

Inhaltsverzeichnis

1 Änderungsbeschreibung.....	2
2 Änderung in gemKPT_Test.....	3
2.1 "Testhub ready" - Begriff, Zielbild und Geltungsbereich.....	3
2.1.1 Entkopplung als Leitprinzip.....	4
2.1.2 Wesentliche Konzepte.....	4
2.2 Anforderungen an „Testhub ready“ Testsuiten.....	5
2.2.1 Technische Basis.....	5
2.2.2 Framework und Observability.....	5
2.2.3 Layering.....	8
2.3 Laufzeitumgebung der Testsuite.....	10
2.4 Compliance.....	11
2.5 Reporting.....	11
2.6 IOP-Testsuite für den TI-Flow Fachdienst.....	11

1 Änderungsbeschreibung

Erweiterung der bestehenden Testanforderungen für Auftragnehmer an Testsuiten.

Ziel ist der Aufbau einer homogenen Testlandschaft, in der die Testsuiten der Auftragnehmer nahtlos in den bestehenden Testhub der gematik integriert werden.

Dadurch sollen Redundanzen bei der Erstellung von Testsuiten vermieden und ein einheitlicher Qualitäts- sowie Strukturstandard gefördert werden, der die Wiederverwendung von Artefakten und die gegenseitige Nutzung bestehender Testlösungen erleichtert.

Das Dokument ergänzt die bisherigen Anforderungen um Kriterien für die vollautomatisierte Ausführbarkeit, die fachliche Verifikation über beobachtbares Systemverhalten, die Wiederverwendbarkeit von Artefakten, die strukturierte Ergebnisdokumentation sowie die Bereitstellung in der Integration und automatisierte Ausführbarkeit in der von der gematik verantworteten CI/CD-Plattform und führt den Begriff "Testhub ready" als eigenständige Qualitäts- und Strukturklasse ein.

2 Änderung in gemKPT_Test

##Der gesamte Änderungstext wird Bestandteil des neuen Kapitels "4.7 Testtools". Die Kapitelstruktur wird entsprechend bei der Einarbeitung in gemKPT_Test angepasst werden.

Die gematik stellt im Rahmen des TI 2.0 Testhub in gitHub (<https://github.com/gematik/ti2.0-testhub>) modulare Komponenten, Simulatoren, Integrationsbausteine und Testsuiten bereit. Der Testhub bildet damit einen zentralen Baustein für die automatisierte Durchführung von Interoperabilitäts-, Integrations- und End-to-End-Tests im gematik-Testökosystem.

Dieses Kapitel erweitert die in diesem Dokument bereits festgelegten Anforderungen an Testsuiten um die Kriterien für die Eigenschaft "Testhub ready" und folgt der dort vorgegebenen Gliederung; bestehende Anforderungen bleiben unverändert und werden durch die nachfolgend beschriebenen normativen Anforderungen ergänzt.

Damit Testsuiten im Kontext des Testhubs und darüber hinaus in der TI 2.0 einheitlich nutzbar, nachvollziehbar auswertbar und langfristig erweiterbar sind, definiert dieses Kapitel Anforderungen an Struktur, Wiederverwendbarkeit und Nachweisführung von Testsuiten. "Testhub ready" bezeichnet dabei eine definierte Konformitätsklasse von Testsuiten. Die Anforderungen adressieren insbesondere die vollautomatisierte Ausführbarkeit, die fachliche Verifikation über beobachtbares Systemverhalten, die Wiederverwendbarkeit von Artefakten, die strukturierte Ergebnisdokumentation sowie die Bereitstellung in der DevOps-Plattform der gematik.

Die Erweiterung stellt sicher, dass Testsuiten von Auftragnehmern so ausgestaltet sind, dass sie in den TI 2.0 Testhub integriert, dort vollautomatisiert ausgeführt, einheitlich ausgewertet und langfristig erweitert werden können. Ziel ist, Testsuiten so auszugestalten, dass sie durch unterschiedliche TI-Teilnehmer entwickelt und durch die gematik oder Dritte betrieben, geprüft und erweitert werden können. Das Tiger-Framework bildet hierfür das bevorzugte strategische Zielbild, ohne alternative Frameworks grundsätzlich auszuschließen.

Eine Testsuite ist „Testhub ready“, wenn sie die in diesem Kapitel festgelegten Anforderungen an Automatisierbarkeit, Wiederverwendbarkeit, fachliche Nachweisführung, Schnittstellen, Lizenzierung und Reporting erfüllt. Die Anforderungen richten sich an Auftragnehmer, die Testsuiten im Rahmen der TI-Entwicklung und der gematik zur Weiternutzung bereitstellen sowie an Dritte, die diese Testsuiten im Testhub einsetzen, prüfen oder erweitern.

2.1 "Testhub ready" - Begriff, Zielbild und Geltungsbereich

"Testhub ready" Testsuiten bilden die einheitliche Basis für ein homogenes Test-Ökosystem der TI 2.0. Sie ermöglichen allen Herstellern – Auftragnehmern, Zulieferern, Primärsystemherstellern und App-Entwicklern – Testsuiten zu weiterzuentwickeln und Artefakte auszutauschen.

2.1.1 Entkopplung als Leitprinzip

Testsuiten bewerten primär den Netzwerkverkehr zwischen Komponenten. Dadurch sind Testsuite und Testumgebung entkoppelt; verschiedene Client- und Serversysteme können ohne Anpassung der Testsuite getestet werden. Dies erschließt Synergien über alle Teststufen hinweg: Eine einmal erstellte Testsuite lässt sich über den gesamten Verlauf – vom Beginn der Entwicklung bis zur abschließenden Qualifizierung – wiederverwenden, sofern die Zielumgebung die definierten Schnittstellen bereitstellt. Die lose Kopplung fördert eine wartbare, zukunftssichere und dem Stand der Technik entsprechende Testarchitektur und macht die Testlandschaft zu einem gemeinsam nutzbaren, stetig wachsenden Fundament für alle Beteiligten.

2.1.2 Wesentliche Konzepte

"Testhub ready" Testsuiten basieren auf vier Konzepten:

- **Festgelegter Beobachtungspunkt:**
Die Bewertung der Testergebnisse erfolgt anhand eines festgelegten Beobachtungspunkts im Datenverkehr oder anhand der zugehörigen Request-/Response-Interaktion.
- **Assertions:**
Fachliche Bewertungen beziehen sich auf konkret identifizierbare, logische Bestandteile von Nachrichten an diesem Beobachtungspunkt und sind über die zugrunde liegende Profilierung – nicht über fest verdrahtete Nachrichtenpfade – adressiert. Die bewertete Nachricht und das zugrunde liegende Datum werden nachvollziehbar dokumentiert. Lage und Funktion des Beobachtungspunkts sind normativ festgelegt; seine Implementierung ist austauschbar und über den Artifact-Layer gekapselt. Die Testsuiten verwenden die spezifizierte Schnittstelle des Beobachtungspunkts und sind nicht an eine bestimmte Implementierung gebunden.
- **Layering:**
Das in Kapitel 2.2.3 beschriebene Schichtenmodell bildet die Grundlage für die klare Schichtenteilung. Dadurch werden wiederverwendbare Teile abgegrenzt und enge Kopplungen reduziert.
- **APIs:**
Die Ansteuerung aller Testkomponenten (z. B. Testtreiber-APIs, Provisionierungs-Schnittstellen) ist transparent und nachnutzbar. Herstellerspezifische Logik zur Zustandssetzung des System under Test (SuT) wird ausschließlich im austauschbaren Artifact-Layer gekapselt; Intent- und Composition-Layer bleiben SuT-neutral. SuT-übergreifende Zustandssetzung setzt eine gematik-seitig spezifizierte Provisionierungs-Schnittstelle voraus.

Primärsystemhersteller können ihre Anwendungen um CI-Testsuiten erweitern.

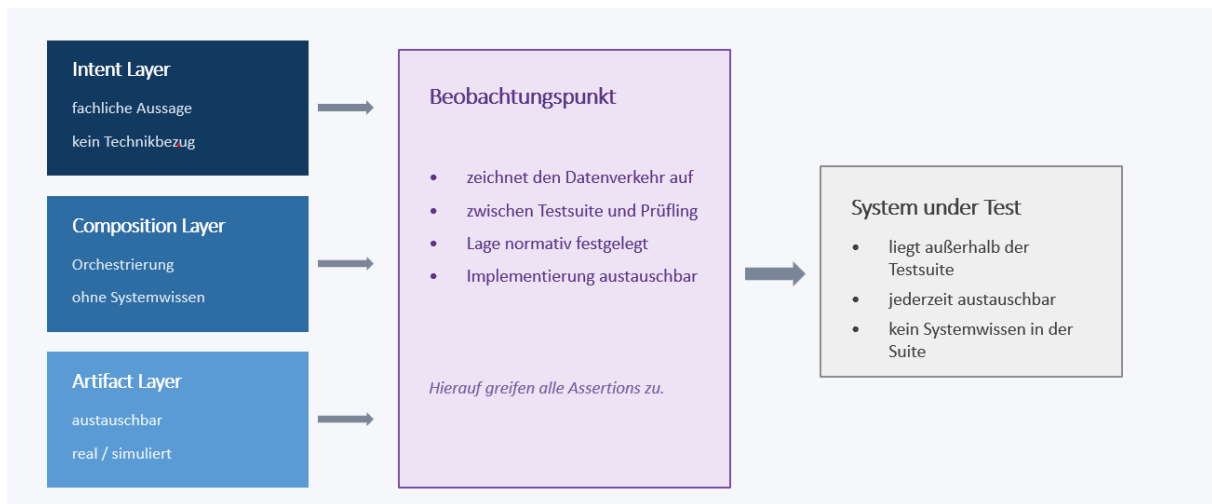


Abbildung 1 : Das Entkopplungsprinzip

2.2 Anforderungen an „Testhub ready“ Testsuiten

2.2.1 Technische Basis

A_30049 -"Testhub ready" Testsuite

Der AN MUSS eine "Testhub ready" Testsuite bereitstellen. Diese MUSS

- voll automatisiert startbar sein und während der Ausführung keine manuelle Interaktion erfordern,
- in der DevOps-Plattform der gematik auf einem Kubernetes-Node betreibbar sein,
- im von der gematik vorgegebenen Git-Repository versioniert bereitgestellt werden,
- unter einer offenen Lizenz stehen, die der gematik eine kostenfreie und kommerzielle Nutzung ermöglicht,
- ausschließlich Abhängigkeiten zu Software enthalten, die durch die gematik kostenfrei und kommerziell genutzt werden darf

[<=]

2.2.2 Framework und Observability

A_30050 -Zieltestarchitektur

Der AN SOLL eine "Testhub ready" Testsuite auf dem Tiger-Testframework der gematik aufbauen.[<=]

A_30150 -Zulässige Entwicklungsausprägung

Für die Entwicklung KANN der AN einen Fork des Tiger-Testframeworks erstellen und um projektspezifische Funktionalität erweitern.[<=]

A_30151 -Abgabefähigkeit

Der AN MUSS bis zur Abgabe der Testsuite Änderungen an einem Fork des Tiger-Testframeworks entweder in den Hauptentwicklungsstand des Tiger-Testframeworks übernehmen oder in einer mit der gematik abgestimmten alternativen technischen Lösung bereitstellen.[<=]

A_30152 -Unzulässige Dauerlösung

Der AN DARF NICHT eine „Testhub ready“ Testsuite von einem nicht durch die gematik genehmigten projektspezifischen Fork abhängig sein lassen. [≤]

Die gematik unterstützt die Entwicklung von Testsuiten im vorgesehenen Test-Ökosystem durch begleitende Angebote. Hierzu gehören insbesondere Workshops zur Einarbeitung in das Tiger-Framework sowie Support-Kanäle über GitHub-Issues und das Anfrageportal der gematik.

Diese Unterstützungsleistungen haben keinen normativen Charakter und sind nicht Bestandteil der Anforderungen an eine "Testhub ready" Testsuite. Sie dienen der Förderung eines gemeinsamen, wiederverwendbaren Test-Ökosystems und der praktischen Unterstützung bei Entwicklung und Integration.

A_30051 -Einsatz alternativer Testframeworks

Der AN MUSS, wenn er für eine "Testhub ready" Testsuite nicht das Tiger-Testframework der gematik verwendet, ein alternatives Testframework einsetzen, das die Anforderungen gemäß [A_30153*] erfüllt. [≤]

A_30153 -Anforderungen an das alternative Framework

Das alternative Testframework MUSS

- unter einer offenen Lizenz bereitgestellt werden, die der gematik eine kostenfreie und kommerzielle Nutzung ermöglicht,
- durch die gematik oder einen von ihr beauftragten Dritten modifiziert und weiterentwickelt werden können,
- ausschließlich Abhängigkeiten zu Software enthalten, für die diese Bedingungen ebenfalls erfüllt sind.

Diese Anforderungen gelten entsprechend für alle zum Betrieb des Testframeworks notwendigen Bestandteile und Abhängigkeiten. [≤]

A_30052 -Fachliche Verifikation aus beobachtbarem Systemverhalten

Der AN MUSS bei "Testhub ready"-Testsuiten die Erfüllung fachlicher Anforderungen ausschließlich auf Grundlage beobachtbaren Systemverhaltens nachweisen. [≤]

Beobachtbares Systemverhalten im Sinne dieser Anforderungen ist der an einem definierten Beobachtungspunkt erfasste Datenverkehr eines von der gematik spezifizierten Kommunikationswegs.

A_30290 -Wahl der Beobachtungspunkt

Der AN MUSS für eine "Testhub ready"-Testsuite einen Beobachtungspunkt wählen, der einen von der gematik spezifizierten Kommunikationsweg betrifft. [≤]

A_30154 -Ausgestaltung des Kommunikationswegs

Der AN KANN einen von der gematik spezifizierten Kommunikationsweg als Request-/Response-basierter Kommunikationsaustausch zwischen Testsystem und Prüfling ausgestalten. [≤]

A_30155 -Zulässige fachliche Assertions

Der AN MUSS ausschließlich fachliche Assertions definieren, die auf konkret identifizierbaren Bestandteilen des am definierten Beobachtungspunkt erfassten Datenverkehrs basieren. Hierzu zählen insbesondere Nachrichteninhalte, Metadaten und Protokollstatus. [≤]

A_30156 -Unzulässige Quellen fachlicher Bewertung

Der AN MUSS eine fachliche Bewertung so gestalten, dass Schnittstellen, Zustände oder Interaktionen außerhalb des definierten Kommunikationswegs unberücksichtigt bleiben. Der AN MUSS Logs, Metriken, Traces, interne Zustände sowie Informationen, die nicht aus

dem am definierten Beobachtungspunkt aufgezeichneten Datenverkehr stammen, bei der fachlichen Bewertung unberücksichtigt lassen. [≤]

A_30157 -Zulässige Hilfsmittel

Der AN KANN Simulatoren, Mocks, Test-APIs sowie sonstige Hilfsmittel zur Teststeuerung, Testdatenerzeugung, Orchestrierung oder technischen Absicherung des Testablaufs einsetzen. Als eigenständige Quelle fachlicher Verifikation sind sie ausgeschlossen. Informationen, die diese Hilfsmittel aus dem am definierten Beobachtungspunkt aufgezeichneten Datenverkehr gewinnen, können für fachliche Assertions herangezogen werden. [≤]

A_30158 -Technische Absicherung ohne fachliche Bewertungswirkung

Der AN MUSS Prüfungen zur Erreichbarkeit von Testsystemen, zum Aufbau der Testumgebung oder zur erfolgreichen Ansteuerung von Simulatoren und Mocks ausschließlich der technischen Absicherung des Testablaufs verwenden. Sie MÜSSEN von der fachlichen Bewertung ausgeschlossen bleiben. [≤]

A_30159 -Nachvollziehbarkeit fachlicher Assertions

Der AN MUSS fachliche Assertions auf einen oder mehrere konkret identifizierbare Bestandteile von Nachrichten im beobachteten Datenverkehr zurückführen können. Alle hierfür verwendeten Daten MÜSSEN nach dem Testlauf strukturiert und maschinenlesbar gemeinsam mit dem Report gespeichert werden. Der AN MUSS die Nachvollziehbarkeit fachlicher Assertions im Testergebnis oder Report durch Referenzierung der jeweils zugrunde liegenden Beobachtungen im aufgezeichneten Datenverkehr sicherstellen. [≤]

A_30053 -Zuordnung der Schnittstellen

Der AN MUSS jede von der Testsuite enutzte Schnittstelle eindeutig der Data Plane oder der Control Plane zuordnen. [≤]

Als Control Plane gelten ausschließlich Schnittstellen zur Herstellung von Testvorbedingungen außerhalb der gematik-spezifizierten oder gematik-erlaubten Anwendungsfälle. Sie dienen ausschließlich der Herstellung, Abfrage oder Veränderung von Zuständen zum Zweck der Testvorbereitung, Testdurchführung oder Testnachbereitung und sind nicht Grundlage fachlicher Nachweise.

Als Data Plane gelten fachliche Aktionen, die über einen gematik-spezifizierten oder gematik-erlaubten Kommunikationsweg ausgeführt werden. Dies gilt auch dann, wenn diese Aktionen den Zustand des SuT verändern.

Ein Kommunikationsweg gilt als gematik-spezifiziert, wenn die gematik das Protokoll oder die Schnittstelle selbst definiert. Er gilt als gematik-erlaubt, wenn eine gematik-Spezifikation die Verwendung eines konkret extern definierten Protokolls vorschreibt oder ausdrücklich zulässt.

A_30160 -Interner Zugriff nur über Control Plane

Der AN MUSS einen direkten Zugriff auf die interne Datenhaltung des SuT über die Control Plane zulassen. Dies gilt insbesondere für Datenbankzugriffe oder vergleichbare interne Zugriffsmechanismen. [≤]

A_30161 -Beobachtungspunkte

Der AN KANN Beobachtungspunkte auf Control-Plane- oder Data-Plane-Schnittstellen legen. [≤]

A_30162 -Fachliche Assertions

Der AN MUSS fachliche Assertions ausschließlich auf Kommunikation über Data-Plane-Schnittstellen beziehen. [≤]

A_30163 -SuT-Neutralität

Der AN MUSS die Testsuite so erstellen, dass diese herstellerspezifische Logik zur Zustandssetzung ausschließlich im austauschbaren Artifact-Layer kapseln. Intent-Layer und Composition-Layer MÜSSEN SuT-neutral bleiben. [≤]

A_30164 -Herstellerspezifische Zustandssetzung über Außenschnittstellen

Der Hersteller des SuT MUSS die von gematik spezifizierte Provisionierungsschnittstelle zur Zustandssetzung verwenden. [≤]

A_30165 -Dokumentation von Control-Plane-Schnittstellen

Der AN MUSS jede Control-Plane-Schnittstelle im Architekturentwurf der Testsuite mit ihrem Scope dokumentieren. Jede Control-Plane-Schnittstelle MUSS:

- über Endpunkte erreichbar sein, die in der DevOps-Plattform ansteuerbar sind,
- vollständig durch ein OCI-Container-Artefakt implementiert sein,
- in der Testkonfiguration nachvollziehbar beschrieben sein,
- im Report nachvollziehbar referenziert sein

[≤]

2.2.3 Layering

Das hier beschriebene Drei-Schichten-Modell (Intent, Composition, Artifact) ist keine Eigenkreation, sondern eine auf unseren Kontext zugeschnittene Variante eines in der Testautomatisierung anerkannten Architekturmusters: Dave Farleys Four-Layer-Model für Akzeptanztests. Seine oberen drei Schichten - Test Cases, Domain-Specific Language, Protocol Driver - können grob den Intent, Composition und Artifact-Layern zugeordnet werden. Die vierte Schicht, das System under Test (SuT), liegt außerhalb der Testsuite selbst und fällt damit raus.

Ziel ist die Etablierung eines Musters, welches zwischen Herstellern und gematik eine gute und einfache Wiederverwendbarkeit erlaubt. Die gematik will durch diese Wahl zum einen sauber abgetrennte Artefakte erzwingen, die durch eine direkte Nachnutzbarkeit durch die gematik und andere Hersteller den überschaubaren Mehraufwand bei der initialen Erstellung aufwiegt. Desweiteren gibt die gematik bewusst Wahlfreiheit auf der obersten Ebene, dem Intent-Layer, um die Vielzahl der möglichen Testsuiten nicht unnötig einzuschränken und den verantwortlichen Testern die nötigen Freiheiten zu geben. Das Ziel sind effektive Testsuiten, deren technische Basis aber dennoch offen und wiederverwendbar ist.

A_30054 -Architektonische Gliederung - Layering

Der AN MUSS "Testhub ready" Testsuiten architektonisch in Intent-, Composition- und Artifact-Layer gliedern. Der AN MUSS die Verantwortlichkeiten dieser Layer eindeutig voneinander abgrenzen. Der AN MUSS die Gliederung der Testsuite in diese Layer durch eine im Repository abgelegte, versionierte und zugängliche Dokumentation beschreiben. Der AN MUSS die Testsuite so gliedern, dass Abhängigkeiten ausschließlich von höheren zu niedrigeren Layern verlaufen. Dabei bildet der Intent Layer den höchsten und der Artifact Layer den niedrigsten Layer. [≤]

A_30055 -Intent Layer- Zweck und Abstraktion

Der AN MUSS in dem Intent Layer die beabsichtigte fachliche oder technische Aussage eines Testfalls in einheitlicher Form beschreiben. Die Beschreibung MUSS innerhalb eines Testfalls auf einer konsistenten Abstraktionsebene erfolgen. [≤]

A_30166 -Intent Layer- Ausschluss technischer Detailtiefe

Der AN MUSS den Intent Layer so bauen, dass dieser frei von testsetup-spezifischen, systemkonkreten und rein implementierungsnahen technischen Details ist. Dies gilt insbesondere für Wissen über konkrete Systeme unter Test, Testclients, Endpunkte oder Proxying. Technische Merkmale, insbesondere Protokoll-, Status- oder Fehlerrückgaben, MÜSSEN im Intent Layer auf solche Fälle beschränkt bleiben, in denen sie Ausdruck des

fachlich erwarteten oder fachlich relevanten Systemverhaltens sind. Weitergehende technische Ausgestaltung MUSS in darunterliegenden Schichten erfolgen. [≤]

A_30167 -Intent Layer - Form der Beschreibung

Der AN MUSS die Testfälle im Intent Layer in einer für alle beteiligten Akteure lesbaren und reviewfähigen Form beschreiben. Der AN MUSS hierfür Gherkin-Feature-Files oder andere gleichwertig strukturierte Beschreibungsformate verwenden. Reiner Quelltext ohne ergänzende standardisierte Testfallbeschreibung ist unzulässig. [≤]

A_30168 -Intent Layer - Konsistenz innerhalb der Testsuite

Der AN MUSS Feature-Files innerhalb einer Testsuite konsistent auf derselben Abstraktionsebene formulieren. Eine Vermischung fachlicher und technischer Detailtiefe innerhalb einer Testsuite ist unzulässig. Der AN MUSS rein technische und rein fachliche Testsuiten getrennt ausführen.

[≤]

A_30056 -Composition Layer

Der AN MUSS den Composition Layer auf die Orchestrierung und Komposition von Artefakten beschränken und dauerhaft darauf beschränkt halten. Die unveränderte Übernahme eines Artefakt-Ergebnisses und dessen Weitergabe an einen nachfolgenden Artefakt-Aufruf gilt als Komposition; dies umfasst insbesondere die Extraktion einzelner Felder zur Parameterübergabe. Fachliche oder technische Logik zur Transformation, Berechnung, Interpretation oder Bewertung von Daten sowie fachliche Assertions MÜSSEN außerhalb des Composition Layers in Artefakten implementiert werden. [≤]

A_30057 -Artifact Layer - Umfang

Der AN MUSS den Artifact Layer so entwerfen, dass dieser alle Artefakte der der Testsuite enthält. Hierzu MÜSSEN mindestens Implementierungen zur Interaktion mit Testsystemen, zur Auswertung von Beobachtungen, für System- und Client-Abstraktionen sowie für Protokoll- und Kommunikationsadapter gehören. [≤]

A_30058 -Artifact Layer - Unabhängigkeit der Artefakte

Der AN MUSS Artefakte frei von Annahmen, Abhängigkeiten und Referenzen auf konkrete Testfälle, Features oder Testsuiten entwerfen. Soweit fachliche, projektspezifische oder szenariobezogene Vorprägungen im Einzelfall erforderlich sind, MUSS der AN diese nachvollziehbar begründen, dokumentieren und gegenüber generischen Artefakten klar abgrenzen. Die Vorprägung DARF die fachliche Bewertung NICHT auf Informationen stützen, die nicht aus dem am definierten Beobachtungspunkt aufgezeichneten Datenverkehr stammen.

Der AN MUSS sicherstellen, dass Artefakte über stabile und klar definierte Schnittstellen verfügen und diese Definitionen maschinenlesbar in dem gemäß [A_30049*] vorgegebenen Repository abgelegt sind. [≤]

A_30059 -Artifact Layer - Ablageort

Der AN MUSS Artefakte in der DevOps-Plattform ablegen. Der AN MUSS diese entweder in Quelltextform innerhalb der Testsuite, in einem eigenen Repository oder in einem mit der gematik abgestimmten Repository zur Ablage von Binaries ablegen. [≤]

A_30060 -Artifact Layer - Eigenständige Akteure

Der AN MUSS Artefakte, die eigenständige Akteure im Testsetup simulieren oder repräsentieren, darunter Hardware-Komponenten sowie Client- oder Serversysteme

- als eigenständige und unabhängig betreibbare OCI-Container bereitstellen,
- über eine dokumentierte und maschinenlesbar beschriebene Steuerschnittstelle ansteuern.

Der AN MUSS die Container entweder im Repository der DevOps-Plattform oder in einem frei im Internet zugänglichen zentralen Repository bereitstellen. [$\leq$]

A_30061 -Artifact Layer - Lose Kopplung

Für die gemäß [A_30060*] bereitgestellten Artefakte gilt: Der AN MUSS jedes Artefakt, das einen eigenständigen Akteur simuliert oder repräsentiert, lose an die ansteuernde Gegenstelle koppeln. Die Umschaltung zwischen einer realen und einer simulierten Ausprägung MUSS ausschließlich über die Testkonfiguration erfolgen. Sie MUSS ohne Anpassung der Testsuite und ohne Anpassung der ansteuernden Gegenstelle möglich sein. [$\leq$]

A_30062 -Artifact Layer - Komplexe und domänenspezifische Prüfungen

Der AN MUSS komplexe oder domänenspezifische Prüfungen (z. B. FHIR-Profilvalidierung, XSD-Validierung, kryptographische Prüfungen) als wiederverwendbare Artefakte im Artifact Layer implementieren. Die Testsuite MUSS die zugrunde liegenden Daten ausschließlich über den Beobachtungspunkt beziehen. Eine Vorverarbeitung oder Transformation der Daten im Composition- oder Intent-Layer ist ausgeschlossen. [$\leq$]

A_30063 -Artifact Layer - Zuordnung zu Data Plane und Control Plane

Der AN MUSS jedes Artefakt eindeutig entweder der Data Plane oder der Control Plane zuordnen. Die Zuordnung MUSS in der Artefakt-Dokumentation explizit ausgewiesen sein. [$\leq$]

A_30299 -Unzulässige Doppelzuordnung

Der AN DARF NICHT ein Artefakt sowohl mit Data-Plane- als auch mit Control-Plane-Schnittstellen ansprechen lassen. [$\leq$]

2.3 Laufzeitumgebung der Testsuite

Die Laufzeitumgebung einer „Testhub ready“ Testsuite besteht aus einem oder mehreren Testgegenständen (System under Test), der Testsuite, dem definierten Beobachtungspunkt und dem benötigten Tooling. Zusätzliche Hilfsartefakte dürfen ergänzend zum Einsatz kommen.

A_30064 -Laufzeitumgebung - Hilfsartefakte

Der AN MUSS Hilfsartefakte

- als Open-Source in einem Git-Repository der DevOps-Plattform,
- als OCI-Container im Repository der DevOps-Plattform
- oder als OCI-Container in einem frei zugänglichen, zentralen Internet-Repository bereitstellen und unter eine offene, durch die gematik kostenfrei und kommerziell nutzbaren Lizenz stellen. [$\leq$]

A_30065 -Laufzeitumgebung - Schnittstellen

Der AN MUSS die Software so entwerfen, dass alle Schnittstellen, die im Testaufbau genutzt werden,

- vollständig von Spezifikationen der gematik abgedeckt sind
- oder über eine OpenAPI-Beschreibung verfügen, die im Rahmen der Testsuite zur Verfügung steht.

[$\leq$]

A_30066 -Laufzeitumgebung - Automatisierung

Der AN MUSS die Software so entwerfen, dass die gesamte Laufzeitumgebung vollständig automatisiert startbar, ausführbar und abbaubar ist. Manuelle Interaktionen während der Ausführung sind unzulässig. Die Bereitstellung und der Betrieb der Umgebungen, in denen die Testsuite ausgeführt wird, bleiben davon unberührt und richten sich nach den hierfür geltenden Anforderungen. [≤]

2.4 Compliance

A_30069 -Compliance - Architekturentwurf

Der AN SOLL die Entwicklung einer „Testhub ready“ Testsuite in kooperativer Abstimmung mit der gematik durchführen. Der AN SOLL hierzu ein Architekturentwurf erstellen, der den Testscope, das geplante Layering, den oder die Beobachtungspunkte innerhalb der Kommunikationsflüsse und die geplanten Artefakte beschreibt. Der AN SOLL den Entwurf mit der gematik abstimmen und im Entwicklungsverlauf fortschreiben. [≤]

2.5 Reporting

A_30070 -Reporting

Der AN MUSS das Reporting einer „Testhub ready“ Testsuite so implementieren, dass das Reporting ein Testergebnis in maschinenlesbarer und menschenlesbarer Form erzeugen kann. Beide Repräsentationen MÜSSEN Folgendes enthalten:

- alle Testfälle mit Name, Status und ggf. aufgetretenem Fehler,
- alle fachlichen Assertions, auf denen das finale Verdikt beruht,
- den zur Bewertung herangezogenen Datenverkehr,
- die Testkomponenten/Artefakte (inkl. Framework) in ihrer jeweils genutzten Version dokumentieren,
- die Konkret genutzte Konfiguration (Konfigurationsdateien, Entwicklungsvariablen ...).

[≤]

2.6 IOP-Testsuite für den TI-Flow Fachdienst

Dieses Kapitel definiert die Anforderungen an die Entwicklung und Implementierung einer IOP-Testsuite für den TI-Flow-Fachdienst. Sie soll modulare, vollautomatisierte Testszenarien umfassen, die sich in die CI/CD-Pipeline der gematik integrieren lassen. Sofern nicht abweichend spezifiziert, gelten für die IOP-Testsuite auch die allgemeinen Anforderungen an „Testhub ready“ Testsuiten.

A_30071 -Ableitung und Abstimmung der Testszenarien

Der AN MUSS die Testszenarien auf Basis der Anwendungsfälle des TI-Flow Fachdienstes ableiten und in modularer Gherkin-Syntax (Given / When / Then) in deutscher Sprache verfassen. Die Ableitung MUSS mit der gematik abgestimmt erfolgen, um fachliche Missverständnisse früh zu erkennen. [≤]

A_30072 -Integration in die CI/CD-Pipeline

Der AN MUSS die IOP-Testsuite in die CI/CD-Pipeline der gematik für den TI-Flow
Fachdienst integrieren und dort ohne Anpassungen seitens der gematik ausführen
können.【<=】

A_30073 -Konfigurierbarkeit

Der AN MUSS die IOP-Testsuite so entwerfen, dass diese über zentrale
Konfigurationsdateien und Umgebungsvariablen steuerbar ist . Zusätzlich MUSS die
Selbstauskunft des TI-Flow Fachdienstes abrufbar und zur Laufzeitkonfiguration
herangezogen werden können.【<=】

A_30074 -Wechsel zwischen Simulation und Echt-Komponenten

Der AN MUSS den Wechsel zwischen simulierten und realen Komponenten ausschließlich
über die Testkonfiguration ermöglichen ohne den Testcodes anzupassen. Dies gilt für
Konnektor-Simulationen, HSKs und EinBox-Konnektoren.【<=】