API Architektur Bund

Klassifizierung	nicht klassifiziert
Status	genehmigt zur Nutzung
Programmname	Digitalisierungsstrategie des Bundes Strategische Initiative 3 «Once Only Prinzip»
Initiativenleiter	Wüst Jürg, DTI
Version	1.0
Datum	17. Januar 2022
Auftraggeber	Digitale Transformation und IKT-Lenkung (DTI)
Autor/Autoren	Glavitsch Michael, Detecon (Schweiz) AG
Beteiligte Fachgruppen	Arbeitsgruppe API Architektur Bund, Architekturboard Bund (ABB)
Genehmigende Stelle	Andreas Spichiger, DTI

Änderungsverzeichnis

Version	Datum	Änderung	Autor
0.1	02.03.2021	Initiale Version	Glavitsch Michael
0.2	26.03.2021	Erster Entwurf für Struktur der Inhalte	Glavitsch Michael
0.3	05.05.2021	Version für Workshop 18.05.2021	Glavitsch Michael
0.4	07.05.2021	Version für Workshop 18.05.2021, interne Rückmeldungen eingearbeitet	Glavitsch Michael
0.5	23.06.2021	Version für Workshop 29.06.2021	Glavitsch Michael
0.6	25.06.2021	Version für Workshop 06.07.2021	Dyer Matthias Glavitsch Michael
0.7	07.07.2021	Gestaltungsempfehlung IAM, Ausgangslage, Einordnung und Zielsetzung	Glavitsch Michael Wüst Jürg
0.8	21.07.2021	Inhalte ausformuliert	Brüning Fabian Glavitsch Michael
0.9	23.07.2021	Befunde aus projekt-interner Review verarbeitet	Glavitsch Michael
0.95	26.10.2021	Befunde aus Review verarbeitet	Brüning Fabian Glavitsch Michael
0.96	01.12.2021	Qualitätssicherung nach Abnahme durch Architektur Board Bund (ABB)	Brüning Fabian Glavitsch Michael
0.97	02.12.2021	Anpassungen aus Meeting 02.12.2012 BK DTI Detecon	Glavitsch Michael
1.0	17.01.2021	Freigegebene Version	Spichiger Andreas

Inhaltsverzeichnis

1 Management Summary

Die API Architektur Bund verfolgt das Ziel, den digitalen Zugang zu Behördenleistungen im Bundesumfeld im Kontext Maschine-zu-Maschine für Unternehmen, Verwaltung und Personen zu standardisieren und zu fördern. Diese Zielsetzungen entsprechen Forderungen seitens der Politik, Wirtschaft und der Digitalisierungsstrategie des Bundes 2020-2023. Die Zielsetzung ist als Vision formuliert, welche sich in folgendem Leitsatz manifestiert:

«Als moderne Verwaltung vereinfachen wir für unsere Partner den Zugang zu unseren Behördenleistungen, indem wir die Leistungen über beliebige elektronische Mittel nutzbar machen.»[1]

Die Vision der API Architektur Bund ist in verschiedene Dimensionen gegliedert, welche in Tabelle 1 dargestellt sind. Sie diente der für die Erstellung der API Architektur Bund einberufenen Arbeitsgruppe bei der Erarbeitung der API Architektur Bund als Orientierung. Die API Architektur Bund wurde unter der Leitung des Bereichs Digitale Transformation und IKT-Lenkung (DTI) zusammen mit Vertretern aus zahlreichen Departementen und Verwaltungseinheiten (VE) erstellt und ist damit im Bundesumfeld breit abgestützt.

Dimension	Vision
Nutzen	Der digitale Zugang zu (offenen) Behördenleistungen im Kontext Maschine-zu-Maschine für Unternehmen, Verwaltung und Personen soll gefördert werden. Auf digitale Behördenleistungen zugreifende Maschinen können Lokale Apps, Fachservices oder Portalanwendungen sein. APIs sollen verwaltet, deren Existenz bekannt gemacht und damit wenn immer möglich wiederverwendet werden.
Zielgruppe	Die Zielgruppe besteht aus Unternehmensarchitekten, IT Architekten, Fachpersonal, Management sowie Orientierungssuchende aus Business und IT, welche Vorhaben zur digitalen Nutzung von Behördenleistungen unterstützen, sowohl auf Seite der Bundesverwaltung als auch auf Seite der Geschäftspartner ausserhalb der Bundesverwaltung.
Umfang	Die API Architektur Bund ist ein Ordnungsrahmen, welcher im Kontext der Bundesverwaltung Empfehlungen für die Entwicklung und den Betrieb von APIs zur digitalen Abwicklung von Verwaltungsgeschäften macht. An dieser Stelle hervorzuheben sind die Gestaltungsempfehlungen zum API Design und zum Identity & Access Management sowie das Führen eines API Verzeichnisses. Auf technische Einschränkungen bei der Gestaltung von Lösungsarchitekturen wird weitgehend verzichtet.
Architekturprinzipien	Architekturprinzipien bieten Steuerungsrichtlinien für die Entwicklung und Betrieb von Behördenleistungen-unterstützenden APIs.
Standards & Good Practices	Die API Architektur basiert auf anerkannten Standards wie z.B. eCH, Tallinn, EIF, W3C sowie Good Practices aus dem Bundesumfeld und wird periodisch weiterentwickelt.

Tabelle 1: Dimensionen der Vision API Architektur Bund

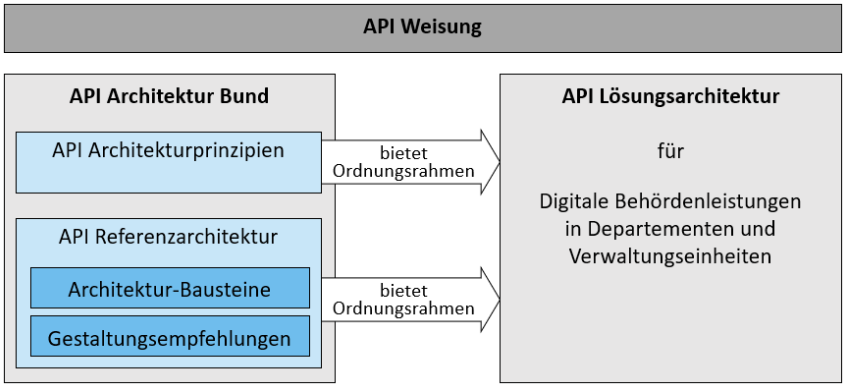

Abbildung 1: Positionierung der API Architektur Bund

[1] Vision API Architektur Bund

Die API Architektur Bund besteht aus einem Satz von Architekturprinzipien und einer an den Architekturebenen (BDAT) nach TOGAF[2] ausgerichteten Referenzarchitektur, wobei nur die Geschäfts-, Daten- und Applikationsarchitekturebenen beschrieben sind, denn die API Architektur Bund verzichtet auf Empfehlungen auf der Technologiearchitekturebene. Gemeinsam bieten Prinzipien und Referenzarchitektur einen Ordnungsrahmen und eine gemeinsame Sprache für den Entwurf von API Lösungsarchitekturen für den digitalen Zugang zu Behördenleistungen in den Departementen und VE (s. Abbildung 1).

Die API Architektur Bund hat Empfehlungscharakter und soll damit die Erarbeitung von API Lösungsarchitekturen unterstützen. Getrennt von der API Architektur Bund können durch den Erlass von Weisungen Aussagen zur Verbindlichkeit von Elementen der API Architektur Bund gemacht werden.

Für den digitalen Zugang zu Behördenleistungen im Kontext Maschine-zu-Maschine verwendet die API Architektur Bund den Begriff der digitalen Behördenleistung. Eine digitale Behördenleistung kann auf einem einzigen API basieren, oder aber auf einem Verbund aus APIs aufbauen. Wenn in der API Architektur Bund der Begriff API verwendet wird, sind damit fachlich gruppierte Verbunde von APIs eingeschlossen.

Die API Architektur Bund kann anhand folgender Merkmale zusammengefasst werden:

- Der Bund ermöglicht den Geschäftspartnern auf Basis der Vorgaben des EMBaG[3] den Bezug von digitalen Behördenleistungen. Zu den digitalen Behördenleistungen werden von den zuständigen Departementen und VE auf Basis der EMBaG-Vorgaben Metadaten zu APIs in einem API Verzeichnis veröffentlicht.
- Der fähigkeitsbasierten Planung nach TOGAF folgend, definiert die API Architektur Bund für die Entwicklung und den Betrieb von digitalen Behördenleistungen Geschäftsfähigkeiten.
- Die Geschäftsfähigkeiten stehen in Beziehung zur übergeordneten Fähigkeit API Management und sind den fünf Prozessstufen Manage API Lifecycle, Discover APIs, Register Client, Operate APIs und Analyse API Operation & Usage zugeordnet. Im Fall von API-spezifischen Geschäftsfähigkeiten stellen sie eine Teil-Fähigkeit von API Management dar. Im Fall von bereits existierenden, API-unspezifischen Geschäftsfähigkeiten werden diese von der Fähigkeit API Management verwendet.
- Die API Architektur Bund baut auf einer serviceorientierten Architektur auf, in der API Clients über ein API Gateway APIs von Fachservices aufrufen. Bieten in digitale Behördenleistungen einzubindende Fachservices noch keine APIs an, so müssen deren Leistungen über eine API Schicht verfügbar gemacht werden.
- Um getreu dem «Once-Only-Prinzip» (s. Kapitel 2.2.1) die Durchgängigkeit von digitalen Behördenleistungen des Bundes sicherzustellen, können Fachservices ihrerseits ebenfalls weitere Fachservices via APIs integrieren. Die API Architektur Bund spricht dabei von «API Integration im Fachservice des Bundes». Ebenso kann auch «API Integration im API Client des Geschäftspartners» stattfinden. Dies ist dann der Fall, wenn die digitale Behördenleistung des Bundes für die digitale Leistung des Geschäftspartners nur eine Teil-Leistung darstellt.
- Für die Entwicklung und den Bezug von digitalen Behördenleistungen ist geplant, dass einzelne Fähigkeiten der Prozessstufen Discover API und Register Client bundesweit zentral oder föderiert umgesetzt werden, z.B. ein zentrales API Verzeichnis oder die Nutzung von zentralen IAM Standard-Diensten.
- Die API Architektur Bund macht keine gesonderten Aussagen oder Vorgaben zu Informationssicherheit und Datenschutz (ISDS). Die Rahmenbedingungen für die Sicherstellung des ISDS im Bundesumfeld sind durch bestehende übergreifende Rechtsgrundlagen und Vorgaben definiert. Zusätzlich zu diesen gelten ggf. weitere Rahmenbedingungen auf Stufe Departement oder Amt. Die API Architektur Bund untersteht den geltenden Verordnung über die Koordination der digitalen Transformation und die IKT-Lenkung in der Bundesverwaltung [4].

[2] https://pubs.opengroup.org/architecture/togaf9-doc/arch/chap02.html
[3] https://www.admin.ch/gov/de/start/dokumentation/medienmitteilungen.msg-id-81580.html (Bundesgesetz über den Einsatz elektronischer Mittel zur Erfüllung von Behördenaufgaben (EMBaG) ist noch nicht abschliessend verabschiedet, Stand 30.11.2021)
[4] SR 172.010.58 - Verordnung vom 25. November 2020 über die Koordination der digitalen Transformation und die IKT-Lenkung in der Bundesverwaltung (Verordnung über die digitale Transformation und die Informatik, VDTI) (admin.ch)

2 Ausgangslage

2.1 Einordnung und treibende Kräfte

Elektronische Schnittstellen (Application Programming Interfaces, APIs) bilden die Grundlage für die digital zur Verfügung gestellten Behördenleistungen der Verwaltung. Aus diesem Grund hat der Bundesrat mit der Digitalisierungsstrategie des Bundes 2020 -2023 die Bundesverwaltung beauftragt, entsprechende digitale Behördenleistungen zur Verfügung zu stellen (s. Kapitel 2.2).

Der digitale Zugang zu Behördenleistungen und -daten der Bundesverwaltung erfolgt einerseits über E-Government-Portale oder Apps Mensch-zu-Maschine (H2M) oder direkt Maschine-zu-Maschine (M2M) über elektronische Schnittstellen (API). M2M-Anbindungen sind als Zugangskanäle für digitale Behördenleistungen zu verstehen, welche auch von E-Government-Portalen oder Apps (H2M) genutzt werden können.

Nachdem die Bundesverwaltung in den vergangenen Jahren zahlreiche E-Government-Portale aufgebaut hat (EasyGov.swiss, E-Portal EFD, eGovernment Portal UVEK, Agate Portal, etc.), wird von der Politik, der Wirtschaft und der Zivilgesellschaft vermehrt auch die Schaffung von Möglichkeiten für die direkte M2M-Anbindung gefordert. Dabei wird an dieser Stelle gewürdigt, dass einzelne Departement und VE bereits den Zugang zu Behördenleistungen via M2M-Anbindungen anbieten. Nachfolgend sind die wichtigsten treibenden Kräfte für die API Architektur Bund beschrieben.

2.1.1 Politische Motionen

Insgesamt drei Motionen aus dem Parlament fordern den Auf- bzw. Ausbau von elektronischen Schnittstellen.

Die Motionen 18.4276 von Ständerat Beat Vonlanthen und 18.4238 von Nationalrat Franz Grütter «Erleichterter Informationsaustausch durch die Einführung von elektronischen Schnittstellen in der Bundesverwaltung» fordern den Aufbau von elektronischen Schnittstellen für den direkten Datenaustausch mit Unternehmen und natürlichen Personen. Die darauf aufbauende Motion 20.4260 aus der Finanzkommission des Nationalrates fordert: «Mittels interoperablen, maschinenlesbaren und auf offenen Standards basierenden Echtzeit-Schnittstellen (sogenannte Microservices und APIs) soll der digitale Austausch zwischen den Bundesbehörden und den Behörden anderer Staatsebenen, der Wirtschaft und der Zivilgesellschaft verbessert werden.»

Sämtliche Motionen wurden vom Parlament und Bundesrat angenommen und die Bundesverwaltung zur Umsetzung beauftragt.

2.1.2 Digitale Verwaltung Schweiz bzw. E-Government Schweiz

Die digitale Verwaltung Schweiz beschreibt in der Agenda «Nationale Infrastrukturen und Basisdienste Digitale Verwaltung Schweiz» für die digitale Transformation im Bundesstaat folgende Ambition:

«Das Potenzial zur Automatisierung und Vereinfachung für die Wirtschaft ist ausgeschöpft: **Die Wirtschaft ist durch den automatisierten Datenaustausch und Schnittstellen mit der Verwaltung administrativ entlastet.** Die dafür erforderlichen gemeinsamen Standards, Infrastrukturen und institutionellen Grundlagen sind bis 2026 schweizweit konsequent umgesetzt.»

Im Rahmen der E-Government Strategie Schweiz 2020 – 2023 wird eine föderale Portalarchitektur im Rahmen des Umsetzungsziel 1 «EasyGov.swiss ausbauen» erarbeitet. Das Vorhaben wird durch das SECO geführt und durch eine Arbeitsgruppe mit Vertretern aus der Bundesverwaltung wie auch diverser Kantonsverwaltungen unterstützt. Die föderale Portalarchitektur fokussiert auf die Interoperabilität zwischen den Portalen. Dabei spielen APIs als Schnittstellen zwischen den Portalen und den Fachapplikationen bzw. Fachservices eine zentrale Rolle.

2.2 Einordnung in die Digitalisierungsstrategie des Bundes 2020-2023 (vormals IKT-Strategie)

2.2.1 Übersicht Digitalisierungsstrategie des Bundes 2020-2023

Die Digitalisierungsstrategie des Bundes[5] beschreibt Massnahmenbereiche entlang der folgenden vier strategischen Stossrichtungen.

[5] IKT-Strategie Bund 2020-2023 und IKT-Strategie Bund 2020-2023, Masterplan

- Stossrichtung A: Informations-, Daten- und Prozessmanagement
- Stossrichtung B: Innovations- und Changemanagement
- Stossrichtung C: Kunden- und Dienstleistungsorientierung
- Stossrichtung D: Zusammenwirken von Geschäft und IKT

Die Stossrichtungen sind mit der Strategie Digitale Schweiz, der Nationalen Strategie zum Schutz vor Cyber-Risiken, der E-Government-Strategie Schweiz sowie dem Schlussbericht EFD/KdK zum Projekt Digitale Verwaltung abgestimmt.

Die API Architektur Bund wird im Rahmen des Massnahmenbereichs «C2 Portale und Schnittstellen bereitstellen» erarbeitet. Der Massnahmenbereich ist Bestandteil der Stossrichtung C, die sich folgendermassen definiert:

Zweck und Wirkung

Behördenleistungen basieren auf dem Gesetzmässigkeits- und Zuständigkeitsprinzip. Daraus ergibt sich eine verwaltungsinterne Arbeitsteilung, die im Widerspruch zur Kundenperspektive stehen kann. Stossrichtung C adressiert diesen Aspekt und dient dem Erreichen der Ziele Serviceinnovation und Prozessinnovation.

Absicht

Die Stossrichtung C (Kunden- und Dienstleistungsorientierung) zielt darauf ab,

- die Kundenorientierung und damit das Denken von aussen nach innen, statt von innen nach aussen als Grundlage für die Digitalisierung zu verankern;
- digitale Interaktionen mit Kantonen, Gemeinden, ausländischen/internationalen Organisationen, Verbänden sowie insbesondere mit Unternehmen und Privatpersonen über die gesamte Bundesverwaltung zu harmonisieren;
- IKT-Organisationen und Anwender von Fragen rund um Infrastrukturen und Plattformen zu entlasten, so dass personelle und finanzielle Kapazitäten für die Digitalisierung des Geschäfts freigemacht werden.

Abbildung 2: Übersicht Strategische Initiativen Digitalisierungsstrategie Bund

2.2.2 Massnahmenbereich C2: Portale und Schnittstellen bereitstellen

Digitale Behördenleistungen können – abhängig davon, wer die Leistungen in Anspruch nimmt und welcher Art die Leistung ist – entweder über Portale oder über elektronische Schnittstellen bereitgestellt werden. Für Behördenleistungen mit vielen, wiederkehrenden Transaktionen bieten elektronische Schnittstellen den höheren Kundennutzen als Portale.

Auf der Stufe Bundesverwaltung ist ein Portfolio[6] der gemeinsam nutzbaren Portale und Schnittstellen-Dienste nach Massgabe des Nutzens zu steuern und zu führen, den die Portale und Schnittstellen-Dienste bei der Bereit-

[6] Die API Architektur Bund definiert als Instrument für die Führung des Portfolios das API Verzeichnis.

stellung von Behördenleistungen erzielen. Zur Übersicht soll ein zentraler Katalog mit den elektronisch angebotenen Behördenleistungen und den dazugehörigen Nutzergruppen[7] bereitgestellt werden, der auch über die Zuständigkeiten Auskunft gibt. Die VE definieren, welche Behördenleistungen über Portale / Schnittstellen sowie für welche Nutzergruppen diese angeboten werden sollen, und sie formulieren ihre Anforderungen an die Portal- und Schnittstellen-Dienste.

2.2.3 Ziel C2-1: Portale und Schnittstellen bereitstellen

Behördenleistungen werden den Nutzenden über Portale (H2M) und/oder elektronische Schnittstellen (M2M) zur Verfügung gestellt. Es bestehen gemeinsam genutzte Portale und Schnittstellen, um Leistungen schneller, kostengünstiger und mit geringeren Risiken erbringen zu können.

3 Ziel der API Architektur Bund

«Als moderne Verwaltung vereinfachen wir für unsere Partner den Zugang zu unseren Behördenleistungen, indem wir die Leistungen über beliebige elektronische Mittel nutzbar machen.»[8]

Entsprechend der Zielsetzung der Digitalisierungsstrategie der Bundesverwaltung und der politischen Forderungen konzentriert sich die API Architektur Bund auf die Bereitstellung von digitalen Behördenleistungen zu den Geschäftspartnern der Bundesverwaltung. Diese Behördenleistungen sollen direkt via API (M2M) zur Verfügung gestellt oder das API gemäss der föderalen Portalarchitektur als interoperabler Service auf einer E-Government-Plattform von einer Portalanwendung genutzt werden.

Zur fachlichen Unterstützung der Erarbeitung der API Architektur Bund wurde innerhalb der Bundesverwaltung eine Arbeitsgruppe mit Vertretern aus zahlreichen Departementen und VE (Leistungsbezüger und Leistungserbringer) zusammengestellt. Die Arbeitsgruppe hat als Zielbild der API Architektur Bund eine Vision entwickelt. Die Dimensionen der Vision und deren Inhalte sind in Tabelle 2 dargestellt.

Dimension	Vision
Nutzen	Der digitale Zugang zu (offenen) Behördenleistungen im Kontext M2M für Unternehmen, Verwaltung und Personen soll gefördert werden. Auf digitale Behördenleistungen zugreifende Maschinen können Lokale Apps, Fachservices oder Portalanwendungen sein. APIs sollen verwaltet, deren Existenz bekannt gemacht und damit wenn immer möglich wiederverwendet werden.
Zielgruppe	Die Zielgruppe besteht aus Unternehmensarchitekten, IT Architekten, Fachpersonal, Management sowie Orientierungssuchende aus Business und IT, welche Vorhaben zur digitalen Nutzung von Behördenleistungen unterstützen, sowohl auf Seite der Bundesverwaltung als auch auf Seite der Geschäftspartner ausserhalb der Bundesverwaltung.
Umfang	Die API Architektur Bund ist ein Ordnungsrahmen, welcher im Kontext der Schweizer Bundesverwaltung Empfehlungen für die Entwicklung und den Betrieb von APIs zur digitalen Abwicklung von Verwaltungsgeschäften macht. An dieser Stelle hervorzuheben sind die Gestaltungsempfehlungen zum API Design und zum Identity & Access Management sowie das Führen eines API Verzeichnisses. Auf technische Einschränkungen bei der Gestaltung von Lösungsarchitekturen wird weitgehend verzichtet.
Architekturprinzipien	Architekturprinzipien bieten Steuerungsrichtlinien für die Entwicklung und Betrieb von Behördenleistungen unterstützende APIs.
Standards & Good Practices	Die API Architektur basiert auf anerkannten Standards wie z.B. eCH, Tallinn, EIF, W3C sowie Good Practices aus dem Bundesumfeld, und wird periodisch weiterentwickelt.

Tabelle 2: Dimensionen Vision API Architektur Bund

[7] Die API Architektur Bund unterteilt die Nutzergruppen in drei Arten von Geschäftspartnern (s. Kapitel 4.2).
[8] Vision API Architektur Bund

4 Grundlagen und Geltungsbereich

4.1 Grundbegriffe

Tabelle 3 definiert einige Grundbegriffe, welche in der API Architektur Bund einheitlich verwendet werden.

Begriff	Beschreibung
Digitale Behördenleistung	Die digitale Behördenleistung ist der digitale Zugang zu einer Verwaltungsleistung. Eine digitale Behördenleistung kann auf einem einzigen API basieren, oder auf einem Verbund aus APIs aufbauen. Als digitale Behördenleistungen gelten Leistungen, die direkt via API oder über eine elektronische Schnittstelle auf einem Behördenleistungsportal bezogen werden können.
Application Programming Interface (API)	Ein API ist eine Programmierschnittstelle für die Kommunikation zwischen Software-Komponenten im Allgemeinen. Die API Architektur Bund verwendet den Begriff API für die Kommunikation zwischen Software-Komponenten über ein Netzwerk.
Geschäftspartner	Der Begriff Geschäftspartner ist der generische Begriff für das die digitale Behördenleistung nutzende Subjekt.
Anbieter	Der Begriff Anbieter ist der generische Begriff für die die digitale Behördenleistung anbietende Behörde.
Endbenutzer	Der Begriff Endbenutzer beschreibt eine natürliche Person, die IT-Produkte bzw. Software nutzt, welche über APIs auf digitale Behördenleistungen zugreifen.
Lokale App	Eine lokale App ist eine lokal beim Endbenutzer laufende Applikation, welche auf eine digitale Behördenleistung zugreift. Eine lokale App kann eine Desktop App, eine Mobile App oder im Browser ausgeführter API-Zugriffscode sein (z.B. JavaScript). Browser selbst werden nicht als lokale Apps betrachtet, da HTML unter Abwesenheit von API-Zugriffscode (s. Kapitel 8.3.2.1 «Code on Demand») nicht fähig ist, auf APIs zuzugreifen und Antworten auszuwerten.
Fachservice	Der Fachservice ist die Software-Komponente, welche die Fachlichkeit einer digitalen Behördenleistung abbildet und diese über eine elektronische Schnittstelle anbietet.
Fachservice-Cluster	Ein Fachservice-Cluster ist ein Verbund von identischen Instanzen eines Fachservices. Das Clustering dient der Steigerung der Performance und/oder der Ausfallsicherheit. Fachservice-Cluster können über mehrere Standorte verteilt sein.
Portalanwendung	Eine Portalanwendung ist eine mit einem Browser ansteuerbare Webapplikation. Der Begriff ist in der föderalen Portalarchitektur definiert, welche im Rahmen des Umsetzungsziels 1 der E-Government Strategie Schweiz 2020-2023 erarbeitet wurde. Eine Portalanwendung ist ein Spezialfall eines Fachservices und kann daher als API Client auftreten.
API Client	Ein API Client ist die Software-Komponente, welche fähig ist, elektronische Schnittstellen anzusteuern und Antworten zu verarbeiten. Ein API Client kann entweder eine Lokale App, ein Fachservice oder eine Portalanwendung sein.
API Gateway	Das API Gateway ist die zwischen API Client und Fachservice vermittelnde Komponente, welche den Fachservice vor den Gefahren der Aussenwelt schützt. Das API Gateway ist ein Reverse-Proxy mit zusätzlichen Fähigkeiten wie z.B. Traffic Management. Die Standorte eines Fachservice-Clusters können über eigene API Gateway-Instanzen verfügen, die individuell ansprechbar sind.

Tabelle 3: Grundbegriffe

4.2 Geschäftspartner

Die Bundesverwaltung bietet nach aussen digitale Behördenleistungen an. Die Geschäftspartner nutzen diese digitalen Behördenleistungen. Die Geschäftspartner existieren in den drei Kontexten Government (G), Business (B) und Citizen (C). Tabelle 4 liefert die Definitionen. Entsprechend gibt es drei Arten von Geschäftsbeziehungen zwischen VE und Geschäftspartnern: G2G, G2B und G2C. Die API Architektur Bund versteht unter dem Begriff Geschäftspartner die Geschäftspartner all dieser drei Arten von Geschäftsbeziehungen, inkl. allfällig nominierter Mitarbeitenden, die im Auftrag des Geschäftspartners agieren.

Geschäftspartner	Beschreibung
Government	Umfasst die internationalen Verwaltungsebenen, Bundesverwaltungseinheiten (VE-übergreifend), Kantone und Gemeinden
Business	Umfasst juristische Personen, d.h. Unternehmungen, aber auch Universitäten, Forschungsinstitute, NGOs oder Vereine
Citizen	Umfasst natürliche Personen, welche digitale Behördenleistungen via lokale Applikationen und Portalanwendungen nutzen

Tabelle 4: Definition der Geschäftspartner

4.3 Intermediäre

An die Stelle von Geschäftspartnern können Intermediäre treten (z.B. Swissdec[9]), welche die digitalen Behördenleistungen für die Geschäftspartner stellvertretend ansprechen. Die API Architektur Bund verwendet den Begriff Intermediär für die Organisation, welche die vermittelnde Dienstleistung gegenüber den Geschäftspartnern erbringt. Intermediäre sind Leistungserbringer, welche ausserhalb der Bundesverwaltung angesiedelt sind. Sie nehmen typischerweise folgende Aufgaben wahr, wobei verschiedene Kombinationen von Dienstleistungen möglich sind:

- Entwicklung und Betrieb einer vermittelnden Plattform (Hub)
- Entwicklung und Betrieb von Authentifizierungsmechanismen für den gesamten Kommunikationsweg vom API Client bis zum API Gateway resp. dem Fachservice
- Entwicklung von Standards für die Vereinheitlichung der elektronischen Datenübermittlung von Unternehmen zu Behörden der verschiedenen föderalen Verwaltungsebenen und Sozialversicherungen
- Entwicklung und Vertrieb von als API Client agierenden eigenen lokalen Apps
- Auditierung von als API Client agierender Standard-Software von Software-Herstellern (z.B. ERP-Software) zwecks Zertifizierung der Konformität mit o.g. Standards

4.4 API Typen

Die API Architektur Bund unterteilt APIs in folgende drei Typen: Public APIs, Partner APIs und Private APIs (s. Tabelle 5). Die API Architektur Bund ist auf Public APIs und Partner APIs ausgerichtet, Private APIs hingegen sind nicht im Fokus. Die API Architektur Bund kann auch für Private APIs angewendet werden, insbesondere dann, wenn diese das Potenzial haben, zu Public APIs oder Partner APIs zu werden. Public und Partner APIs können jeweils alle drei Geschäftspartner-Typen gemäss Kapitel 4.3 bedienen.

API Typen	Beschreibung
Public API	- Public APIs sind öffentlich und machen die von den VE angebotenen Daten und Leistungen intern und extern leicht zugänglich. - Sie können von Geschäftspartnern entweder OHNE oder MIT Registrierung einer Identität genutzt werden, wobei die Nutzung ohne Registrierung mit anonymer Nutzung gleichbedeutend ist. - Bei Nutzung mit Registrierung ist eine Identität zu hinterlegen. Danach erfolgt der Zugriff auf das Public API mittels Credential und Secure Token. - Nutzungsbedingungen (auch Nutzungsvereinbarung genannt) sind definiert, kommuniziert und müssen eingehalten. Bei Nutzung mit Registrierung kann ein explizites Akzeptieren der Nutzungsbedingungen eingefordert werden. - Bei Nutzung mit Registrierung wird, sofern die Identität authentifiziert werden kann, jeder Zugriffsantrag akzeptiert. Selbstregistrierung ist die Regel.
Partner API	- Partner APIs verlangen vom Geschäftspartner IMMER die Registrierung einer Identität. - Partner APIs können nur von berechtigten Partnern genutzt werden, mit denen die Bundesverwaltung eine Geschäftsbeziehung führt. Zugriffsanträge werden folglich immer geprüft und können akzeptiert oder abgelehnt werden. - Bei Partner APIs ist immer eine Identität zu hinterlegen. Danach erfolgt der Zugriff auf das Partner API mittels Credential und Secure Token. - Bei Partner APIs ist für die Autorisierung ein Berechtigungssystem für die individuelle Steuerung der einzelnen Zugriffe erforderlich. Für die Authentisierung muss ein dem Schutzbedarf der Daten entsprechendes Verfahren gewählt werden, das die Prüfung der Vertrauensstellung der Identität einbezieht.
Private API	- Private APIs sind auf die Verwendung innerhalb von VE ausgerichtet. Sie sind von aussen nicht zugänglich.

Tabelle 5: Definition der API Typen

[9] Swissdec ist ein als Verein organisiertes, nicht gewinnorientiertes Gemeinschaftsprojekt mehrerer unabhängiger Partner. Träger und Mitglieder des Vereins sind die schweizerische Steuerkonferenz (SSK), der Verein eAHV/IV als Vertreter der Ausgleichskassen, das Bundesamt für Statistik (BFS) als Vertreter des Bundes, der schweizerische Versicherungsverband (SVV) sowie die Suva. Swissdec regelt und organisiert die Übermittlung von Lohndaten von den Unternehmen zu den Sozialversicherungen und Behörden sowie die Prozessierung von Schadensereignissen. Swissdec betreibt unter dem Begriff Distributor eine Vermittler-Plattform.

4.5 Integrationspfade

Beim Erbringen von digitalen Behördenleistungen werden Fachservices verschiedenartig in API Clients und andere Fachservices integriert. Dies erfolgt über verschiedene Integrationspfade. Dieses Kapitel beschreibt fünf Integrationspfade und zeigt auf, welche davon im Fokus der API Architektur Bund stehen und welche API Typen mit diesen Integrationspfaden verbunden sind (s. Tabelle 6). Abbildung 3 visualisiert die fünf verschiedenen Integrationspfade vor dem Hintergrund der drei in Kapitel 4.2 eingeführten Geschäftspartner Government, Business und Citizen und deckt dabei in einer abstrakten Sicht die Gesamtheit der Anwendungsfälle für API Integration ab, in denen digitale Behördenleistungen vom Bund erbracht werden. Darüber hinaus liefert Kapitel 4.6 eine Übersicht zu sechs typischen, bei digitalen Behördenleistungen im Bundesumfeld beobachteten Anwendungsfällen.

Die API Architektur Bund liefert einen Ordnungsrahmen für die Integrationspfade IP3 und IP4. In Abbildung 3 sind diese blau markiert. Die anderen Integrationspfade IP1, IP2 und IP5 werden in der API Architektur Bund nicht betrachtet. IP3 ist relevant, da es sich dabei um die VE-übergreifende Nutzung von digitalen Behördenleistungen handelt, und die damit verknüpften API Aufrufe die Netzzonen des Bundes nicht verlassen. IP4 ist im Fokus, da die damit verbundenen API Aufrufe Bundes-Netzzonen-Grenzen überschreiten. Nicht behandelt werden sowohl IP1 und IP2, da die auf diesen Integrationspfaden liegenden APIs Private APIs sind, als auch IP5, da alle dabei involvierten Kommunikationspartner in Netzzonen ausserhalb des Bundes liegen.

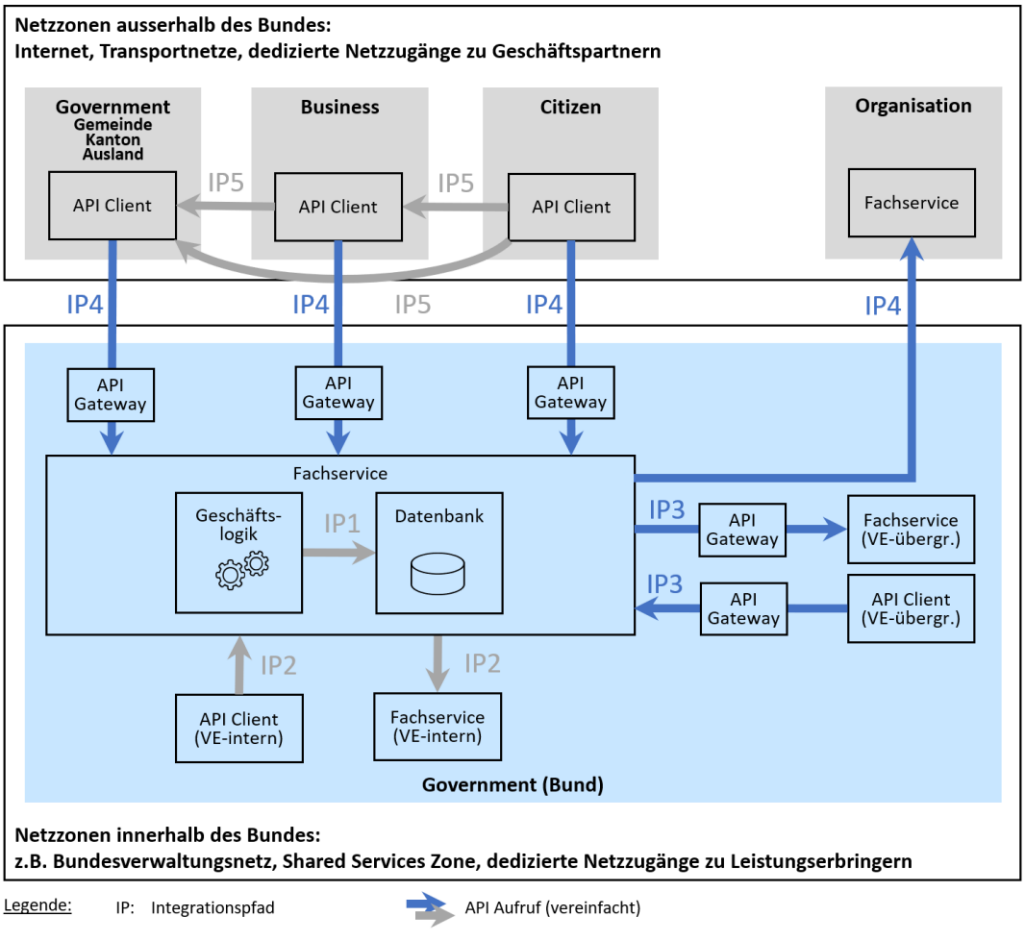

Abbildung 3: Integrationspfade

Gemäss Abbildung 3 kann eine sog. Organisation gegenüber einer VE des Bundes eine digitale Leistung erbringen, welche wiederum für das Erbringen einer digitalen Behördenleistung gegenüber dem Geschäftspartner erforderlich ist. Diese Organisation kann einem Geschäftspartner gemäss Tabelle 4 zugeordnet werden, ist in Abbildung 3 aber von den Geschäftspartnern begrifflich getrennt, da Geschäftspartner digitale Behördenleistungen beziehen, während Organisationen digitale Leistungen erbringen. In jedem Fall muss für das Integrieren einer digitalen Leistung einer Organisation in eine digitale Behördenleistung des Bundes eine gesetzliche Grundlage vorhanden sein (Legal Compliance).

Auf den Integrationspfaden IP3 und IP4 sieht die API Architektur Bund als vermittelnde Systemkomponente zwischen API Client und Fachservice den Einsatz eines API Gateways vor. Bei Integrationspfad IP4 kann ein Intermediär

an die Stelle des Geschäftspartners treten und damit eine Vermittler-Plattform die Rolle des API Clients wahrnehmen (s. Abbildung 4). Vermittler-Plattformen können bei Public APIs und Partner APIs gleichermassen zum Einsatz kommen.

Eine digitale Behördenleistung kann erfordern, dass ein Fachservice ohne unmittelbar vorangehenden Request des Geschäftspartners ein API eines Geschäftspartners aufruft (z.B. eine Erinnerung an eine zu erfüllende behördliche Pflicht). Die Integrationspfade IP3 und IP4 decken diese in die Gegenrichtung gerichteten Kommunikation ab, ohne diese in Abbildung 3 explizit darzustellen.

Abbildung 4 hingegen stellt diesen Rückwärtspfad dar. Der Rückwärtspfad führt nicht zwingend über das (dem Fachservice zugeordnete) API Gateway, denn die primäre Funktion des API Gateways besteht darin, die Ressourcen des Fachservices zu schützen und nicht diejenigen des (ein API anbietenden) API Clients. Der API Client darf seine Ressourcen durch ein eigenes API Gateway schützen, so wie auch die Organisation ihren Fachservice anhand eines API Gateways schützen darf. Abbildung 3 und Abbildung 4 verzichten auf die Visualisierung dieser API Gateways, da diese nicht im Fokus der API Architektur Bund stehen. Die nach aussen gerichteten API Aufrufe müssen aber in jedem Fall die im Bundesumfeld geltenden Richtlinien der Informationssicherheit und des Datenschutzes erfüllen.

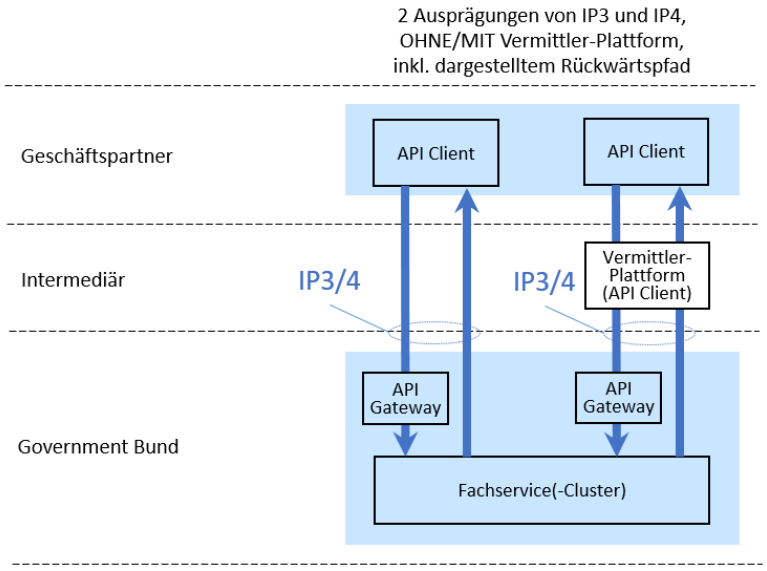

Abbildung 4: Zwei Ausprägungen der Integrationspfade IP3 und IP4, Darstellung des Rückwärtspfads

Pfad	Bezeichnung	Beschreibung	API Typen
IP1	FA-intern	Fachservice-interne Kommunikation	Private API
IP2	VE-intern	VE-interne Kommunikation zwischen zwei Fachservices oder zwischen einem API Client und einem Fachservice	Private API
IP3	Netzzonen innerhalb des Bundes	VE-übergreifende Kommunikation zwischen zwei Fachservices, wobei ein Fachservice auch als API Client auftreten kann.	Public API Partner API
IP4	Bundes-Netzzonen-Grenzen-überschreitend	Kommunikation zwischen einer lokalen App oder einem Fachservice ausserhalb der Bundesverwaltung und einem Fachservice innerhalb der Bundesverwaltung	Public API Partner API
IP5	Netzzonen ausserhalb des Bundes	Kommunikation zwischen Anwendungen in Netzzonen ausserhalb der Bundesverwaltung	Public API Partner API

Tabelle 6: Integrationspfade

4.6 Anwendungsfälle für API Integration

Die API Architektur Bund definiert sechs typische, bei digitalen Behördenleistungen im Bundesumfeld auftretende Anwendungsfälle für API Integration. Sie unterscheidet dabei zwei Gruppen: Solchen, bei denen die API Integration im API Client stattfindet, und solchen, bei denen die API Integration im Fachservice stattfindet. Abbildung 5 zeigt die Visualisierungen der sechs Anwendungsfälle, während Tabelle 7 die dazugehörigen Beschreibungen liefert.

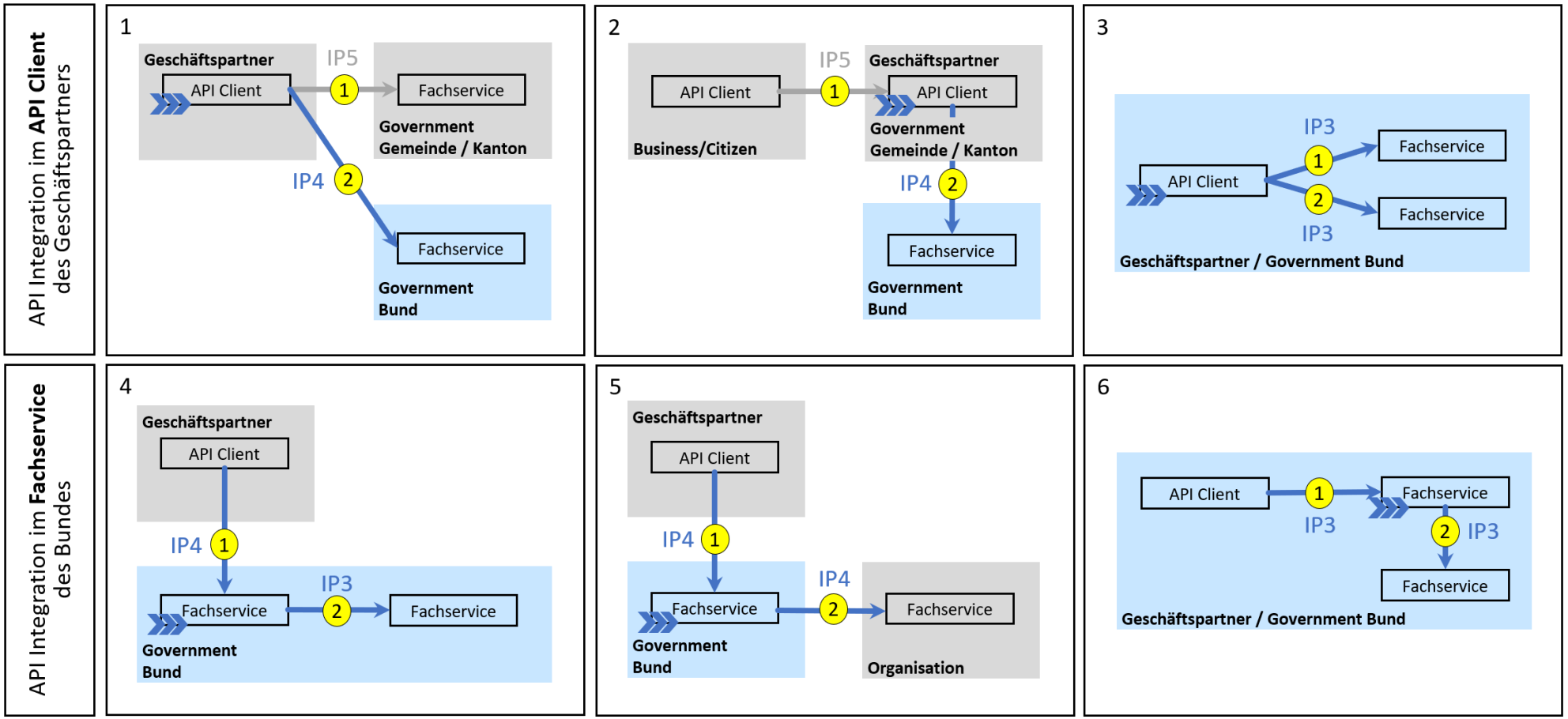

Abbildung 5: Anwendungsfälle für API Integration

API Integration im API Client des Geschäftspartners		API Integration im Fachservice des Bundes	
1	Ein Geschäftspartner ausserhalb des Bundes integriert 1-N API Aufrufe von Fachservices ausserhalb und innerhalb des Bundes im API Client.	4	Der Fachservice einer VE wird von einem Geschäftspartner ausserhalb des Bundes aufgerufen und integriert 1-N APIs anderer VEs des Bundes.
2	Ein Geschäftspartner ausserhalb des Bundes wird von einem Drittsystem aufgerufen und integriert für die Leistungserbringung gegenüber dem Drittsystem 1-N APIs von VEs des Bundes.	5	Der Fachservice einer VE wird von einem Geschäftspartner ausserhalb des Bundes aufgerufen und integriert 1-N APIs von Organisationen ausserhalb des Bundes.
3	Der Geschäftspartner ist eine VE des Bundes, dessen API Client 1-N APIs anderer VEs des Bundes integriert.	6	Der Fachservice einer VE wird von einem Geschäftspartner innerhalb des Bundes, d.h. einer anderen VE, aufgerufen und integriert 1-N APIs anderer VEs.

Tabelle 7: Anwendungsfälle für API Integration

In allen sechs der in Abbildung 5 abgebildeten Anwendungsfälle sind die Integrationspfade IP3 und/oder IP4 zu erkennen, auf welche die API Architektur Bund fokussiert. Jeder der sechs Anwendungsfälle lässt sich in Abbildung 3 wiedererkennen, indem in Abbildung 3 der entsprechende Ausschnitt gewählt wird. Die sechs Anwendungsfälle können auch kombiniert werden, wobei ein Fachservice auch als API Client auftreten kann.

Abbildung 5 markiert bei jedem der sechs Anwendungsfälle, in welcher Systemkomponente die Integrationsleistung verortet ist. Die Integrationsleistung kann dabei auch aus einem mehrstufigen Workflow an API Aufrufen bestehen.

Abbildung 5 veranschaulicht auch, dass ein Anwendungsfall typischerweise mit einer bestimmten Reihenfolge an API Aufrufen verknüpft ist. Während die Anwendungsfälle 2, 4, 5 und 6 auf den in Abbildung 5 dargestellten Reihenfolgen an API Aufrufen basieren, muss die Reihenfolge der API Aufrufe bei den Anwendungsfällen 1 und 3 nicht zwingend wie abgebildet definiert sein. Insbesondere können die API Aufrufe in diesen Fällen je nach fachlicher Anforderung auch parallel abgesetzt werden.

Während bei API Integration im API Client der Geschäftspartner für das Design der API Integration verantwortlich ist, ist es bei API Integration im Fachservice in jedem Fall die VE des Bundes. Bei den Anwendungsfällen 3 und 6 ist der Bund selbst der Geschäftspartner.

4.7 Positionierung der API Architektur Bund gegenüber OGD und I14Y

Der Bundesrat genehmigte am 30. November 2018 die Open Government Data (OGD) Strategie Schweiz 2019-2023[10]. Mit dieser Strategie sollen der Öffentlichkeit auf dem vom Bundesamt für Statistik (BFS) betriebenen Portal opendata.swiss[11] offene und frei nutzbare Verwaltungsdaten von Anbieter aller föderalen Verwaltungsebenen zur Verfügung gestellt werden. In Fällen, wo das Portal als Portalanwendung auftritt und APIs des Bundes ansteuert, ist die API Architektur Bund auch auf ins Portal opendata.swiss integrierte API Clients anwendbar.

Die I14Y Interoperabilitätsplattform[12] ist der Nationale Datenkatalog der Schweiz und unterstützt den effizienten Datenaustausch zwischen Behörden, Unternehmen und Bürgern. Darin wird eine Übersicht der Datensammlungen und Schnittstellen von Bund, Kantonen und Gemeinden laufend ausgebaut und deren Metadaten zentral zur Verfügung gestellt. Die I14Y Interoperabilitätsplattform bietet sich für die Umsetzung des API Verzeichnisses an.

[10] https://www.bfs.admin.ch/bfs/de/home/dienstleistungen/ogd.html
[11] https://opendata.swiss
[12] https://www.i14y.admin.ch

5 Strukturübersicht

5.1 Struktur API Architektur Bund

Die API Architektur Bund verfolgt das Ziel, im Bundesumfeld den digitalen Zugang zu Behördenleistungen im Kontext M2M für Unternehmen, Verwaltung und Personen zu standardisieren und zu fördern.

Die API Architektur Bund besteht aus einem Satz von Architekturprinzipien und einer an den vier Architekturebenen (BDAT) nach TOGAF[13] ausgerichteten Referenzarchitektur. Gemeinsam bieten Prinzipien und Referenzarchitektur einen Ordnungsrahmen und eine gemeinsame Sprache für den Entwurf von Lösungsarchitekturen für digitale Behördenleistungen in den Departementen und VE. Die API Referenzarchitektur ist weiter unterteilt in Architekturbausteine und Gestaltungsempfehlungen (GE), welche entlang der Architekturebenen gegliedert sind.

Die API Architektur Bund selbst macht keine Aussagen zur Verbindlichkeit. Getrennt von der API Architektur Bund können durch den Erlass von Weisungen Aussagen zur Verbindlichkeit von Elementen der API Architektur Bund gemacht werden. Der Erlass von und Umgang mit API Weisungen ist im Kapitel 7.5 beschrieben. Abbildung 6 veranschaulicht, wie sich die API Architektur Bund gegenüber von Lösungsarchitekturen und API Weisungen aufstellt.

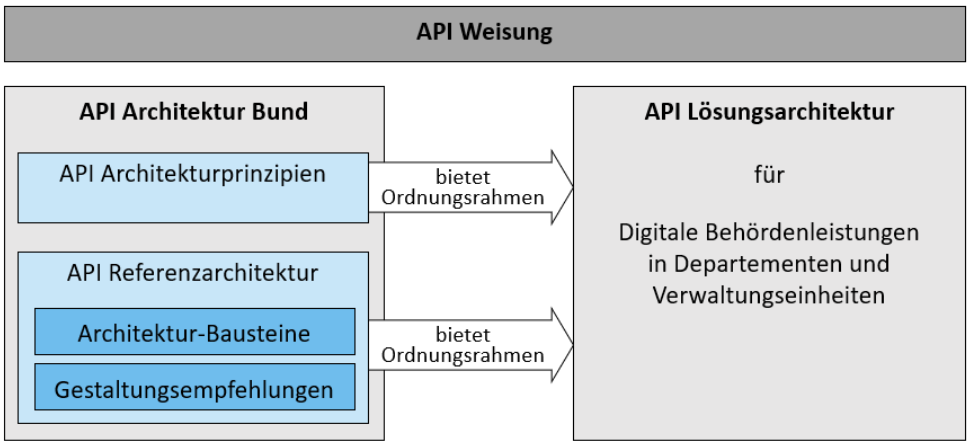

Abbildung 6: Positionierung der API Architektur Bund

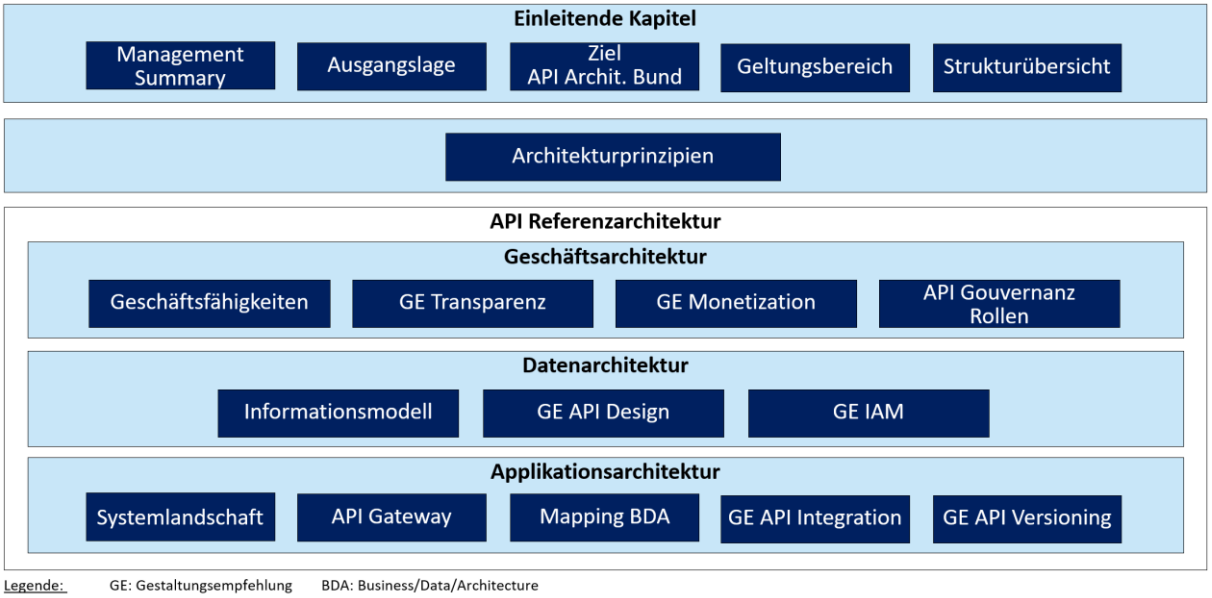

Abbildung 7: Strukturübersicht API Architektur Bund, Architekturebenen nach TOGAF

[13] https://www.opengroup.org/togaf

Abbildung 7 veranschaulicht die Gesamtstruktur der API Architektur Bund und gibt eine Übersicht zu den Inhalten. Jeder Architekturebene wird ein Kapitel gewidmet, mit Ausnahme der Technologieebene. Auf die Technologiearchitektur wird verzichtet, da die API Architektur Bund keine Technologieempfehlungen macht. Den Architekturprinzipien und der Referenzarchitektur gehen die einleitenden Kapitel 1-5 voraus.

5.2 API Referenzarchitektur

Um den Ordnungsrahmen der API Architektur Bund abzustecken, definiert die API Architektur Bund eine API Referenzarchitektur (Kapitel 7-9), welche alle relevanten Architekturbausteine einer API Architektur beschreibt. Die API Referenzarchitektur vereinigt diese Architekturbausteine zu einem Gesamtbild, wobei diejenigen Architekturbausteine definiert werden, welche für das Erbringen von digitalen Behördenleistungen sowie für die effiziente Interoperabilität als notwendig betrachtet werden.

Je nach Anwendungsfall im Bundesumfeld werden bei der Entwicklung einer API Lösungsarchitektur nur diejenigen Architekturbausteine verwendet, welche zur Erfüllung der gegebenen Geschäftsanforderungen einen Beitrag leisten. Die API Referenzarchitektur ist so definiert, dass für die Gestaltung der Lösungsbausteine weitgehende Freiheiten bleiben. Zwecks Nutzung von Synergien im Bundesumfeld wird, wo dies möglich und sinnvoll ist, der Einsatz von IKT-Standarddiensten oder zentralen Anwendungen empfohlen.

Die API Architektur macht keine Vorgaben. Sie macht jedoch Gestaltungsempfehlungen. Sie unterscheidet zwischen Architekturbausteinen, bei denen auf eine Gestaltungsempfehlung verzichtet wird und überlässt dabei die Ausgestaltung den Leistungserbringern und VE, und solchen, bei denen sie eine konkrete Gestaltungsempfehlung macht.

Diese Gestaltungsempfehlungen beruhen zwecks Sicherstellung von Interoperabilität und Nachhaltigkeit immer entweder auf anerkannten Standards oder Good Practices.

6 Architekturprinzipien

Die Architekturprinzipien der API Architektur Bund sind eingebettet in eine Architekturprinzipien-Hierarchie mit drei Ebenen:

1. Übergreifende für die Bundesverwaltung geltende (Architektur-)Prinzipien wie (Liste nicht abschliessend):
    - Tallinn Declaration on eGovernment: Die Deklaration enthält fünf zentrale Prinzipien für den Bereich E-Government[14].
    - SOA-Policies[15]: Richtlinien für den Einsatz von Services gemäss den Konzepten der serviceorientierten Architektur (SOA) in der Bundesverwaltung
    - Prinzipien, die aus der IAM Bund Rahmenarchitektur hervorgehen (s. Kapitel 8.4)
2. Die Architekturprinzipien der API Architektur Bund (dieses Kapitel), welche als der API Referenzarchitektur übergeordnete Architekturprinzipien zur Gestaltung von API Lösungsarchitekturen dienen.
3. Die in der API Referenzarchitektur den drei Architekturebenen zugeordneten Gestaltungsempfehlungen (s. Kapitel 5.1).

Weitere API Guidelines für REST- und ereignisorientierte APIs sind von Zalando (Zalando RESTful API and Event Guidelines[16]) und den Schweizerischen Bundesbahnen (API Principles[17]) publiziert worden. Diejenigen der SBB basieren auf denjenigen von Zalando. Einzelne Architekturprinzipien der API Architektur Bund sind in denjenigen von Zalando und der SBB wiederzufinden. Meist sind diese API Guidelines auf sehr detaillierter Ebene formuliert und dürfen daher als Ergänzung zu den nachfolgend formulierten Architekturprinzipien verstanden werden.

Die API Architektur Bund definiert zehn Architekturprinzipien, die sich um drei Themenbereiche gruppieren: Kunden- und Service-Orientierung, API Design / Interoperabilität sowie Publizieren / Kommunizieren von APIs (s. Abbildung 8). Tabelle 8 beschreibt die Architekturprinzipien im Detail, gegliedert nach Name, Aussage, Begründung und Auswirkung.

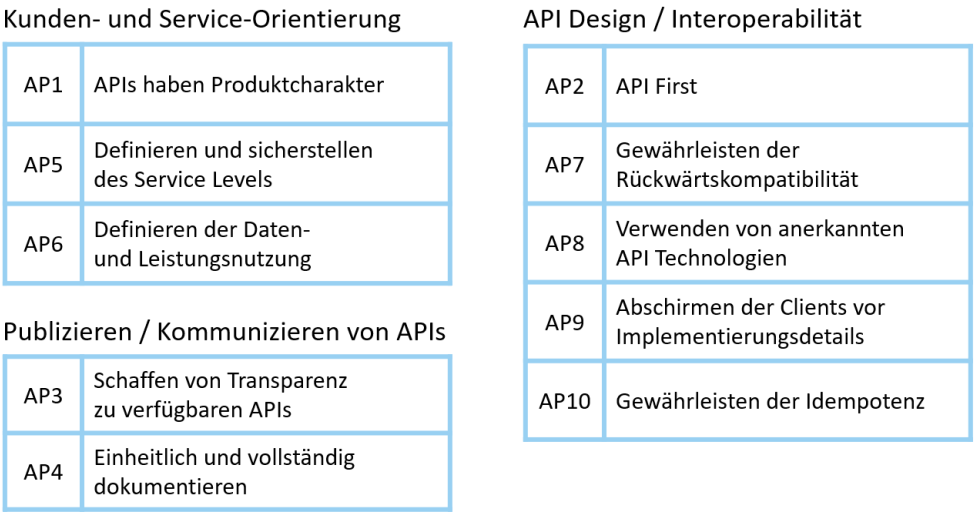

Abbildung 8: Gruppierung der Architekturprinzipien der API Architektur Bund

[14] https://www.admin.ch/gov/de/start/dokumentation/medienmitteilungen.msg-id-68342.html
[15] https://intranet.dti.bk.admin.ch/isb_kp/de/home/ikt-vorgaben/architekturen/r016-soa-policies.html
[16] https://opensource.zalando.com/restful-api-guidelines
[17] https://schweizerischebundesbahnen.github.io/api-principles

Ref.	Name	Aussage	Begründung	Auswirkung
AP1	**APIs haben Produktcharakter**	*APIs werden Geschäftspartnern wie Produkte mit zugehörigen Dienstleistungen angeboten. Jedes API hat einen bei der VE angesiedelten API Product Owner. Das API ist ein Teilprodukt des Gesamtpakets an Daten und Leistungen, welche das API zugänglich macht.*	Die Bereitstellung von digitalen Behördenleistungen ist kundenzentriert sowie daten- und leistungsgetrieben. Die Handhabung von APIs als Produkt ermöglicht es, die Behördenpflichten effizient zu leisten und dabei die Anforderungen der Geschäftspartner optimal zu erfüllen. Durch diesen Produktansatz wird eine hohe Akzeptanz und Nutzung der APIs bei den Geschäftspartnern erreicht.	Wird ein API als Produkt behandelt, so stellt dies die Wahrnehmung von Produktmanagement-spezifischen Aufgaben sicher. Dies beinhaltet: - Sicherstellung von Einfachheit und Benutzerfreundlichkeit - Bewirtschaften der Geschäftspartner-Beziehungen - Bereitstellen der API Dokumentation - Bereitstellen von fachlichem und technischen Support - Bereitstellen einer API Testversion - Einfache Registrierungsprozesse - Auswertung der API Nutzung - Weiterentwicklung der API Mit der Handhabung von APIs als Produkt und übergeordnet damit auch von Daten und Leistungen als Produkt wird auch die Haftung für die Richtigkeit der Daten und Dienste durch den API Product Owner und Data Owner gewährleistet. Der Data Owner definiert die rechtlichen Rahmenbedingungen zur Datennutzung, der API Produkt Owner verantwortet die Vorgaben-konforme Bereitstellung der APIs.
AP2	**API First**	*Das funktionale Design von nach aussen gerichteten APIs wird als zentrales Element einer digitalen Behördenleistung definiert. Das Design richtet sich am Geschäftsfall der digitalen Behördenleistung sowie den benötigten Daten und Leistungen aus und wird in einem standardisierten Spezifikationsformat dokumentiert.*	Die vorgängige Planung von APIs bei einem API-First-Ansatz mit konsistenten, verständlichen API Spezifikationen stellt sicher, dass der Geschäftsfall optimal abgedeckt ist und die funktionalen Anforderungen an die Schnittstelle bekannt und dokumentiert sind. Dadurch wird die effiziente Bereitstellung digitaler Behördenleistungen ermöglicht. Ein API-First-Ansatz stellt die Stabilität der APIs sicher. Insbesondere dann, wenn die dahinterliegende Anwendungsarchitektur / Service-Implementierung ersetzt wird. Er stellt damit die Interoperabilität mittels API mit allen das API nutzenden Geschäftspartnern sicher. Der API-First-Ansatz unterstützt somit den Trend zu serviceorientierten Architekturen.	Die standardisierte API Spezifikation ist die gültige und aktuelle Dokumentation des API. Dazu gehören Vereinbarungen zu Schnittstellentypen, Datenformaten, Message Exchange Patterns, Message Typen und API Protokollen, welche sich am Geschäftsfall orientieren. Zu API Spezifikationen wird von Geschäftspartnern und Entwicklern zu einem frühen Zeitpunkt Feedback eingeholt und berücksichtigt. Bei Fällen, wo bereits Fachservices bestehen, wird die Technologiearchitektur des API möglichst unabhängig vom Fachservice entworfen. Dies führt ggf. zu neuen Anforderungen an den Fachservice. Bei Fällen, wo Fachservices nicht existieren, richten sich deren Ausgestaltung am Design der APIs aus. Hinter jedem nach aussen gerichteten API steht ein Fachservice, aber nicht jeder Fachservice verfügt über nach aussen gerichtete APIs.
AP3	**Schaffen von Transparenz zu verfügbaren APIs**	*Die API Metadaten und Anlaufstellen zu den in der Bundesverwaltung verfügbaren, nach aussen gerichte-*	Mit der Publikation wird die Nutzung und Wiederverwendung der in der Bundesverwaltung verfügbaren, nach aussen gerichte-	Eine gute Pflege der API Metadaten ist erforderlich, um sicherzustellen, dass Informationen zu APIs für die Zielgruppe der API Architektur Bund jederzeit zugänglich sind.

Ref.	Name	Aussage	Begründung	Auswirkung
		ten APIs sind nach den Bestimmungen des EMBaG[18] in einem zentralen API Verzeichnis publiziert.	ten APIs gefördert. Das Schaffen von Transparenz ist damit auch ein Mittel zur Förderung von Kundenorientierung.	
AP4	**Einheitliche und vollständige Dokumentation von APIs**	*Die API Dokumentation von API Releases innerhalb der Bundesverwaltung ist einheitlich und vollständig.*	Die API Dokumentation ist der Grundstein für eine gute Erfahrung bei der Nutzung und der Weiterentwicklung einer API. Eine vollständige und einheitliche API Dokumentation erleichtert das Arbeiten mit der API und sorgt für Zufriedenheit und Akzeptanz bei Geschäftspartnern.	Die API Dokumentation beinhaltet die Dokumentation eines (veröffentlichten) API Releases. Sie umfasst alle technischen Informationen, welche notwendig sind, um einen Bezugsentscheid fällen und das API aus einem API Client heraus korrekt anzusprechen zu können. Wo immer möglich wird die API Dokumentation automatisiert erstellt. Dieses Prinzip gilt nicht für APIs, deren Daten gemäss EMBaG nicht veröffentlich werden, z.B. auf Grund von Bestimmungen zum Daten- oder Informationsschutz.
AP5	**Definieren und sicherstellen des Service Levels**	*Jedes API verfügt über definierte und transparente Service Level Objectives (SLO). Garantieren die API Dokumentation oder bei kostenpflichten APIs der Lizenzvertrag SLOs, so stellt der API Product Owner deren Einhaltung sicher.*	Dieses Prinzip stellt den Service Level der angebotenen digitalen Behördenleistungen sicher. Dafür werden sowohl für Public APIs als auch für Partner APIs SLOs definiert.	Die Service Level Objectives sind definiert und transparent kommuniziert. Darin enthalten sind Service Level Parameter wie Verfügbarkeit, Antwortzeit, Durchsatz oder Wiederherstellungszeit. Garantieren die API Dokumentation oder bei kostenpflichten APIs der Lizenzvertrag SLOs, so erfordert die Einhaltung der SLOs auch deren Messung. Die gegenüber den Kunden kommunizierte Servicequalität basiert auf der Vererbung aller zugrundeliegenden SLOs, auch bei API Integration im Fachservice des Bundes. Es gelten die schwächsten SLOs. Grundsätzlich geben die SLOs des Fachservice die SLOs des APIs vor. Ein Departement oder eine VE kann für ihre digitalen Behördenleistungen einheitliche SLOs festlegen, z.B. «Best Effort».
AP6	**Definieren der Daten- und Leistungsnutzung**	*Bei jedem API definiert eine Nutzungsvereinbarung, wie die dazugehörenden Daten und Leistungen genutzt werden dürfen. Die Nutzungsvereinbarung ist ein Bestandteil des Gesamtpakets an Daten und Leistungen, welche das API zugänglich macht.*	Der rechtliche Rahmen für die Nutzung der über die digitale Behördenleistung ausgetauschten Daten muss festgelegt sein. Die Nutzung ist grundsätzlich in den Verordnungen geregelt. Die Nutzungsvereinbarung kann auf diese verweisen und schliesst zudem allfällige Lücken. Damit sind die rechtlichen Grundlagen für die Nutzung der Behördenleistungen und der dazugehörigen Daten geschaffen.	Die Definition der in der Nutzungsvereinbarung festgehaltenen Datennutzung enthält die Bedingungen, unter denen die über APIs zwischen Anbieter und Geschäftspartner ausgetauschten Daten als Teil von digitalen Behördenleistungen genutzt werden dürfen. Die Regeln der Datennutzung sind abhängig vom Geschäftsfall und der zugehörigen Leistung und/oder den entsprechenden Daten.
AP7	**Gewährleisten der Rückwärtskompatibilität**	*Bei der Weiterentwicklung von APIs wird Rückwärtskompatibilität sichergestellt.*	APIs werden laufend weiterentwickelt. Die Aufrechterhaltung der Rückwärtskompatibili-	Ist die Rückwärtskompatibilität gewährleistet, so kann bei der Publikation von neuen API Releases auf Seite des Geschäftspartners auf die Anpassung des API Clients verzichtet werden.

[18] https://www.admin.ch/gov/de/start/dokumentation/medienmitteilungen.msg-id-81580.html (Bundesgesetz über den Einsatz elektronischer Mittel zur Erfüllung von Behördenaufgaben (EMBaG) ist noch nicht in Kraft, Stand 30.11.2021)

Ref.	Name	Aussage	Begründung	Auswirkung
			tät zwischen verschiedenen API Releases vermeidet die Notwendigkeit, auf Seite des Geschäftspartners bei neuen API Releases am API Client Anpassungen vornehmen zu müssen.	Erzwingt ein Anwendungsfall den Bruch der Rückwärtskompatibilität (z.B. Legal Compliance), so ist dem Geschäftspartner eine sich am Anwendungsfall orientierte Frist einzuräumen, um den API Client an einen im Betrieb stehenden API Release anzupassen. Das Erzwingen von Migrationen hat immer den Parallelbetrieb von API Releases zur Folge. Die Anzahl parallel betriebener API Releases wird aus Gründen der Betriebskostenreduktion tief gehalten.
AP8	**Verwenden von anerkannten API Technologien**	*APIs in der Bundesverwaltung verwenden nur breit anerkannte API Technologien.*	APIs sind neben der fachlichen Funktionalität immer auch ein Abbild der Sprache, in der zwei Systeme oder Systeme in einem Systemverbund miteinander kommunizieren. Um die Interoperabilität zu fördern, werden nur breit anerkannte API Technologien verwendet. Damit wird auch eine höhere Wiederverwendbarkeit von APIs erreicht.	Die API Technologie manifestiert sich in den Vereinbarungen zum Datenaustausch, zu denen die API Architektur Bund Empfehlungen gibt. Dazu gehören: - API Protokoll - Schnittstellentyp - Message Exchange Pattern - Datenformat - Message-Typ
AP9	**Abschirmen der API Clients vor Implementierungsdetails**	*Implementierungsdetails von Fachservices bleiben vor den Geschäftspartnern verborgen.*	Getreu dem Prinzip «API First» stellt die API Spezifikation eine gegen aussen konstante Spezifikation dar. Das API implementiert diese API Spezifikation.	APIs dürfen keine Details über die dahinterliegende Implementierungstechnik verraten. Diese kann sich von API Release zu API Release oder von Fachservice Release zu Fachservice Release ändern, ohne dass dies gegen aussen sichtbar wird.
AP10	**Gewährleisten der Idempotenz**	*Mehrfach aufeinander folgende identische API Aufrufe haben im Fachservice immer die gleiche Wirkung.*	Mehrfach aufeinander folgende, identische, schreibende Aufrufe durch API Clients können beabsichtigt oder unbeabsichtigt erfolgen. Sie dürfen ihre Wirkung im Fachservice nur einmal entfalten. Da das API Gateway die Fachlichkeit des Fachservices gegen aussen repräsentiert, ist das Prinzip der Idempotenz auch für das API Gateway anzuwenden.	Während mehrfach aufeinander folgende, identische Lesezugriffe mehrfach ausgeführt werden, dürfen mehrfach aufeinander folgende, identische Schreibzugriffe nur einmal ausgeführt werden. Das API erkennt redundante Schreibzugriffe und informiert den API Client korrekt über das im Fachservice hinterlassene Verarbeitungsergebnis.

Tabelle 8: Architekturprinzipien der API Architektur Bund

7 Geschäftsarchitektur

7.1 Fähigkeit API Management

7.1.1 Übersicht

Die API Architektur Bund verwendet für die Definition von Geschäftsfähigkeiten das Instrument der fähigkeitsbasierten Planung nach TOGAF (s. Anhang Kapitel 10.1). Sie definiert die Geschäftsfähigkeit **API Management** als Top-Level-Fähigkeit für API-spezifische Geschäftsfähigkeiten. Darüber hinaus nutzt die Geschäftsfähigkeit API Management weitere Geschäftsfähigkeiten, die im Kontext der API Architektur notwendig sind, jedoch auch in API-fernen IKT-Vorhaben benötigt werden. Um ein vollständiges Bild der für eine API Architektur benötigten Geschäftsfähigkeiten zu zeichnen, beschreibt die API Architektur Bund die Gesamtheit dieser Geschäftsfähigkeiten. Die API Architektur Bund unterscheidet 20 Geschäftsfähigkeiten, welche fünf API-bezogenen Prozessstufen zugordnet sind.

Die Geschäftsfähigkeit API Management wird umgesetzt, indem die API Lösungsarchitektur die API-spezifischen Geschäftsfähigkeiten umsetzt und die API-unspezifischen Geschäftsfähigkeiten einbindet. Daran sind alle Akteure beteiligt, sowohl der Anbieter, der Geschäftspartner als auch die Leistungserbringer.

Obwohl die Modellierung von Prozessen erst nach der fähigkeitsbasierten Planung erfolgt, sind bereits an dieser Stelle Prozessstufen definiert. Die Prozessstufen dienen der Gruppierung der Geschäftsfähigkeiten und erlauben die Modellierung eines sequenziellen Ablaufs. Eine Iteration dieses Ablaufs startet infolge einer Initiative zur Entwicklung, Veränderung oder Stilllegung einer digitalen Behördenleistung immer bei der Prozessstufe Manage API Lifecycle und wird in der in Abbildung 9 dargestellten Reihenfolge durchlaufen. Die Prozessstufe Register Client wird unter bestimmten Bedingungen übersprungen.

Dieses Kapitel beschreibt die Prozessstufen dieses Ablaufs. Abbildung 10 zeigt die Geschäftsfähigkeiten und die Zuordnung zu den Prozessstufen. Eine Geschäftsfähigkeit kann zu mehreren Prozessstufen einen Beitrag leisten. Die API-spezifischen Geschäftsfähigkeiten sind in Abbildung 10 markiert.

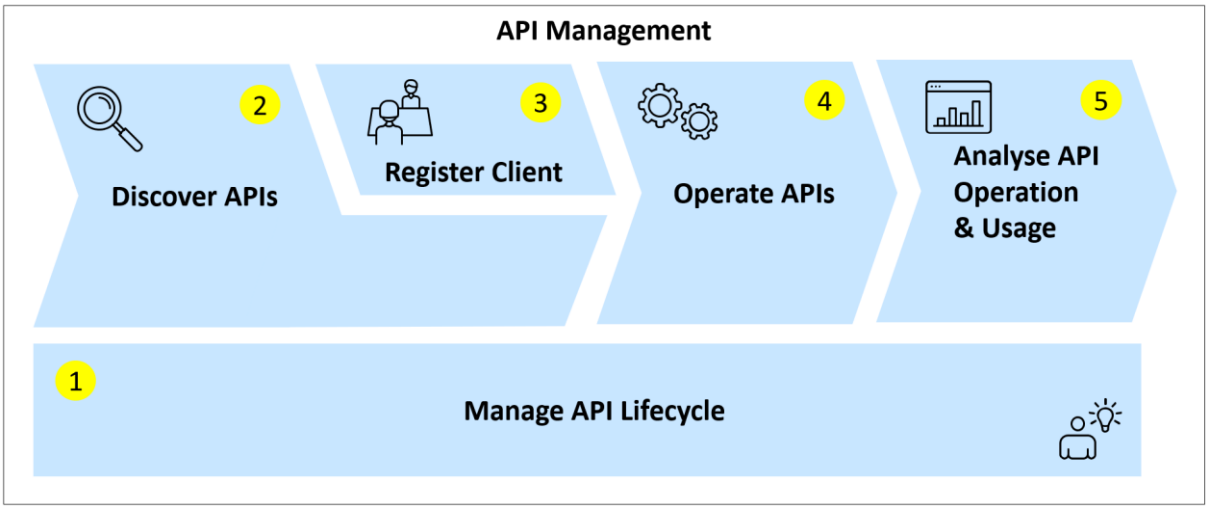

Abbildung 9: Prozessstufen zur Gruppierung von Geschäftsfähigkeiten

Der Ablauf ist in die Prozessstufen Manage API Lifecycle (1), Discover APIs (2), Register Client (3), Operate APIs (4) und Analyse API Operation & Usage (5) unterteilt, welche in Beziehung zueinanderstehen. Zu Beginn steht die Initiative, eine Behördenleistung zu entwickeln. Eine Initiative kann beispielsweise getrieben sein durch eine parlamentarische Initiative, eine gesetzliche Vorgabe, oder durch das Bestreben, eine existierende Behördenleistung digital anzubieten.

Der Beginn der Initiative ist auch der Beginn der Prozessstufe **Manage API Lifecycle**, womit der Lebenszyklus eines APIs beginnt. Das API wird konzipiert, entwickelt, getestet, dokumentiert, schliesslich publiziert, genutzt und am Ende des Lebenszyklus wieder zurückgebaut. Die Prozessstufe ist damit ein übergeordneter Prozess, der mit der Planung der Entwicklung einer digitalen Behördenleistung beginnt und nach der Stilllegung der digitalen Behördenleistung endet. APIs werden ständig weiterentwickelt. Auslöser für die Weiterentwicklung können Verände-

rungen in den gesetzlichen Rahmenbedingungen, neue Kundenanforderungen oder über die Zeit gesammelte Optimierungswünsche sein. Das API Lifecycle Management deckt dabei alle API Versionen und API Releases ab, welche im Laufe des Lebenszyklus erstellt und publiziert werden. Ein API Release ist eine API Version, die im API Verzeichnis publiziert wird.

Die Publikation zielt darauf ab, einen API Release dem relevanten Zielpublikum - namentlich Unternehmensarchitekten, IT Architekten, Fachpersonal, Management sowie Orientierungssuchende aus Business und IT – zur Verfügung zu stellen. Dies umfasst das Bereitstellen von installierbaren Software-Paketen (API Build) und/oder API Konfigurationen in einem Repository resp. in einer Registry[19] sowie das Deployment von daraus abgeleiteten API Instanzen in die Betriebsumgebung. Zudem werden API Metadaten zum API Release in einem API Verzeichnis eingetragen und damit die Verfügbarkeit des APIs publiziert. Bei Public APIs ist die Publikation immer an ein Deployment einer API Instanz in die Betriebsumgebung geknüpft, bei Partner APIs, die naturgemäss eine Registrierung erfordern, kann das Deployment der API Instanz an die Freigabe des Registrierungsantrags geknüpft sein.

Das API Verzeichnis ist der zentrale Ort einer Organisation, wo sich die Zielgruppe über die veröffentlichten APIs informiert und gezielt nach verfügbaren APIs sucht, um diese für die Entwicklung eigener APIs zu nutzen. Diese Suche nach geeigneten wiederverwendbaren APIs ist die Kernfähigkeit der Prozessstufe **Discover APIs.** Bei APIs, welche die Registrierung einer Identität voraussetzen, ist das API Verzeichnis der Startpunkt für die Prozessstufe Register Client.

Die Prozessstufe **Register Client** beinhaltet das Beantragen, Freigeben und Terminieren des Zugriffs auf ein API. Wenn die Geschäftspartner Stammdaten-relevant sind, gehört dazu auch das Registrieren eines Geschäftspartners im Stammdaten-Management. Die Prozessstufe wird bei Public APIs ohne Registrierung übersprungen.

Kernstück des APIs ist das API Gateway, welches eine Vermittler-Rolle zwischen API Client und Fachservice einnimmt (s. Kapitel 9.2). Es stellt das zentrale Eingangstor zur digitalen Behördenleistung dar. Es steuert die Zugriffe auf die dahinterliegenden Fachservices und schirmt diese gegen die Aussenwelt ab. Der Betrieb des API Gateways erfolgt in der Prozessstufe **Operate APIs**. Die Geschäftsfähigkeiten Logging und Monitoring sind das Bindeglied zur Prozessstufe Analyse API Operation & Usage.

Die Prozessstufe **Analyse API Operation & Usage** umfasst das Messen von Key Performance Indicators (KPIs), deren Auswertung und das Erstellen von Reports. Reports können betrieblicher oder fachlicher Natur sein und aus einfachen Listen oder komplexen Auswertungen bis hin zu Elementen künstlicher Intelligenz bestehen. Die Prozessstufe Analyse API Operation & Usage liefert auch die Basis, um aus dem Betrieb von APIs Einnahmen zu generieren. Insbesondere können die Fähigkeiten API Reporting und API Monetization Einfluss auf den Betrieb eines APIs nehmen, indem Datenverkehrsprofile und Nutzungspläne automatisiert erstellt und damit API Instanzen (s. Kapitel 9.1) in der Betriebsumgebung dynamisch gesteuert werden.

Die Prozessstufen sind mit verschiedenen Zeitbegriffen verknüpft. Da einige Geschäftsfähigkeiten bei mehreren Prozessstufen einen Beitrag leisten, ordnet die API Architektur Bund diesen Prozessstufen verschiedene Zeitbegriffe zu (s. Tabelle 9), um mit diesen Zeitbegriffen die Leistungen voneinander abgrenzen zu können. Diese Zeitbegriffe kommen bei der Beschreibung der Geschäftsfähigkeiten und des Informationsmodells zur Anwendung (s. Kapitel 7.1.2 und 8.2).

Prozessstufe	Zeitbegriffe
Manage API Lifecycle	- Entwicklungszeit (Creation Time, CT) - Definitionszeit (Definition Time, DT) - Laufzeit (Run Time, RT)
Discover APIs	- Entwicklungszeit
Register Client	- Definitionszeit
Operate APIs	- Laufzeit
Analyse API Operation & Usage	- Laufzeit

Tabelle 9: Prozessstufen und Zeitbegriffe

[19] API Code und API Konfigurationen werden typischerweise eine einem Code Repository mit Versionskontrolle, während API Builds in einer Registry (Software-Store) geführt werden.

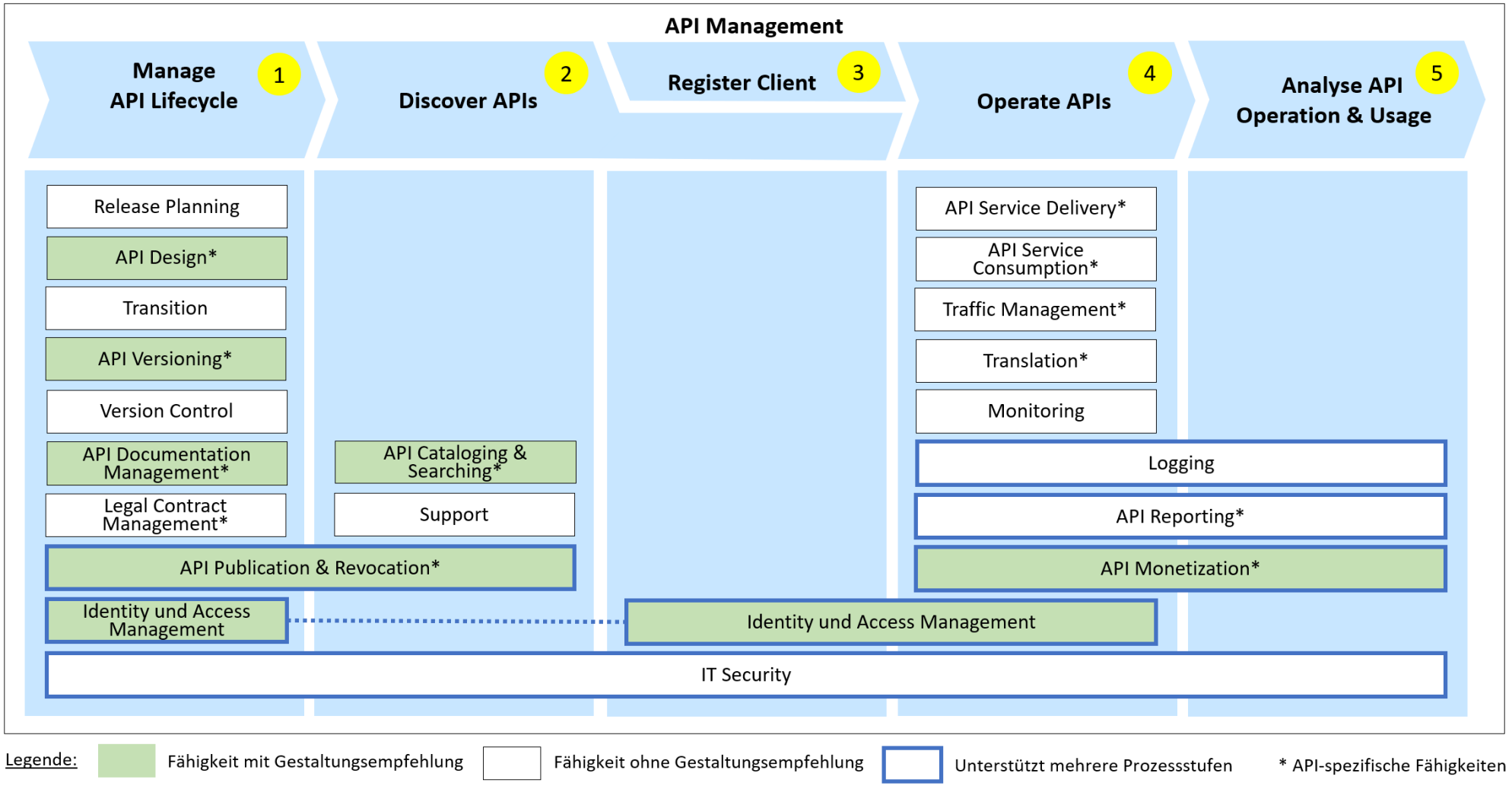

Abbildung 10: Landkarte der Geschäftsfähigkeiten

7.1.2 Beschreibung der Geschäftsfähigkeiten

7.1.2.1 Manage API Lifecycle

Fähigkeit	Definition
Release Planning	Im Rahmen des API Lebenszyklus müssen API Releases geplant werden. Die Fähigkeit umfasst das Sammeln und Bündeln von anstehenden Änderungen, das Festlegen der neuen Version mit begleitender Kommunikation (besonders relevant bei erzwungener Migration, s. Kapitel 9.5.5), das Bereitstellen von Entwicklungs- und Betriebsressourcen sowie das Vorbereiten der Supportorganisation.
API Design	In einer (API) Architektur kommt der Fähigkeit API Design erhöhte Aufmerksamkeit zu. Design-Elemente gibt es auf allen vier BDAT-Architektur-Ebenen, wobei die API Architektur Bund zur Unterstützung der Fähigkeit API Design einen strukturellen Rahmen mit Architekturbausteinen und Gestaltungsempfehlungen für APIs bietet. Eine zentrale Grundlage des API Designs ist die Anforderung, die für die digitale Behördenleistung relevanten gesetzlichen Rahmenbedingungen einzuhalten (Legal Compliance). Die Fähigkeit API Design umfasst auch die Fähigkeit, Standards zur API Spezifikation, API Dokumentation und zur automatisierten Generierung von Code einzusetzen (z.B. OpenAPI Specification für RESTful APIs[20] oder auf WSDL basierende Tools).
Transition	Bei der Fähigkeit Transition handelt es ich um die Fähigkeit, das gewählte API Design Ressourcen-schonend umzusetzen und zu testen. APIs können das Produkt einer Eigenentwicklung sein, oder aber sie werden anhand von Open Source Produkten oder kommerziellen Produkten umgesetzt. Entwicklungsrichtlinien vereinfachen dabei die Zusammenarbeit im Team und tragen zu einer hohen API Qualität bei. Das Testen dient dem Zweck, Aussagen zur Qualität der entwickelten APIs zu machen, d.h. ob die APIs den Anforderungen entsprechen und den Geschäftszweck erfüllen. Tests an den APIs können manuell oder automatisch durchgeführt werden, wobei im Rahmen der digitalen Transformation das automatisierte Testen mehr und mehr an Bedeutung gewinnt. In modernen Entwicklungs- und Betriebsumgebungen werden API Inkremente im Rahmen von DevOps-Prozessen kontinuierlich automatisch getestet, integriert und deployt (Continuous Integration / Continuous Deployment, CI/CD).
API Versioning	Für die Geschäftspartner hat die der API Architektur Bund zugrundeliegende API Versionierungsstrategie eine zentrale Bedeutung (s. Kapitel 9.5). Sie schafft Klarheit darüber, ob eine Änderung am API die API Integration unterbricht oder ob über Versionen hinweg ein konsistentes API Verhalten erwartet werden darf. Eine durchdachte Versionierungsstrategie wird dazu beitragen, dass Geschäftspartner den Schritt zum nächsten API Release vollziehen.
API Documentation Management	Die Fähigkeit API Documentation Management beinhaltet das Erstellen und Nachführen der Fachdokumentation inkl. Nutzungsvereinbarung, der API Metadaten und der API Dokumentation im Rahmen des API Lifecycle Managements. Sie umfasst auch die Nutzung von Standards zur automatischen Erzeugung von API Dokumentation (s. Fähigkeit API Design).
API Publication & Revocation	Nach erfolgreicher Entwicklung eines API Release wird der API Release in einem API Verzeichnis (s. Fähigkeit API Cataloging & Searching) publiziert, so dass sich die Geschäftspartner und die von ihnen beauftragten Leistungserbringer über die durch die Bundesverwaltung bereit gestellten digitalen Behördenleistungen informieren und für deren Nutzung den Zugriff beantragen können. Die Veröffentlichung umfasst alle relevanten Informationen (s. Kapitel 7.2), um Entscheide zum Bezug von APIs sicher fällen zu können. Die Publikation umfasst auch die Hinterlegung des API Builds und der API Konfiguration im Repository resp. der Registry. Am Ende des Lebenszyklus eines API Release wird dieser stillgelegt. Die Stilllegung (Revocation) eines API hat grosse Auswirkung auf Geschäftspartner, Entwicklungs- und Betriebsressourcen und erfordert daher wie die Publikation eine sorgfältige Planung.
Version Control	Die Fähigkeit Version Control (Versionskontrolle) bietet die Fähigkeit, Versionen eines API so zu verwalten, dass alle für die digitale Behördenleistung relevanten Dateien und Dokumente zusammen mit den daran vorgenommenen Änderungen und der Benutzerkennung chronologisch persistiert und wiederhergestellt werden können. API Entwickler arbeiten typischerweise lokal mit Kopien der Dateien und Dokumenten und gleichen ihre lokalen Versionen in regelmässigen Abständen mit dem Repository ab.
Legal Contract Management	Das Legal Contract Management umfasst die Erstellung und Verwaltung des Lizenzvertrags, der Nutzungsvereinbarung, der Service Level Objectives (SLO) und des Service Level Agreements (SLA). - Der Lizenzvertrag wird zwischen Anbieter und Geschäftspartner abgeschlossen und kommt bei kostenpflichtigen digitalen Behördenleistungen zum Tragen.

[20] https://swagger.io

Fähigkeit	Definition
	- Die Nutzungsvereinbarung regelt zwischen Anbieter und Geschäftspartner die Nutzung der dem API zugrundeliegenden Daten auf beiden Seiten. Bei Public APIs wird sie publiziert und bei Nutzung des APIs implizit vom Geschäftspartner akzeptiert. Bei Partner APIs werden sie entweder im Rahmen des Registrationsprozesses explizit akzeptiert, oder aber die auf das API zugreifende Identität gehört einem Nutzerkreis an, der bereits im Vorfeld der Registrierung pauschal die Nutzervereinbarung akzeptiert hat. - Das SLA wird zwischen dem API-betreibenden LE und der API-verantwortenden VE abgeschlossen, legt den Service Level und die SLOs fest sowie den vereinbarten Preis für den API Betrieb. SLOs werden via API Dokumentation publiziert und bei Nutzung des APIs vom Geschäftspartner akzeptiert.
Identity & Access Management (IAM)	Die Fähigkeit Identity & Access Management kommt bei der API Architektur Bund dann zum Tragen, wenn ein API die Registrierung einer Identität erfordert. Sie leistet bei den folgenden drei Prozessstufen einen Beitrag: - Manage API Lifecycle - Register Client - Operate APIs Zur Entwicklungszeit bietet IAM die Möglichkeit, Rollen zu definieren und zu erfassen. Zur Definitionszeit erlaubt IAM das Ausstellen von Identitäten, das Zuweisen von Rollen zu Identitäten (Zugriffsantrag), das Freigeben von Zugriffsanträgen bis hin zum Ausstellen von Identitätsnachweisen. Zur Laufzeit bietet IAM die Föderation von IAM Services und das Ausstellen von Secure Tokens an. IAM dient auch dem Zweck der Selbstregistrierung durch den Geschäftspartner.
IT Security	Die Fähigkeit IT Security umfasst alle Prozessstufen und soll den Schutz vor Cyberangriffen sicherstellen. Dazu gehört der Schutz von Computersystemen, Netzwerken und anderen cyber-physischen Systemen und somit sowohl der umfassende Schutz vor Unterbrechung oder Missbrauch der von ihnen bereitgestellten Dienste und Funktionen als auch der Schutz der Vertraulichkeit, Integrität und Verfügbarkeit der von ihnen verarbeiteten Daten. Es gelten die Rahmenbedingungen für die Sicherstellung von Informationssicherheit und Datenschutz (ISDS) im Bundesumfeld, welche durch bestehende übergreifende Rechtsgrundlagen und Vorgaben definiert sind. Dazu gehören unter anderem die Informatiksicherheitsvorgaben Bund des Nationalen Zentrums für Cybersicherheit (NCSC)[21]. Die übergeordnete Rechtsgrundlage des NCSC ist die Verordnung über den Schutz vor Cyberrisiken in der Bundesverwaltung (CyRV)[22]. Zusätzlich zu diesen gelten ggf. weitere Rahmenbedingungen auf Stufe Departement oder Amt.

Tabelle 10: Fähigkeiten im Bereich der Prozessstufe Manage API Lifecycle

7.1.2.2 Discover APIs

Fähigkeit	Definition
API Cataloging & Searching	Die Katalogisierung ist die Fähigkeit, API Metadaten zu den von der Bundesverwaltung bereitgestellten APIs nach den Bestimmungen des EMBaG zu sammeln, diese zu veröffentlichen und die Bezugsmöglichkeit von APIs anzubieten. Sie dient dem Zweck, die Geschäftspartner und die von ihnen beauftragten Leistungserbringer mit allen relevanten Informationen zu versorgen, um Entscheide zum Bezug von APIs fällen zu können. Das API Verzeichnis bietet der Zielgruppe der API Architektur Bund die Möglichkeit, dieses nach bestimmten APIs zu durchsuchen. Die Suchkriterien sind die Felder der API Metadaten.
Support	Die Fähigkeit Support beinhaltet die Unterstützung der API Nutzer beim Sammeln von Informationen für Bezugsentscheide, beim Entwickeln von Clients, beim Abwickeln der Client Registration sowie in allen Belangen, bei denen die Bordmittel der bereitgestellten Systeme und/oder die Online-Dokumentation nicht ausreichen (s. Kapitel 7.2).
API Publication & Revocation	S. Beschreibung in Kapitel 7.1.2.1
IT Security	S. Beschreibung in Kapitel 7.1.2.1

Tabelle 11: Fähigkeiten im Bereich der Prozessstufe Discover APIs

[21] https://www.ncsc.admin.ch/ncsc/de/home/dokumentation/sicherheitsvorgaben-bund.html
[22] https://www.fedlex.admin.ch/eli/cc/2020/416/de

7.1.2.3 Register Client

Fähigkeit	Definition
Identity & Access Management (IAM)	Die Fähigkeit Identity & Access Management leistet bei den folgenden drei Prozessstufen einen Beitrag: - Manage API Lifecycle - Register Client - Operate APIs S. Beschreibung in Kapitel 7.1.2.1
IT Security	S. Beschreibung in Kapitel 7.1.2.1

Tabelle 12: Fähigkeiten im Bereich der Prozessstufe Register Client

7.1.2.4 Operate APIs

Fähigkeit	Definition
API Service Delivery	API Service Delivery stellt die Fähigkeit dar, eine digitale Behördenleistung gegen aussen anzubieten. Dazu sind die API Instanzen sowie eine API Infrastruktur in mehreren Laufzeitumgebungen zu betreiben. Im Bundesumfeld sind die Laufzeitumgebungen typischerweise unterteilt in eine Referenz-, eine Abnahme- und eine Produktionsumgebung. Referenz- und Abnahmeumgebung können jedoch auch zu einer Laufzeitumgebung vereint sein. Eine Laufzeitumgebung kann in der Cloud, On-Premise oder Hybrid aufgesetzt sein.
API Service Consumption	API Service Consumption stellt die Fähigkeit dar, eine digitale Behördenleistung zu konsumieren. Dazu sind die API Clients selbst sowie eine API Client Infrastruktur in mehreren Laufzeitumgebungen zu betreiben (s. Fähigkeit API Service Delivery zu Informationen betreffend den im Bundesumfeld typischerweise betriebenen Laufzeitumgebungen). API Clients werden aber auch ausserhalb des Bundesumfelds eingesetzt. Eine Laufzeitumgebung kann in der Cloud, On-Premise oder Hybrid aufgesetzt sein, oder aber ein Mobile Device sein.
Identity & Access Management (IAM)	Die Fähigkeit Identity & Access Management leistet bei den folgenden drei Prozessstufen einen Beitrag: - Manage API Lifecycle - Register Client - Operate APIs S. Beschreibung in Kapitel 7.1.2.1
Traffic Management	Die Fähigkeit Traffic Management deckt folgende Aufgaben ab: - Lastausgleich: Gleichmässiges verteilen von API Zugriffen auf API Instanzen am selben Standort oder anbieten von auf mehrere Standorte verteilte, zu einem Fachservice zugeordnete API Gateways mit individuellen Endpunkten[23]; - Priorisierung: Sicherstellen, dass Zugriffe auf APIs nach einer definierten Reihenfolge priorisiert werden; - Begrenzung/Drosselung: Sicherstellen von Quotas pro Zeiteinheit für Zugriffe auf einen API Endpunkt - Skalierung: Automatische Bereitstellung oder Entzug von Ressourcen im laufenden Betrieb durch Starten/Stoppen von zusätzlichen Instanzen bzw. Containern. Die Wahrnehmung dieser Aufgaben wirkt sich auch positiv auf die Latenzzeiten aus. Insgesamt stellen sie die Performanz des APIs sicher.
Translation	Die Fähigkeit Translation ist die Fähigkeit, API Protokolle zu übersetzen. Sie schliesst das Konvertieren von Datenformaten (z.B. JSON<>XML) und Verarbeiten von Nutzlasten (z.B. Datenfilterung) explizit aus.
Logging	Die Fähigkeit Logging leistet bei den folgenden zwei Prozessstufen einen Beitrag. - Operate APIs - Analyse API Operation & Usage Die Fähigkeit Logging erlaubt, betriebliche[24] und fachliche Ereignisse (Synonym: Events) zu loggen, die mit dem Betrieb von Fachservices und APIs in Verbindung stehen. Die Ereignisse können somit sowohl vom Fachservice als auch vom API Gateway erzeugt werden. Die Geschäftsfähigkeit Logging deckt auch die Protokollierung[25] gemäss Art. 10 der

[23] Ist ein Fachservice-Cluster über mehrere Standorte verteilt, so können sich bei schreibenden API Zugriffen via über mehrere Standorte verteilte API Gateways Fragen der Datenspiegelung und der Transaktionssicherheit stellen, welche bei der Gestaltung der Lösungsarchitektur beantwortet werden müssen.
[24] Betriebliche Ereignisse sind SLA-relevante Ausnahme-Ereignisse.
[25] Synonym: Audit-Trail

Fähigkeit	Definition
	Verordnung zum Bundesgesetz über den Datenschutz (VDSG)[26] ab, welche ausschliesslich im Fachservice stattfindet. Zu den fachlichen Ereignissen zählen insbesondere sowohl API Zugriffe der Öffentlichkeit sowie durch die IAM Services kontrollierte, akzeptierte und abgewiesene Zugriffe auf API Endpunkte als auch Ereignisse, die im Rahmen der Protokollierung erzeugt werden. Bei APIs, die eine Registrierung erfordern, ist bei API Aufrufen neben anderen relevanten Log-Parametern insbesondere die aufrufende Identität zu loggen. Es müssen in jedem Fall die geltenden Datenschutz-Bestimmungen eingehalten werden. Daher werden der Protokollierung dienende Log-Daten typischerweise in dedizierten Logging-Systemen verwaltet. Die Fähigkeit Logging bietet die Grundlage für die Fähigkeiten Monitoring und Reporting. Der Prozessstufe Operate APIs sind die Unterfähigkeiten Erkennen, Filtern und Senden von Ereignissen zugeordnet, der Prozessstufe Analyse API Operation & Usage die Unterfähigkeiten Empfangen und Persistieren von Ereignissen. Die Fähigkeit Logging ermöglicht, das Logging der API Infrastruktur in die generische Logging-Infrastruktur der VE zu integrieren.
Monitoring	Monitoring stellt die Fähigkeit dar, auf betriebliche Ereignisse zu reagieren und an Zielsysteme und/oder Akteure weiterzuleiten. Das Monitoring benachrichtigt die Betriebsorganisation über Betriebsereignisse. Die Fähigkeit Monitoring ermöglicht, das Monitoring der API Infrastruktur in die generische Monitoring-Infrastruktur der VE zu integrieren.
IT Security	S. Beschreibung in Kapitel 7.1.2.1

Tabelle 13: Fähigkeiten im Bereich der Prozessstufe Operate APIs

7.1.2.5 Analyse API Operation & Usage

Fähigkeit	Definition
API Reporting	Die Fähigkeit Reporting umfasst die Fähigkeit, die relevanten betrieblichen und fachlichen Ereignisse zu sammeln, auszuwerten und aufzubereiten. Die Fähigkeit Reporting kann auch künstliche Intelligenz beinhalten und so z.B. Ereignis-Mustererkennung betreiben und Optimierungsmassnahmen vorschlagen. In Verbindung mit Fähigkeiten aus der Prozessstufe Operate API kann z.B. automatische Skalierung erfolgen. Zielgruppe der Reports sind der API-betreibende LE und die API-verantwortende VE. Die Reports versetzen: - API Product Owner der VE in die Lage, den Business Value zu beziffern und anhand von KPIs das API weiterzuentwickeln und am Geschäftspartner auszurichten; - den API-betreibenden LE in die Lage, den Grad der SLO-Erfüllung auszuweisen.
API Monetization	Monetization umfasst die Fähigkeit, aus dem Betrieb von APIs Einnahmen zu generieren (s. Kapitel 7.3). Dies beinhaltet nicht nur die Verbindung von API Konsum mit Preismodellen zwecks Ermittlung der Einnahmen, sondern auch die Erstellung von Nutzungsplänen mit den zugehörigen Preismodellen bis hin zur Verbindung mit Fähigkeiten aus der Prozessstufe Operate API, um z.B. Quotas für Zugriffe durchzusetzen, wenn Leistungsbezugslimiten überschritten sind. Die Fähigkeit Monetization setzt dabei auf den im Rahmen der Fähigkeit Reporting erstellten Berichte und berechneten KPIs auf.
Logging	Die Fähigkeit Logging leistet bei den folgenden zwei Prozessstufen einen Beitrag. - Operate APIs - Analyse API Operation & Usage S. Beschreibung in Kapitel 7.1.2.4
IT Security	S. Beschreibung in Kapitel 7.1.2.1

Tabelle 14: Fähigkeiten im Bereich der Prozessstufe Analyse API Operation & Usage

7.2 Gestaltungsempfehlung Transparenz

Die Gestaltungsempfehlung Transparenz beinhaltet Empfehlungen zu den folgenden drei Geschäftsfähigkeiten:

- API Documentation Management
- API Cataloging & Searching
- API Publication & Revocation

[26] https://www.fedlex.admin.ch/eli/cc/1993/1962_1962_1962/de

Die Inhalte zu diesen drei Geschäftsfähigkeiten werden an dieser Stelle unter der Gestaltungsempfehlung Transparenz zusammengefasst, da diese nur in Kombination für das Zielpublikum der API Architektur Bund einen Nutzen stiften.

7.2.1 API Documentation Management

Die API Architektur Bund empfiehlt, die Fähigkeit API Documentation Management wie folgt auszuprägen:

- Die Artefakte der **Dokumentation** der digitalen Behördenleistung (s. Informationsmodell in Abbildung 15) werden **erstellt** und fortlaufend **gepflegt**. Der API Product Owner (s. Kapitel 7.4.1) ist für die Erstellung und Pflege der API Metadaten und der API Dokumentation verantwortlich. Die Fachdokumentation wird von der zugehörigen Fachabteilung der VE verantwortet.
- Die **API Dokumentation** umfasst alle Informationen, welche notwendig sind, um einen Bezugsentscheid fällen und das API aus einem API Client heraus korrekt ansprechen zu können. Insbesondere enthält die API Dokumentation auch die API Spezifikation, die API Release Notes und die SLOs. Für die API Spezifikation wird die englische Sprache gewählt.
- Die **Fachdokumentation** ist die Voraussetzung für die API Dokumentation. Die API Dokumentation verweist auf die Fachdokumentation, um zu beschreiben, welche Aspekte der Fachlichkeit ein bestimmter API Release abdeckt. Die Fachdokumentation beinhaltet auch die Nutzungsvereinbarung. Alle Teile der Fachdokumentation, welche den Geschäftspartnern und Endbenutzern zugänglich gemacht werden, sind je nach Bedarf in den lokalen Sprachen bereitzustellen. Dazu gehören auch sprachliche, fachliche Inhalte, welche das API überträgt.
- Die **API Metadaten** stellen eine strukturierte, maschinell verarbeitbare Beschreibung des API Profils dar. Sie dienen dem Zweck, Informationen über die Existenz und die Eigenschaften von APIs im Bundesumfeld sowie die anbietenden VE in **API Verzeichnissen online** zu veröffentlichen. Die Veröffentlichung von API Metadaten in einem API Verzeichnis folgt den Bestimmungen des Bundesgesetzes über den Einsatz elektronischer Mittel zur Erfüllung von Behördenaufgaben (EMBaG).
- Auch die **Fachdokumentation** und **API Dokumentation** werden **online** verwaltet und bereitgestellt. Eine vom Anbieter verwaltete **Landing Page** (in Standard eCH-0200 als RDF Property dcat:landingPage[27] definiert, zu RDF s. Anhang Kapitel 10.4) stellt diese dezentral geführte Online-Dokumentation zur Verfügung.
- Die API Dokumentation wird so weit wie möglich **automatisiert** erstellt. Dazu werden anhand Programmiersprachen-unabhängiger Schnittstellenbeschreibungssprachen (Interface Definition Language, IDL) Schnittstellenspezifikationen erstellt, aus denen automatisiert Dokumentationen, Server-/Client-Code-Vorlagen sowie Test Code und/oder Test Cases erstellt werden können. Etablierte Standards zur Spezifikation von RESTful APIs sind die OpenAPI Specification (OAS)[28] oder AsyncAPI[29] für APIs mit asynchronen, (ereignisorientierten) Message Exchange Patterns (für Erläuterungen zu MEPs s. Kapitel 8.3.2).

7.2.2 API Cataloging & Searching

7.2.2.1 API Metadaten

Die Fähigkeit API Cataloging & Searching umfasst das Bereitstellen der API Metadaten in standardisierter Form. Die API Architektur Bund baut bei der Bereitstellung der API Metadaten auf den Standards «Interoperability solutions for public administrations, businesses and citizens»[30] (ISA) und eCH auf. Diese Standards dienen dem Zweck, API Metadaten anhand eines Metadaten-Modells einheitlich zu beschreiben, so dass die API Metadaten zwischen API Verzeichnissen ausgetauscht werden können.

Die API Architektur Bund unterscheidet zur Beschreibung von API Metadaten die in Tabelle 15 beschriebenen Sichten auf ein API Verzeichnis. Das API Verzeichnis enthält dabei sowohl einen Daten- wie auch einen Leistungskatalog. Die beiden Kataloge enthalten Einträge, die einem katalogspezifischen API Metadaten-Standard folgen. Tabelle 16 beschreibt die beiden API Metadaten-Standards.

[27] https://www.w3.org/TR/vocab-dcat-3
[28] https://swagger.io
[29] https://www.asyncapi.com
[30] Ein Standard der Europäischen Kommission.

Sichten zur Unterscheidung von API Metadaten	Beschreibung	Empfohlener API Metadaten-Standard
Datensicht (Datasets)	Geschäftspartner, die insbesondere Daten der Bundesverwaltung beziehen wollen, suchen die APIs über den Datenkatalog. Typischerweise werden die Daten von API Clients lediglich gelesen.	eCH-0200 DCAT-Anwendungsprofil für Datenportale in der Schweiz 4[31, 32]
Leistungssicht (Public Services)	Geschäftspartner, die eine digitale Leistung der Bundesverwaltung beziehen bzw. «einen Geschäftsfall» über das API abwickeln wollen, suchen die APIs über den Leistungskatalog für digitale Behördenleistungen. Typischerweise erlauben API Clients, Daten zu erstellen, zu lesen, zu verändern und zu löschen.	ISA-Standard Core Public Service Vocabulary Application Profile (CPSV-AP)[33]

Tabelle 15: Sichten zur Unterscheidung der API Metadaten

Digitale Behördenleistungen und ihre APIs können entweder nur im Datenkatalog, nur im Leistungskatalog, oder aber in beiden Katalogen aufgeführt sein. Dies hängt vom Charakter der digitalen Behördenleistung ab und korreliert nicht mit dem API Typ (Public API und Partner API). Die Datenmodelle der beiden API Metadaten-Standards haben eine Schnittmenge und können damit auch vereint werden.

Standard	Beschreibung
eCH-0200 DCAT-Anwendungsprofil für Datenportale in der Schweiz 4	- Ein eCH-Standard, der auf dem ISA-Standard «DCAT Application Profile for data portals in Europe (DCAT-AP)»[34] basiert - eCH-0200 konkretisiert DCAT-AP und erweitert das Datenmodell um einige Attribute.
ISA-Standard Core Public Service Vocabulary Application Profile (CPSV-AP)	- Ein API Metadaten-Standard für die Standardisierung von Metadaten von digitalen Behördenleistungen auf allen föderalen Verwaltungsstufen - Es existiert (noch) kein entsprechender eCH-Standard.

Tabelle 16: API Metadaten-Standards

Beide API Metadaten-Standards beinhalten ein eigenes API Metadaten-Modell, dessen Klassen und Attribute mithilfe eines RDF-Vokabulars beschrieben sind (s. Anhang Kapitel 10.4). Das RDF-Vokabular definiert die Bedeutung (Semantik) der Klassen und Felder sowie deren Beziehung untereinander. Die Nutzung dieser API-Metadaten-Standards erlaubt es, API Metadaten zwischen API Verzeichnissen auszutauschen oder auch API Metadaten zu durchsuchen, welche über föderierte API Verzeichnisse verteilt sind.

7.2.2.2 Zentrales API Verzeichnis und Landing Pages

Die API Architektur Bund sieht vor, im Bundesumfeld ein zentrales API Verzeichnis zu führen. Somit wird das API Verzeichnis bei einer Publikation einer digitalen Behördenleistung um die API Metadaten ergänzt. Bei der Stilllegung einer digitalen Behördenleistung werden die API Metadaten aus dem API Verzeichnis entfernt. API Metadaten enthalten gemäss den o.g. Standards keine API Versionierungsinformationen. Stattdessen sind es die in den API Metadaten referenzierten, dezentral gehaltenen Landing Pages mit den API Dokumentationen, welche die publizierten, versionierten API Releases beschreiben. Das API Verzeichnis ist öffentlich zugänglich. Die Veröffentlichung von API Metadaten im API Verzeichnis folgt den Bestimmungen des EMBaG.

Werden zu einem API aufgrund der EMBaG-Bestimmungen keine API Metadaten publiziert, so empfiehlt die API Architektur Bund den Anbietern, eine Landing Page zu betreiben, welche nur einem kontrollierten Nutzerkreis zugänglich gemacht wird. Der Zugriff auf die Landing Pages unterliegt in diesem Fall strengen Auflagen, da die API Dokumentation potenziell sensitive Informationen über eine digitale Behördenleistung preisgibt. Sprechen wirtschaftliche Gründe gegen den Betrieb einer Landing Page, so kann der Anbieter darauf verzichten.

[31] https://www.ech.ch/index.php/de/standards/60609 (dieser Standard wird derzeit - Stand 30.11.2021 - überarbeitet und dem ISA-Standard angeglichen)
[32] Das Portal https://opendata.swiss für frei zugängliche Daten der Schweizer Behörden (Open Government Data, OGD) nutzt diesen API Metadaten-Standard.
[33] https://ec.europa.eu/isa2/solutions/core-public-service-vocabulary-application-profile-cpsv-ap_en
[34] https://ec.europa.eu/isa2/solutions/dcat-application-profile-data-portals-europe_en

Der Zugriff auf die Landing Pages durch Geschäftspartner und deren beauftragte Leistungserbringer erfolgt über einen der beiden von der BK DTI geführten IKT-Standarddienste des IAM Bund (s. eIAM-Glossar[35] «Was ist das SSO-Portal EJPD und wie ist der Zusammenhang mit eIAM?»:

- eIAM[36], dem zentralen Zugriffs- und Berechtigungssystem der Bundesverwaltung für Webapplikationen, und native Mobile Apps, betrieben durch das Bundesamt für Informatik und Telekommunikation (BIT)
- SSO-Portal[37], das Single-Sign-On-Portal EJPD für Applikationen der Justizbehörden und Affiliierter, betrieben durch das Informatik Service Center des Eidg. Justiz und Polizei Departements (ISC-EJPD)

In den Landing Pages ist im Verbund mit eIAM oder dem SSO-Portal ein Rollenkonzept implementiert, so dass die unter Kapitel 7.4 beschriebenen Rollen auf Seite des Anbieters und des Geschäftspartners ihre Rollen wahrnehmen können.

7.2.2.3 Föderiertes API Verzeichnis

Der Anbieter kann bei Bedarf auch sein eigenes, dezentral geführtes API Verzeichnis betreiben. Die API Architektur Bund empfiehlt in diesem Fall, dieses dezentral geführte API Verzeichnis in das zentrale API Verzeichnis einzubinden. Durch das Einbinden entsteht ein föderiertes API Verzeichnis, welches der Benutzende als ein zentrales API Verzeichnis wahrnimmt.

Die Suche nach digitalen Behördenleistungen in einem API Verzeichnis basiert auf den API Metadaten. Jedes Datenfeld der API Metadaten kann dabei als Suchkriterium dienen. Dabei muss sichergestellt sein, dass eine in einem föderierten API Verzeichnis abgesetzte Suche nach einer digitalen Behördenleistung im Suchraum aller API Metadaten des Bundesumfelds durchgeführt wird. Dies kann auf zwei Arten erreicht werden:

- Das zentrale API Verzeichnis erhält periodisch Kopien der API Metadaten der dezentral geführten API Verzeichnisse.
- Ausgehend von einem definierten Master-Verzeichnis wird die Suche über alle dezentral geführten API Verzeichnisse durchgeführt.

7.2.2.4 Verweis auf web-basierte Behördenleistungsportale

Das API Verzeichnis stellt nicht nur Informationen zur Nutzung von digitalen Behördenleistungen bereit, sondern auch Informationen zu web-basierten Behördenleistungsportalen, welche ebenfalls digitale Behördenleistungen anbieten. Dies ist dadurch sichergestellt, dass das Datenmodell des API Metadaten-Standards CPSV-AP die Möglichkeit bietet, die URL des Behördenleistungsportals zu hinterlegen[38]. Im Weiteren verweist die API Architektur Bund für die Katalogisierung von web-basierten Behördenleistungsportalen auf die föderale Portalarchitektur (s. Kapitel 2.1.2).

7.2.3 API Publication & Revocation

Die API Architektur Bund empfiehlt, die Fähigkeit API Publication & Revocation wie folgt zu gestalten:

- Der API Product Owner veröffentlicht die API Metadaten im API Verzeichnis und die zum API Release zugehörige Online-Dokumentation (s. Kapitel 7.2.1) auf der Landing Page. Der API Product Owner kann diese Aufgabe an einen sog. Publisher delegieren. Publisher sind natürliche Person, welche die VE des API Product Owners vertreten.
- Der API Product Owner stellt durch Rollenzuweisungen in eIAM sicher, so dass alle natürliche Personen, welche Zugriff auf die Landing Page benötigen, diesen Zugriff Rollen-konform erhalten (s. Kapitel 7.4).
- Wird ein API Release publiziert oder stillgelegt, so verantwortet der API Product Owner folgende Aufgaben:
    - Publizieren, anpassen und entfernen der API Metadaten. Die API Metadaten sind allgemein gehalten und referenzieren daher keine API Versionen (s. Kapitel 7.2.2.2). Sie können aber aufgrund eines publizierten API Releases Änderungen erfahren.

[35] https://www.eiam.admin.ch/pages/f!eiamglossary!pub_de.html?c=f!eiamglossary!pub&l=de
[36] https://www.eiam.admin.ch
[37] https://www.isc-ejpd.admin.ch/isc/de/home/dienstleistungen/sso-portal.html
[38] Die optionale Klasse *Channel* repräsentiert zum API alternative Nutzungskanäle, wie z.B. ein Behördenleistungsportal. Beim Attribut *Identifier* kann die URL hinterlegt werden.

- Publizieren, anpassen und entfernen der zum API Release zugehörigen Online-Dokumentation auf der Landing Page.
- Bei der Publikation, sicherstellen des Betriebs von API Instanzen durch abschliessen oder anpassen von Nutzungsvereinbarungen. Bei der Stilllegung, beenden der Betriebsvereinbarungen.
- Bei der Publikation, Deployment von API Instanzen auf der produktiven Umgebung. Bei der Stilllegung, entfernen von API Instanzen von der produktiven Umgebung.
- Aufbau, Betrieb und Abbau einer Support-Organisation für die Unterstützung der Rollen API Client Entwickler und API Client Nutzer auf Seite des Geschäftspartners (s. Kapitel 7.4.2).

7.3 Gestaltungsempfehlung API Monetization

7.3.1 Rechtlicher Rahmen

Der Bundesrat genehmigte am 30. November 2018 die Open Government Data (OGD) Strategie Schweiz 2019-2023[39]. Mit der OGD Strategie werden offene und frei nutzbare Verwaltungsdaten der Bundesverwaltung grundsätzlich gebührenfrei zur Verfügung gestellt.

Bei Einzelfällen von APIs, in denen hohe Datenvolumina transferiert werden und/oder eine hohe Servicequalität erwartet wird, kann eine Übertragung der Kosten auf den Geschäftspartner stattfinden. Voraussetzung für eine solche Übertragung ist eine bundesrätliche Verordnung[40]. Ist diese Bedingung erfüllt, so dürfen die VE, die den Verordnungen unterliegen, für die Bereitstellung von digitalen Behördenleistungen nach den Regelungen der Verordnungen Gebühren erheben.

7.3.2 Monetarisierungsmodelle

Bei der Fähigkeit der Monetarisierung unterscheidet die API Architektur Bund grundsätzlich zwei API Monetarisierungsmodelle: Das indirekte und das direkte API Monetarisierungsmodell[41] (s. Tabelle 17).

API Monetarisierungsmodell	Definition
Indirekt	Ein indirektes API Monetarisierungsmodell basiert auf der Grundlage, dass die API Interaktion zwischen Anbieter und Geschäftspartner auf beiden Seiten Wert generiert. Die Kosten für Entwicklung und Betrieb des APIs trägt jedoch (zumindest in der Anfangsphase des API Betriebs) vollumfänglich der Anbieter. Die Monetarisierung entsteht durch Kosteneinsparungen als Folge der Bereitstellung eines APIs beim Erbringen einer Behördenleistung.
Direkt	Ein direktes API Monetarisierungsmodell basiert darauf, dass primär der Geschäftspartner von der Bereitstellung eines APIs profitiert und daher vollumfänglich für die Kosten von Entwicklung und Betrieb des APIs aufkommt.

Tabelle 17: Direktes und indirektes API Monetarisierungsmodell

Das Entwickeln der Fähigkeit der Monetarisierung ist ein stufenweiser Prozess. Die erste Stufe stellt die Phase nach API Betriebsaufnahme dar, in der der Anbieter mit dem indirekten API Monetarisierungsmodell Erfahrungen sammelt, dabei Nutzungsdaten auswertet und die Einhaltung von SLOs überwacht. Die nächste Stufe besteht in einem Mix des indirekten und direkten Modells, in dem die Kosten der Leistungserbringung teilweise an den Geschäftspartner übertragen werden. Dieser Mix kann Stufe für Stufe in Richtung der direkten Monetarisierung entwickelt werden. Sobald der Umsatz aus dem Vertrieb des APIs vollumfänglich die Gestehungskosten für Entwicklung und Betrieb des APIs abdeckt, ist die Stufe der direkten API Monetarisierung erreicht. Welche Stufe als Zielstufe definiert wird, entscheidet die VE vor dem Hintergrund der Frage nach der geteilten Wertschöpfung (s. Kapitel 7.3.3) und der gegebenen Verordnungen.

[39] https://www.bfs.admin.ch/bfs/de/home/dienstleistungen/ogd.html
[40] Beispiel einer solchen Bundesratsverordnung ist die Verordnung SR 510.620 über Geoinformation, Geoinformationsverordnung, GeoIV.
[41] Vgl. https://www.gartner.com/en/documents/3903563/choose-the-right-api-monetization-and-pricing-model (Artikel hinter Bezahlschranke)

7.3.3 Preismodelle

Bei der API Monetarisierung können verschiedene Preismodelle eingesetzt werden. Tabelle 18 zeigt eine Übersicht über wählbare Preismodelle[42]. Die Preismodelle können auch kombiniert werden.

Preismodell	Beschreibung
Kostenfreie Nutzung	- Die kostenfreie Nutzung wird oft für öffentliche Daten verwendet. Dabei kann auch vorgesehen sein, mit den kostenfrei angebotenen digitalen Behördenleistungen den Verkauf von verwandten digitalen Behördenleistungen anzuregen (Cross-Selling). - Die Nutzungsvereinbarung muss in jedem Fall eingehalten werden. Bei Überbeanspruchung kann diese eine vorbehaltliche Drosselung der Zugriff pro Zeiteinheit enthalten.
Bezahlung pro API Zugriff	- Pro API Zugriff wird ein fixer Einheitspreis verrechnet.
Abonnement	- Geschäftspartner abonnieren auf Monatsbasis ein vordefiniertes Nutzungspaket, mit Warnungen und/oder Überschreitungsgebühren für den Fall der Limitenüberschreitung. - Das Abonnement kann auf einer Anzahl API Zugriffe und/oder auf Datenvolumen basieren.
Staffelung	- Die Nutzung ist frei bis zu einem gewissen Limit. Danach ist die Nutzung (gestaffelt) kostenpflichtig. Es kann mehrere Staffelungsstufen geben. - Die Limite kann auf einer Anzahl API Zugriffe und/oder auf Datenvolumen basieren. - Bei Kombination mit dem Preismodell «Bezahlung pro API Zugriff» können bei Erreichen der nächsten Stufe tiefere Einheitspreise wirksam werden. - Das Modell erfordert ein Cockpit, bei dem der Geschäftspartner seinen Konsum pro Zeiteinheit einsehen kann.
Aufteilung der Einnahmen	- Die Einnahmen, die vom Endbenutzer stammen, werden zwischen Anbieter und Geschäftspartner aufgeteilt. - Wenn der Endbenutzer für die Nutzung der Client Application bezahlt, dann tritt der Geschäftspartner einen Teil der Einnahmen an den Anbieter ab. - Wenn der Endbenutzer für die Nutzung der digitalen Behördenleistung bezahlt, dann tritt der Anbieter einen Teil der Einnahmen an den Geschäftspartner ab.

Tabelle 18: Preismodelle für API Monetarisierung

7.3.4 Voraussetzungen

API Monetarisierung stellt erhöhte Anforderungen an den API Betrieb, da dem bezahlten Preis für die API Nutzung eine garantierte Leistung gegenübersteht. Die kommunizierten SLOs sind in jedem Fall einzuhalten. Ist eine VE bestrebt, sich in Richtung der direkten API Monetarisierung zu bewegen, dann sind im Rahmen von API Entwicklung und Betrieb folgende Voraussetzungen zu erfüllen[43]:

- **APIs sind so konzipiert, dass die Akzeptanz der Geschäftspartner gewährleistet ist.**
  Dem API Design kommt hier eine besondere Rolle zu, da APIs bei der Abwicklung von digitalen Behördenleistungen geschäftspartnerorientiert und anwendungsfallorientiert durchgängige Prozesse unterstützen sollen. Der Fokus liegt auf der Benutzerfreundlichkeit der APIs durch einen hohen Automatisierungsgrad.
- **Für API Entwickler ist der Zugang zu APIs ohne Hindernisse möglich.**
  Der Zugang zu APIs für Entwicklungszwecke ist für Entwickler so einfach wie möglich. Dies wird erreicht durch Zugriff auf die API Dokumentation, einfache Registrierung von Test-Clients in Sandbox-Umgebungen mit API Testversionen und API Monetarisierungsmodelle, welche die API Nutzung fördern.
- **Die Leistungen von APIs sind klar definiert und können erfüllt werden.**
  API Monetarisierung erfordern SLOs, auf die jederzeit Verlass ist.
- **Für die API Nutzung sind Supportleistungen bereitzustellen.**
  Über den gesamten Lebenszyklus eines APIs wird sichergestellt, dass Geschäftspartner bei der API Nutzung über geeignete Kommunikationskanäle individuell unterstützt werden, insbesondere auch dann,

[42] Vgl. https://www.gartner.com/en/documents/3903563/choose-the-right-api-monetization-and-pricing-model (Artikel hinter Bezahlschranke)
[43] Dito

wenn eine Migration zum nächsten API Release aufgrund eines anstehenden Rückwärtskompatibilitätsbruchs erzwungen wird.

API Infrastrukturen, welche die Anforderung der direkten API Monetarisierung erfüllen sollen, müssen folgende Funktionen anbieten:

- Erstellen von Nutzungsplänen für APIs sowie nutzungsbasierte Zuweisung von Preislisten
- Bereitstellen von Self-Service-Funktionen für die Client Registrierung und die Wahl des gewünschten Preismodells
- Überwachen der API Nutzung pro Kunde in einem Cockpit
- Einschränken der API Nutzung bei Überschreitung der in den Preismodellen festgelegten Limiten
- Protokollieren von API Zugriffen sowie Änderungen an Preismodellen und SLAs
- Abrechnen der API Nutzung auf Basis der festgelegten Preismodelle

7.4 Rollenbeschreibung

In den folgenden Unterkapiteln werden die Rollen *API Product Owner*, *API Architekt*, *API Entwickler* und *API Betreiber* beschrieben und den Bereichen *API Infrastruktur*, *API Instanz* und *API Client* zugeordnet. Der Bereich API Infrastruktur umfasst alle übergeordneten Dienste und Systemkomponenten (technische Plattform), welche den VE für die Führung und Ausführung ihrer API Instanzen zur Verfügung gestellt werden. Die Rollen im Bereich Daten (*Data Owner* und *Lokaler Datenhalter*) werden hier nicht beschrieben, da sie im Rollenmodell des Programms Nationale Datenbewirtschaftung (NaDB)[44] beschrieben und nicht API-spezifisch sind.

Gemäss Rollenmodell entscheidet der Data Owner beispielsweise, basierend auf der gesetzlichen Grundlage, über eine Publikation der Daten als Open Government Data, d.h. in kostenloser, maschinenlesbarer und offener Form, während der API Product Owner für die Umsetzung dieses Entscheids verantwortlich ist. Der Lokale Datenhalter ist u.a. verantwortlich, dass die API Metadaten korrekt und vollständig im API Verzeichnis abgebildet sind. Insofern arbeiten Data Owner und Lokaler Datenhalter eng mit dem API Product Owner zusammen.

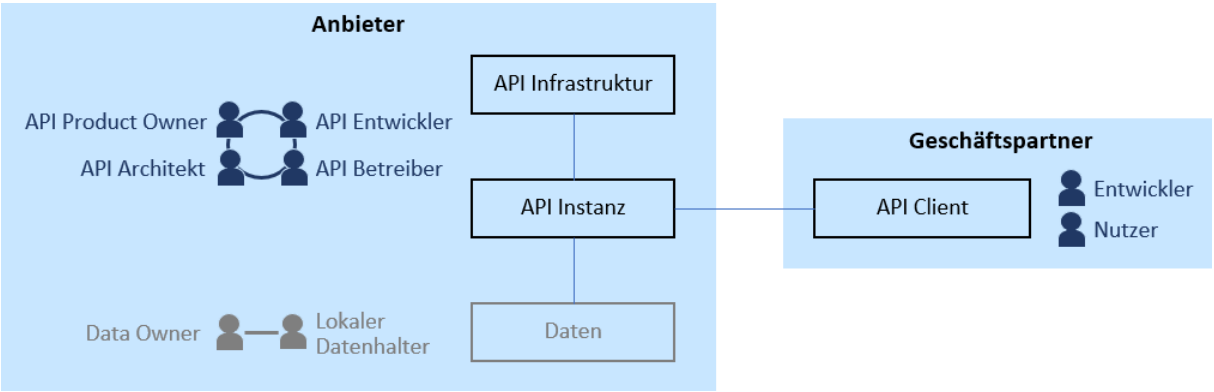

Abbildung 11: API-spezifische Rollen

Diese Rollen können auf verschiedenen Verwaltungsebenen besetzt werden. So kann es z.B. einen API Entwickler in einer VE oder auf Stufe Departement, z.B. beim departementalen Leistungserbringer geben. Die API-spezifischen Rollen sind vom Umfang her eher zu klein, um dafür eine eigene Stelle zu schaffen. Zudem sind die Aufgaben darin sehr ähnlich wie bei jenen für Applikationsmanagement. Eine mögliche und effiziente Rollenbesetzung wäre demnach die Erweiterung der Stellen für Applikationsmanagement um entsprechende API-spezifische Rollen (z.B. ein Anwendungsverantwortlicher übernimmt zusätzlich die Rolle des API Product Owners).

7.4.1 Rollen des Anbieters

Die Rollen im Bereich API Infrastruktur sind typischerweise auf Stufe Departement oder Stufe Bund angesiedelt, da nicht jede VE eine eigene API Infrastruktur aufbauen soll. Die Rollen im Bereich API Instanz haben einen fachlichen und applikatorischen Bezug und sind demnach hauptsächlich auf Stufe VE angesiedelt. Grundsätzlich können diese Rollen für einzelne API Instanzen besetzt werden. Bei API Architekt, API Entwickler und API Betreiber ist es je nach Grösse der VE sinnvoller, die Rolle für sämtliche API Instanzen einmal in der VE zu besetzen.

[44] https://www.bfs.admin.ch/bfs/de/home/nadb/nadb.assetdetail.14965606.html

Rolle	Verantwortung, Aufgaben und Kompetenzen
API Product Owner (API Infrastruktur, API Instanz)	- Verantwortet als Product Owner die dauerhafte Weiterentwicklung und den Betrieb der API Infrastruktur (technische Plattform) oder einer oder mehrerer API Instanzen. Dies deckt auch Application Management des API Gateways ab. - Verantwortung der digitalen Leistungserbringung gegenüber den Geschäftspartnern - Verantwortung für die korrekte und vollständige Beschreibung der API Inhalte und Strukturen sowie deren Qualität in API Verzeichnissen und Online-Dokumentationen - Vertretung der Interessen der Geschäftspartner und anderen Stakeholdern - Zusammenarbeit mit API Architekt und API Entwickler - Identifiziert und formuliert die Anforderungen an die API Infrastruktur bzw. an API Instanzen - Priorisiert die Anforderungen und entscheidet (nach Konsultation) über deren Umsetzung - Beantragt das Jahresbudget für Weiterentwicklung und Betrieb der API Infrastruktur bzw. API Instanzen - Schliesst mit API Betreiber Betriebsverträge (SLA) ab *Siehe auch Rolle Produktverantwortliche/r in P000 Informatikprozesse in der Bundesverwaltung*[45]
API Architekt (API Infrastruktur, API Instanz)	- Entwickelt und dokumentiert eine konforme API Lösungsarchitektur - Beteiligung bei und Review von Inkrementen der API Architektur Bund *Siehe auch Rolle Lösungsarchitekt/in in P000 Informatikprozesse in der Bundesverwaltung*
API Entwickler (API Infrastruktur, API Instanz)	- Entwickelt die konforme API Infrastruktur bzw. API Instanzen
API Betreiber (API Infrastruktur, API Instanz)	- IKT-Betrieb (inkl. Support) der API Infrastruktur bzw. API Instanzen *Siehe auch Prozess «P06: IKT-Infrastruktur und -Services betreiben» in P000 Informatikprozesse in der Bundesverwaltung*

Tabelle 19: Rollen des Anbieters

7.4.2 Rollen des Geschäftspartners

Auf der Seite des Geschäftspartners sind nur zwei Rollen beschrieben, welche Kontakt zum Anbieter seitens Bund haben.

Rolle	Verantwortung, Aufgaben und Kompetenzen
API Client Entwickler	- Entwickelt den API Client seitens des Geschäftspartners mit Anbindung / Integration über ein API der Bundesverwaltung
API Client Nutzer	- Die API Client Nutzer sind autorisierte Personen oder autorisierte Systeme, welche über das API Zugriff auf digitale Behördenleistungen bzw. die Daten haben.

Tabelle 20: Rollen des Geschäftspartners

7.5 API Gouvernanz

7.5.1 API Gouvernanzmodell

Die Steuerung von APIs in der Bundesverwaltung erfolgt gemäss dem IKT-Lenkungsmodell auf Basis der Verordnung über die digitale Transformation und die Informatik (VDTI)[46]. Die Departemente und deren VE sowie die Bundeskanzlei sind für die Führung ihrer API Infrastrukturen und APIs selbst verantwortlich, unter Einhaltung übergeordneter Vorgaben. Der Bereich DTI ist für die API Architektur Bund verantwortlich und erlässt ggf. Weisungen (gemäss Art. 17, VDTI), welche die Verbindlichkeit der vorliegenden API Architektur oder Teile daraus regelt.

[45] P000 Informatikprozesse in der Bundesverwaltung, Intranet: https://intranet.dti.bk.admin.ch/isb_kp/de/home/ikt-vorgaben/prozesse-methoden/p000-informatikprozesse_in_der_bundesverwaltung.html
[46] https://www.fedlex.admin.ch/eli/cc/2020/988/de

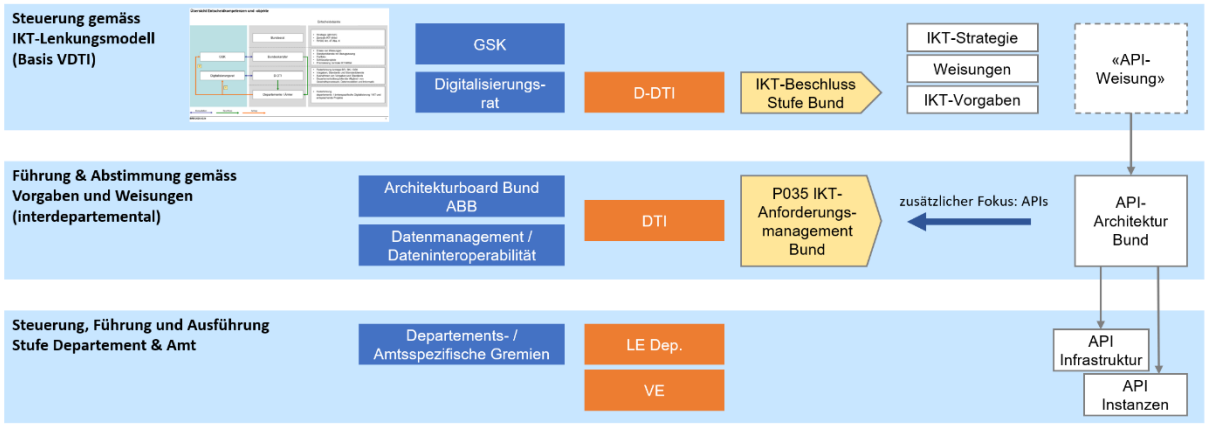

Abbildung 12: API Gouvernanz gemäss IKT-Lenkungsmodell

Abbildung 12 beschreibt die API Gouvernanz Bund auf drei Ebenen:

- **Steuerung** gemäss IKT-Lenkungsmodell (Basis VDTI): Müssen zu API Architektur im Bund und deren Anwendung übergeordnete Entscheide gefällt werden (z.B. aufgrund eines Bezugs zur IKT-Strategie, Portfolio, zentrale IKT-Mittel, Weisungen, IKT-Vorgaben, etc.), erfolgt die Steuerung über die im IKT-Lenkungsmodell und in der VDTI beschriebenen Gremien, Prozesse und Entscheidungsträger.
- **Führung & Abstimmung** gemäss Vorgaben und Weisungen (interdepartemental): Auf dieser Ebene erfolgt die fachliche Abstimmung und Konsultation zu interdepartementalen API Fragestellungen (z.B. zu API Architektur Bund oder gemeinsame API Infrastruktur). Dazu werden bestehende Gremien und Prozesse genutzt. Zum einen ist dies der Prozess P035 IKT-Anforderungsmanagement Bund, über welchen Departemente und die BK, die Leistungserbringer oder übergreifende Gremien IKT-Anforderungen zur Abstimmung und Entscheid einreichen können. Bei Bedarf wird das Architekturboard Bund (ABB) oder das Fachgremium zum Datenmanagement und Dateninteroperabilität (im Aufbau) konsultiert. Es wird zum Thema API keine eigene Organisation aufgebaut. Die bestehenden Gremien und Prozesse werden um das Thema API ergänzt.
- **Steuerung, Führung und Ausführung Stufe Departement & Amt:** Die Departemente und die BK regeln in ihrem Zuständigkeitsbereich die IKT-Lenkung selbst. Das betrifft auch API Infrastrukturen und APIs. Die Entwicklung und der Betrieb von API Lösungsarchitekturen erfolgt damit dezentral in den Departementen und VEs. Zur Förderung des interdepartementalen Austauschs und der Interoperabilität können als Empfehlung die Rollen aus Kapitel 7.4 verwendet werden.

7.5.2 Weiterentwicklung der API Architektur Bund

In der Weiterentwicklung der vorliegenden API Architektur Bund gibt es folgende Zuständigkeiten:

- Der Bereich DTI ist verantwortlich für die Weiterentwicklung der API Architektur Bund.
- Der Bereich DTI erstellt in Abstimmung mit den API Product Ownern und API Architekten auf Stufe Departement (inkl. LE) neue Inkremente der API Architektur Bund.
- Der Bereich DTI erstellt ggf. Weisungen, in welcher die Verbindlichkeit der vorliegenden Architektur oder Teile daraus geregelt wird.
- Das Architekturboard Bund und das Gremium zu Datenmanagement/Dateninteroperabilität wird bei grösseren Änderungen der API Architektur Bund und der Digitalisierungsrat bei Änderungen an allfälligen Weisungen konsultiert.
- Die API Product Owner und API Architekten auf allen Stufen kommunizieren die API Architektur Bund und ihre Änderungen in ihren Zuständigkeitsbereichen.

Eine Anpassung der API Architektur Bund soll mit Bedacht und langfristiger Perspektive erfolgen, damit die Stabilität für eine Zielverfolgung auf Architekturebene gewährleistet werden kann.

8 Datenarchitektur

8.1 Informationsmodell

Die Datenarchitektur wird anhand des Informationsmodells beschrieben. Das Informationsmodell wird als vereinfachtes UML-Klassendiagramm dargestellt. Es beschreibt die Informationsobjekte der API Referenzarchitektur und deren Beziehungen zueinander. Die Informationsobjekte sind unterteilt nach Akteuren, Serviceobjekten und Datenobjekten. Tabelle 21 beschreibt die Informationsobjekt-Typen, Tabelle 22 zeigt die Zuordnung der Informationsobjekte zu den Informationsobjekt-Typen.

Informationsobjekt-Typ	Beschreibung
Akteure	- Sind aktiv - Interagieren mit anderen Akteuren - Sind beteiligt am definieren, konsumieren und verarbeiten von Datenobjekten - Nutzen Serviceobjekte
Serviceobjekte	- Sind gekennzeichnet durch ein definiertes Verhalten - Produzieren, konsumieren oder verarbeiten Datenobjekte - Nutzen andere Serviceobjekte
Datenobjekte	- Sind passiv - Können Aggregationen von Datenobjekten sein - Auf Datenobjekten können Operationen ausgeführt werden.

Tabelle 21: Informationsobjekt-Typen

Akteure	Serviceobjekte	Datenobjekte	
- API Anbieter - LE Anbieter - API Geschäftspartner - LE Geschäftspartner - Subjekt	- Katalog Service - Provisioning Service - Stammdaten Service - IAM Services - Client - Gatekeeper Service - Ressource Service - Logging Service - Monitoring Service - Reporting Service - Monetization Service	- Geschäftspartner-objekt - Geschäftspartnerrolle - API Release - API Version ID - API Code - API Konfiguration - API Build - API Metadaten - API Dokumentation - Fachdokumentation - Nutzungsvereinbarung - Identität	- API Rolle - Secure Token - Ressource - Ereignis - Betriebsereignis - Fachereignis - Key Performance Indicator (KPI) - Preismodell - Report - Leistungsnachweis - Business Value Report - KPI Report - Rechnung

Tabelle 22: Informationsobjekte nach Informationsobjekt-Typen

Das Informationsmodell zeichnet sich dadurch aus, dass es die Informationsobjekte den Prozessstufen zuordnet. Abbildung 13 und Abbildung 14 zeigen das Informationsmodell in zwei Teile unterteilt, wobei die Prozessstufe Operate APIs diejenige Prozessstufe darstellt, welche die Abbildungen verbindet. Das Informationsmodell zeigt der Vereinfachung halber diejenige Teilmenge der Informationsobjekte, welche für das übergeordnete Verständnis der Beziehungen zwischen den Informationsobjekten relevant ist[47]. Kapitel 8.2 gibt einen Überblick über die Zusammenhänge zwischen den Informationsobjekten.

Abbildung 15 und Abbildung 16 vervollständigen das Informationsmodell durch Ergänzung um Datenobjekte der Prozessstufen Manage API Lifecycle und Analyse API Operation & Usage.

[47] Die Datenobjekten *API Build*, *API Konfiguration*, *Geschäftspartnerobjekt* und *Geschäftspartnerrolle* sind in Abbildung 13 zwecks Übersichtlichkeit doppelt abgebildet.

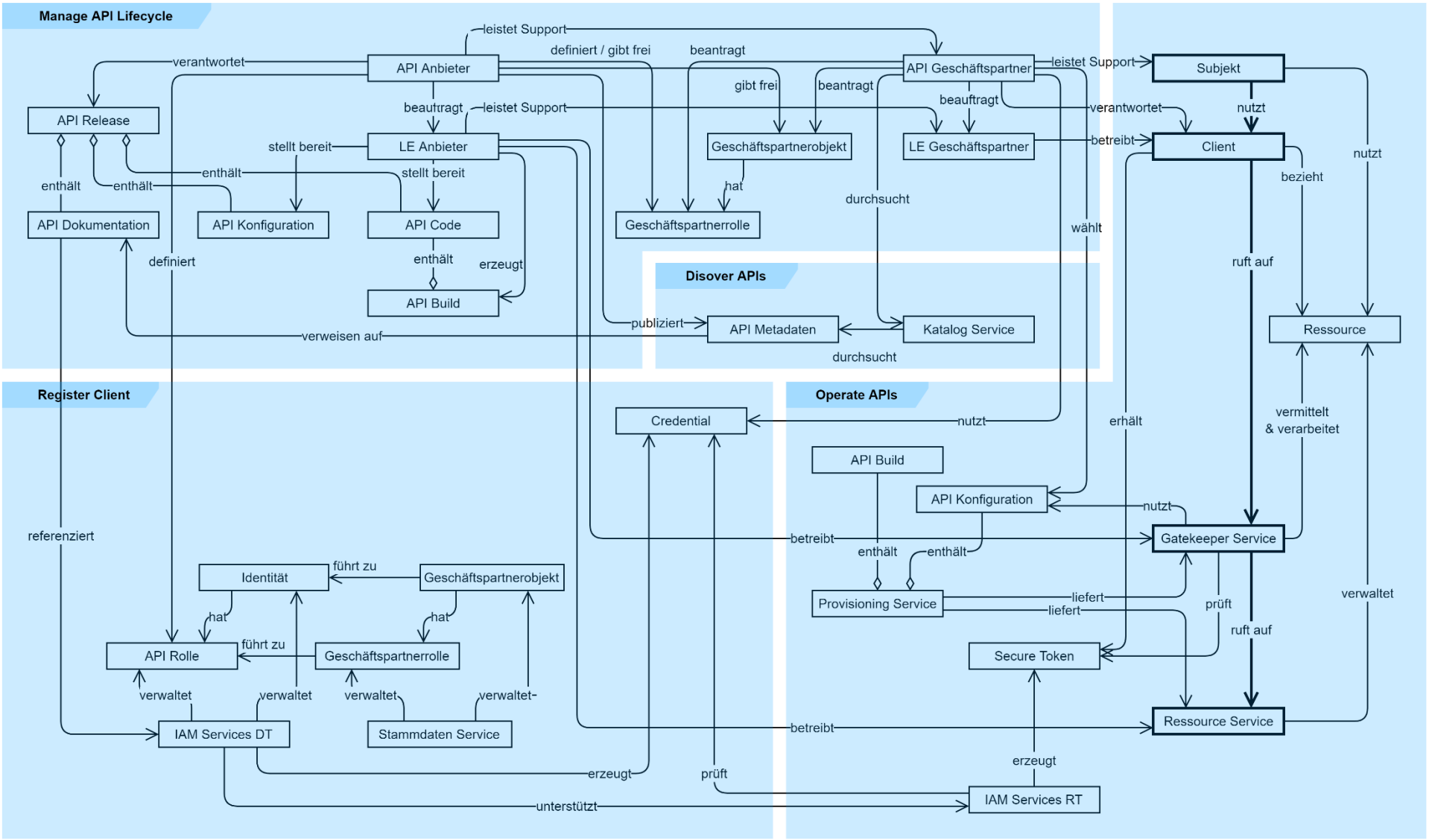

Abbildung 13: Informationsmodell Teil 1

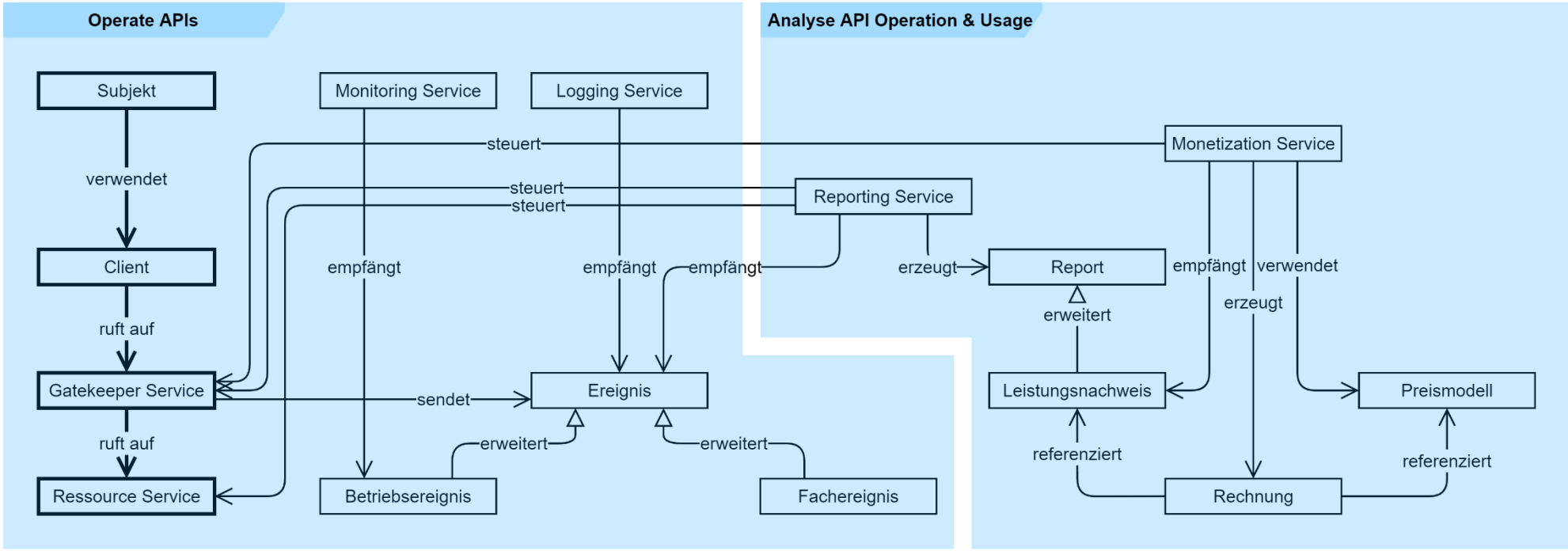

Abbildung 14: Informationsmodell Teil 2

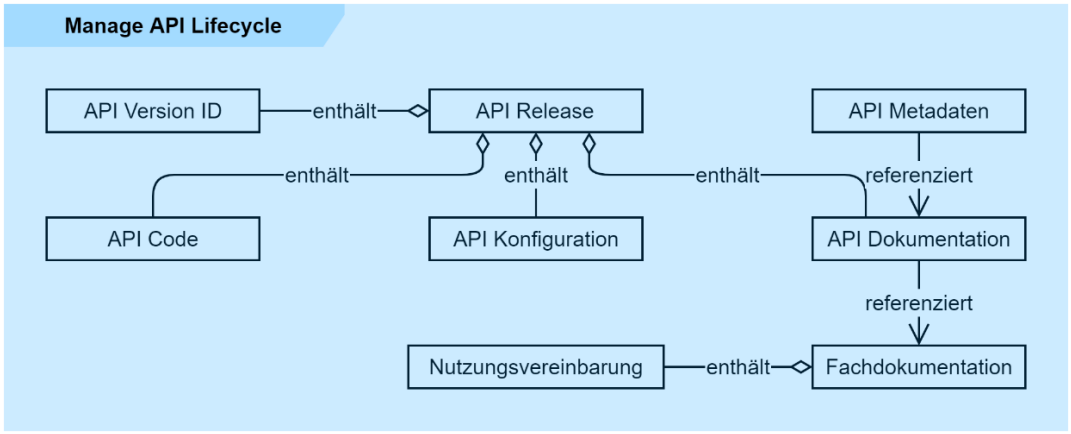

Abbildung 15: Informationsmodell mit Datenobjekten der Prozessstufe Manage API Lifecycle

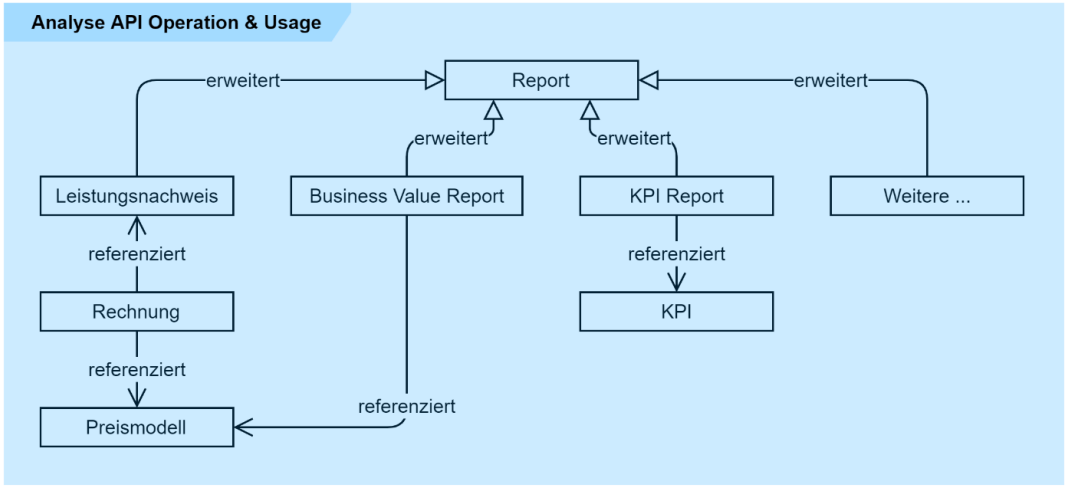

Abbildung 16: Informationsmodell mit Datenobjekten der Prozessstufe Analyse API Operation & Usage

8.2 Beschreibung der Informationsobjekte

Dieses Kapitel beschreibt die fundamentalen Zusammenhänge zwischen den Informationsobjekten der API Architektur Bund. Darüber hinaus liefert Kapitel 10.2 im Anhang eine detaillierte Beschreibung jedes einzelnen Informationsobjektes.

Der Kern einer digitalen Behördenleistung besteht im Zugriff eines Clients auf einen Ressource Service, der eine Leistung oder Daten anbietet. Der Zugriff wird durch einen Gatekeeper Service vermittelt. Mit der Unterstützung von IAM Services schützt der Gatekeeper Service die Ressourcen des Ressource Service[48].

Der API Anbieter ist die VE, welche die Entwicklung und den Betrieb der digitalen Behördenleistung verantwortet. Der API Anbieter kann die Entwicklung und/oder den Betrieb der digitalen Behördenleistung an einen Leistungserbringer delegieren. Der API Geschäftspartner ist der Partner, der die digitale Behördenleistung nutzt. Dabei handelt es sich entweder um eine Person, ein Unternehmen, oder eine andere VE auf einer föderalen Verwaltungsebene. Auch der API Geschäftspartner kann die Entwicklung und/oder den Betrieb des Clients an einen Leistungserbringer delegieren.

Ein API Release ist eine lauffähige Version eines APIs, die via Katalog Service veröffentlicht und den API Geschäftspartnern zur Nutzung bereitgestellt wird. API Releases umfassen typischerweise folgende Datenobjekte: den API Code, die API Konfiguration und die API Dokumentation.

API Builds sind auslieferbare Software-Pakete. Sie stellen die Grundlage für die API Infrastruktur und potenziell auch für die API Instanzen dar (s. Kapitel 9.1). API Builds werden entweder kostenfrei oder kostenpflichtig aus dem

[48] Der Ressource Service stellt sowohl Daten als auch Leistungen zur Verfügung, was im Kontext des Informationsmodells in beiden Fällen als Ressource bezeichnet wird und nicht mit der Ressourcenorientierung und damit Datenorientierung von REST APIs in Verbindung gebracht werden darf.

Internet bezogen oder aus dem API Code heraus erzeugt. Eine API Konfiguration parametrisiert sowohl Gatekeeper Service als auch Ressource Service. Der Provisioning Service liefert sowohl die API Builds als auch die API Konfigurationen aus. Wenn für den Fachservice eine API Schicht oder eine Systemkomponente für API Integration ausgeliefert wird (s. Abbildung 20), wird auch der Ressource Service mit einer API Instanz und/oder einer API Konfiguration versorgt werden.

Der Katalog Service ist das Kernelement der API Verzeichnisses. Er verwaltet die von den VE veröffentlichten API Metadaten und bietet eine Suchfunktion an.

Public APIs mit Registrierung und Partner APIs verlangen von Clients, dass sich diese anhand von Identitäten ausweisen. Typischerweise sind diesen Identitäten API Rollen für den Zugriff auf die Ressourcen zugewiesen. Bei Partner APIs sind Geschäftspartnerobjekte und Geschäftspartnerrollen als Teil des Stammdaten-Managements Voraussetzung für das Erzeugen von Identitäten und das Zuweisen von API Rollen zu diesen Identitäten. Identitäten und Rollenzuweisungen müssen beantragt und freigegeben werden.

Die IAM Services verwalten die Identitäten, die API Rollen und die Zuweisungen von API Rollen zu Identitäten. Sie generieren Credentials, Secure Tokens und führen Zugriffskontrollen durch.

Der Betrieb der digitalen Behördenleistung wird durch Logging und Monitoring Services unterstützt. Er versorgt auch die Reporting und Monetization Services mit betrieblicher und fachlicher Information. Die Monetization Services erlauben, anhand von Reports und Preismodellen die durch die API Nutzung entstehenden Einnahmen zu quantifizieren. In reifen Betriebsumgebungen erlauben Reporting Service und/oder Monetization Service, dynamisch auf den Betrieb der digitalen Behördenleistung zwecks Sicherstellung eines stabilen Betriebs Einfluss zu nehmen.

8.3 Gestaltungsempfehlung API Design

8.3.1 Geschäftsanforderungen

Wird eine Initiative zur Entwicklung einer digitalen Behördenleistung gestartet, so ist in einem ersten Schritt die Geschäftsarchitektur der digitalen Behördenleistung zu entwickeln. Typischerweise kann für die Ableitung der Geschäftsarchitektur der digitalen Behördenleistung die Geschäftsarchitektur der Behördenleistung in der physischen Welt herangezogen und darauf aufgebaut werden. Das Ergebnis sind Geschäftsarchitekturartefakte wie die Geschäftsfähigkeitenlandkarte, Geschäftsprozessbeschreibungen, Rollenkonzepte oder Geschäftsanforderungen, die anschliessend erlauben, für die digitale Behördenleistung Datenmodelle und Workflows von API-Aufrufen zu erarbeiten oder IAM Rollen zu modellieren.

Diese Geschäftsarchitektur steuert zusammen mit möglicherweise bereits im Bundesumfeld wiederverwendbaren APIs die Wahl der Vereinbarungen zum Datenaustausch (s. Kapitel 8.3.2).

8.3.2 Vereinbarungen zum Datenaustausch

Im Rahmen der Geschäftsfähigkeit API Design liefert die API Architektur Bund einen Ordnungsrahmen für Vereinbarungen zum Datenaustausch zwischen dem API Client und dem Fachservice. Dieser Ordnungsrahmen spannt ein Feld von möglichen Merkmalen eines einzelnen APIs auf. Die Wahl der Vereinbarungen wird beeinflusst von den Geschäftsanforderungen und den massgebenden technischen Standards des Geschäftsfeldes[49], im Rahmen dessen eine digitale Behördenleistung bereitgestellt werden soll.

Die API Architektur Bund gliedert diese Vereinbarungen entlang folgender Dimensionen:

- API Protokoll
- Schnittstellentyp
- Message Exchange Pattern (MEP)
- Datenformat
- Message-Typ

In der Praxis sind bei der Entwicklung von APIs bestimmte Kombinationen von Schnittstellentypen, Datenformaten, Message Typen, MEPs und API Protokollen verbreiteter als andere, oder es kommt vor, dass bei der Fixierung einer Dimension nur noch eine beschränkte Anzahl an Kombinationen auszumachen ist. Die API Architektur Bund

[49] Z.B. Standard eCH-0051 Standard für den Austausch von Daten im polizeilichen Anwendungsbereich oder Standard eCH-0056 Anwendungsprofil Geodienste

lässt grundsätzlich alle die Geschäftsanforderungen erfüllenden und mit einem günstigen Kosten-/Nutzen-Verhältnis umsetzbaren Kombinationen zu. Am Ende entscheiden die API Entwickler und damit der Markt über die Akzeptanz der APIs. Entsprechend wichtig ist ein sorgfältig nach Geschäftsfall abgestimmtes API Design, wobei die nachfolgenden Kapitel als Orientierungshilfe dienen.

Tabelle 23 stellt die empfohlenen Vereinbarungen zum Datenaustausch dar[50]. Diese Empfehlungen sind häufig anzutreffende Kombinationsgruppierungen von Vereinbarungen zum Datenaustausch und damit als Good Practice zu verstehen. Die verschiedenen Merkmale, welche die Vereinbarungen zum Datenaustausch festlegen, sind im Anhang in Kapitel 10.3 beschrieben. Bei der Gestaltung einer API Lösungsarchitektur sind diese Vereinbarungen nicht nur entlang des augenfälligsten Kommunikationswegs vom Client über den Gatekeeper Service zum Ressource Service zu treffen, sondern z.B. auch im Kontext der IAM Services oder des Logging Service.

API Protokoll	Schnittstellentyp	MEP	Datenformat	Message-Typ
HTTP	Ressourcenorientiert Methodenorientiert	Request-Response	Keine Einschränkung	Request Response
SOAP/WSDL	Ressourcenorientiert Methodenorientiert	Keine Einschränkung	XML	Request Command Response
gRPC	Ressourcenorientiert Methodenorientiert	Keine Einschränkung	Binärformat[51]	Keine Einschränkung
WebSockets	Ressourcenorientiert	Keine Einschränkung	Keine Einschränkung	Keine Einschränkung
MQTT	Ressourcenorientiert	Messaging	Keine Einschränkung	Request Response Event
AMQP	Ressourcenorientiert	Messaging	Binärformat[dito]	Request Response Event

Tabelle 23: Good Practices zu Vereinbarungen zum Datenaustausch

Die in den nachfolgenden Kapiteln sowie im Kapitel 10.3 verwendeten Begriffe Client und Server stellen die generischen Begriffe für fragende resp. sendende und antwortende resp. empfangende Informationsobjekte (Serviceobjekte) dar, welche im Informationsmodell an verschiedenen Stellen in Erscheinung treten.

8.3.2.1 RESTful APIs[52]

RESTful APIs erfüllen die Bedingungen des Representational-State-Transfer-Architektur-Stils (REST). RESTful APIs sind ressourcenorientiert. Sie stellen den dominierenden API Typ im Web dar und umfassen auch die Web-Technologie HTML über HTTP. Alle Kombinationen der Vereinbarungen zum Datenaustausch in Tabelle 23, welche die Bedingungen des REST-Architektur-Stils erfüllen, dürfen als RESTful APIs bezeichnet werden. Es handelt sich dabei um die sechs Bedingungen in Tabelle 24Tabelle 24, wobei eine Bedingung optional ist.

Die API Architektur Bund empfiehlt, in digitale Behördenleistungen einzubindende ressourcenorientierte Fachservices anhand von RESTful APIs zu realisieren.

Bedingung	Beschreibung
Client/Server Architecture	- RESTful APIs sind immer in eine Client/Server-Architektur eingebettet, in welcher der Client an den Server einen Request absetzt und vom Server eine Response erhält. - Die Rollen des Clients und des Servers sind klar definiert und statisch.
Uniform Interface	- Ein RESTful API implementiert eine einheitliche Schnittstelle zwischen Client und Server, die es erlaubt, Client und Server entkoppelt zu entwickeln. - RESTful APIs erlauben Create/Read/Update/Delete-Operationen (CRUD) auf Ressourcen, wie es z.B. die fünf Methoden POST, GET, PUT, PATCH und DELETE des HTTP-Protokolls ermöglichen.
Stateless Operations	- Aufeinanderfolgende Zugriffe von Clients auf Server sind immer unabhängig voneinander. Der Server stützt sich nicht auf Informationen aus früheren Anfragen.

[50] Partiell in Anlehnung an https://www.gartner.com/en/documents/3917098/a-guidance-framework-for-designing-a-great-api (Artikel hinter Bezahlschranke)
[51] Typisierte Daten, Strings können formatierte Inhalte sein, z.B. XML oder JSON
[52] Roy Thomas Fielding, PhD "Architectural Styles and the Design of Network-based Software Architectures", 2000, https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Bedingung	Beschreibung
	- Der Client ist für die Speicherung und Handhabung aller anwendungsbezogenen Informationen auf der Client-Seite verantwortlich.
Layered System of Servers	- Für den Client darf es keine Rolle spielen, ob er mit dem Server selbst, oder mit einem Vermittler, wie z.B. einem Gatekeeper Service (API Gateway), spricht. - Der Kommunikationsweg vom Client zum Server darf mehrere Vermittler aufweisen.
Resource Cache	- Auf dem Kommunikationsweg zwischen Client und Server dürfen Ressourcen im Client oder im Vermittler zwecks Steigerung der Performanz des Gesamtsystems zwischengespeichert werden. - Die Aktualität der Ressourcen muss von den Clients und den Vermittlern selbst verwaltet werden.
Code on Demand (optional)	- Der Server kann dem Client Programmcode für den Zugriff auf seine Ressourcen bereitstellen. - Der Client kann diesen Programmcode abrufen und lokal ausführen. - Diese Bedingung ist optional.

Tabelle 24: Bedingungen des REST-Architektur-Stils

8.3.2.2 GraphQL

Bei GraphQL handelt es sich um eine Open Source Data Query Language für APIs und eine Laufzeitumgebung für die Ausführung von Requests auf bereits vorhandenen Daten[53]. GraphQL bietet als Kombination von Abfragesprache und Laufzeitumgebung sowie eigenem Typen-System ein flexibles API Gerüst.

Eine auf GraphQL basierte Schnittstellen-Architektur stellt 1-N verschiedenartige Datenquellen über 1 API für den API-Client zur Verfügung, wobei es keine Limitierung für die Zahl abgefragter Ressourcen gibt. Mit GraphQL können somit auch komplexe Anfragen in einem einzelnem Request formuliert und beantwortet werden und es können exakt die in der Abfrage definierten Informationen ausgeliefert werden und somit Under- und Over-Fetching von Daten vermieden werden. Die Requests erfolgen in der GraphQL-spezifischen Syntax, welche drei Operationen erlaubt: *query* (Read), *mutation* (Create, Update, Delete), *subscription* (Publish/Subscribe). GraphQL über HTTP erlaubt dabei *query*- und *mutation*-Operationen, während GraphQL über WebSockets auch *subscription*-Operationen ermöglicht.

Die Nutzung der Vorteile von GraphQL gegenüber REST APIs geht mit einer Steigerung der Komplexität bei der Entwicklung von APIs einher.

API Protokoll	Schnittstellentyp	MEP	Datenformat	Message-Typ
HTTP	Ressourcenorientiert	Request-Response	GraphQL Query Syntax, JSON	Request Response
WebSockets	Ressourcenorientiert	Request-Response Messaging	GraphQL Query Syntax, JSON	Request Response Event

Tabelle 25: Vereinbarungen zum Datenaustausch bei GraphQL

8.3.2.3 Linked Data

Die API Architektur Bund behandelt das Thema Linked Data, da Linked Data ein Konzept für die Veröffentlichung von Daten ist, das auch bereits im Bundesumfeld zum Einsatz kommt. Beispiele von im Bundesumfeld betriebenen Linked Data Services sind derjenige vom Bundesarchiv (LINDAS)[54] und derjenige von swisstopo (Linked Data Service GeoDaten)[55]. Insbesondere empfiehlt die API Architektur Bund, API Metadaten von API Verzeichnissen auf einer Linked Data Plattform zu veröffentlichen (s. Kapitel 7.2.2.1), womit die API Metadaten des API Verzeichnisses selbst über ein Public API abgerufen werden können.

Unter dem Begriff Linked Data versteht man auf Basis des Resource Data Frameworks (RDF)[56] strukturierte Daten, welche zur Verwendung in Software-Anwendungen veröffentlicht werden (M2M-Kommunikation). Linked Data wird über die Abfragesprache (Query Language) SPARQL von SPARQL Servern resp. sog. SPARQL Endpunkten bezogen. Linked Data Services sind im Web weit verbreitet. Eine SPARQL Query kann mehrere Linked Data Services

[53] https://graphql.org
[54] https://lindas.admin.ch
[55] https://www.geo.admin.ch/de/geo-dienstleistungen/geodienste/linkeddata.html
[56] https://www.w3.org/TR/rdf11-primer

in einer einzigen SPARQL Query ansprechen. Da Linked Data auf dem API Protokoll HTTP basiert, fällt Linked Data in die Gruppe der ressourcenorientierten, HTTP-basierten Vereinbarungen zum Datenaustausch (s. Tabelle 26). Linked Data wird in Form von RDF Triples in sog. RDF Triplestores verwaltet. SPARQL Endpunkte bieten Zugang zu diesen RDF Triples. Weitere Erläuterungen zu Linked Data sind im Anhang Kapitel 10.4 zu finden.

API Protokoll	Schnittstellentyp	MEP	Datenformat	Message-Typ
HTTP	Ressourcenorientiert	Request-Response	JSON, XML, CSV	Request Response

Tabelle 26: Vereinbarungen zum Datenaustausch bei Linked Data

8.3.2.4 sedex – Secure Data Exchange[57]

sedex ist eine Dienstleistung des Bundesamts für Statistik (BFS). sedex steht nicht nur für definierte Vereinbarungen zum Datenaustausch, sondern auch für eine Vermittler-Plattform – analog zur Vermittler-Plattform von Swissdec -, welche auf Basis dieser Vereinbarungen arbeitet. Das BFS tritt mit sedex somit als Intermediär auf. sedex ist vergleichbar mit einem Postboten, der eingeschriebene Briefe elektronisch übermittelt. Die Dienstleistung kann von allen Geschäftspartner-Typen verwendet werden. Sie kann insbesondere auch von zwei Kommunikationspartnern ausserhalb der Bundesverwaltung genutzt werden, wobei alle Nachrichten die Netzzone(n) des Bundes passieren. Dabei können beide Kommunikationspartner Daten senden und empfangen.

API Protokoll	Schnittstellentyp	MEP	Datenformat	Message-Typ
HTTP	Ressourcenorientiert	Request-Response	XML	Request Response
SOAP/WSDL	Ressourcenorientiert	Keine Einschränkung	XML	Request Response

Tabelle 27: Vereinbarungen zum Datenaustausch bei sedex

Die Kommunikation über sedex erfordert immer eine gesetzliche Grundlage.

sedex ist ressourcenorientiert und kombiniert die API Protokolle HTTP und SOAP/WSDL, wobei der XML-Inhalt Dokumente in verschiedenen Formaten (csv, docx, jpg, pdf, xml, zip, etc.) enthalten kann. Tabelle 27 zeigt, wie sedex bei den Vereinbarungen zum Datenaustausch einzuordnen ist.

Die Systemkomponenten der API Referenzarchitektur auf den Integrationspfaden IP3 und IP4 finden folgende Entsprechungen in der sedex-Architektur[58]:

API Client	←→	Application + sedex Client
Vermittler-Plattform + API Gateway	←→	sedex Server
Fachservice	←→	Application + sedex Client

Es sei hier erwähnt, dass bei sedex der im Bund angesiedelte Leistungserbringer mit dem sedex Client auch einen Teil des üblicherweise vom API Geschäftspartner verantworteten API Clients ausliefert.

8.3.3 Fehlerbehandlung

Die API Architektur Bund empfiehlt, den Gatekeeper Service so zu gestalten, dass dieser dem API Client bei fehlerhaft formulierten API Zugriffen und Nicht-Verfügbarkeit von für den Betrieb der digitalen Behördenleistung notwendigen Services in jedem Fall aussagekräftige Fehlermeldungen zukommen lässt. Fehlermeldungen können auch in Form von Fehlercodes ausgegeben werden, welche in der API Dokumentation spezifiziert sind. Werden Fehlercodes von Aufruf-Parametern begleitet, so unterstützt dies die Aussagekraft der Fehlermeldungen. Die Fehlermeldungen erlauben es dem API Client geeignet zu reagieren, oder befähigen den API Entwickler den API Client geeignet anzupassen.

[57] https://www.sedex.ch
[58] https://www.bfs.admin.ch/bfs/de/home/register/personenregister/sedex/downloads.assetdetail.8826825.html

8.4 Gestaltungsempfehlung Identity & Access Management

8.4.1 Einleitung

Ein Identity- und Access Management (IAM) stellt eine wesentliche Komponente für integriertes, durchgängiges und elektronisches E-Government dar, insbesondere auch dann, wenn VE der Bundesverwaltung ihren API Geschäftspartnern digitale Behördenleistungen über APIs anbieten. Die Geschäftsfähigkeit IAM kommt in der API Architektur Bund dann zum Tragen, wenn ein API die Registrierung einer Identität erfordert, was im Minimum bedeutet, dass der API Client eine Identität präsentiert, die im IAM System hinterlegt ist und welcher der Fachservice vertraut.

Bei Public APIs mit Registrierung und Partner APIs müssen Identitäten authentifiziert werden, bevor sie auf die APIs zugreifen dürfen. Bei Public APIs mit Registrierung ist die Qualität der Authentifizierung i.d.R. tiefer als bei Partner APIs, da auf Public APIs jeder zugreifen darf, während bei Partner APIs (zur Definitionszeit) entschieden wird, welche Identitäten auf ein API zugreifen dürfen. Die Qualität der Authentisierung wird anhand des Level of Assurance (LoA, vgl. eCH-0170 Qualitätsmodell zur Authentifizierung von Subjekten[59]) definiert, dessen Einhaltung die Geschäftsfähigkeit IAM bei Zugriffen von Identitäten auf Ressourcen sicherstellen muss.

Bei Partner APIs werden Zugriffsanträge immer geprüft und können akzeptiert oder abgelehnt werden. Das Einschränken der Zugriffe auf Partner APIs bedeutet daher, dass Identitäten Rollenzuweisungen haben, die definieren, ob die Identität für den Zugriff auf ein API autorisiert ist.

Im IAM Kontext sind in jedem Fall die Vorgaben zur Netzwerksicherheit (NCSC, Si003)[60] einzuhalten, welche definiert, unter welchen Bedingungen Identitäten auf Netzwerkzonen zugreifen dürfen.

Am 19. Oktober 2016 trat die Verordnung über Identitätsverwaltungssysteme und Verzeichnisdienste des Bundes (IAMV)[61] in Kraft. Sie regelt für die Identitätsverwaltungssysteme, die Verzeichnisdienste und den zentralen Identitätsspeicher des Bundes die Zuständigkeiten, die Bearbeitung und Bekanntgabe von Personendaten und die Anforderungen an die Informationssicherheit. Gemäss Art. 3 ist der Zweck eines IAM Systems, Daten über die Identität und die Berechtigungen von Personen, Maschinen und Systemen gebündelt zu verwalten, um sie nachgelagerten Systemen und anderen IAM Systemen auf Anfrage zur Verfügung zu stellen. In diesem Sinne gilt die IAM Verordnung auch für digitale Behördenleistungen.

8.4.2 API Zugriff mittels Secure Token

Partner APIs können nur von berechtigten Partnern genutzt werden, mit denen die Bundesverwaltung eine Geschäftsbeziehung führt. Bei Partner APIs ist immer eine Identität zu hinterlegen. Danach erfolgt der Zugriff auf das Partner API mittels Credential und typischerweise einem Secure Token. Auch Public APIs können Secure Tokens verwenden, nämlich dann, wenn lediglich die Identität authentifiziert werden muss.

Die API Architektur Bund empfiehlt, Authentisierung und Autorisierung bei API Zugriffen auf Public APIs und Partner APIs mittels Secure Tokens zu implementieren. Secure Tokens haben gegenüber der ausschliesslichen Verwendung von Credentials folgende Vorteile:

- Secure Tokens stellen ein Instrument dar, um Authentisierungs- und Autorisierungsinformationen vom Client zum Gatekeeper Service und vom Gatekeeper Service zum Ressource Service zu übertragen.
- Durch Verwendung von typischerweise begrenzt gültigen Secure Tokens kann das permanente Speichern von Credentials im Client und Server vermieden werden.
- Durch Verwendung von Secure Tokens kann serverseitiges Session-Management vermieden werden (keine Sticky Sessions), da die Prüfung der Secure Tokens via IAM Services erfolgt (föderiertes IAM, s. Kapitel 8.4.3).
- Die verbreiteten Standards SAML[62] und OAuth/OIDC[63], welche Secure Tokens definieren, sind mit HTTP einfach nutzbar.

Darüber hinaus hat das Secure Token folgende Eigenschaften:

[59] https://www.ech.ch/de/standards/60593
[60] https://www.ncsc.admin.ch/ncsc/de/home/dokumentation/sicherheitsvorgaben-bund/sicherheitsverfahren/grundschutz.html
[61] https://www.fedlex.admin.ch/eli/cc/2016/610/de
[62] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security
[63] https://openid.net/connect

- Es stellt eine Ableitung des Credentials einer bekannten Identität dar.
- Es enthält die Bestätigung, dass eine Identität authentifiziert wurde.
- Es kann Autorisierungsinformationen tragen.

Secure Tokens werden immer nur für ein bestimmtes Zielpublikum ausgestellt, entweder den Client oder den Gatekeeper Service. Der Gatekeeper Service darf das Secure Token des Clients nicht an den Ressource Service weiterreichen.

8.4.3 IAM Bund Rahmenarchitektur

Den IAM Services kommt im Informationsmodell eine zentrale Bedeutung zu, denn es ist ein Verbund von IAM Services, der den Zugriff auf die Ressourcen steuert. Bei der Definition der IAM Services setzt die API Architektur Bund auf dem DTI-Standard *IAM Bund Rahmenarchitektur*[64] auf. Generell unterscheidet die API Architektur Bund zwei Gruppen von IAM Services: IAM Services zur Definitionszeit und IAM Services zur Laufzeit. Die IAM Bund Rahmenarchitektur nimmt bei diesen IAM Services weitere Unterteilungen in IAM Teil-Services vor, für deren Beschreibung die API Architektur Bund auf die Referenz verweist.

Die IAM Bund Rahmenarchitektur unterscheidet grundsätzlich zwei Arten von IAM:

- Föderiertes IAM
  Die IAM Services sind vom Gatekeeper Service getrennt und können über föderierte IAM Teil-Services verteilt sein.
- Nicht-föderiertes IAM
  Das IAM System ist im Gatekeeper Service integriert. Der Gatekeeper Service verfügt über seine eigenen IAM Services.

Um bestehende Identitäten und API Rollen im Bundesumfeld für mehrere digitale Behördenleistungen verwenden zu können, empfiehlt die API Architektur Bund föderiertes IAM zu verwenden. Das Informationsmodell in Abbildung 13 modelliert daher föderiertes IAM. Nicht-föderiertes IAM ist ebenfalls zugelassen, führt jedoch bei den API anbietenden VE über alle VE hinweg zu mehr Verwaltungsaufwand.

Die IAM Bund Rahmenarchitektur definiert die Prinzipien, die Regeln und den Ordnungsrahmen für die IAM Systemgestaltung, welche beim Bereitstellen von föderierten IAM Lösungen in der Bundesverwaltung berücksichtigt werden müssen. Insbesondere beschreibt sie den Umgang mit Secure Tokens.

Aufgrund des in der IAM Bund Rahmenarchitektur definierten Umgangs mit Secure Tokens bei föderiertem IAM sowie der mit Secure Tokens verbundenen Vorteile ist die Gestaltungsempfehlung IAM auf API Protokolle anwendbar, welche den Umgang mit Secure Tokens beherrschen, was bei Verwendung von HTTP gegeben ist.

Präsentiert ein Client dem Gatekeeper Service ein Secure Token, so stellt der Gatekeeper Service die vier in Tabelle 28 dargestellten, für jeden IAM Kontext geltenden Fragen, auf welche das Informationsmodell der API Architektur Bund die in Tabelle 28 dargestellten Antworten liefert. Die Fragen 3 und 4 werden nur von Partner APIs gestellt.

	Fragen	Antwort des Informationsmodells
1	Wer bist Du?	- Die IAM Services stellen für den API Geschäftspartner zur Definitionszeit eine Identität aus.
2	Wie erkenne ich Dich?	- Die IAM Services stellen für den API Geschäftspartner zur Definitionszeit ein Credential aus. - Die IAM Services prüfen zur Laufzeit vor dem Ausstellen des Secure Tokens für den API Geschäftspartner dessen Credential. - Der Gatekeeper Service prüft das Secure Token des Clients zwecks Authentifizierung der Identität.
3	Wozu hat Dich «jemand» autorisiert?	- Der API Anbieter gibt zur Definitionszeit die für die Identität beantragte(n) API Rolle(n) frei.
4	Wie setze ich die Grenzen Deiner Autorisation durch?	- Der Gatekeeper Service prüft das Secure Token des Clients zwecks Autorisierung des Zugriffs. - Der Ressource Service prüft das Secure Token des Gatekeepers zwecks Autorisierung des Zugriffs (nicht das gleiche Secure Token, s. Kapitel 8.4.4)

Tabelle 28: Antworten des Informationsmodells auf IAM-relevante Fragen

[64] https://intranet.dti.bk.admin.ch/isb_kp/de/home/themen/iam-bund/dokumente-steuerung-iam-bund.html

Die IAM Bund Rahmenarchitektur arbeitet mit den für die API Architektur Bund relevanten Informationsobjekten *Identitätslieferant*, *Federation Provider*, *Web-PEP* und *Web-PEP-On-Behalf*. Tabelle 29 liefert dazu die Beschreibung und die Entsprechungen der IAM-relevanten Informationsobjekte der API Architektur Bund. Diese Zuweisung der Informationsobjekte stellt sicher, dass API Architektur Bund und IAM Bund Rahmenarchitektur konsistent sind.

Informationsobjekte IAM Bund Rahmenarchitektur	Beschreibung	Informationsobjekte API Architektur Bund
Identitätslieferant	- Stellt Identitäten, Credentials und Secure Tokens aus und verwaltet diese	IAM Services (DT und RT)
Federation Provider	- Authentifiziert Identitäten bei föderiertem IAM - Konsumiert das Secure Token des Clients - Bezieht ein neues Secure Token - Gewährt Zugriff auf die Zielnetzwerkzone des Ressource Service	Gatekeeper Service
Web Policy Enforcement Point (Web-PEP)	- Fällt (die letzte) Entscheidung über den Zugriff auf die Ressource	Ressource Service
Web Policy Enforcement Point On Behalf (Web-PEP-On-Behalf)	- Fällt (die letzte) Entscheidung über den Zugriff auf die Ressource	Gatekeeper Service

Tabelle 29: Informationsobjekte der IAM Bund Rahmenarchitektur

8.4.4 Ausprägungen der IAM-relevanten API Architektur

Die API Architektur Bund unterscheidet vier typische, in Abbildung 17 dargestellte Ausprägungen bei der Gestaltung der IAM-relevanten API Architektur, welche sich in der Art der Client-Nutzung und der Art der Zugriffskontrolle unterscheiden.

	Client Nutzung durch (anonymes) Subject	Client Nutzung durch API Geschäftspartner
Zugriffskontrolle beim Gatekeeper Service	Ausprägung 1 Client-Nutzung durch (anonymes) **Subjekt** mit Zugriffskontrolle beim **Gatekeeper Service** (s. Abbildung 18)	Ausprägung 2 Client-Nutzung durch **API Geschäftspartner** mit Zugriffskontrolle beim **Gatekeeper Service**
Zugriffskontrolle beim Ressource Service	Ausprägung 3 Client-Nutzung durch (anonymes) **Subjekt** mit Zugriffskontrolle beim **Ressource Service**	Ausprägung 4 Client-Nutzung durch **API Geschäftspartner** mit Zugriffskontrolle beim **Ressource Service** (s. Abbildung 19)

Abbildung 17: Ausprägungen der IAM-relevanten API Architektur

Zwei der vier Ausprägungen visualisiert die API Architektur Bund in Abbildung 18 und Abbildung 19. Die anderen beiden Visualisierungen können davon abgeleitet werden.

Tabelle 30 beschreibt die beiden Arten der Client-Nutzung. Entweder wird der Client durch ein – möglicherweise anonymes – Subjekt oder durch den API Geschäftspartner selbst genutzt. An dieser Stelle wird vorweggenommen, dass ein Client immer nur mit einer einzigen Identität auf ein API zugreifen kann, womit das API auch immer nur eine einzige Identität zu authentifizieren braucht.

Art der Client-Nutzung	Beschreibung
Client-Nutzung durch Subjekt	- Der API Geschäftspartner verantwortet den Client. - Der API Geschäftspartner muss die den Client nutzenden Subjekte nicht zwingend kennen. - Die Subjekte können den Client auch anonym nutzen. - Die Identität repräsentiert den API Geschäftspartner und ist damit eine juristische Person (z.B. ein Unternehmen, das als Intermediär auftritt).
Client-Nutzung durch API Geschäftspartner	- Der API Geschäftspartner verantwortet den Client. - Die Subjekte vertreten den API Geschäftspartner (z.B. Mitarbeitende eines Unternehmens). - Das Subjekt kann nicht anonym auftreten. - Die Identität repräsentiert sowohl API Geschäftspartner als auch Subjekt und ist damit entweder eine natürliche oder juristische Person. - Repräsentiert das Subjekt eine juristische Person, so kann bei Bedarf die Identität der natürlichen Person der Identität der juristischen als Attribut mitgegeben werden.

Tabelle 30: Arten der Client-Nutzung

Tabelle 31 beschreibt in Abstimmung mit der IAM Bund Rahmenarchitektur die zwei Arten der Zugriffskontrolle bei API Zugriffen mittels Secure Token. Sie unterscheiden sich dadurch, dass die Kontrolle des Zugriffs auf den Ressource Service im einen Fall vom Gatekeeper Service, im anderen Fall vom Ressource Service wahrgenommen wird.

Art der Zugriffskontrolle	Schritte zur Laufzeit
Zugriffskontrolle beim Gatekeeper Service mit Web-PEP-On-Behalf durch Einsatz eines Secure Tokens	1. Der Client greift mittels Secure Token auf den Gatekeeper Service zu. 2. Der Gatekeeper Service authentifiziert die Identität. 3. Der Gatekeeper prüft die Autorisierung im Auftrag des Ressource Service und entscheidet über den Zugriff auf den Ressource Service.
Zugriffskontrolle beim Ressource Service mit Web-PEP durch Einsatz zweier verschiedener Secure Tokens	1. Der Client greift mittels Secure Token A auf den Gatekeeper Service zu. 2. Der Gatekeeper Service authentifiziert die Identität. 3. Der Gatekeeper bezieht mittels Secure Token A, das in diesem Anwendungsfall als Credential wirkt, bei den IAM Services ein Secure Token B mit der Autorisierung, die derjenigen von Secure Token A entspricht. 4. Der Gatekeeper Service greift mittels Secure Token B auf den Ressource Service zu. 5. Der Ressource Service authentifiziert die Identität des Gatekeeper Services. 6. Der Ressource Service prüft die Autorisierung und entscheidet über den Zugriff auf den Ressource Service.

Tabelle 31: Arten der Zugriffskontrolle

Varianten:

- Bei nicht föderiertem IAM sind die IAM Services (DT und RT) in Abbildung 18 und Abbildung 19 nicht vom Gatekeeper Service und Ressource Service getrennt, sondern ein Teil davon.
- Secure Tokens können – wie das Credential auch - mit einem Ablaufdatum versehen werden. Auf diese Weise lässt sich eine Registrierung beenden.
- Eine natürliche Person kann sich an einer Portalanwendung mit ihrer Identität auch via eIAM einloggen, welche dann mit dem Secure Token, welches der Client dem Gatekeeper Service zukommen lässt, verknüpft wird.
- Grundsätzlich schwächt föderiertes IAM das Sicherheitsniveau eines Ressource Services, da die Verantwortung für Informationssicherheit und Datenschutz auf mehr Akteure verteilt wird als bei nicht-föderiertem IAM. Daher kann es nützlich sein, Clients mit Client-Zertifikaten zu versehen, welche z.B. auf eine Client-Software-Version ausgestellt sind und damit erlauben, über die Identität hinaus, auch die Quelle eines API Aufrufs zu authentifizieren.

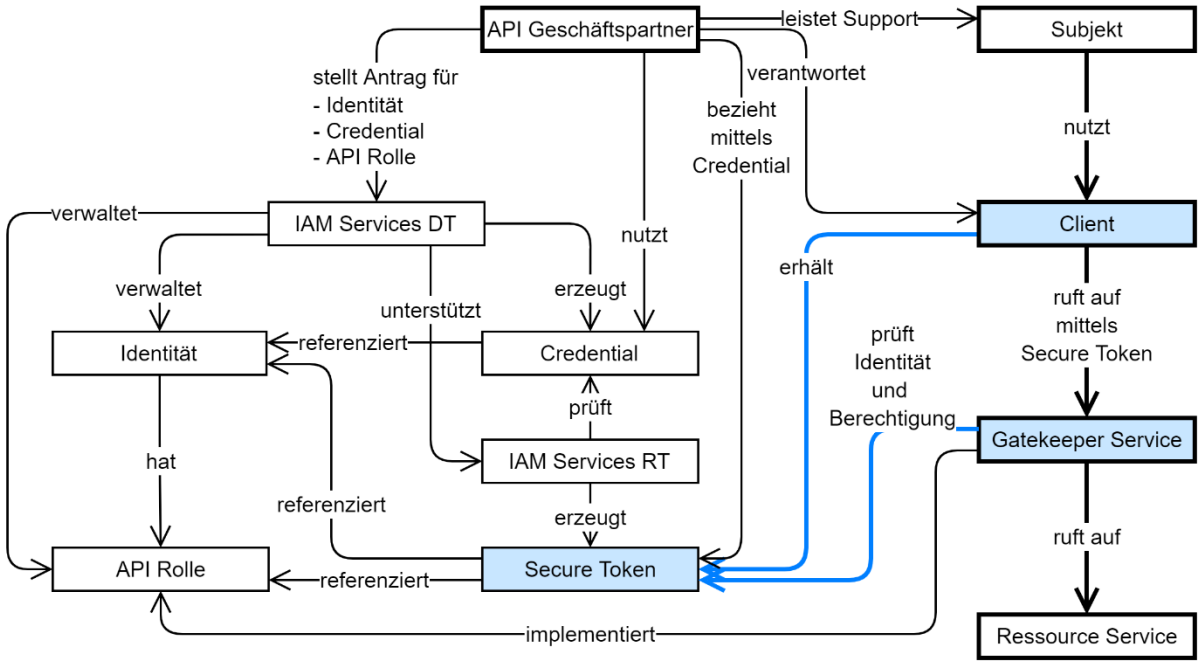

Abbildung 18: IAM-relevantes Informationsmodell Ausprägung 1

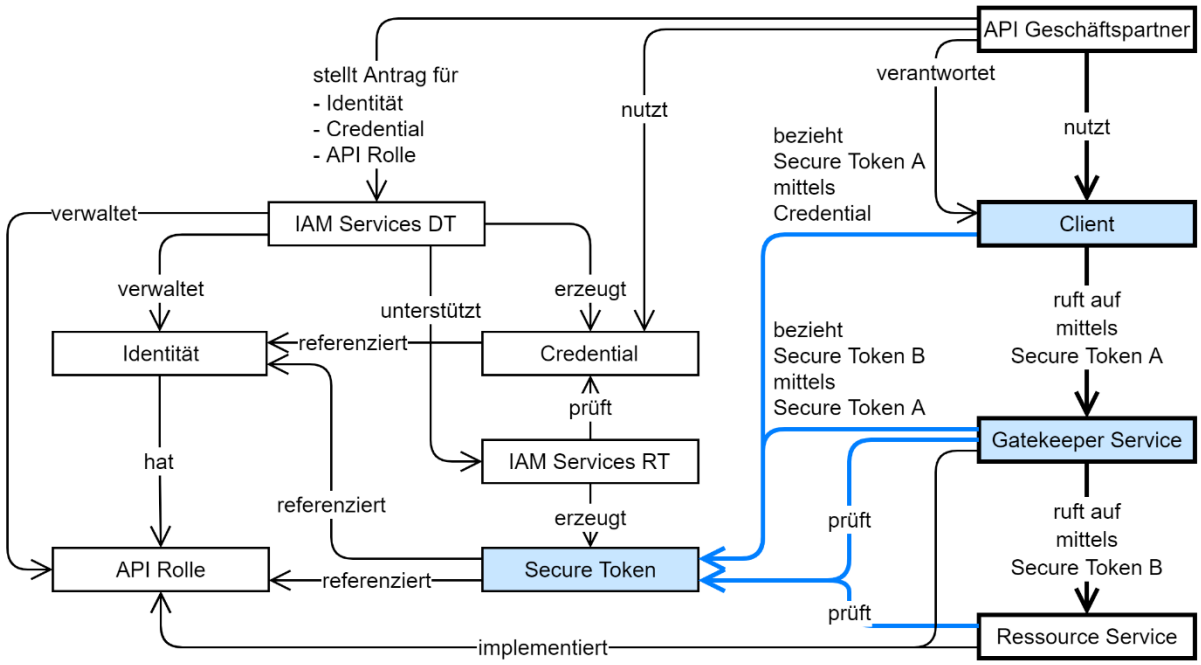

Abbildung 19: IAM-relevantes Informationsmodell Ausprägung 4

Im Bundesumfeld sind bereits Architekturen im Einsatz, welche Zugriffskontrollen im Gatekeeper Service und im Ressource Service umsetzen: Sowohl das Bundesamt für Zoll und Grenzsicherheit (BAZG)[65, 66] als auch das Eidg. Justiz- und Polizeidepartement (EJPD)[67, 68, 69] setzen einen Gatekeeper Service ein. Zusätzlich findet in der Lösungsarchitektur des EJPD eine weitere Stufe der Zugriffskontrolle im Ressource Service statt. In beiden Fällen ist der API Geschäftspartner auch der Nutzer des Clients.

[65] Es handelt sich um eine für das Programm DaziT entwickelte Lösungsarchitektur, in der die IAM Services vom Gatekeeper Service getrennt sind und durch die Systemkomponenten SAP MDG, Connex, PAMS und ePortal wahrgenommen werden.
[66] vgl. eIAM-Glossar https://www.eiam.admin.ch/pages/f!eiamglossary!pub_de.html?c=f!eiamglossary!pub&l=de «Was ist PAMS? Wie ist der Zusammenhang mit eIAM?»
[67] Es handelt sich um den E-Service SSO-Portal EJPD, welcher sowohl den Gatekeeper Service als auch die IAM Services abdeckt.
[68] https://www.bk.admin.ch/bk/de/home/digitale-transformation-ikt-lenkung/e-services-bund/services/sso-ejpd.html
[69] vgl. eIAM-Glossar https://www.eiam.admin.ch/pages/f!eiamglossary!pub_de.html?c=f!eiamglossary!pub&l=de «Was ist das SSO-Portal EJPD und wie ist der Zusammenhang mit eIAM?»

8.4.5 Single-Sign-On

Ein Secure Token kann Autorisierungen für mehrere Ressource Services enthalten. Diese Ressourcen Services sind Fachservices im Rahmen derselben digitalen Behördenleistung. Die API Architektur Bund spricht von Single-Sign-On (SSO), wenn der Client im Rahmen

- einer digitalen Behördenleistung; und
- eines definierten Zeitfensters (Time-Out)

für den Zugriff auf die N Ressource Services (N > 1) innerhalb des definierten Zeitfensters nur beim Zugriff auf den ersten Ressource Service authentifiziert wird. Für den Zugriff auf jeden weiteren Ressource Service innerhalb des Zeitfensters entfällt bei SSO die Authentifizierung, sofern die Identität die LoA-Anforderungen aller N Ressource Services erfüllt. Die API Architektur Bund empfiehlt, in den IAM Services SSO zu implementieren.

9 Applikationsarchitektur

9.1 Systemlandschaft der API Referenzarchitektur

9.1.1 Übersicht

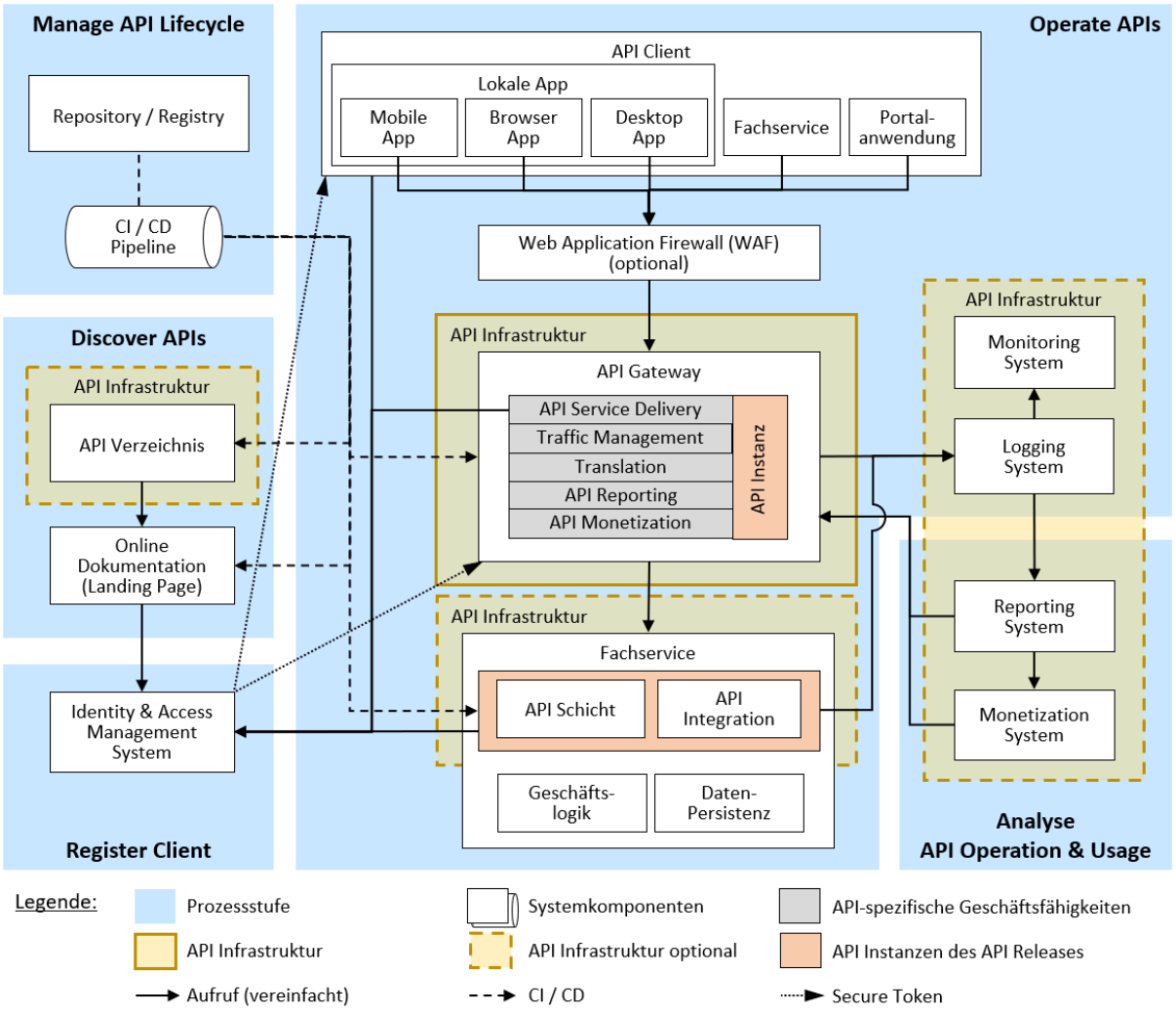

Abbildung 20: Systemlandschaft der API Referenzarchitektur

Die API Architektur Bund umfasst die in Abbildung 20 nach Prozessstufen gruppierten Systemkomponenten. Bei Entwicklung und Betrieb von APIs unterscheidet die API Architektur Bund zwischen Services und Komponenten der API Infrastruktur und den API Instanzen, welche die API-spezifischen Systemkomponenten der digitalen Behördenleistung darstellen. Entwicklung und Betrieb von Systemkomponenten der API Infrastruktur können von

einer anderen Organisationseinheit wahrgenommen werden als Entwicklung und Betrieb der API Instanzen. Insbesondere dürfen Entwicklung und Betrieb von Systemkomponenten der API Infrastruktur zentralisiert werden, während Entwicklung und Betrieb der API Instanzen dezentral wahrgenommen werden sollen.

9.1.2 API Infrastruktur

Die API Infrastruktur – typischerweise eine technische Plattform – bietet die generischen Systemkomponenten und Services für den Betrieb von APIs. Die in der Open Source Community oder auf dem Markt verfügbaren Produkte bieten dabei unterschiedliche Funktionen an. Je nach gewähltem Produkt und Anwendungsfall kann die API Infrastruktur-Plattform (d.h. das gewählte Produkt) unterschiedlich viele Systemkomponenten abdecken. Dies ist in Abbildung 20 durch die optionalen API Infrastruktur-Komponenten veranschaulicht. Wenn das API-Infrastruktur-Produkt eine für den Geschäftsfall relevante Systemkomponente nicht abdeckt, so muss die generische IKT-Infrastruktur der VE die Funktion dieser Systemkomponente anbieten (z.B. ein Logging-Service), und die API Infrastruktur kann an die generische IKT-Infrastruktur der VE angebunden werden.

Eine API Infrastruktur kann auf mehrere geografische Standorte verteilt sein und damit eine API Instanz zwecks Traffic Management auch an mehreren geografischen Standorten unter verschiedenen Adressen anbieten. Eine API Infrastruktur erlaubt die zentrale Verwaltung aller darin laufenden zum selben API gehörenden API Instanzen.

9.1.3 API Instanzen

API Instanzen kommen an zwei Stellen zum Einsatz:

- Sie laufen im API Gateway, konfigurieren dieses als Teil der API Infrastruktur oder stellen eigenständige ausführbare Objekte dar.
- Sie laufen im Fachservice entweder in Form einer API Schicht, welche die Funktionen eines Fachservices über APIs gegen aussen anbietet, oder in Form einer Systemkomponente für API Integration, welche ihrerseits als API Client auftritt und weitere APIs ansteuert.

API Instanzen exponieren immer die Fachlichkeit einer digitalen Behördenleistung, während die API Infrastruktur die Betriebsumgebung dafür bieten. Das Repository resp. die Registry liefert die API Builds und API Konfigurationen für beide Arten von API Instanzen. Die Geschäftslogik und die Datenpersistenz der Fachwendung sind nie Teil der API Instanz. Für das Implementieren der Fachlichkeit ist immer die Geschäftslogik zuständig. Zwecks Sicherstellung der Skalierung können API Instanzen vervielfacht und an mehreren geografisch verteilten Standorten laufen. In reifen Umgebungen kann die Skalierung dynamisch durch das Reporting System gesteuert werden.

9.2 API Gateway und Message Exchange Pattern

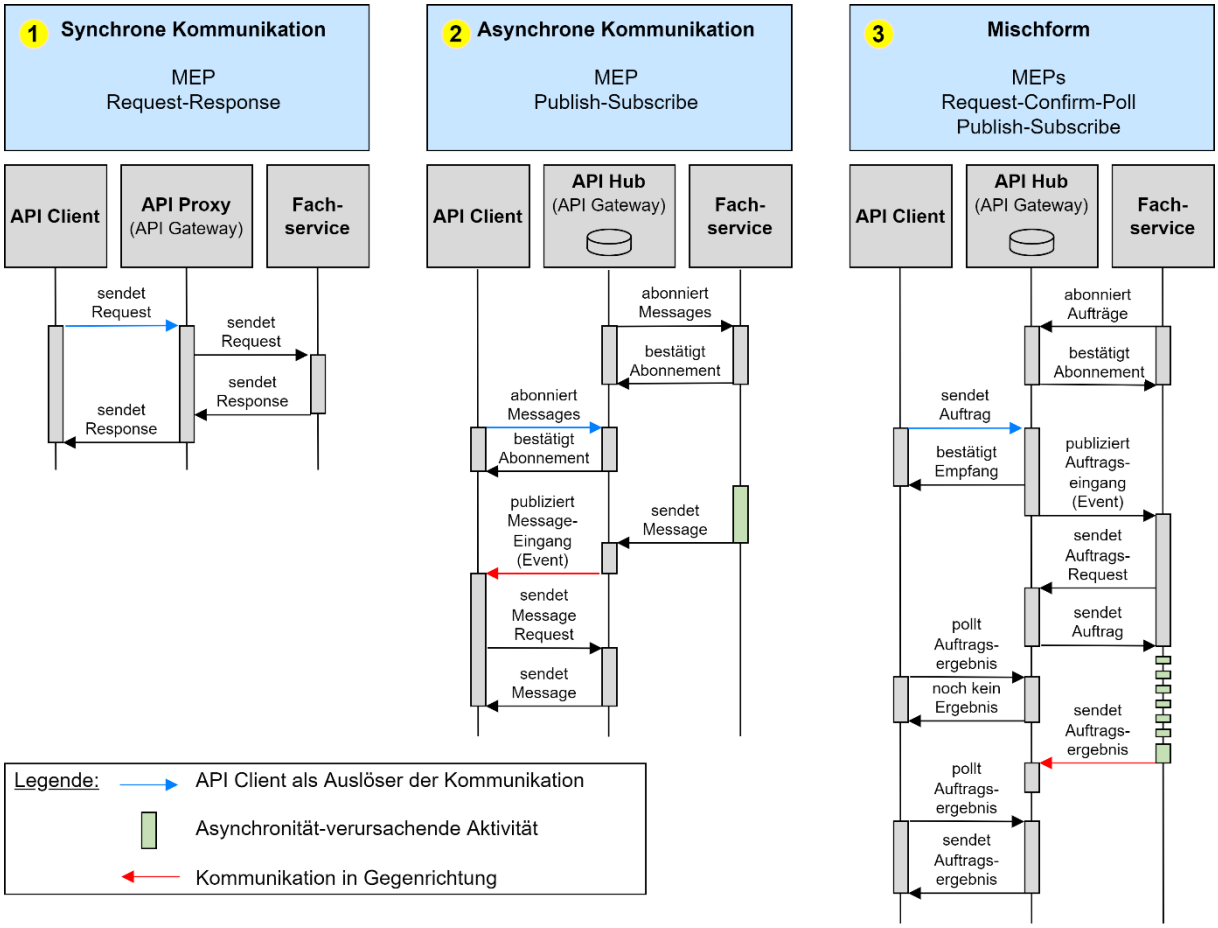

Abbildung 21: API Gateway und Message Exchange Pattern

Das API Gateway kann verschiedene Ausprägungen haben. Insbesondere kann es als API Proxy oder als API Hub gestaltet sein, wobei die einfachste Ausprägung eines API Gateways ein Reverse-Proxy ist. Ist das Gateway als API Proxy definiert, so wird die Kommunikation über ein synchrones MEP abgewickelt. Ist es als API Hub definiert, so kommt ein asynchrones MEP zur Anwendung. Bei asynchroner Kommunikation werden Messages im API Hub zwischengespeichert, da diese nicht in Echtzeit zwischen API Client und Fachservice ausgetauscht werden. Daher verfügt der API Hub im Gegensatz zum Proxy über einen Cache (flüchtiger Speicher) oder eine Datenbank (nicht-flüchtiger Speicher).

Abbildung 21 zeigt beispielhaft in Form von Sequenzdiagrammen drei Anwendungsfälle von API Gateways mit den Ausprägungen API Proxy oder API Hub sowie synchroner und/oder asynchroner Kommunikation.

1. API Proxy mit synchroner Kommunikation in Richtung API Client sowie in Richtung Fachservice
2. API Hub mit asynchroner Kommunikation in Richtung API Client sowie in Richtung Fachservice
3. API Hub mit synchroner Kommunikation in Richtung API Client und asynchroner Kommunikation in Richtung Fachservice (Mischform)

Die Ausprägung des API Gateways als API Proxy oder API Hub hat Auswirkungen auf die zwei MEPs zwischen API Client und API Gateway und zwischen API Gateway und Fachservice. Die beiden MEPs müssen nicht identisch sein. Ist eines der beiden MEPs asynchron, so kommt wegen der Anforderung der Zwischenspeicherung ein API Hub zum Einsatz.

Eine Ausprägung eines asynchronen MEP ist das MEP Publish/Subscribe, welches vom API Client verlangt, vom Server generierte Events empfangen zu können, was die Anforderung der Bidirektionalität an das API Protokoll stellt. Das MEP Publish/Subscribe bedingt, dass der API Client beim Server Messages abonniert, die der API Client nach Empfang eines Events vom Server anschliessend beim Server abholt.

Bei Anwendung des MEP Publish/Subscribe in Kombination mit Public APIs gibt es keine Prüfung von Registrationsanträgen, so dass der API Anbieter keine Kontrolle über die Anzahl API Clients hat, die bei bereitstehenden Messages benachrichtigt werden müssen und danach Messages abholen. Um das Risiko der Erzeugung von übermässigem Traffic entgegenzuwirken, kann die in Abbildung 21 dargestellte Mischform eingesetzt werden, welche zwischen API Hub und Fachservice Asynchronität zulässt und zwischen API Client und API Hub ein (synchrones) Polling-Verfahren vorsieht. Da nicht alle API Clients mit der gleichen Frequenz «pollen», ergibt sich dadurch eine zeitliche Verteilung der Anfragen.

Um die Anzahl API Clients steuern zu können, empfiehlt die API Architektur Bund, das MEP Publish/Subscribe zwischen API Client und API Hub nur mit Partner APIs zu kombinieren, da bei diesen Registrierungsanträge geprüft werden.

Obwohl bei gewissen asynchronen MEPs (s. Anhang Kapitel 10.3.3) der API Client Events empfangen können muss, ist es sowohl bei synchronen als auch bei asynchronen MEPs immer der API Client, der die Kommunikation anstösst (s. Abbildung 21).

Die API Architektur Bund empfiehlt, Anforderungen, die einen Daten- resp. Informationsfluss vom Fachservice in Richtung des API Clients erfordern, umzusetzen, indem der API Client anstatt des Servers die Kommunikation beginnt. Denn die API Architektur Bund verzichtet auf das Formulieren von Vorgaben zu APIs, welche von als Server agierenden API Clients angeboten werden. Bei tieffrequenten derartigen Datenflüssen[70], wo sich aus wirtschaftlichen Gründen Entwicklung und Betrieb eines APIs nicht lohnen, kann auf alternative Instrumente wie Email-Kommunikation zurückgegriffen werden.

9.3 Beschreibung der Systemkomponenten durch Mapping der Architekturebenen

Die Geschäftsarchitektur und die Datenarchitektur in Kapitel 7 und 8 definieren die Geschäftsfähigkeiten und die Informationsobjekte der API Architektur Bund. Die Beschreibung der Systemkomponenten erfolgt nun durch Zuordnung der Geschäftsfähigkeiten und Informationsobjekte zu den Systemkomponenten. Diese Zuordnung ist in Tabelle 32 abgebildet. Sie ist von links nach rechts zu lesen. Die Zuordnung einer Geschäftsfähigkeit zu einer Systemkomponente ist so zu verstehen, dass die Systemkomponente die Geschäftsfähigkeit oder Teile davon implementiert. Die Zuordnung eines passiven Informationsobjekts zu einer Systemkomponente bedeutet, dass die Systemkomponente das passive Informationsobjekt, konsumiert, ausgibt und/oder verarbeitet.

[70] Z.B. die jährliche Erinnerung eines API Geschäftspartners an die termingerechte Erfüllung einer behördlichen Pflicht.

Applikationsarchitektur	Datenarchitektur		Geschäftsarchitektur
Systemkomponente	**Serviceobjekte**	**Passive Informationsobjekte**	**Geschäftsfähigkeiten**
Repository / Registry	- Provisioning Service	- API Release - API Version ID - API Code - API Build - API Konfiguration - API Dokumentation - API Metadaten	- Release Planning - Transition - API Versioning - Version Control - API Documentation Management - API Publication & Revocation
CI/CD Pipeline	- Provisioning Service	- API Build - API Konfiguration - API Dokumentation - API Metadaten	- Transition
API Verzeichnis	- Katalog Service	- API Metadaten	- API Publication & Revocation - API Cataloging & Searching
Online-Dokumentation	- Katalog Service	- API Dokumentation	- API Documentation Management
IAM System	- IAM Services DT - IAM Services RT	- Identität - API Rolle - Credentials - Secure Token	- Identity & Access Management
API Client	- Client	- Ressource - Credential - Secure Token	- API Service Consumption
Web Application Firewall	- Gatekeeper Service	- Ressource	- API Service Delivery
API Gateway	- Gatekeeper Service	- API Build - API Konfiguration - Ressource - Ereignis - Secure Token	- API Service Delivery - Traffic Management - Translation - Monitoring - Logging - API Reporting - API Monetization - Identity & Access Management
Fachservice	- Ressource Service	- API Build - API Konfiguration - Ressource - Ereignis - Secure Token	- API Service Delivery - Monitoring - Logging - API Reporting - API Monetization - Identity & Access Management
Logging System	- Logging Service	- Ereignis	- Logging
Monitoring System	- Monitoring Service	- Betriebsereignis	- Monitoring
Reporting System	- Reporting Service	- Ereignis - Key Performance Index - Report	- API Reporting
Monetization System	- Monetization Service	- Preismodell - Leistungsnachweis - Rechnung	- API Monetization
n.a.	- Stammdaten Service	- Geschäftspartnerobjekt - Geschäftspartnerrolle	n.a.

Tabelle 32: Verknüpfung der Applikationsarchitektur, der Datenarchitektur und der Geschäftsarchitektur

Bemerkungen zu Tabelle 32:

- Die Geschäftsfähigkeit Release Planning führt zu einem API Release im Repository, geht aber weit über die Persistenz des API Releases im Repository hinaus. Die weiteren für Release Planning notwendigen Systemkomponenten und Informationsobjekte werden in der API Architektur Bund nicht beschrieben, da sie API-unspezifisch sind.
- Auf die Darstellung der Geschäftsfähigkeit API Design wird verzichtet, da diese auf alle Systemkomponenten und Informationsobjekte einen Einfluss hat.
- Die Informationsobjekte Stammdaten Service, Geschäftspartnerobjekt und Geschäftspartnerrolle sind Architekturbausteine der Master-Data-Government-Infrastruktur, welche mit dem Programm SUPERB[71] bereitgestellt wird.
- Die Systemkomponente CI/CD Pipeline ist häufig auch im Repository integriert.

9.4 Gestaltungsempfehlung API Integration

Geschäftsprozesse benötigen Fachservices. Beim Erbringen von digitalen Behördenleistungen greifen API Clients via APIs auf die Fachservices zu und integrieren auf diese Art Fachservice-Funktionen in den API Client.

Falls das Erbringen einer digitalen Behördenleistung des Bundes eine Verkettung von digitalen Behörden-Teil-Leistungen ist, empfiehlt die API Architektur Bund - getreu dem „Once-Only-Prinzip“ (s. Kapitel 2.2.1) - die Durchgängigkeit der digitalen Behördenleistungen sicherzustellen, indem der angesprochene Fachservice weitere Fachservices via APIs integriert.

Die API Architektur Bund spricht dabei von «API Integration im Fachservice des Bundes» (s. Kapitel 4.6). Ein vom API Client ausgelöster API Aufruf kann daher im Fachservice zu einem Durchlaufen eines mehrstufigen und verzweigten Baums an API Aufrufen führen. Bewegt sich ein solcher API Aufruf auf den Integrationspfaden IP3 oder IP4, so ist der Einsatz eines API Gateways angezeigt, falls nicht, kann auf den Einsatz eines API Gateways verzichtet werden. Die Kette von Aufrufen ist aus Gründen der Performance, des Unterhalts und des Testings möglichst kurz und einfach zu halten.

Falls im Rahmen von «API Integration im Fachservice des Bundes» geschützte, nicht anonymisierte Daten von anderen VE genutzt werden, muss eine vorgängige datenschutzrechtliche Abklärung erfolgen und eine Einwilligung bei der Datenerhebung eingeholt werden, oder der Verwendungszweck wird in der jeweiligen gesetzlichen Grundlage ergänzt.

Ebenso kann auch «API Integration im API Client des Geschäftspartners» stattfinden. Dies ist dann der Fall, wenn die digitale Behördenleistung des Bundes für die digitale Leistung des Geschäftspartners nur eine Teil-Leistung darstellt.

Bei der API Integration orchestriert eine vermittelnde Systemkomponente das Durchlaufen eines mehrstufigen und verzweigten Baums an API Aufrufen. Voraussetzung dafür ist immer, dass die einzelnen Teil-APIs von der vermittelnden Systemkomponente über das Netzwerk angesprochen werden können. Die vermittelnde Systemkomponente kann eine Integration Middleware sein (z.B. WSO2[72]), auf einem Data Service Framework basieren (z.B. ein mehrere Datenquellen integrierendes GraphQL API) oder eine anderweitige Individualentwicklung mit API Unterstützung sein. Diese Aufzählung ist nicht abschliessend. Die Grenzen zwischen dem API Gateway, der API Integration und der API Schicht sind je nach den gewählten Software-Produkten fliessend.

Bei Teil-APIs, die im Rahmen der API Integration eine Registrierung erfordern, ist für jeden Integrationsschritt zur Definitionszeit die Prozessstufe Register Client zu durchlaufen.

Beinhaltet eine Kette von API Aufrufen mehrere schreibende Operationen, so ist besonderes Augenmerk auf Transaktionssicherheit zu legen. Die API Architektur Bund unterscheidet an dieser Stelle zwischen den in Abbildung 22 dargestellten Abläufen «Parallele, schreibende Operationen» und «Kaskadierte, schreibende Operationen». Um die zur Gewährleistung von Transaktionssicherheit notwendige Komplexität tief zu halten, empfiehlt die API Architektur Bund «Parallele, schreibende Operationen» zu vermeiden.

[71] https://www.efd.admin.ch/efd/de/home/digitalisierung/superb.html
[72] https://wso2.com

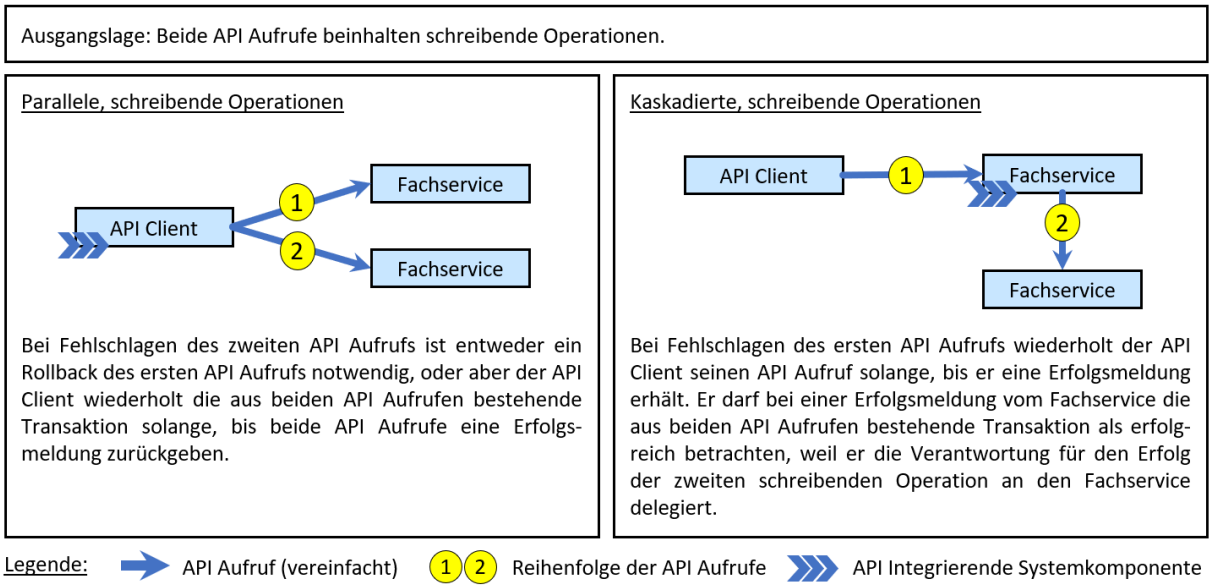

Abbildung 22: Parallele und kaskadierte, schreibende Operationen

9.5 Gestaltungsempfehlung API Versioning

9.5.1 Versionierung von API-spezifischen Lieferobjekttypen und Standards

Bei der Versionierung von APIs stellt sich die Frage, welche API-spezifischen Lieferobjekttypen und Standards innerhalb der Applikationsarchitektur einer Versionierung unterliegen. Die API Architektur Bund unterscheidet dazu die drei in Tabelle 33 abgebildeten API-spezifischen Lieferobjekttypen und Standards.

Lieferobjekttypen / Standards	Beschreibung	Empfohlene Handhabung bei der Gestaltung von API Lösungsarchitekturen
API Release	Der API Release bildet die Fachlichkeit einer digitalen Behördenleistung ab (s. Definition in Kapitel 10.2.1)	- Die Versionierung der Fachlichkeit manifestiert sich in der Versionierung des API Release. - Die Versionierung der Fachlichkeit wird allein durch den API Anbieter gesteuert.
API Infrastruktur	Unter API Infrastruktur werden API Infrastruktur-Plattformen oder API Frameworks verstanden (s. Definition in Kapitel 9.1). Beispiele sind API Infrastrukturprodukte wie WSO2 API Manager 4.0.0 oder RabbitMQ 3.8.18.	- Die Versionierung der API Intrastruktur wird durch die Open Source Community resp. kommerziellen API Infrastruktur-Anbietern gesteuert. - Die API Entwickler auf Seite der API Anbieter wählen diejenige Produktversion, welche die fachlichen und technischen Anforderungen am besten erfüllt.
API Protokolle	API Protokolle sind Standards, die versioniert werden (s. Kapitel 10.3.5). Beispiele sind HTTP/2.0 oder WSDL 2.0.	- Die Versionierung von etablierten API Protokollen werden von internationalen Gremien resp. grossen Internetkonzernen gesteuert. - Die API Entwickler auf Seite der API Anbieter wählen diejenige Protokollversion, welche die fachlichen und technischen Anforderungen am besten erfüllt oder übernehmen die Version, welche die API Infrastruktur bietet.

Tabelle 33: API-spezifische Lieferobjekttypen / Standards

Es sind damit die API Releases und damit deren fachliche Versionierung, welche im Folgenden in dieser Gestaltungsempfehlung behandelt wird.

9.5.2 Versionierung von API Releases

Für die Versionierung von API Releases empfiehlt die API Architektur Bund den Standard Semantic Versioning 2.0.0 (SemVer 2.0.0)[73]. Dieser weit verbreitete Standard basiert auf einer dreistelligen Versionsnummer, welcher auch optionale Extensions erlaubt:

[73] https://semver.org

Versionsnummer von API Releases in der Versionskontrolle:	MAJOR.MINOR.PATCH(-Extension)
Beispiele:	1.4.1, 2.0.0-rc1

Tabelle 34 erläutert die Bedeutung der vier Komponenten der Versionsnummer.

Stelle	Beschreibung
MAJOR	- Hauptversion, numerisch - Wird inkrementiert, wenn die Rückwärtskompatibilität gebrochen wird
MINOR	- Nebenversion, numerisch - Wird inkrementiert, wenn Erweiterungen vorgenommen werden - Rückwärtskompatibilität bleibt erhalten
PATCH	- Patchversion, numerisch - Wir inkrementiert, wenn Defect Fixes ausgeliefert werden - Rückwärtskompatibilität bleibt erhalten
Extension	- Optionales alphanumerisches Suffix - Dient der Bezeichnung von Alpha-/Beta-Releases bzw. Releases Candidates - Kann ggf. die Build-Nummer reflektieren

Tabelle 34: Komponenten der Versionsnummer nach SemVer 2.0.0

Werden API Versionen publiziert, welche damit zu API Releases werden, so wird der API Code im Repository im Rahmen der Versionskontrolle (s. Geschäftsfähigkeit Version Control) mit einem Label versehen (Tagging), welches die Versionsnummer gemäss SemVer 2.0.0 abbildet. Ein Label bildet die API Version ID ab (s. Informationsobjekt API Version ID in Abbildung 15) und wird in der Online-Dokumentation für die Referenzierung des API Releases verwendet. Das Tagging kann auch zur Kennzeichnung von API-Anbieter-internen Entwicklungsständen verwendet werden. Die entsprechenden API Version IDs sind in diesen Fällen gegen aussen nicht sichtbar, da interne Entwicklungsstände von APIs nicht publiziert werden.

9.5.3 Freiheitsgrade

Dem API Anbieter ist freigestellt, ob er die Stellen MINOR, PATCH und Extension in der Online-Dokumentation publiziert und technisch abbildet (s. Kapitel 9.5.4). Der Hintergrund dieses Freiheitsgrades ist, dass bei Betrieb eines einzigen API Releases beim Inkrementieren von Neben- und Patchversion die Rückwärtskompatibilität immer garantiert bleibt und der API Geschäftspartner durch Erweiterungen und Defect Fixes der digitalen Behördenleistung nur profitieren kann. Daher empfiehlt die API Architektur Bund für die Veröffentlichung von Versionsnummern folgendes Pattern:

Veröffentlichte Versionsnummern von API Releases:	(v)MAJOR(.MINOR.PATCH-Extension)
Beispiele:	v2, 1.2, v1.3.3

Betreibt ein API Anbieter mehrere API Releases parallel, so drängt sich die Publikation weiterer Stellen der Versionsnummer auf. Auf die Publikation der Patchversion und der Erweiterung kann jedoch verzichtet werden.

Es ist zulässig, bei APIs Versionen weder zu publizieren noch technisch abzubilden, wenn von vornherein angenommen wird, dass die Evolution eines APIs nur Erweiterungen und Defect Fixes hervorbringt, welche keine Anpassungen des API Clients erfordern.

9.5.4 Technische Abbildung der Versionsnummer

Ein Beispiel einer technischen Abbildung ist die Abbildung der Versionsnummer in der URL oder die Nutzung eines HTTP Headers. Unabhängig davon, wie viele API Releases einer digitalen Behördenleistung parallel betrieben werden und wie viele Stellen der Versionsnummer publiziert sind, die in der Online-Dokumentation publizierten Versionsnummern manifestieren sich in den publizierten API Releases derart, dass diese für den API Client bei jedem API Request eindeutig erkennbar sind, sei es in der URL, einem HTTP Header oder bei nicht HTTP-basierten APIs an einer anderen Stelle in der Applikationsschicht des OSI Modells (Schicht 7).

Tabelle 35 gibt einen Überblick, wo die Versionsnummer vor dem Hintergrund der in Kapitel 10.3.5 genannten API Protokolle typischerweise in der Applikationsarchitektur technisch verortet sein kann.

API Protokoll	Verortung der Versionsnummer
HTTP (Ausprägung RESTful APIs)	- Die API Architektur Bund beschreibt fünf Optionen: 1. URL Namespace: https://api.v2.admin.ch/services 2. URL Resource Level: https://api.admin.ch/v2/services 3. URL Query Parameter: https://api.admin.ch/services?version=2 4. HTTP Headers, Custom Headers - Accept-version: v2 5. HTTP Headers, Content Negotiation - Accept: application/myapi.v2+json - Accept: application/myapi+json;version=2 - Die Wahl der Option hat einen Einfluss auf die technische Komplexität des API Clients. Ist z.B. die Versionsnummer nicht in der URL abgebildet, so muss die Versionsnummer im API Client ausgewertet werden. - Option 4 und 5 sind in der Handhabung ähnlich.
Linked Data (Spezialfall von HTTP)	- SPARQL erlaubt, die Struktur der zugrundeliegenden Daten samt Semantik abzufragen, womit das Führen einer Versionsnummer nicht zwingend erforderlich ist. API Entwickler machen davon Gebrauch und erstellen zum Datenmodell passende SPARQL Queries. - Ist die Unterscheidung von API Versionen gewünscht, so muss die API Version im Datenmodell abgebildet und in den SPARQL Queries abgefragt werden. - Der Datenlieferant entscheidet über die Versionierungsstrategie.
Messaging (WebSockets, AMQP, MQTT)	- Das Message-Format weist ein Feld für die Versionsnummer aus. - Aufgrund der Verschiedenartigkeit der Anforderungen wird auf weitere Empfehlungen zur Versionierung von Messages verzichtet.
gRPC	- Versionierung der protobuf-Schnittstellen-Spezifikation (IDL) - Nutzung der Package ID der protobuf-Datei
SOAP/WSDL	- Versionierung der WSDL-/XSD-Schnittstellen-Spezifikation (IDL) - Aufgrund der Verschiedenartigkeit der Anforderungen wird auf weitere Empfehlungen zur Versionierung von WSDL-/XSD-Dateien verzichtet.

Tabelle 35: Technische Verortung der Versionsnummer

9.5.5 Erzwungene Migration

Die Empfehlung des Versionierungsstandards SemVer 2.0.0 impliziert, dass es beim Schritt zum nächsten API Release zulässig ist, die Rückwärtskompatibilität zu brechen (s. Kapitel 6 Architekturprinzip AP7). Ist die Publikation einer neuen Hauptversion geplant und soll aus Gründen der Kosteneffizienz eine API Hauptversion ausser Betrieb genommen werden, sollte dem API Geschäftspartner eine am Anwendungsfall orientierte Übergangsfrist für die Umstellung seines API Clients auf eine im Betrieb stehende API Hauptversion gewährt werden.

Diese Übergangsfrist hat zur Folge, dass der API Anbieter bei erzwungener Migration vorübergehend mehrere API Releases betreibt. Die API Release Planung wird in der Online-Dokumentation übergeordnet oder als Teil der API (Release) Dokumentation publiziert.

9.5.6 API Gateway und Fachservice

Die API Architektur Bund geht davon aus, dass die Kommunikation zwischen API Client und Fachservice bei einer digitalen Behördenleistung immer über ein API Gateway führt. Das API Gateway besteht dabei aus der API Infrastruktur und einer API Instanz. Der API Release manifestiert sich in der API Instanz des Gateways und der API Instanz des Fachservices mit den beiden Systemkomponenten API Schicht und API Integration.

Die API Architektur Bund empfiehlt, in der Online-Dokumentation immer die Version des API Releases und damit die fachliche Versionierung abzubilden, da es der API Release mit den genannten Systemkomponenten ist, der sich auf die Implementation des API Clients auswirkt. Es ist darauf zu achten, dass ein Anheben der Version der API Infrastruktur auf die API Releases möglichst keine Auswirkungen hat.

Die API Architektur Bund empfiehlt grundsätzlich auch, Entwicklung und Betrieb der API Infrastruktur und der API Instanzen unter die gleiche technische Verantwortung zu stellen. Die Infrastruktur-Verantwortung kann, insofern dies organisatorische Hürden beseitigt und eine effizientere Bereitstellung von APIs ermöglicht, von der technischen Verantwortung der einzelnen Fachservices und dazugehörigen API Instanzen getrennt sein.

10 ANHANG

10.1 Fähigkeitsbasierte Planung nach TOGAF

Geschäftsfähigkeiten können mit dem Konzept der fähigkeitsbasierten Planung gezielt geplant und mit KPIs gemessen werden. Dabei lassen sich die Fähigkeiten in vier verschiedenen Dimensionen beschreiben (s. Abbildung 23): Mitarbeiter, Infrastruktur, Prozesse und Informationen. Als Entwicklungsschritte für den Aufbau von Fähigkeiten werden sog. Fähigkeitsinkremente definiert, welche einer der vier Dimensionen zugeordnet werden. Die auf den drei Architekturebenen definierten Architekturbausteine können verwendet werden, um Fähigkeitsinkremente zu definieren und damit die Fähigkeit API Management Schritt für Schritt zu entwickeln. Tabelle 36 zeigt Beispiele von Fähigkeitsinkrementen für die Entwicklung der Fähigkeit API Monetization.

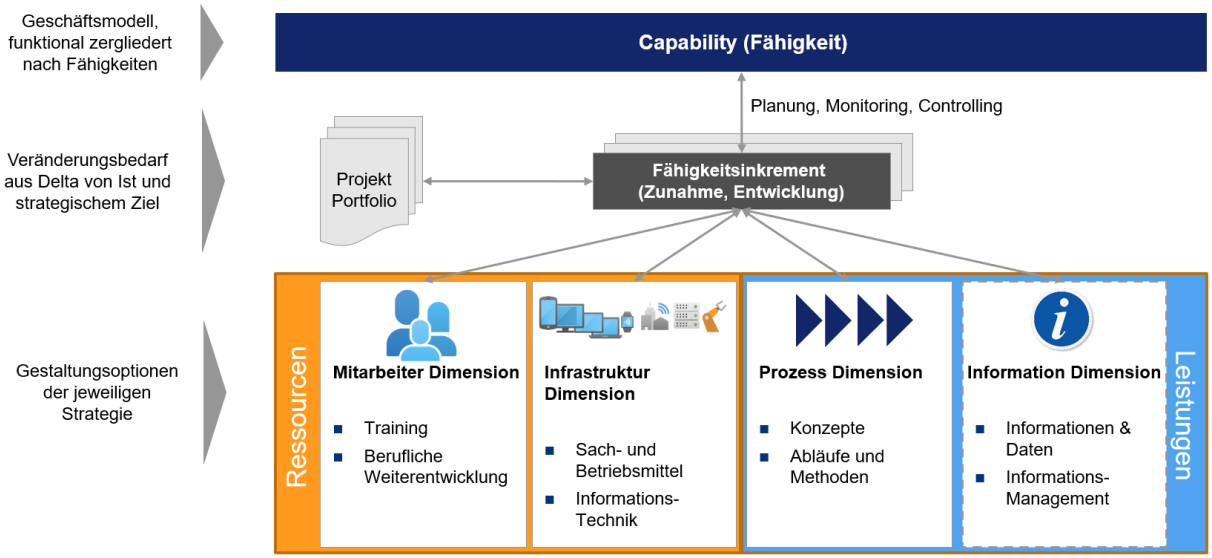

Abbildung 23: Fähigkeitsinkremente und Fähigkeitsdimensionen

Fähigkeitsdimensionen		Beispiele von Fähigkeitsinkrementen
Ressourcen	Mitarbeiter	Aufbau der Monetization-Kompetenz, Gewinn von IT Mitarbeitenden eines bestimmten Profils
	Infrastruktur	Evaluation eines Monetarisierungsprodukts
Leistungen	Prozesse	Definition eines Prozesses für das Planen von Einnahmen aus dem API Betrieb
	Informationen	Entwicklung von Datenmodellen für Preismodelle

Tabelle 36: Beispiele von Fähigkeitsinkrementen für die Fähigkeit API Monetization

10.2 Beschreibung der Informationsobjekte

10.2.1 Manage API Lifecycle

Informationsobjekt	Beschreibung
API Anbieter	Der API Anbieter ist die VE, welche die digitale Behördenleistung anbietet. Bei APIs, die die Registrierung einer Identität erfordern, vertritt der API Anbieter die Interessen der Ressource im IAM System. Daher ist bei Partner APIs der API Anbieter auch die Relying Party[74].
LE Anbieter	Der API Anbieter kann die Entwicklung und/oder den Betrieb der digitalen Behördenleistung an einen Leistungserbringer delegieren.
API Geschäftspartner	Der API Geschäftspartner ist der Partner, der die digitale Behördenleistung nutzt. Dabei handelt es sich entweder um eine Person, ein Unternehmen, oder eine andere VE auf einer föderalen Verwaltungsebene.
LE Geschäftspartner	Der API Geschäftspartner kann die Entwicklung und/oder den Betrieb des Clients an einen Leistungserbringer delegieren.
API Release	Eine API Version ist eine lauffähige Version[75] eines APIs. Ein API Release ist eine API Version, welche via Katalog Service veröffentlicht und dem API Geschäftspartner in Form von laufenden API Instanzen zur Nutzung bereitgestellt wird. Ein API Release umfasst typischerweise die Datenobjekte API Code und/oder API Konfiguration sowie API Dokumentation.
API Version ID	Die API Version ID bezeichnet die Version eines API Release. Jeder API Release hat eine API Version ID. Die API Version ID folgt einem Versionierungskonzept (s. Kapitel 9.5).
API Code	Der API Code stellt den lauffähigen Code des APIs dar. Er wird entweder vom Leistungserbringer entwickelt oder kostenfrei/kostenpflichtig erworben und bereitgestellt. Aus dem API Code wird der API Build erzeugt. Im Anwendungsfall, in dem ein API Infrastrukturprodukt als API Build bezogen wird, kann der API Code entfallen.
API Konfiguration	Die API Konfiguration erlaubt, den Gatekeeper Service und den Ressource Service zu parametrisieren. Die Parametrisierung kann z.B. in einer Anzahl API Instanzen, in der Memory-Ausstattung oder in Quotas für API Zugriffe bestehen. Die API Konfiguration des Gatekeeper Service wird zur Entwicklungszeit vom LE Anbieter bereitgestellt, zur Definitionszeit vom API Geschäftspartner gewählt (die Wahl bedingt die Registrierung einer Identität) und zur Laufzeit vom Gatekeeper Service genutzt. Die API Konfiguration des Ressource Service hängt typischerweise von der API Konfiguration des Gatekeeper Service ab. In reifen Betriebsumgebungen können API Konfigurationen vom Reporting Service und/oder vom Monetization Service gesteuert werden.
API Dokumentation	Die API Dokumentation umfasst alle technischen Informationen, welche notwendig sind, um einen Bezugsentscheid fällen und das API aus einem API Client heraus korrekt anzusprechen zu können. Insbesondere enthält sie auch die API Spezifikation und die API Release Notes sowie die SLOs, auf deren Darstellung als Informationsobjekte aus Gründen der Vereinfachung verzichtet wird.
Fachdokumentation	Die Fachdokumentation beschreibt die digitale Behördenleistung aus fachlicher Sicht. Sie verweist auch auf die gesetzlichen Grundlagen und enthält einen Masterplan mit Informationen zum API Lebenszyklus. Auch eine potenzielle Stilllegung des API wird anhand dieses Masterplans kommuniziert. Zur Fachdokumentation gehört auch die Nutzungsvereinbarung (s. Fähigkeit Legal Contract Management). Die Fachdokumentation ist die Voraussetzung für die API Dokumentation. Die API Dokumentation verweist auf die Fachdokumentation, um zu beschreiben, welche Aspekte der Fachlichkeit ein bestimmter API Release abdeckt.
API Build	API Builds stellen auslieferbare Software-Pakete dar. Sie sind die Grundlage für die API Infrastruktur und potenziell auch für die API Instanzen (s. Kapitel 9.1). API Builds werden kostenfrei/kostenpflichtig bezogen oder aus dem API Code heraus erzeugt und im Repository resp. einer Registry persistiert[76]. Wenn für den Fachservice eine API Schicht oder eine Systemkomponente für API Integration ausgeliefert wird (s. Abbildung 20), wird auch der Fachservice über eine API Instanz verfügen.

Tabelle 37: Informationsobjekte der Prozessstufe Manage API Lifecycle

[74] https://intranet.dti.bk.admin.ch/isb_kp/de/home/themen/iam-bund/dokumente-steuerung-iam-bund.html
[75] Aus Gründen der Vereinfachung wird auf die Modellierung der API Version im Informationsmodell verzichtet. API Versionen von APIs werden nur für Leistungserbringer-interne Zwecke verwendet.
[76] Ein Beispiel eines API Builds ist ein in eine Container Registry hochgeladener Container.

10.2.2 Discover APIs

Informationsobjekt	Beschreibung
API Metadaten	Die API Metadaten stellen im Sinne eines Katalogeintrags eine strukturierte, maschinell verarbeitbare Beschreibung des API Profils dar. Sie dienen dem Zweck, Informationen über die Existenz und die Eigenschaften von APIs im Bundesumfeld sowie die anbietenden VE in API Verzeichnissen online zu veröffentlichen. Die API Metadaten referenzieren die API Dokumentation auf der Landing Page.
Katalog Service	Der Katalog Service ist das Kernelement des API Verzeichnisses. Er verwaltet die von den VE veröffentlichten API Metadaten und bietet eine Suchfunktion an.

Tabelle 38: Informationsobjekte der Prozessstufe Discover APIs

10.2.3 Register Client

Informationsobjekt	Beschreibung
Geschäftspartner-objekt	Eine VE bietet API Geschäftspartnern digitale Behördenleistung an. Die API Geschäftspartner können dabei juristische oder natürliche Personen sein. Falls die Identifizierung der API Geschäftspartner für das Beziehen einer Behördenleistung notwendig ist, so manifestieren sich die API Geschäftspartner im Informationsmodell in dedizierten Datenobjekten im Stammdaten-Management.
Geschäftspartner-rolle	Existiert ein Geschäftspartnerobjekt im Stammdaten-Management, so können ihm dort Geschäftspartnerrollen zugewiesen werden, die mit dem Bezug von Behördenleistungen verbunden sind (nicht nur digitale Behördenleistungen). Bezieht der API Geschäftspartner digitale Behördenleistungen, so werden entsprechende Geschäftspartnerrollen vorausgesetzt.
Stammdaten Service	Der Stammdaten Service ist das Serviceobjekt, welches die Geschäftspartnerobjekte und Geschäftspartnerrollen verwaltet.
Identität	Die Identität ist die Repräsentation des Subjekts in der digitalen Welt. Eine Identität kann eine natürliche Person oder einen technischen Benutzer repräsentieren. Die Identität wird zur Definitionszeit erstellt. Sie besitzt Attribute und API Rollen, kann zudem Credentials tragen und einem Vertrauenslevel (Level of Assurance/LoA) unterliegen. Bei Partner APIs muss ein LoA festgelegt werden, dessen Einhaltung bei Zugriffen durch IAM sichergestellt werden muss, wobei das verwendete Credential die LoA-Anforderungen zu erfüllen hat. Den Identitäten werden Accounts der Fachservices zugeordnet. Falls die Identifizierung der API Geschäftspartner für das Beziehen einer erforderlich ist, so wird für die Erstellung einer Identität ein Geschäftspartnerobjekt vorausgesetzt.
API Rolle	Die Rolle ist das applikatorische Gegenstück zu einem rollenbasiert-modellierten Recht. Aus einer Applikationssicht können Rollen hierarchisch angeordnet werden und Regeln zur (automatischen) Zuweisung zu einer Identität enthalten. Falls die Identifizierung der API Geschäftspartner für das Beziehen einer Behördenleistung erforderlich ist, so wird für die Erstellung einer API Rolle eine Geschäftspartnerrolle (das o.g. Recht) vorausgesetzt. API Rollen werden zur Entwicklungszeit definiert, implementiert und zur Definitionszeit einer Identität zugewiesen.
IAM Services DT/RT	Das Serviceobjekt IAM Services repräsentiert zwei definierte Gruppen von IAM Services: - IAM Services, die zur Definitionszeit erbracht werden (z.B. Credential Service) → IAM Services Definition Time (DT), als Teil der Prozessstufe Register Client - IAM Services, die zur Laufzeit erbracht werden (z.B. Authentication Service). → IAM Service Run Time (RT), als Teil der Prozessstufe Operate APIs Die IAM Services zur Definitionszeit verwalten u.a. die Identitäten, die API Rollen und die Zuweisungen von API Rollen zu Identitäten. Die IAM Services zur Laufzeit generieren u.a. das Secure Token. IAM Services zur Definitionszeit erlauben einer natürlichen Person, sich (via die IKT-Standarddienste eIAM oder SSO-Portal) einzuloggen und einen Client resp. seine Identität mit einem in den API Rollen abgebildeten API zu verknüpfen.
Credential(s)	Das Credential ist die applikatorische Repräsentanz eines Identitätsnachweises, welches exakt einer Identität zugeordnet wird.

Tabelle 39: Informationsobjekte der Prozessstufe Register Client

10.2.4 Operate APIs

Informationsobjekt	Beschreibung
Subjekt	Das Subjekt ist eine natürliche Person, welche die Ressource nutzt. Das Subjekt nutzt die Ressource entweder über die Clients direkt oder indirekt über ein Behördenleistungsportal.
Ressource	Die Ressource stellt diejenigen Daten und Leistungen dar, für die die digitale Behördenleistung konzipiert wurde.

Informationsobjekt	Beschreibung
Client	Der Client ist das Ding und/oder ein Stück Software, welches über den Gatekeeper Service auf den Ressource Service zugreift. Typische Beispiele von API Clients sind Mobile Apps, Browser Apps und Desktop Apps. Ein Client kann auch in eine Portalanwendung integriert sein, welches Behördenleistungen anbietet. Das Serviceobjekt Client ist auf den API Zugriff ausgerichtet und blendet Aspekte der Benutzerführung aus.
Gatekeeper Service	Der Gatekeeper Service ist der zwischen Client und Ressource Service vermittelnde Service, der die Ressourcen vor Gefahren aus der Aussenwelt (z.B. unbefugte Zugriffe, Überlast) schützt und den Service-Level-konformen Betrieb sicherstellt. Der Gatekeeper Service wird anhand der API Konfiguration parametrisiert. Die Web Application Firewall (WAF) kann dem Gatekeeper Service zugeordnet werden und ist optional, da das API Gateway darauf ausgelegt ist, die Funktion der Web Application Firewall abzudecken.
Ressource Service	Der Ressource Service stellt den Backend-Service dar, z.B. ein Fachservice, auf den der Gatekeeper Service zugreift. Der Ressource Service stellt alle mit der digitalen Behördenleistung verbundenen Ressourcen bereit, auch dann, wenn dieser dafür weitere Ressource Services aufrufen muss. Der Ressource Service wird anhand der API Konfiguration parametrisiert.
API Build	S. Beschreibung in Kapitel 10.2.1
API Konfiguration	S. Beschreibung in Kapitel 10.2.1
Provisioning Service	Der Provisioning Service ist das Serviceobjekt, welches die API Builds und API Konfigurationen für die Provisionierung (Auslieferung) bereithält. Der Provisioning Service liefert die API Infrastruktur und die API Instanz(en) für den Gatekeeper Service und den Ressource Service aus. Die Auslieferung wird im Rahmen der API Publication angestossen.
IAM Services DT/RT	S. Beschreibung in Kapitel 10.2.3
Secure Token	Das zentrale Element einer Föderation ist das Secure Token, welches alle Informationen der Identität selbst, ihrer Authentisierung und API Rollen-Informationen enthält. Das Secure Token kann wiederum als eine Art Identitätsnachweis oder Credential betrachtet werden, welches von einem Secure Token-Konsument akzeptiert und ausgewertet wird. Das Secure Token wird nur nach erfolgreicher Authentifizierung der Identität ausgestellt und kryptographisch sicher übergeben.
Ereignis	Ein Ereignis ist ein Ereignis, das geloggt werden soll.
Betriebsereignis	Ein Betriebsereignis ist ein Ereignis, das als Basis für die Messung von betrieblichen KPIs oder dem Logging von sonstigen betrieblichen Informationen dient. SLA-relevante Ereignisse sind Betriebsereignisse.
Fachereignis	Ein Fachereignis ist ein Ereignis, welches als Basis für die Messung von fachlichen KPIs und der Ermittlung des Business Values dient.
Logging Service	Der Logging Service ist des Serviceobjekt, welches alle Ereignisse loggt und der API-anbietenden LE zur Verfügung stellt.
Monitoring Service	Der Monitoring Service ist das Serviceobjekt, welches SLA-relevante Ereignisse detektiert und diese dem API-anbietenden LE über geeignete Benachrichtigungskanäle meldet.

Tabelle 40: Informationsobjekte der Prozessstufe Operate APIs

10.2.5 Analyse API Operation & Usage

Informationsobjekt	Beschreibung
Key Performance Indicator (KPI)	Der KPI ist eine Messgrösse, um die Performanz einer digitalen Behördenleistung zu messen. Ereignisse liefern die Daten für das Messen des KPIs. KPIs können sich sowohl auf betriebliche als auch auf fachliche Ereignisse beziehen.
Reporting Service	Der Reporting Service erbringt folgende Leistungen: - Er erlaubt, Auswertungen vorzunehmen und Reports zu erzeugen. - Dies können auch auf künstlicher Intelligenz basierende Auswertungen sein. - Er kann über die API Konfiguration auch auf den Gatekeeper Service und/oder den Ressource Service steuernd einwirken.
Report	Ein Report ist ein Bericht über Ereignisse. Es gibt zahlreiche Report-Typen. Die API Architektur Bund nennt deren drei. Je nach Anwendungsfall sind weitere zulässig.
Leistungsnachweis	Ein Leistungsnachweis ist eine Spezialisierung eines Reports, welcher den Konsum einer digitalen Behördenleistung über einen bestimmten Zeitraum ausweist. Der Leistungsnachweis wird von der Rechnung referenziert.
Business Value Report	Der Business Value Report ist eine Spezialisierung eines Reports, welcher den Geschäftswert einer digitalen Behördenleistung ausweist. Der Business Value Report referenziert auf ein oder mehrere Preismodelle.
KPI Report	Der KPI Report ist eine Spezialisierung eines Report, der die Entwicklung von KPIs über einen Zeitraum hinweg ausweist.

Informationsobjekt	Beschreibung
Preismodell	Das Preismodell dient dem Zweck, die digitale Behördenleistung nach einer bestimmten Methode zu verrechnen, um damit Einnahmen zu erzielen.
Rechnung	Die Rechnung wird auf Basis des Leistungsnachweises und durch Anwendung eines Preismodells erstellt.
Monetization Service	Der Monetization Service erbringt folgende Leistungen: - Er empfängt den Leistungsnachweis, wendet darauf ein Preismodell an und erzeugt damit die Rechnung. - Er dient der Erstellung von Nutzungsplänen. - Er kann über die API Konfiguration auf den Gatekeeper Service und/oder den Ressource Service steuernd einwirken.

Tabelle 41: Informationsobjekte der Prozessstufe Analyse API Operation & Usage

10.3 Merkmale der Vereinbarungen zum Datenaustausch

10.3.1 Schnittstellentypen

Die Art des Fachservices beeinflusst den Schnittstellentyp, welcher bei der Entwicklung eines API angewendet wird. Bei Fachservices mit angebundener Datenbank bestehen Anwendungsfälle typischerweise in Daten- oder Dokument-Abfragen, bei Fachservices ohne angebundener Datenbank typischerweise im Aufrufen von Funktionen oder Ausführen von Operationen. Schnittstellentypen können daher in die zwei in Tabelle 42 dargestellten Arten unterteilt werden.

Schnittstellentyp	Beschreibung
Ressourcenorientiert	Die Funktion des Fachservices besteht in der Bereitstellung von Ressourcen, die über das API abgerufen und/oder verändert werden können. Clients kommunizieren mit dem Server durch den Austausch von Ressourcen in einem gewählten Datenformat (s. Kapitel 10.3.2). Ressourcen können auch Dokumente sein (csv, docx, jpg, pdf, xml, zip, etc.). Bei ressourcenorientierten APIs kommt der REST-Architektur-Stil zur Anwendung (s. Kapitel 8.3.2.1). REST APIs können jedes Datenformat übertragen und kommunizieren immer über HTTP. REST APIs sind weit verbreitet und daher sehr geeignet für die Anwendung in Public APIs.
Methodenorientiert	Die Funktion des Fachservices besteht in der Bereitstellung von Operationen (Methoden) auf Ressourcen, welche von den Clients aufgerufen werden können. Zu den gängigen methodenorientierten APIs gehören WS-* Webservices, welche SOAP/WSDL nutzen (s. Kapitel 10.3.5). Bei methodenorientierten APIs ist XML das am weitesten verbreitete Datenformat.

Tabelle 42: Schnittstellentypen

10.3.2 Datenformate

Tabelle 43 zeigt eine Übersicht über die beim Datenaustausch via APIs verwendeten Datenformate. Die Datenformate unterscheiden sich u.a. im Overhead, der Typisierung, dem Stil und der Lesbarkeit. Werden bei der Wahl der Datenformate die Good Practices gemäss Tabelle 23 befolgt, so wird das API bei den Entwicklern von API Clients mehr Akzeptanz finden.

Datenformat	Beschreibung
JSON[77]	JavaScript Object Notation (JSON) ist ein leichtgewichtiges[78], textbasiertes Datenformat, das von der Skriptsprache JavaScript abgeleitet ist. Es unterstützt Key/Value-Paare (Objects), Listen (Arrays) und Verschachtelungen. Da es im Vergleich zu XML mit weniger Overhead auskommt, findet es immer breitere Anwendung.
XML[79]	Extensible Markup Language (XML) ist ein textbasiertes Datenformat, das wie HTML mit Tags, Attributen und Verschachtelungen arbeitet. Tags, Attribute und Struktur können in einer XML Schema Definition (XSD) spezifiziert werden. Die Möglichkeit der Schemadefinition erlaubt auch die Validierung von XML-Inhalten und macht die Verarbeitung von XML eher aufwändig.

[77] https://www.json.org
[78] Der Begriff der Leichtgewichtigkeit bezieht sich auf das «Gewicht» der Nutzlast, die mit einem Datenformat oder einem Protokoll übertragen wird. Je geringer der Overhead ist, der mit einer Datenformat-Syntax oder einem API Protokoll-Syntax einhergeht, desto leichtgewichtiger ist das Datenformat resp. das API Protokoll.
[79] https://www.w3.org/TR/xml

Datenformat	Beschreibung
CSV	Comma Separated Values ist ein textbasiertes, zeilenorientiertes Datenformat, welches die Werte einer Zeile mit Kommas trennt (alternativ Semicolon oder Colon). Es können damit lediglich tabellenartige Strukturen dargestellt werden.
Binärformat	Binärformate werden für Anwendungsfälle verwendet, in denen eine hohe Performanz gefordert ist. Beispiele sind Audio/Video-Codecs beim Streaming oder Protocol Buffers (protobuf) beim Einsatz des API Protokolls gRPC.

Tabelle 43: Datenformate

JSON, XML und CSV sind Programmiersprachen-unabhängige Datenformate und können sowohl mit dem ressourcenorientierten als auch dem methodenorientierten Schnittstellentyp kombiniert werden. Mit REST-APIs (ressourcenorientierte APIs) arbeiten meist mit JSON. An zweiter Stelle folgt XML. Breit genutzte REST-APIs bieten daher oft beides, JSON und XML.

10.3.3 Message Exchange Patterns (MEPs)

Ein MEP beschreibt den zeitlichen Ablauf und die Reihenfolge der Interaktionen zwischen dem Client und dem Server. Bei der Kombination von MEPs mit Schnittstellentypen bestehen grosse Freiheiten. Hingegen sind MEPs eng an API Protokolle geknüpft. Tabelle 44 zeigt eine Übersicht über die verschiedenen MEPs.

Message Exchange Pattern	Sync/Async	Beschreibung
Request-Response	Synchron	Dieses MEP stellt die einfachste Form der Kommunikation zwischen Client und Server dar und definiert das Standard-Verhalten von HTTP. Der Client stellt eine Anfrage an den Server und setzt seine Arbeit erst dann fort, nachdem er vom Server die Antwort erhalten hat. Daher wird dieses MEP als synchrones MEP bezeichnet. Das MEP wird dann eingesetzt, wenn vom Server kurze Antwortzeiten erwartet werden dürfen.
Request-Confirm-Poll	Asynchron	Bei diesem asynchronen MEP besteht die auf den Request folgende Response lediglich in der Bestätigung der Annahme des Requests. Die Bestätigung enthält eine Request ID. Der Server stellt den Request zur Verarbeitung in eine Warteschlange und bearbeitet diesen erst, wenn freie Kapazitäten zur Verfügung stehen. Nachdem der Client die Bestätigung erhalten hat, beginnt er den Server in einem festgelegten Intervall mit der Request ID zu pollen. Das Polling ist erst dann erfolgreich, wenn der Server die Verarbeitung des Requests abgeschlossen hat und die Antwort in der Polling-Response an Client übertragen wurde. Das MEP wird dann eingesetzt, wenn die Verarbeitung im Server mehr Zeit in Anspruch nimmt und die Verantwortung für den Transfer des Verarbeitungsergebnisses beim Client bleiben soll.
Request-Confirm-Callback	Asynchron	Dieses asynchrone MEP ist an das MEP «Request-Confirm-Poll» angelehnt, überträgt jedoch die Verantwortung für den Transfer des Verarbeitungsergebnisses an den Server. Der Request des Clients beinhaltet eine Callback-Adresse, welche vom Server nach Abschluss der Verarbeitung aufgerufen wird. Der Client kann damit auf das Polling verzichten, und die Kommunikation zwischen Client und Server wird bidirektional. Dieses MEP ist vor allem dann von Vorteil, wenn auf Serverseite zwecks Steigerung der Performanz mehrere Instanzen für die Request-Verarbeitung eingesetzt werden und die Quelle des Verarbeitungsergebnisses daher unbekannt ist.
Point-to-Point (Messaging)	Asynchron	Das MEP «Point-to-Point» bietet einen einfachen asynchronen Message-Transfer vom Client zum Server über eine Message Queue. Bei diesem MEP wird beim Client vom Sender und beim Server vom Empfänger gesprochen. Der Sender kennt den Empfänger. Obwohl die Message Queue potenziell mehrere Empfänger erreicht, ist die Message für nur genau einen Empfänger bestimmt. Die Ereignisse über Message-Eingänge in der Queue werden immer zugestellt. Messages werden in der Queue gespeichert, bis der Empfänger bereit ist, diese abzurufen.
Publish-Subscribe (Messaging)	Asynchron	Das MEP «Publish-Subscribe» ermöglicht ereignisorientierte Kommunikation zwischen Client und Server, welche bidirektionaler Natur ist. Der Server ermöglicht mehreren Clients zur Definitionszeit das Abonnieren von Ereignissen zu spezifischen Themen (Subject). Zur Laufzeit benachrichtigt der Server die Clients via Ereignisse, sobald zu einem Subject neue Daten vorliegen. Der Server braucht seine Clients nicht zu kennen.

Message Exchange Pattern	Sync/Async	Beschreibung
		Je nach Informationsgehalt des Ereignisses reagiert der Client auf dessen Empfang auf eine der folgenden drei Arten[80]: - Der Event enthält eine ID, mit der ein bestimmter Datensatz beim Server abgeholt werden kann. - Der Event enthält IDs und/oder Attribute, die steuern, wie der Client seine Verarbeitung fortzusetzen hat. Dies kann den Bezug von weiteren Daten beim Server beinhalten. - Es besteht kein Bedarf, weitere Daten vom Server zu beziehen, denn das Ereignis enthält alle für den Client relevanten Daten (z.B. Logging Events)
Streaming	Asynchron	Beim Streaming handelt es sich um den Fall des MEPs «Publish-Subscribe», bei dem das Ereignis alle für den Client relevanten Daten enthält: Der Client stellt zum Server eine Verbindung her mit Angaben zum Subject. Danach hält er die Verbindung zum Server aufrecht. Jedes Mal, wenn beim Server neue Daten verfügbar werden, werden diese über die stets offene Verbindung dem Client zugestellt.

Tabelle 44: Message Exchange Patterns

Obwohl bei den MEPs «Request-Confirm-Callback» und «Publish-Subscribe» die Kommunikation bidirektional ist, ist es immer der Client, der die Kommunikation anstösst und der Server, der darauf reagiert. Beim MEP «Request-Confirm-Callback» besteht der Anstoss in der Anfrage, beim MEP «Publish-Subscribe» in der Abonnierung.

10.3.4 Message Typen

Messages bilden bei APIs die Grundeinheit der Kommunikation zwischen Client und Server. Sie können buchstäblich alles sein. Eine ID, eine Zeichenfolge, ein Objekt, ein Befehl (Command), ein Ereignis, etc. Messages können also vieles sein und haben konzeptuell keinen eingegrenzten Zweck. Dies macht Messages generisch. Die API Architektur Bund arbeitet mit den in Tabelle 45 definierten Message Typen.

Message Type	Beschreibung	Merkmale
Request (Synonym: Anfrage)	Ein Request ist eine Message, mit der Daten resp. Informationen abgefragt werden. Es handelt sich um eine 1:1 Kommunikation zwischen Sender und Empfänger.	- Ein Request ist an exakt einen Empfänger gerichtet. Der Sender kennt den Empfänger. - Ein Request ist eine Abfrage, d.h. der Sender sagt, was er vom Empfänger wissen muss. - Der Sender erwartet in jedem Fall eine Antwort: Entweder das Abfrageresultat oder eine Meldung, dass die Anfrage nicht bearbeitet werden kann. - Die Fehlerbehandlung ist in der Verantwortung des Senders. Er überwacht die Verarbeitung auf Seite des Empfängers.

Command (Synonym: Befehl)	Ein Command ist eine Message in Form einer 1:1 Kommunikation zwischen einem Sender und einem Empfänger.	- Ein Command ist an exakt einen Empfänger gerichtet. Der Sender kennt den Empfänger.
		- Ein Command ist ein Befehl, d.h. der Sender sagt, was der Empfänger tun muss.
		- Der Sender erwartet 0 – N Antworten
		  - 0 = Fire & Forget
		  - 1 = Normal
		  - N = Wenn der Empfänger einen Prozess abarbeitet und mehrere Antworten generiert
		- Die Fehlerbehandlung ist in der Verantwortung des Senders. Er überwacht die Verarbeitung auf Seite des Empfängers (ausgenommen Fire & Forget).
		- Der Empfänger kann den Command abweisen, wenn er ihn nicht verarbeiten kann.

Response (Synonym: Antwort)	Eine Response ist eine Message, die auf einen Request oder einen Command antwortet.	- Eine Response ist eine Antwort auf genau einen Request oder einen Command. - Auf jeden Request und jeden Command muss eine Response folgen (ausgenommen Fire & Forget).
Event	Ein Event ist eine Message, die über irgendeine Veränderung informiert. Den Publisher eines	- Ein Event zeigt eine Veränderung an. - Events verwenden immer das MEP «Publish/Subscribe».

[80] https://martinfowler.com/articles/201701-event-driven.html

Message Type	Beschreibung	Merkmale
(Synonym: Ereignis)	Events interessiert es nicht, wer den Event zur Kenntnis nimmt und darauf reagiert.	- Der Publisher muss nicht wissen, ob es Subscriber gibt. - Der Subscriber entscheidet, ob und wie er reagiert. - Es gibt keine Response. Der Publisher kümmert sich nicht, ob jemand reagiert. - Die Fehlerbehandlung ist in der Verantwortung des Subscribers. - Der Subscriber kann einen Event nicht zurückweisen – ein Event ist ein Fakt.

Tabelle 45: Message Typen

10.3.5 API Protokolle

API Protokolle sind eng an MEPs geknüpft. Wie die anderen Dimensionen der Datenaustausch-Vereinbarungen ist deren Wahl am Zielpublikum auszurichten. Die verbreitetsten API Protokolle sind in Tabelle 46 aufgeführt.

API Protokolle	Beschreibung
HTTP	Das Hypertext Transfer Protocol (HTTP) ist das Standard-Protokoll des Internets. Die jüngste Version ist HTTP/2, welche im Jahr 2015 veröffentlicht wurde und mittlerweile von den namhaften Browsern unterstützt wird. HTTP ist ein Application Layer Protokoll, wird aber teilweise von anderen Protokollen, wie z.B. gRPC oder SOAP/WSDL, auch als Transport Layer Protokoll genutzt. Es ist auf des MEP «Request-Response» beschränkt.
WebSockets[81]	WebSockets ist ein Application Layer Protokoll der Internet Engineering Task Force (IETF), das Full-Duplex-Streaming-Kommunikation über TCP ermöglicht und von den namhaften Browsern unterstützt wird. Es handelt sich dabei um ein Protokoll, bei dem im Gegensatz zu HTTP beide Seiten gleichberechtigt sind. WebSockets ermöglichen eine anhaltende bidirektionale Kommunikation zwischen Client und Server bei stets offener Verbindung. WebSockets eignet sich daher für das Übertragen von Events vom Server an den Client.
MQTT[82]	Message Queuing Telemetry Transport (MQTT) ist ein Standard der «Organization for the Advancement of Structured Information Standards» (OASIS) für IoT-Anwendungen. Es ist als Transport Layer Protokoll auf Basis des MEPs «Publish-Subscribe» konzipiert, das mit minimaler Netzwerkbandbreite auskommt und daher als sehr leichtgewichtig gilt.
AMQP[83]	Advanced Message Queuing Protocol (AMQP) ist ein OASIS-Standard auf Basis der MEPs «Point-to-Point» und «Publish-Subscribe» (Messaging) für den Austausch von Business Messages.
gRPC[84]	gRPC ist ein modernes, leistungsstarkes Open-Source-Framework für Remote Procedure Calls (RPC), das in jeder Umgebung ausgeführt werden kann. Es kann Services effizient verbinden und bietet Unterstützung für Load Balancing, Tracing, Health Checking und Authentifizierung. gRPC basiert auf HTTP/2 und nutzt die IDL protobuf.
SOAP/WSDL[85]	Das Simple Object Access Protokoll (SOAP) ist ein Protokoll des World Wide Web Consortiums (W3C), das für den Austausch von strukturierten Informationen in einer dezentralen Umgebung gedacht ist. SOAP verwendet XML-Technologien zur Definition eines erweiterbaren Messaging-Frameworks, das erlaubt, Messages über eine Vielzahl von zugrundeliegenden Protokollen auszutauschen. SOAP nutzt XML als Datenformat und meistens HTTP als Transport Layer Protokoll, um Clients auf Servern Methoden von SOAP Web Services auszuführen zu lassen. Zur Spezifikation der Methoden dient die Web Service Description Language (WSDL) in Kombination mit einer XML Schema Definition (XSD) als IDL.

Tabelle 46: API Protokolle

10.4 Resource Data Framework / Linked Data[86]

Unter dem Begriff Linked Data versteht man auf Basis des Resource Data Frameworks (RDF) strukturierte Daten, welche zur Verwendung in Software-Anwendungen veröffentlicht werden (M2M-Kommunikation). Das Resource Description Framework ist ein Rahmen zur Darstellung von Informationen über Ressourcen. Ressourcen können alles sein, einschließlich Dokumente, Personen, physische Objekte und abstrakte Konzepte. RDF ist für Situationen gedacht, in denen Informationen im Web von Software-Anwendungen verarbeitet und nicht nur für Personen angezeigt werden müssen.

[81] https://datatracker.ietf.org/doc/html/rfc6455
[82] https://mqtt.org
[83] https://www.amqp.org
[84] https://grpc.io
[85] https://www.w3.org/TR/soap12-part1
[86] https://www.w3.org/TR/rdf11-primer

Linked Data basiert auf dem Konzept, Datenobjekte netzwerkartig zu verknüpfen, Verbindungen zu folgen, Daten zu aggregieren und ein gemeinsames Verständnis über die Bedeutung (Semantik) von Datenobjekten zu definieren. RDF ermöglicht es, Aussagen über Ressourcen zu machen. Das Format dieser Aussagen ist einfach. Eine Aussage hat immer die folgende Struktur:

<Subjekt> <Prädikat> <Objekt>

Diese Struktur wird RDF Triple genannt. Eine RDF Aussage (auch Ausdruck) drückt eine Beziehung zwischen zwei Ressourcen aus. Das Subjekt und das Objekt repräsentieren die beiden Ressourcen, die miteinander in Beziehung stehen. Das Prädikat stellt die Art ihrer Beziehung dar. Die Beziehung ist richtungsweisend formuliert (vom Subjekt zum Objekt) und wird als RDF Property bezeichnet. Die Bedeutung von Subjekt, Prädikat und Objekt ist in sog. RDF Vokabularen anhand von International Resource Identifiers (IRI) definiert.

Ein IRI identifiziert eine Ressource. Die Uniform Resource Locators (URL), die Menschen als Web-Adressen verwenden, sind eine Form von IRIs. Diesen stehen für das Festlegen von Semantik andere Formen von IRIs gegenüber, welche Identifikatoren für Ressourcen darstellen, ohne deren Standort oder den Zugang zu diesen zu implizieren. Der Begriff IRI ist eine Verallgemeinerung des Uniform Resource Identifiers (URI), der die Verwendung von Nicht-ASCII-Zeichen in der IRI-Zeichenfolge ermöglicht.

Linked Data wird über die Abfragesprache (Query Language) SPARQL von SPARQL Servern resp. sog. SPARQL Endpunkten bezogen. Linked Data Services sind im Web weit verbreitet. Eine SPARQL Query kann mehrere Linked Data Services in einer einzigen SPARQL Query ansprechen. Linked Data wird in Form von RDF Triples in sog. RDF Triplestores verwaltet. SPARQL Endpunkte bieten Zugang zu diesen RDF Triples.

10.5 Abkürzungsverzeichnis

Abkürzung	Bedeutung
ABB	Architekturboard Bund
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface, der geläufige Begriff für eine elektronische Schnittstelle
BDAT	Business, Data, Application, Technology (TOGAF Architekturebenen)
CI/CD	Continuous Integration / Continuous Deployment
CPSV-AP	Core Public Service Vocabulary Application Profile
CRUD	Create, Read, Update, Delete
DCAT-AP	Data Catalogue Application Profile
DTI	Bereich Digitale Transformation und IKT-Lenkung der Bundeskanzlei
G2G	Government to Government, womit Interaktionen zwischen öffentlichen Verwaltungen gemeint sind, auch A2A genannt, wobei A für Administration steht
G2B	Government to Business, womit Interaktionen zwischen öffentlichen Verwaltungen und juristischen Personen gemeint sind, auch A2B genannt
G2C	Government to Citizen, womit Interaktionen zwischen öffentlichen Verwaltungen und natürlichen Personen gemeint sind, auch A2C genannt
GE	Gestaltungsempfehlung
H2M	Mensch-zu-Maschine
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
gRPC	gRPC Remote Procedure Calls
IAM	Identity & Access Management
IDL	Interface Definition Language
IKT	Informations- und Kommunikationstechnologie
IoT	Internet of Things
IRI	International Resource Identifier
ISA	Interoperability Solutions for public Administrations, businesses and citizens
JSON	JavaScript Object Notation
LB	Load Balancer
LE	Leistungserbringer
LoA	Level of Assurance
M2M	Maschine-zu-Maschine

Abkürzung	Bedeutung
MDG	Master Data Government
MEP	Message Exchange Pattern
MQTT	Message Queuing Telemetry Transport
NCSC	National Cyber Security Centre
OAS	OpenAPI Specification
OASIS	Organization for the Advancement of Structured Information Standards
OGD	Open Government Data
PAMS	Polymorphic Access Management System
RDF	Resource Description Framework
REST	Representational State Transfer
SLA	Service Level Agreement
SLO	Service Level Objective
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SPARQL	SPARQL Protocol and RDF Query Language
TOGAF	The Open Group Architecture Framework
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VDTI	Verordnung über die digitale Transformation und die Informatik
VE	Verwaltungseinheit
W3C	World Wide Web Consortium
WAF	Web Application Firewall
WSDL	Web Service Description Language
XML	Extensible Markup Language
XSD	XML Schema Definition

Tabelle 47: Verwendete Abkürzungen

10.6 Abbildungsverzeichnis

10.7 Tabellenverzeichnis