Bericht

BayernCloud

Gesamtreferenzarchitektur

fortiss

BayernCloud –
Gesamtreferenzarchitektur

Herausgeber

Prof. Dr. Helmut Krcmar

Technische Universität München
Boltzmannstr. 3
85748 Garching bei München
helmut.krcmar@tum.de

Prof. Dr. Claudia Eckert

Fraunhofer-Institut für Angewandte
und Integrierte Sicherheit AISEC
Lichtenbergstraße 11
85748 Garching bei München
Claudia.Eckert@aisec.fraunhofer.de

Autoren

Julius Baecker

fortiss GmbH
Guerickestr. 25
80805 München
baecker@fortiss.org

Martin Engert

fortiss GmbH
Guerickestr. 25
80805 München
engert@fortiss.org

Julian Müller

fortiss GmbH
Guerickestr. 25
80805 München
mueller@fortiss.org

Stefan Neubig

fortiss GmbH
Guerickestr. 25
80805 München
neubig@fortiss.org

Norman Schaffer

fortiss GmbH
Guerickestr. 25
80805 München
schaffer@fortiss.org

Immanuel Kunz

Fraunhofer-Institut für Angewandte
und Integrierte Sicherheit AISEC
Lichtenbergstraße 11
85748 Garching bei München
immanuel.kunz@aisec.fraunhofer.de

Angelika Schneider

Fraunhofer-Institut für Angewandte
und Integrierte Sicherheit AISEC
Lichtenbergstraße 11
85748 Garching bei München
angelika.schneider@aisec.fraunhofer.de

Weitere Mitwirkende:
Dian Balta, Alejandro Arreola Gonzalez und Matthias Pfaff.

Gefördert durch

Bayerisches Staatsministerium für
Wirtschaft, Landesentwicklung und Energie

BayernCloud Gesamtreferenzarchitektur

Grundlagenteil

Herausgeber: Helmut Krcmar & Claudia Eckert

Datum der Erstellung	Datum letztes Update
03.06.2019	28.09.2020

Autoren (alphabetisch)
Julius Baecker (fortiss GmbH)
Martin Engert (fortiss GmbH)
Julian Mueller (fortiss GmbH)
Stefan Neubig (fortiss GmbH)
Norman Schaffer (fortiss GmbH)

Immanuel Kunz (Fraunhofer AISEC)
Angelika Schneider (Fraunhofer AISEC)

Beteiligte Partner
fortiss GmbH
Fraunhofer AISEC

Aktuelle Version
V.1.4

Versionshistorie

28.09.2020	V1.4	Grafische Aufbereitung
08.09.2020	V1.3	Erweiterung um Abschnitt 1.1 „Zur Nutzung des Dokuments" sowie eine Übersichtsgrafik zu den BayernCloud Use Cases
12.08.2020	V1.2	Ergänzung neuer Module in der technisch-funktionalen Sicht basierend auf Validierungsfeedback (#6 Dublettenprüfung, #11 Data Information Dashboard)
10.06.2020	V1.1	Ergänzung Ökosystemsicht um übergreifende Wertflüsse
31.03.2020	V1.0	Interne Reviews; Ergänzung weiterer Evaluationsmaßnahmen; Ergänzung Management Summary; Vervollständigung von Referenzen; formale Überarbeitungen und finale Anpassungen
25.03.2020	V.0.9	Interne Reviews; formale Überarbeitungen; Ergänzung Ökosystemsicht; Anpassungen in Kapitel 3; Management-Summary hinzufügen
18.03.2020	V.0.8	Konkretisierung funktional-technische Sicht (insb. BayernCloud Services); Zertifizierungsmodell als eigenes Kapitel; Ergänzung Ökosystemsicht und Verteilungssicht um weitere Use Cases; Erste formale Überarbeitung
27.02.2020	V.0.7	Review Referenzarchitekturen, Kontext der BayernCloud, Zusammenfassung und Ausblick
12.02.2020	V.0.6	Einleitung, Ergänzung Definition Referenzarchitektur, Kontext der BayernCloud
20.12.2019	V.0.5	Überarbeitung Prozesssicht
04.12.2019	V.0.4	Ergänzung Ökosystemsicht, Prozesssicht, Verteilungssicht

05.11.2019	V.0.3	Überarbeitung Struktur / Sichten der Referenzarchitektur; Ökosystemsicht
30.08.2019	V.0.2	Ergänzung Servicelandschaft
03.06.2019	V.0.1	Ersterstellung

Management Summary

Die Bereitstellung digitaler Produkte, Services und Angebote ist branchenübergreifend zu einem wesentlichen Bestandteil heutiger Wertschöpfung geworden. Daten und Informationen werden über das Internet in einer stetig zunehmenden Menge, Vielfalt und Geschwindigkeit bereitgestellt, ausgetauscht und für immer neue Anwendungsbereiche verwertet. Diese Referenzarchitektur, die im Forschungsprojekt „BayernCloud" entstanden ist, dient als Rahmenwerk zur Gestaltung digitaler Plattform-Ökosysteme, die die technische Grundlage zum Austausch von Information, Ressourcen und sonstigen Artefakten sowie der Entwicklung neuer Services bilden. Auf diese Weise können vor allem kleine und mittelständische, bayerische Unternehmen digital angebunden und vernetzt werden, um schließlich neue Geschäftsmodelle zu initialisieren.

Dementsprechend richtet sich die BayernCloud Referenzarchitektur insbesondere an kleine und mittlere Unternehmen verschiedener Branchen, die an digitalen Plattform-Ökosystemen partizipieren möchten, ebenso wie an Entwickler und Systemarchitekten und potentielle Plattformbetreiber. Eine solche Referenzarchitektur stellt umfassendes Wissen, Informationen und Rahmenbedingungen zur Entwicklung eines digitalen Plattform-Ökosystems bereit, um eine anschließende Umsetzung zu unterstützen. Es wird sowohl aus technischer, als auch betriebswirtschaftlicher und organisatorischer Perspektive aufgezeigt, welche Entscheidungen für eine Umsetzung getroffen werden müssen. Konkrete Empfehlungen für eine technische Architektur, für Governancemechanismen zur Steuerung entsprechender Ökosysteme, zur Erstellung neuartiger Geschäftsmodelle und zur Zertifizierung digitaler Plattformen helfen bei der Umsetzung. Somit wird ein systematischer Leitfaden für die Erstellung konkreter technischer Architekturen, sowie von Geschäfts- und Betreibermodellen bereitgestellt.

Die BayernCloud Referenzarchitektur ist branchenunabhängig und kann auf verschiedene Domänen übertragen werden. Hierfür können domänenspezifische Anwendungsfälle auf die Referenzarchitektur übertragen werden, um so einen Branchenbezug anhand konkreter und realer Use Cases herzustellen. Die Umsetzung dessen wird anhand des bayerischen Tourismussektors, der als Pilotdomäne dient, aufgezeigt. Von der Ermittlung der aktuellen Touristenzahl im Ort anhand von Buchungsdaten bis hin zur tagesaktuellen Kapazitätsplanungsanwendung für Berghüttenbetreiber, welcher diese Informationen nutzt, ergeben sich exemplarische Innovationspotentiale, die derzeit nicht ausgeschöpft werden. Durch eine Instanziierung dieser Referenzarchitektur im Tourismus kann ein Ökosystem geschaffen werden, um diese Potentiale in der Breite verfügbar zu machen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Die Bereitstellung digitaler Produkte, Services und Angebote ist branchenübergreifend zu einem wesentlichen Bestandteil heutiger Wertschöpfung geworden. Daten und Informationen werden über das Internet in einer stetig zunehmenden Menge, Vielfalt und Geschwindigkeit bereitgestellt, ausgetauscht und für immer neue Anwendungsbereiche verwertet. Sie dienen als maßgeblicher Treiber einer Entwicklung, die auch in etablierten Gewerben zu neuem Wettbewerb um einen nachhaltigen Erfolg führt. Als treffendes Beispiel bemühen sich etwa etablierte Unternehmen im bayerischen Tourismus darum, den sich verändernden Bedürfnissen ihrer Kundengruppen in einer zunehmend informationsgetriebenen Gesellschaft gerecht zu werden. Einerseits müssen Informationen zu Angeboten, Öffnungszeiten oder sonstigen Leistungen akkurat und stets aktuell zur Verfügung gestellt werden. Andererseits betreten immer neue Akteure mit der Entwicklung innovativer Apps, Webseiten oder Dienstleistungen den Markt und beeinflussen das Kundenverhalten oder verändern ganze Prozesse der Branche. Dieser Trend macht die Auseinandersetzung mit digitalen Angeboten zu einer bedeutsamen Herausforderung für Unternehmen, kann sich jedoch selbst für kleinere Betriebe zu einem Wettbewerbsvorteil entwickeln. So ist die Verfügbarkeit der saisonalen Öffnungszeiten einer Berghütte in gängigen Suchmaschinen ebenso von wesentlicher Bedeutung wie eine gute Bewertung auf häufig besuchten Erfahrungswebseiten, Reise-Apps oder in sozialen Medien. Es bedarf jedoch einer entsprechenden Anpassung an diese Entwicklung.

Trotz aller Herausforderungen besteht für Unternehmen auch die Chance, neue Kooperationen, Synergien und Leistungen zu schaffen, die ohne den adäquaten Einsatz moderner Technologien nicht möglich wären. Oft sind es gerade kleine und mittlere Unternehmen (KMU), die in bestimmten Nischenmärkten agieren und hierbei neue innovative Angebote und Geschäftsmodelle hervorbringen (Krcmar, Leimeister, Roßnagel, & Sunyaev, 2016). Doch die Adaptionshürden zur Anpassung an technologische Veränderung ebenso wie die Herausforderungen für einen nachhaltigen Einsatz neuer Lösungen können gerade für KMU sehr hoch sein. Es erfordert die Aufwendung von zeitlichen und finanziellen Ressourcen mit entsprechenden Risiken ebenso wie die Bereitschaft sich auf neue, digitale Ansätze und Wege einzulassen. Eine gute Möglichkeit, die Bereitstellung und Nutzung von IT-Leistungen zu erleichtern, scheint durch den Einsatz von Cloud-Services gegeben zu sein, durch die eine neue Leistungsqualität gerade für Mittelständler erschwinglicher und greifbarer werden könnte (Krcmar et al., 2016). Richtig eingesetzt bestehen durch die Nutzung von Cloud-Services weitreichende Potenziale auf individueller Ebene und durch die stärkere Vernetzung von Daten und Leistungen auch innerhalb gesamter Branchen und Domänen. Die hierbei auftretende Verbindung von Akteuren, Diensten, Informationen und sonstigen Interaktionen über eine zentrale Cloud-Plattform führt zur Entstehung sogenannter *digitaler Plattform-Ökosysteme* (DPÖ). Ein solches DPÖ ermöglicht es den beteiligten Akteuren, Daten in standardisierter Weise bereitzustellen, auszutauschen und sonstige Serviceleistungen entsprechend ihres Bedarfs einzusetzen.

Die zentrale Frage des Forschungsvorhabens BayernCloud lautet daher, wie geeignete Lösungen für bayerische Mittelständler beschaffen sein müssen, damit auch sie trotz bisheriger Marktzurückhaltung bzgl. aktueller Lösungsanbieter von branchenspezifischen Cloud-Technologien profitieren können. Dementsprechend steht im Mittelpunkt dieses Forschungsvorhabens die konzeptionelle Entwicklung zur Ausgestaltung eines

cloudbasierten DPÖ, welches mittelständische Unternehmen die Nutzung digitaler (Cloud) Technologien erleichtern und gleichermaßen Synergien fördern soll. Hierdurch sollen insbesondere Adaptions- und Akzeptanzschwellen von bayerischen KMU gesenkt werden, wodurch ein erweiterter Einsatz in unterschiedlichen Branchen durch das Aufzeigen neuer Potentiale ermöglicht werden soll.

Gekoppelt an dieses Vorhaben steht die Entwicklung einer Referenzarchitektur, anhand derer die Implementierung eines DPÖ unter Berücksichtigung der zentralen Anforderungen aus dem Mittelstand erfolgen kann. Eine solche Referenzarchitektur stellt umfassendes Wissen, Informationen und ein entsprechendes Rahmenwerk zur Entwicklung eines DPÖ bereit, um eine anschließende Umsetzung zu unterstützen. Konkreter bildet die im Rahmen dieser Grundlagenforschung zu erarbeitende *BayernCloud Referenzarchitektur* die Basis zur Entwicklung eines digitalen Plattform Ökosystems, welches auf die Bedürfnisse bayerischer KMU zugeschnitten ist. Hierbei steht speziell die Förderung von (neuem) Wettbewerb auf Basis einer BayernCloud Infrastruktur im Vordergrund. Die BayernCloud Referenzarchitektur stellt eine Grundlage für die Instanziierung in verschiedenen Domänen bereit. Als Pilotdomäne dient der bayerische Tourismus, der durch ein weiteres, kooperierendes Forschungsvorhaben, dem BayernCloud Anwendungsteil, gesondert gefördert und erarbeitet wird.

1.1. Zur Nutzung dieses Dokuments

Dieses Dokument umfasst die Ergebnisse der BayernCloud Grundlagenforschung. Im folgenden **Kapitel 2** wird hierfür zunächst eine Einordnung des Forschungsvorhabens BayernCloud in den zeitlichen und technischen Kontext gegeben. Es dient daher im Speziellen interessierten Lesern zur Motivation des thematischen Hintergrundes. Im Anschluss erfolgt die Definition und Diskussion des Referenzarchitekturbegriffs als Methodik zur Formalisierung der Ergebnisdokumentation in **Kapitel 3**. Hier wird auch ein Rückblick und Vergleich bzgl. thematisch angrenzender Referenzarchitekturen gegeben – das Kapitel thematisiert somit eher den wissenschaftlichen Hintergrund zur Erarbeitung dieses Dokuments.

Kapitel 4 beinhaltet die zentralen Ergebnisse der BayernCloud Grundlagenforschung im Rahmen der zu entwickelnden Referenzarchitektur. Es richtet sich je nach inhaltlichem Schwerpunkt an verschiedene Adressaten zur möglichen Instanziierung des BayernCloud DPÖs in einer Domäne:

- **Abschnitt 4.1** empfiehlt sich zur Erläuterung der Ergebnisdarstellung allen potenziellen Nutzern der BayernCloud Referenzarchitektur. Entsprechend wird hier sowohl das methodische Vorgehen als auch die Konzeption der Ergebnisse im Detail erläutert.
- In **Abschnitt 4.2** werden die BayernCloud Use Cases des Grundlagenteils vorgestellt. Diese reflektieren die (domänenübergreifenden) Anforderungen, die durch die entwickelten Konzepte der Referenzarchitektur adressiert werden sollen.
- Die **funktional-technische Sicht in Abschnitt 4.3** gibt den logischen Aufbau und die Funktionsweise der Referenzarchitektur in möglichst technologieneutraler Weise wieder. Hierfür wird die Architektur in einzelne Module untergliedert, deren Funktionalität für die Beantwortung der Anforderungen aus den BayernCloud Use Cases genutzt werden kann. Sie richtet sich im Speziellen an Systemarchitekten und Entwickler für eine Adaption der Referenzarchitektur.

- Die **Ökosystemsicht in Abschnitt 4.4** beinhaltet die primär nicht technischen Komponenten und Eigenschaften digitaler Plattform Ökosysteme. Sie behandelt hierbei die Interaktionen und Wertströme zwischen Akteuren eines solchen DPÖs im Detail und dient daher insbesondere möglichen Plattformbetreibern für eine mögliche Ausgestaltung innerhalb einer Domäne.
- Die **Verteilungssicht in Abschnitt 4.5** modelliert die Abbildung von Softwarekomponenten auf Hardware Ressourcen. Sie adressiert die Sichtweise von Administratoren und Plattformbetreibern und stellt hierbei das Bindeglied zwischen Implementierungslogik und physischer Infrastruktur dar.
- In der **Prozesssicht in Abschnitt 4.6** werden die dynamischen Aspekte des Systems verdeutlicht. Der Zusammenhang zwischen den Modulen aus der funktional-technischen Sicht, deren Zuordnung zu Kontrollflüssen, Kommunikationswegen und der notwendigen Synchronisation werden beschrieben. Sie adressiert in erster Linie einen möglichen Plattformbetreiber sowie Drittentwickler, die sich primär mit den vorhandenen Prozessen im Falle einer Nutzung auseinandersetzen müssen.

Anschließend behandelt **Kapitel 5** das Thema Zertifizierung im BayernCloud Kontext. In **Kapitel 6** wird die Evaluation dieser Referenzarchitektur zusammengefasst und ein abschließendes Fazit sowie ein Ausblick in **Kapitel 7** gegeben.

2. Kontext der BayernCloud und wesentliche Konzepte

Kurzübersicht des Kapitels

In diesem Kapitel wird das Forschungsvorhaben in einen zeitlich/technischen Kontext gesetzt. Dies ermöglicht die Einordnung dieser Forschung rund um das zu konzipierende BayernCloud DPÖ hinsichtlich aktueller und hoch relevanter Fragestellungen und Forschungsbereiche. Die folgenden Abschnitte spiegeln hierbei einerseits aktuelle Ergebnisse aus der Literatur wider und sind darüber hinaus durch die gemachten Erfahrungen im Rahmen der Projektlaufzeit – insbesondere im Kontext der Pilotdomäne des bayerischen Tourismus – motiviert.

2.1. Potenziale von Open Data und Data Sharing

In Zukunft wird eine zunehmende Anzahl von Diensten und Anwendungen auf der Grundlage von „Open Data" entwickelt werden (Immonen, Palviainen, & Ovaska, 2014). Der Begriff „Open Data" bezeichnet Daten, die von jedermann ohne Einschränkungen verwendet werden können (Dapp, Balta, Palmetshofer, & Krcmar, 2016). Gerade im Bereich öffentlicher Verwaltungen und Behörden gibt es länderübergreifend einen starken Trend in Richtung einer grundsätzlichen Freigabe von Daten, sogenannter Open Government Data, die ein enormes volkswirtschaftliches Potential aufweisen, dessen Wert im Bereich hoher Milliardenbeträge alleine in Deutschland prognostiziert wird (Dapp et al., 2016). Allgemein bietet Open Data zahlreiche Vorteile etwa bei der Förderung von Innovation, Effizienz, Transparenz, Zufriedenheit, Bürgerbeteiligung oder Wirtschaftswachstum (Dapp et al., 2016; Zuiderwijk, Janssen, & Davis, 2014). Solcher Nutzen tritt dabei in allen Branchen und Domänen auf, in denen Open Data zum Tragen kommt, kann jedoch in seiner Stärke variieren (Dapp et al., 2016).

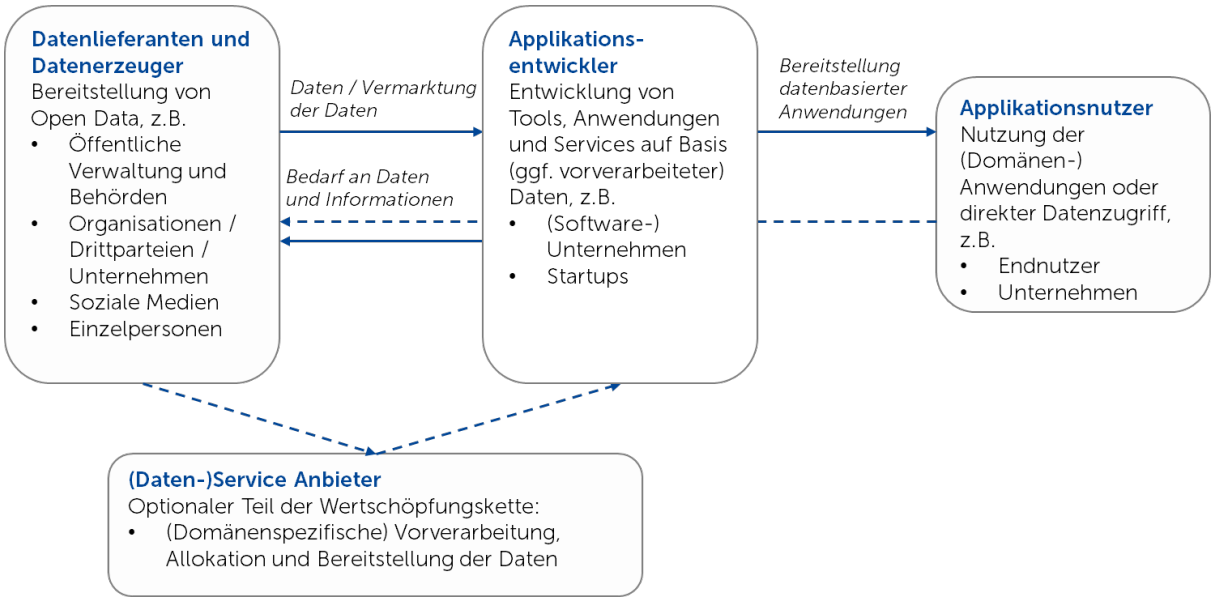

Abbildung 1: Akteure eines Open Data Ökosystems in Anlehnung an (Immonen et al., 2014)

Neben der Bereitstellung von Open Government Data stellt auch das Teilen und gemeinsame Nutzen von Domänendaten („Data Sharing") ein großes Potenzial für die Innovations- und Wettbewerbsfähigkeit von Branchen dar. So kann beispielhaft die flächendeckende Verfügbarkeit von Daten im bayerischen Tourismus (wie etwa Angaben zu Öffnungszeiten, aktuellen Veranstaltungen oder Parkplatzauslastungen) zur Attraktivität und zu einem Alleinstellungsmerkmal einer gesamten Region führen, da einerseits akkurate Informationen für Touristen verfügbar werden und andererseits neue Dienstleistungen auf Basis dieser Daten entstehen können. Hierfür müssen in einem sogenannten Open Data Ökosystem (siehe Abbildung 1) einerseits entsprechende Datenlieferanten zur Bereitstellung ihrer Daten bereit bzw. von einem entsprechenden Mehrwert überzeugt sein. Andererseits bedarf es häufig an gesonderten Möglichkeiten, diese Daten in standardisierter Form, zentral und vertrauenswürdig über geeignete Anbieter zur Verfügung stellen zu können. Unter Nutzung solcher Daten können dann Applikationsentwickler Services für mögliche Endnutzer oder Unternehmen bereitstellen.

Gerade im Bereich verteilter Unternehmensdaten (wie Öffnungszeiten, Auslastungen o.ä.) benötigt es jedoch Anreize sowie geeignete Möglichkeiten zum Teilen von Daten, welche insbesondere entsprechend kommuniziert werden müssen. Das zu konzeptionierende BayernCloud DPÖ bietet hierbei die Möglichkeit, diese Punkte auch im Kontext von Open Data zu adressieren. Hierbei kann eine entsprechende Plattform u.a. die zentrale Rolle als Zwischeninstanz zwischen Datenlieferanten bzw. Datenerzeugern und Entwicklern mit Fokus auf dem bayerischen Mittelstand in verschiedenen Branchen einnehmen. Dies verdeutlicht die Notwendigkeit und aktuelle Zweckdienlichkeit dieses Vorhabens, um die oben beschrieben Vorteile heben zu können.

2.2. Value Co-Creation und ihre Steuerung in Ökosystem

Value Co-Creation, die gemeinschaftliche Generierung von Wert ist eines der zentralen Forschungsthemen der Wirtschaftsinformatik, aber auch der Forschung in den Bereichen Marketing und Service (Hein et al., 2020; Storbacka, 2019). Dieser Forschungstrend beruft sich auf die mittlerweile zentrale Bedeutung von plattformbasierten Ökosystemen in der heutigen Welt. Die bekanntesten Beispiele wie Google, Facebook und Apple finden sich im B2C-Bereich, aber auch im B2B-Bereich finden sich bekannte Plattformökosysteme wie etwa Salesforce, SAP und Microsoft Azure. Charakterisiert werden diese digitalen Plattformen durch einen Plattformbetreiber, der Governanceregeln implementiert, um Wertschöpfungsmechanismen auf der Plattform zwischen Plattformbetreiber und einem Ökosystem aus autonomen Komplementoren und Konsumenten zu erleichtern (Hein et al. 2019). Die beteiligten Komplementoren spielen eine zentrale Rolle im Prozess der Wertgenerierung, der sogenannten Value Co-Creation (Schreieck & Wiesche, 2019). Sie profitieren von der Teilnahme an der Plattform, indem sie die vom Plattformbetreiber zur Verfügung gestellten Ressourcen in der Regel unentgeltlich nutzen können (Ghazawneh & Henfridsson, 2013). Neben vielen anderen Faktoren, welche Komplementoren zur Teilnahme bewegen, gilt eine große Kundengruppe als zentrale Einflussgröße (Engert, Pfaff, & Krcmar, 2019).

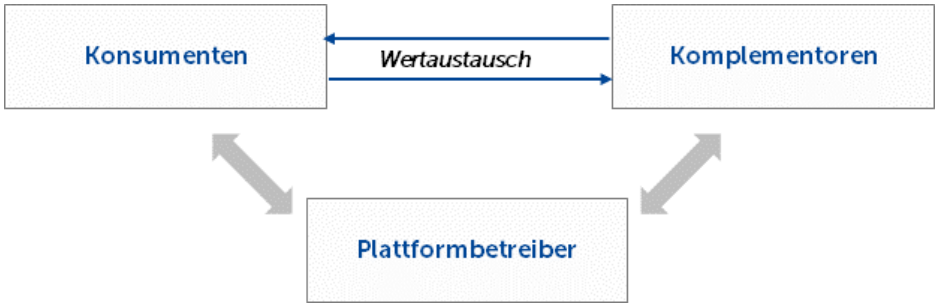

Abbildung 2: Elemente von Digitalen Plattform Ökosystemen nach (Eisenmann, Parker, & van Alstyne, 2008)

Zu den Kernaufgaben des Plattformbetreibers gehört es, einerseits eine passende Architektur zur Verfügung zu stellen, andererseits effektive Governance-Regeln zu definieren, welche die Rollen und die Interaktionen zwischen den Ökosystemteilnehmern festlegen (Schreieck, Wiesche, & Krcmar, 2016; Tiwana, 2014). Aus diesem Grund ist das Governancemodell einer der Kernaspekte von digitalen Plattformökosystemen und die Blaupause für alle Aktivitäten innerhalb des Ökosystems. Grundlegend für effektive Governance ist die Überwachung der Leistung der einzelnen Komplementoren, aber auch des Ökosystems als Ganzem (Tiwana, 2014). Aus diesem Grund müssen dem Plattformbetreiber Metriken und Tools zur Verfügung gestellt werden, die diesem das Monitoring, aber auch die aktive Steuerung ermöglichen.

Für das aus der BayernCloud Referenzarchitektur zu instanziierende DPÖ ist demnach ein Schlüsselbestandteil die Steuerung der verschiedenen Stakeholder, insbesondere der wertgenerierenden Komplementoren. Die Bereitstellung geeigneter Governancemechanismen und Steuerungsmetriken für den stufenweisen Aufbau eines umfassenden DPÖ ist somit unerlässlich für dieses Forschungsvorhaben.

2.3. Semantische Auszeichnung von Daten

Die nächste Evolutionsstufe des Word Wide Web und seiner Services sieht vor, Daten mit Semantik zu versehen um diese maschinenlesbar und maschinenverständlich zu machen und damit das namensgebende Semantic Web (Berners-Lee, Hendler, & Lassila, 284) zu erschaffen, mit dem intelligente Services und Applikationen großflächig verfügbar werden (Handschuh S., 2003). Die dazu notwendigen technischen Standards sind bereits vorhanden und werden im Rahmen der BayernCoud adaptiert.

Zunächst ist eine Syntax im Trägerformat für die Daten erforderlich, die eine semantische Annotation erlaubt. Im bekanntesten Fall ist das die JSON-LD (Sporny, Longley, Kellogg, Lanthaler, & Lindström, 2014) Notation, welches die weitverbreitete JSON-Notation um semantische Beschreibungsstrukturen erweitert. Nachdem APIs über JSON als üblichen Content Type verfügen, können diese mit minimalem Aufwand an JSON-LD angepasst werden. Zur semantischen Kontextualisierung kommt dann ein Vokabular zum Einsatz, welches aus einer Reihe von vorgefertigten Bausteinen zur Beschreibung eines Kontexts besteht. Jedes Attribut aus dem übermittelten Datensatz erhält demnach eine URI, die das Attribut spezifiziert und darüber hinaus den Kontext des Attributes erläutert. Da Daten eingepflegt und ausgetauscht werden, muss deren Gültigkeit überprüft werden. Zur Validierung können Überprüfungsstrukturen eingesetzt werden, wie etwa die Shapes

Constraint Language (SHACL), die die Konformität der Datensätze hinsichtlich vordefinierter Constraints überprüft.

Zusammen erlauben diese Technologien die Kontextualisierung der Services hinsichtlich ihrer Verwendung im Sinne des Semantic Web, dessen Entwicklung Hand in Hand mit den aktuellen Entwicklungen im Bereich der künstlichen Intelligenz einhergeht. Einerseits lässt sich das funktionale Potential der KI-Anwendungen durch die Verwendung semantisch annotierter Daten deutlich steigern (Handschuh & Staab, 2003) andererseits sind KI-Ansätze wie Machine Learning dazu geeignet, Rohdatensätze mit Annotationen zu versehen (d'Amato, 2020). Die Schnittmenge des Semantic Web und der KI findet sich im intelligenten Verhalten von Systemen, was durch semantische Annotation fundiert und aus KI-Forschung resultiert.

Es wird durch die semantische Annotation insbesondere die Grundlage zur maschinellen Auffinden eines relevanten Services durch Agenten anhand dessen Bedarfs geschaffen (Benaboud, Maamri, & Sahnoun, 2013). Dies könnte beispielsweise durch die Anbindung der semantischen Services an einen Knowledge Graph geschehen. KI-Anwendungen der Zukunft sind so beispielsweise in der Lage, domänenspezifische (z.B. touristische) Fragestellungen zu beantworten, indem ein Dienst der BayernCloud anhand der durch semantische Annotation nunmehr vorliegenden Bedeutung als relevant für die Beantwortung der Fragestellung erachtet wird und anschließend entsprechend genutzt wird.

Die semantische Annotation der gesamten Datenbasis der BayernCloud mittels standardisierter Beschreibungsschemata dient zusammenfassend also dazu, die Interoperabilität mit anderen Systemen zu fundieren, indem Maschinenlesbarkeit und Auffindbarkeit der Services gestärkt werden. Die BayernCloud etabliert sich damit als zukunftsstarkes Vorreiterprojekt für die nächste Generation des Webs.

2.4. Cloud Computing im Mittelstand

Unter Cloud Computing versteht man im Allgemeinen ein Netzwerk basiertes, on demand Modell zur Verfügung Stellung von Ressourcen wie Servern, Speicher, Netzwerk aber auch Applikationen und Services auf eine einfache und ortsunabhängig verfügbare Art und Weise (Mell & Grance, 2011). Damit hielt in den letzten Jahren eine grundlegend neue Art der Bereitstellung von Ressourcen im IT Umfeld von Unternehmen Einzug. Cloud basierte Dienste führten zu einem starken Wandel der IT Services welcher sich auf die Effizienz und Dynamik der Bereitstellung aber auch auf das insgesamt verfügbare Angebot auswirkte. So können Dienstleister beispielsweise dadurch einfacher standardisierte Lösungen anbieten und durch Skalierungseffekte auch kleinere und mittelständische Unternehmen effizient adressieren (Krcmar et al., 2016). Für spezialisierte Domänen können dadurch ebenfalls einfacher Synergiepotenziale und Innovationen im Markt geschaffen werden. Dennoch zeigte sich von Anfang an eine starke Zurückhaltung im Markt was die Adaption von Cloud basierten Angeboten angeht, wie bereits 2010 im Rahmen einer Erhebung unter deutschen CIOs ermittelt wurde (Krcmar & Leimeister, 2010). Als limitierende Faktoren wurden zwei Aspekte identifiziert. Die Informationsasymmetrie im Markt, sowie ein Mangel an Vertrauen in die Sicherheit der angebotenen Leistungen. Insbesondere die Informationsasymmetrie erschwert es den Anwendern, Cloud Angebote hinsichtlich ihrer Vor- und Nachteile

passend zu bewerten. Außerdem hindert es Anbieter von Services die Bedürfnisse der Kunden entsprechend zu adressieren (Krcmar et al., 2016).

Dieses Bild hat sich im Verlaufe der letzten Jahre zwar teilweise gewandelt, allerdings gibt es noch viel ungenutztes Potential. Sicherheitsaspekte sind nach wie vor stets Gegenstand von Diskussionen allerdings wurden bereits deutliche Fortschritte in diesem Bereich erzielt. Untersuchungen und Mechanismen zu Zertifizierungen können dabei unterstützen (Sunyaev & Schneider, 2013). Ein Vergleich mittels des Cloudmonitors im Jahr 2019 zeigt, dass die adaptionsrate von Clouddiensten in den letzten Jahren deutlich angestiegen ist, allerdings werden die Mehrwerte bisher noch nicht signifikant zur Unterstützung bei der Entwicklung neuer digitaler Geschäftsmodelle genutzt (KPMG, 2019). Unternehmen sind heutzutage gezwungen immer schneller und effizienter Innovationen im Markt voranzutreiben und ihre Geschäftsstrategie anzupassen. Dabei können entsprechend Cloud Services einen wichtigen Beitrag leisten um beispielsweise mittels open Innovation (Reichwald & Piller, 2009) branchenspezifische Lösungen zu erstellen und kontinuierlich weiter zu entwickeln.

Im Rahmen der BayernCloud ist es demnach wichtig den Fokus auf ein Cloud Konzept zu legen, das Informationen sicher, vertrauenswürdig und einfach für alle Parteien zugänglich macht. Aufbauend auf dieser Informationsinfrastruktur werden dann weitere branchenspezifische Mehrwert Dienste umsetzbar.

2.5. Veränderung und Anpassung von Geschäftsmodellen

Die Digitalisierung führt zur Veränderung bzw. vollständigen Transformation von bestehenden Geschäftsmodellen. Durch Plattform-Ökosysteme, wie dem Vorhaben BayernCloud, entstehen mehrseitige Märkte, in denen verschiedene Gruppen von Akteuren Daten und Dienste untereinander austauschen. Die um ein Vielfach erhöhte Informationsverfügbarkeit als auch die günstigere Verfügbarkeit großer Rechenkapazitäten führen verstärkt zu primär digitalen bzw. datengetrieben Geschäftsmodellen. Akteure nutzen die verfügbaren Daten und Ressourcen, um neuartige digitale Services anzubieten. Die Wertschöpfungskette wird transparenter und kürzer, wodurch teilweise Player ersetzt werden. Diese Entwicklung führt zu einer erhöhten Innovationsgeschwindigkeit, und somit auch zu einer erhöhten Frequenz von Geschäftsmodellveränderungen.

Hierbei müssen einerseits bestehende Player schnell und frühzeitig auf neue Innovationen oder neue Geschäftsmodelle reagieren. Gleichzeitig entstehen eine Vielzahl von StartUps, die in kürzester Zeit neue Geschäftsmodelle testen und ausrollen. Insgesamt müssen Geschäftsmodelle durch die Zunahme von Plattform-Ökosystem und der Ressourcenverfügbarkeit dynamischer werden. Unternehmen müssen dazu fähig sein, ihr Geschäftsmodell auf schnelle und einfache Weiße anzupassen, um optimal auf die sich kontinuierlich verändernden Bedürfnisse reagieren zu können. Gleichzeitig müssen Unternehmen, die an entsprechenden Ökosystem wie der BayernCloud teilnehmen, die Auswirkungen auf das eigene Geschäftsmodell verstehen und ggf. frühzeitige Anpassungen vornehmen. Insbesondere durch sog. datengetrieben Geschäftsmodelle, die primär auf der Analyse großer Datenmengen sowie die Aggregation heterogener Daten beruhen und individuelle und nutzerzentriete Wertversprechen herstellen, erhöht sich die notwendige Veränderungsgeschwindigkeit abermals.

Im Rahmen der BayernCloud werden die Anpassungen der Wertschöpfungsnetzwerke und Geschäftsmodelle untersucht. Darüber hinaus werden Methoden vorgeschlagen und angewendet, um schnell eine Anpassung der Geschäftsmodelle vorzunehmen. Es wird ein Tool entwickelt, mit dem die Dynamik der individuellen Geschäftsmodelle abgebildet werden kann und Simulationen bzgl. verschiedener strategischer Geschäftsmodellentscheidungen durchgeführt werden können, um potentielle Anwender optimal auf diese Veränderungen vorzubereiten und Entscheidungen transparent und proaktiv zu fördern.

3. Merkmale von Referenzarchitekturen

Kurzübersicht des Kapitels

In diesem Kapitel wird der abstrakte Begriff einer Referenzarchitektur eingeführt und diskutiert. Hierunter fällt insbesondere die Definition einer Referenzarchitektur (Abschnitt 3.1.1) ebenso wie die Einordnung der BayernCloud Referenzarchitektur anhand eines geeigneten Klassifikationsschemas (Abschnitt 3.1.2). Zudem wird das methodische Vorgehen zur Erarbeitung der BayernCloud Referenzarchitektur anhand der wesentlichen Prozessschritte in Abschnitt 3.2 vorgestellt. Dieses enthält insbesondere einen Vergleich mit bestehenden Vorarbeiten.

3.1. Begriffliche und theoretische Grundlagen

Referenzarchitekturen eignen sich zur Beschreibung komplexer Systeme und Artefakte. Hierbei sollen sie Wissen Richtlinien oder geeignete Blaupausen vorgeben, um die Umsetzung und Entwicklung zu erleichtern (Reidt, Pfaff, & Krcmar, 2018). Dies ist vor allem dann von Bedeutung, wenn verschiedene Interessengruppen oder Unternehmen in dem zu entwickelnden Artefakt adressiert werden sollen. Der Entwicklungsprozess wird dann dadurch erleichtert, dass sich alle Beteiligten an den Richtlinien orientieren können und Teile der bereitgestellten Inhalte wiederverwenden oder adaptieren können. Dies fördert insbesondere die domänenübergreifende Kommunikation mit potentiellen Stakeholdern (Angelov & Grefen, 2008).

3.1.1. Definition Referenzarchitektur

Der Begriff einer Referenzarchitektur weist in der Literatur eine große Heterogenität auf. Entsprechend gibt es hierzu bislang keine einheitliche Definition, sondern wird durch die verschiedenen Autoren mit unterschiedlichem Fokus belegt. So kann er hinsichtlich der Stärke der technischen Ausrichtung variieren oder über einen (praxisnahen) Domänenbezug definiert sein. (Reidt et al., 2018) geben einen Überblick über verschiedene Definitionen zum Begriff einer Referenzarchitektur und leiten eine eigene Definition her, welche sich am Referenzmodellbegriff von (Vom Brocke, 2015) orientiert und für die Definition einer Referenzarchitektur adaptiert wird. Die in diesem Forschungsprojekt entwickelte Referenzarchitektur folgt dieser konsolidierten Definition:

„Eine Referenzarchitektur ist eine abstrakte Architektur, die den Menschen die Entwicklung von Systemen, Lösungen und Applikationen erleichtern soll, indem sie Wissen bereitstellt und einen Rahmen zur Entwicklung vorgibt. Die Beziehung zwischen Referenzarchitektur und konkreter Architektur ist dadurch gekennzeichnet, dass Gegenstand oder Inhalt der Referenzarchitektur bei der Konstruktion der konkreten Architektur des jeweiligen zu entwickelnden Systems (wieder-)verwendet werden. Die Referenzarchitektur besitzt einen technischen Fokus, verbindet diesen jedoch mit dem dazugehörigen Fachwissen der jeweiligen Domäne. Sie bildet durch ihre Ausprägung und ihren Inhalt ein gemeinsames Rahmenwerk, um die detaillierten Diskussionen aller bei der Entwicklung beteiligten Stakeholder geführt werden können" (Reidt et al., 2018).

Die eingeführte Definition eignet sich gut für die angestrebten Zielsetzungen zur Konzeptionierung eines DPÖs im Rahmen der BayernCloud Grundgenforschung. Der Fokus dieser Referenzarchitektur umfasst dementsprechend die folgenden Schwerpunkte:

- Die Bereitstellung von Wissen und eines umfassenden Rahmenwerks zur Entwicklung eines DPÖs in Form von unterschiedlichen Referenzbausteinen (u.a. eine technische Architektur, ein Geschäfts- und Governancemodell sowie rechtliche Rahmenbedingungen), die sich für die Anwendung in einer Zieldomäne (wieder-) verwenden lassen.
- Die Fokussierung auf technische Aspekte unter Berücksichtigung und Hinzunahme domänenspezifischer Anforderungen und Use Cases (siehe Abschnitt 4.2).
- Trotz des technischen Fokus ermöglicht sie durch die Erarbeitung verschiedener Sichten die Diskussion und Kommunikation aller an der Entwicklung beteiligter Stakeholder.

Gemäß dieser Definition soll in einer Referenzarchitektur der Teil an Informationen vorhanden sein, der für die konkretere Weiterentwicklung einen entsprechenden Rahmen zur Verfügung stellt. Sie ist damit ein Referenzmodell (Vom Brocke, 2015), jedoch mit einem technischen Charakter, da sie konkrete Architekturentscheidungen und ggf. auch Softwareelemente beinhaltet.

3.1.2. Einordnung der BayernCloud Referenzarchitektur

Aufbauend auf der Arbeit von (Reidt, 2019) lassen sich Referenzarchitekturen anhand verschiedener Charakteristika näher klassifizieren. Für die BayernCloud Referenzarchitektur ist eine solche Klassifikation in Tabelle 1 dargestellt.

Charakteristik	Ausprägung		
Abstraktionsgrad	Detailliert (Codebasis)	Mischform	Abstrakt
Technologieneutralität	Ja	Teilweise	Nein
Industriefokus	Industriespezifisch		Industrieübergreifend
Produktfokus	Fokus auf ein Produkt	Produktfamilie	Produkt-übergreifend
Unternehmensfokus	Unternehmensspezifisch		Unternehmensübergreifend
Referenzcharakter	Bezugspunkt	Allgemeingültigkeit	Empfehlungs-charakter
Variationspunkt	Enthalten		Nicht enthalten
Technischer Fokus	Rein technisch		Technisch mit Domäneninformationen
Vollständigkeit	Vollständig		Unvollständig
Praxis- oder Forschungsgetrieben	Praxis		Forschung
Ziel	Standardisierung		Erleichterung
Vorgehensweise	Induktiv	Kombination	Deduktiv
Enthaltenes Wissen	Architekturwissen	Softwareelemente	Richtlinien
	Weiteres Wissen:		

Tabelle 1: Klassifikationsschema der BayernCloud Referenzarchitektur

Anhand des Klassifikationsschemas lässt sich die BayernCloud Referenzarchitektur tiefergehend charakterisieren: Sie besitzt einen hohen Abstraktionsgrad, welcher durch die erforderliche Technologieneutralität und die Gewährleistung einer domänen- bzw.

industrieübergreifenden Allgemeingültigkeit untermauert wird. Das heißt, dass sie für verschiedene Domänen eingesetzt und verwendet werden kann. Eine zentrale Anforderung an die BayernCloud Referenzarchitektur ist die Unabhängigkeit von spezifischen Betreibern oder Produkten. Diese Unabhängigkeit gilt insbesondere für mögliche Anbieter von Cloud-Infrastruktur. Es besteht keine Einschränkung auf ein bestimmtes Unternehmen oder eine wesentliche Steakholdergruppe. Die Referenzarchitektur soll als Bezugspunkt für verschiedene Domänen mit Empfehlungscharakter dienen und gewährt für die wesentlichen Komponenten Allgemeingültigkeit. Hierbei kann die Referenzarchitektur von möglichen Verwendern instanziiert und hinsichtlich ihrer Empfehlungen evaluiert werden. Sie liefert dabei mehrere Variationspunkte (z.B. die Notwendigkeit einer domänenspezifischen Instanziierung des Governancemodells), bietet jedoch neben technischen keine Domäneninformationen. Aufgrund der Möglichkeiten zur Erweiterung der Referenzarchitektur ist sie nicht vollständig. Darüber hinaus ist sie sowohl forschungs- als auch praxisgetrieben. Die Ziele der Referenzarchitektur liegen in einer Erleichterung bei der Schaffung von DPÖs auf Basis der vorhandenen Konzepte und insbesondere in der Standardisierung und Vereinheitlichung für die Bereitstellung von Wissen innerhalb einer Instanziierung der Referenzarchitektur (z.B. in Form von einheitlichen Datenschemata). Durch die Kombination wissenschaftlicher Ergebnisse für die Verwendung der Referenzarchitektur in der Praxis und der Erhebung von empirischen Anforderungen aus der Domäne, weist die BayernCloud Referenzarchitektur eine induktive und deduktive Vorgehensweise auf. Insgesamt stellt die BayernCloud Referenzarchitektur umfangreiches Architekturwissen sowie Softwareelemente zur Verfügung und gibt die wesentlichen Richtlinien und Rahmen für die Entwicklung eines DPÖs vor.

3.2. Methodik zur Erstellung der BayernCloud Referenzarchitektur

Referenzarchitekturen haben sich bereits in früherer Forschung zur Beschreibung und Konzeption komplexer Systeme oder sonstiger Artefakte bewährt, um eine Entwicklung in der Praxis zu erleichtern. Damit einhergehend nimmt die Anzahl an veröffentlichten Referenzarchitekturen insbesondere durch Trends wie Industrie 4.0 oder Digitalisierung stetig zu (Reidt 2019). Die Erstellung dieser Referenzarchitektur orientiert sich an bestehenden Referenzarchitekturen, wird jedoch hinsichtlich der spezifischen Anforderungen an die Entwicklung eines DPÖ angepasst.

3.2.1. Vergleich und Abgrenzung zu bestehenden Referenzarchitekturen

Als maßgebliche Vorarbeiten dieser Arbeit sind die Referenzarchitektur nach (Reidt, 2019) und des International Data Spaces (IDS) (Otto, Steinbuß, Teuscher, & Lohmann, 2019) zu nennen. Aus diesen wurden wesentliche Strukturkomponenten übernommen, um damit die Inhalte, Ergebnisse und Konzepte der BayernCloud Grundlagenforschung in bewährter Weise zu dokumentieren. Neben diesen Referenzarchitekturen wurden weitere Veröffentlichungen verglichen und für eine mögliche Adaption herangezogen. **Fehler! V erweisquelle konnte nicht gefunden werden.** gibt einen Überblick über thematisch und z.T. methodisch angrenzende Referenzarchitekturen. Diese stellen nur einen Auszug der umfassenden Literatur zu diesem Themengebiet heraus und sollen insbesondere den Fokus auf möglichst relevante Vorarbeiten legen.

Referenz-architektur	Autoren	Jahr	Zielbereich	Zusammenfassung
International Data Spaces (IDS) Reference Architecture Model	Otto et al.	2019	Unterstützung von sicherem und standardisiertem Daten Austausch bzw. Verknüpfungen auf Unternehmensebene u.a. zur Erleichterung von (unternehmens-übergreifenden) Smart-Services	Die IDS Referenzarchitektur gibt nach einer kurzen Einleitung mit Überblick über das Gesamtdokument einen Einblick in den aktuellen thematisch / technologischen Kontext. Danach werden in Anlehnung an das 4+1 Sichtenmodell nach Kruchten verschiedene Ebenen vorgestellt: Im "Business Layer" werden insbesondere alle beteiligten Akteure im Detail beschrieben. Das "Functional Layer" gibt eine abstrakte Beschreibung (ohne technische Elemente) zu Services des IDS ab. Im "Process Layer" werden zentrale Prozesse (z.B. Onboarding oder Datenaustausch) in BPMN dargestellt. Das "Information Layer" spezifiziert Vereinbarungen zum IDS zwischen den (Projekt-) Teilnehmern zur Kompatibilität und Interoperabilität. Hierbei werden verschiedene Aspekte sehr umfassend beschrieben, um "digitale Ressourcen" im IDS zu repräsentieren (Inhalt, Kontext, Kommunikation etc.). Innerhalb des "System Layers" wird eine Daten- und Servicearchitektur vorgestellt. Hierbei wird insbesondere auf die drei wichtigen Komponenten des "Functional Layer" eingegangen. Abschließend werden "Layer"-übergreifend die Perspektiven zu Security, Zertifikation und Governance behandelt.

Referenzarchitektur eines integrierten Informations-systems zur Unterstützung der Instandhaltung	Reidt	2019	Informationssysteme in der Instandhaltung im produzierenden Gewerbe	Die Referenzarchitektur (RAII) nach Reidt fokussiert sich auf Informationssysteme zur Unterstützung der Instandhaltung. Hierfür werden allgemeine Richtlinien für die Entwicklung solcher Systeme bereitgestellt. Die Referenzarchitektur gibt eine Blaupause eines solchen Informationssystems aus vier Sichten wieder: Use Case-Sicht, Prozesssicht, Funktionale Sicht sowie Verteilungssicht. Hieraus können Anforderungen, Module und Prozesse abgeleitet werden.
A Reference Architecture for Digital Ecosystems	Averian	2018	Digitale Ökosysteme	Das Paper liefert eine anbieter- und technologieunabhängige Referenzarchitektur für digitale Ökosysteme inkl. Use Case. Zunächst werden die Komponenten eines digitalen Ökosystems definiert. Anschließend werden allgemeine Leitprinzipien von digitalen Ökosystemen (z.B. Heterogenität, Skalierbarkeit etc.) oder zur Kommunikation in Ökosystemen gegeben, die als Voraussetzung an ein solches Ökosystem verstanden werden können. Insgesamt werden sehr abstrakte Beschreibungen zu Komponenten, Interaktionen sowie Adaptionen eines solchen Systems aufgeführt. Die Referenzarchitektur wird beispielhaft anhand eines Supply Chain Ökosystems verdeutlicht.
Cloud-RA: A Reference Architecture for Cloud Based Information Systems	Kiswani et al.	2018	Cloud Computing; Microservice Architektur	Die Referenzarchitektur von Kiswani et al. liefert eine eher High-Level Beschreibung einer (generischen) Cloud Microservice Architektur bestehend aus einem Schichtenmodell. Dieses wird anhand einer Service Komposition für die verschiedenen Schichten beispielhaft instanziiert.

Service-Oriented Reference Architecture for SmartCities	Clement et al.	2017	IoT; Verknüpfung Cloud Computing und Edge Computing	Clement et al. geben eine Beschreibung aktueller Themen rund um die Entwicklung von Smart Cities und den (technischen) Übergang zu diesen. Zudem wird eine Einteilung der Systeme in verschiedene Bereiche (z.B. Mensch, Logistik, Versorger & Energie etc.) bereitgestellt. Die Referenzarchitektur ist ein High-Level Schichtenmodell für verschiedene Domänen.
Data Driven Reference Architecture for Smart City Ecosystems	Abu-Matar / Davies	2017	Ökosysteme, Fokus auf die Datenperspektive	Aufgrund inhaltlicher Überschneidungen, können beide Veröffentlichungen zusammen betrachtet werden. In den Publikationen wird eine abstrakte Referenzarchitektur zu Smart City "Capabilities" vorgestellt, d.h. welche Blickwinkel man zu einer Capability (bspw. smart Parking) einer Smart City mitbringen sollte (z.B. Teilnehmer-/ Nutzerperspektive, Businessperspektive, Datenperspektive usw.). Angelehnt an das methodische Vorgehen nach Kruchten (1995) entstehen auf diese Weise sieben Sichten mit einer vereinheitlichenden Sicht.
Towards a Software Defined Reference Architecture for Smart City Ecosystems	Abu-Matar	2016		
Building a security reference architecture for cloud systems	Fernandez et al.	2014	Sicherheit in der Cloud	Die Publikation stellt einen systematischen Ansatz vor, um Cloud Security Referenzarchitekturen zu entwickeln. Hierbei wird auf Sicherheitsanforderungen für Cloud Systeme eingegangen, wobei Stakeholder (z.B. Service Konsument, Service Provider, Cloud Administrator etc.) und für die Sicherheit relevante Use Cases (z.B. Login, User anlegen, Usern Rechte zuweisen etc.) aus Cloud Systemen eingeführt werden. Des Weiteren werden "Security Patterns" für Cloud Systeme präsentiert, d.h. dass einerseits mögliche Bedrohungen aufgeführt werden und entsprechende

				Gegenmaßnahmen betreffender Akteure.
CCRA: Cloud Computing Reference Architecture	Liu et al.	2012	Cloud Computing	In dieser Publikation wird eine abstrakte Cloud Computing Referenzarchitektur mit beispielhaften Instanziierungen vorgestellt. Ziel ist zunächst die Bereitstellung einer architektonischen Cloud Computing Blaupause zur Entwicklung für Unternehmen bzw. deren Software Architekten (z.B. für Design Entscheidungen). Hierfür wird ein Modell mit verschiedenen Schichten bereitgestellt ebenso wie deren architektonische Bausteine. Anschließend werden beispielhaft eine PaaS- und eine SaaS-Architektur auf Basis der Referenzarchitektur hergeleitet, die entsprechende Schichten und architektonische Bausteine nutzen.

Tabelle 2: Übersicht thematisch angrenzender Referenzarchitekturen

Die aufgeführten Referenzarchitekturen können sowohl als inhaltliche als auch als methodische Vorlagen für die BayernCloud Referenzarchitektur herangezogen werden. Bewährt hat sich in diesem Kontext die geeignete Adaption eines Sichten Modells in Anlehnung an Kruchten (1995), welches etwa in den Arbeiten von (Reidt, 2019), (Otto et al., 2019) oder (Abu-Matar & Davies, 2017) verwendet wurde. Ähnlich wie in diesen Vorarbeiten, wurde das Sichten Modell im Kontext der BayernCloud durch das Einführen einer spezifischen „Ökosystemsicht" angepasst (siehe insbesondere Abschnitt 4.4), um dem Fokus auf DPÖs in ausreichender Form gerecht zu werden. Darüber hinaus steuerten diese Arbeiten wertvolle Inhalte zur Erarbeitung und Strukturierung von Use Cases bei (vgl. Cockburn, 2006).

Inhaltliche Überschneidungen ergeben sich vor allem auf architektonischer Ebene im Kontext von Microservice Architekturen im Bereich Cloud Computing (Cheng & Liu, 2012, (Kiswani, Dascalu, & Harris Jr, 2018) oder mit dem Fokus auf Daten in digitalen Ökosystemen (Abu-Matar & Davies, 2017; Otto et al., 2019). Abgrenzungen zu den bestehenden Arbeiten liegen insbesondere in der Fokussierung auf DPÖs für den Mittelstand: Viele Themen, die in den vorhandenen Referenzarchitekturen angeschnitten werden, eignen sich für eine Adaption (z.B. Cloud Microservice Architekturen oder einheitlichen (Daten-)Standards), müssen jedoch in diesem spezifischen Kontext aufgegriffen, konsolidiert und ggf. umgestaltet werden. Zudem gehen die im Rahmen der BayernCloud untersuchten Wissensbausteine für die Konzeptionierung eines DPÖs für den Mittelstand in wesentlichen Bereichen über bestehende Arbeiten hinaus (z.B. im Hinblick auf adaptierbare Governance- oder Geschäftsmodelle). Entsprechend liefert keine der bisherigen Arbeiten den Fokus und

hierbei die thematische Detailtiefe, die den Zielsetzungen des Forschungsvorhabens BayernCloud entsprechen.

3.2.2. Entwicklung der BayernCloud Referenzarchitektur

Das Vorgehen zur Erstellung der BayernCloud Referenzarchitektur baut auf zwei wesentlichen Vorarbeiten auf. Zum einen gibt der Projektantrag der BayernCloud Grundlagenforschung einen strukturierten Rahmen zur Bearbeitung und zur Zusammenstellung der Ergebnisse vor. Diese Vorgaben gliedern sich jedoch eher in konkrete Arbeitspakete, die es im Projektkontext zu bearbeiten gilt. Andererseits wurden für die finale Dokumentation verschiedene, bereits veröffentlichte Referenzarchitekturen herangezogen und hinsichtlich ihrer strukturellen und methodischen Eignung bewertet (vergleiche Abschnitt 3.3.1). Die BayernCloud Referenzarchitektur basiert auf einer Synthese dieser Vorgaben, indem sie einerseits die Erarbeitung der verschiedenen Referenzbausteine (Geschäfts- und Governancemodell, die technische Architektur sowie die rechtlichen Rahmenbedingungen) anhand der im Antrag vorgesehenen Anforderungen erbringt. Andererseits wird für die finale Konsolidierung der Ergebnisse bei der Erstellung der Referenzarchitektur auf wissenschaftlichen Arbeiten aufgebaut. Zu nennen sind hierbei insbesondere die in Abschnitt 3.3.1 aufgeführten Referenzarchitekturen, an denen sich diese Dokumentation methodisch orientiert. Das im folgenden Kapitel behandelte 4+1 Sichtenmodell nach Kruchten (1995) ist ebenfalls ein Ergebnis dieser Synthese und wurde für den Fokus dieser Arbeit hinsichtlich der spezifischen Entwicklung eines DPÖs angepasst.

Die untersuchten Referenzarchitekturen bieten sich als grundsätzlicher Rahmen für die Erstellung der BayernCloud Referenzarchitektur an. Für die unabhängige Bearbeitung der verschiedenen Arbeitspakete über die gesamte Projektlaufzeit, lässt sich jedoch keine der gefundenen, methodischen Vorgehensweisen direkt auf den vorliegenden Anwendungsfall übertragen. Aus diesem Grund folgt die Methodik zur Erstellung der BayernCloud Referenzarchitektur einer adaptierten Version der unterschiedlichen, zuvor untersuchten Vorgehensweisen. Dieses Vorgehen umfasst dabei die folgenden Schritte:

1. Festlegung des Ziels der BayernCloud Referenzarchitektur: Insbesondere gegeben durch den Forschungsantrag. Zudem wurden Anforderungen aus der Pilotdomäne im Tourismus erhoben.
2. Literaturrecherche: Betrachtung wissenschaftliche Vorarbeiten aus angrenzenden Bereichen (Referenzarchitekturen, Open Data, Geschäftsmodelle, Cloud Computing, Plattform Governance etc.).
3. Situationsbeschreibung unter der Erhebung von Anforderungen aus der Praxis: Diese wurden insbesondere zur Formulierung der BayernCloud Use Cases herangezogen (siehe Abschnitt 4.2).
4. Erstellung von Wissen und Konzepten im Rahmen der BayernCloud Arbeitspakete: Die Ergebnisse der Projektlaufzeit werden in Kapitel 4 sowie in den Abschlussberichten aller einzelnen Arbeitspakete zur Verfügung gestellt.
5. Konsolidierung und Synthese der Arbeitsergebnisse zu generischen Referenzbausteinen: Zur Erstellung der Referenzarchitektur wurden die verschiedenen Arbeitspakete für die Integration in das Zieldokument angepasst – d.h. die Ergebnisse in einem gewissen Rahmen konsolidiert. Detailliertere Einblicke ergeben sich in den Abschlussberichten der einzelnen Arbeitspakete.

6. Zusammenstellung der Ergebnisse in der BayernCloud Referenzarchitektur: In Vorbereitung des Projektabschlusses, wurden alle Ergebnisse in eine methodisch bewährte (vergl. Abschnitt 3.3.1), finale Dokumentation überführt.
7. Feedbackzyklen durch Experten und Anwendungsteil: Um die Ergebnisse zu validieren, wurde ein Validierungskonzept entwickelt, welches den iterativen Austausch mit Experten und Anwendungsteil zur Bewertung der Ergebnisse vorsieht (vergleiche hierzu auch Abschnitt 6.1).

4. BayernCloud Referenzarchitektur

Kurzübersicht des Kapitels

Dieses Kapitel umfasst die zentralen Ergebnisse der BayernCloud Referenzarchitektur. Hierunter fallen die wesentlichen Projektergebnisse aus den Arbeitspaketen der BayernCloud Grundlagenforschung. Abschnitt 4.1 gibt eine Einleitung in den Aufbau und das Vorgehen zur Vorstellung der Ergebnisse. Anschließend werden in Abschnitt 4.2 die BayernCloud Use Cases eingeführt und ihre Verwendung im Kontext der verschiedenen Sichten erläutert. Hierauf aufbauend befassen sich die Abschnitte 4.3, 4.4, 4.5 und 4.6 mit den verschiedenen Sichten der BayernCloud Referenzarchitektur. Sie reflektieren insbesondere die bereitgestellten Use Cases aus unterschiedlichen Blickwinkeln.

4.1. Einleitung

Die BayernCloud Referenzarchitektur umfasst die wesentlichen Forschungsergebnisse der BayernCloud Grundlagenforschung. Der Antragsstellung entsprechend, werden diese im Rahmen einer Referenzarchitektur zur Konzeption eines digitalen Plattform Ökosystems bereitgestellt (siehe zum Thema „Referenzarchitekturen" insbesondere Abschnitt 3).

Für die BayernCloud Referenzarchitektur wurde eine Darstellungsart gewählt, welche sich an das 4+1-Sichten Softwarearchitekturmodell von Kruchten (1995) und an eine angepasste Version von Reidt et al. (2019) anlehnt. Diese Art von Architekturdarstellung, die auf verschiedenen Sichten bzw. Sichtweisen beruht, ist eine typische Darstellungsform zur Entwicklung komplexer IT-Systeme. Sie wurde insbesondere bereits in verschiedenen Referenzarchitekturen als methodische Grundlage herangezogen (siehe Abschnitt 3.2.1).

Abbildung 3 zeigt den grundsätzlichen Aufbau des 4+1 Sichtenmodells. Hierbei werden die zentralen Ergebnisse der BayernCloud Grundlagenforschung anhand vier wesentlicher Blickwinkel strukturiert, die insbesondere auch verschiedene Stakeholder adressieren. Das „+1" im Sichtenmodell umfasst die BayernCloud Use Cases (vergl. Abschnitt 4.2), welche im Rahmen jeder dieser vier Sichten diskutiert werden. Die Use Cases befinden sich damit inhaltlich auf einer den Sichten übergeordneten Betrachtungsebene, da sie zur Analyse der weitgehend autarken Ergebnisse übergreifend herangezogen werden (siehe Abbildung 3).

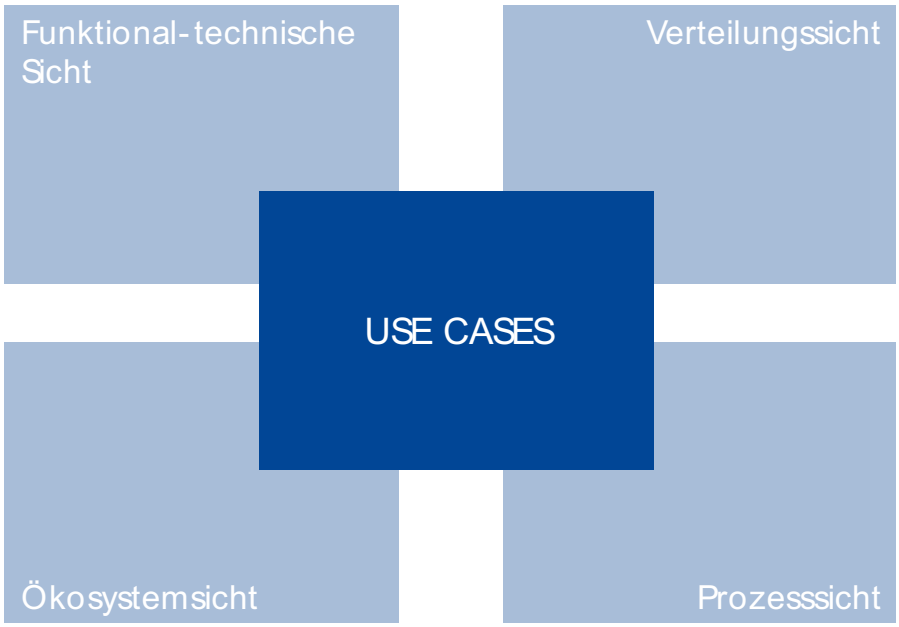

Abbildung 3: 4+1 Sichten der BayernCloud Referenzarchitektur in Anlehnung an Reidt et al. (2019) und Kruchten (1995)

Die vier Sichten werden in den Abschnitten 4.3, 4.4, 4.5 und 4.6 unabhängig voneinander vorgestellt – kurz zusammengefasst ergeben sich die folgenden Schwerpunkte und Adressaten:

1. **Funktional-technische Sicht:**
   Unter Anpassung der logischen Sicht nach Kruchten werden der logische Aufbau und die Funktionsweise der Referenzarchitektur in abstrakter, möglichst technologieneutraler Weise in dieser Sicht beschrieben. Zu diesem Zweck wird die Architektur in einzelne Bausteine, Module und Anforderungen untergliedert. Der Kern dieser Sicht umfasst die Module, die Beschreibung der Funktionalität dieser Module sowie die Darstellung der Verbindungen der Module zueinander. Dies adressiert insbesondere mögliche **Entwickler und Systemarchitekten** für eine Adaption der Referenzarchitektur.

2. **Ökosystemsicht:**
   Die Ökosystemsicht adressiert verschiedene, primär nicht technische Komponenten und Eigenschaften digitaler Plattform Ökosysteme. Im Detail behandelt diese Sicht der Referenzarchitektur die Interaktionen und Wertströme zwischen Akteuren eines auf Basis der Referenzarchitektur instanziierten Plattform-Ökosystems. Entsprechend dient sie insbesondere einem möglichen **Plattformbetreiber** als zentralem Element eines solchen DPÖs zu konkreten Ausgestaltung in einer Domäne. Durch den Fokus auf die Interaktion verschiedener Akteure, ist diese Sicht auch speziell für die weiteren **(Domänen-) Stakeholder** von großer Bedeutung (z.B. Drittentwickler oder KMU).

3. **Verteilungssicht:**
   Die Verteilungssicht der Referenzarchitektur modelliert die Abbildung von Softwarekomponenten auf Hardware Ressourcen, entsprechend des 4+1 Sichten Modells nach Kruchten. Neben der Aufteilung einzelner Module werden dabei auch Kommunikationsmuster und Schnittstellen zwischen diesen berücksichtigt. Sie adressiert die Sichtweise von **Administratoren und Plattformbetreibern** innerhalb des Ökosystems und stellt hierbei das Bindeglied zwischen Implementierungslogik und physischer Infrastruktur dar.

4. **Prozesssicht:**
   In der Prozesssicht werden die dynamischen Aspekte des Systems verdeutlicht. Der Zusammenhang zwischen den Modulen aus der funktional-technischen Sicht, deren Zuordnung zu Kontrollflüssen, Kommunikationswegen und der notwendigen Synchronisation werden beschrieben. Dadurch wird das Laufzeitverhalten ersichtlich und zusätzlich nichtfunktionale Anforderungen wie Parallelität, Verteilung, Integration, Performance und Skalierbarkeit hervorgehoben. Auf diese Weise werden in der Prozesssicht die Prozesse der BayernCloud Referenzarchitektur abgebildet. Sie adressiert hiermit in erster Linie einen möglichen **Plattformbetreiber sowie Drittentwickler**, die sich primär mit den vorhandenen Prozessen im Falle einer Nutzung auseinandersetzen müssen.

Im Detail sind in den Sichten folgende Themenkomplexe und Modelle enthalten (siehe Abbildung 4).

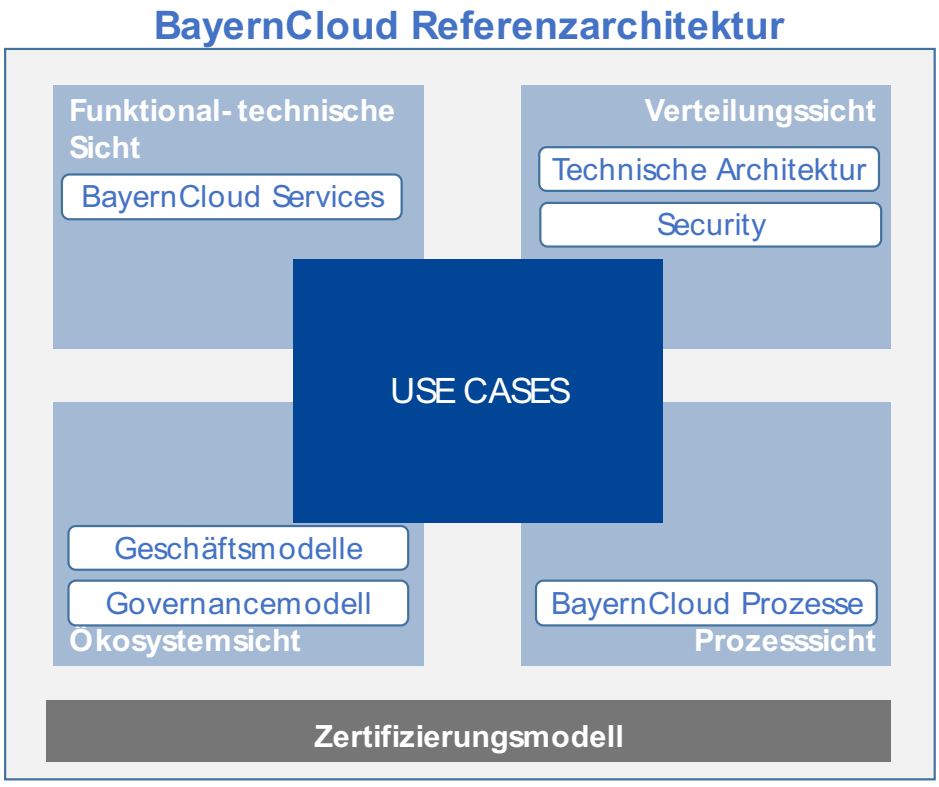

Abbildung 4: BayernCloud Referenzarchitektur

Neben den vier Sichten erfolgt die Entwicklung eines Zertifizierungsmodells für die BayernCloud Referenzarchitektur. Im Gegensatz zu den Sichten, die auf Use Case Spezifika eingehen, behandelt das Zertifizierungsmodell Use-Case übergeordnet die Zertifizierung eines auf Basis dieser Referenzarchitektur instanziierten digitalen Plattform-Ökosystems.

4.2. Use Cases

In diesem Kapitel werden die BayernCloud Use Cases als ein zentrales Element der zu entwickelnden Referenzarchitektur eingeführt und ihre Verwendung im Rahmen dieser Dokumentation erläutert. Entsprechend liefert Abschnitt 4.2.1 eine Definition sowie die wesentlichen Punkte zum Aufbau, zur Strukturierung und zur Beschreibung von Use Cases. Anschließend gibt Abschnitt 4.2.2 eine Erklärung zur Verwendung der BayernCloud Use Cases im Rahmen dieser Referenzarchitektur im Kontext des 4+1 Sichtenmodells von Kruchten (1995). Die Use Cases zur Entwicklung des BayernCloud DPÖ werden dann einzeln in den restlichen Abschnitten dieses Kapitels vorgestellt.

4.2.1. Aufbau und Beschreibung von Use Cases

Im Kontext der Entwicklung dieser Referenzarchitektur beschreibt ein Use Case eine grundlegende, domänenübergreifende und elementare Anforderung an das später zu instanziierende digitale Plattform Ökosystem bzw. an die zugrundeliegende Referenzarchitektur. Die Use Cases dienen als Vorgaben zur Ausgestaltung der Referenzarchitektur (siehe hierzu insbesondere Abschnitt 4.2.2) und werden aus Anforderungen des Projektantrages und aus Anforderungen des Anwendungsteils, d.h. durch Erkenntnisse aus der zu pilotierenden Anwendungsdomäne, gewonnen. Der Fokus von Use Cases liegt darin, eine übersichtliche und möglichst allgemeingültige Darstellung der Anforderungen an das DPÖ zu ermöglichen. Auf diese Weise schaffen sie eine Formalisierung der zu adressierenden Anforderung, auf deren Basis verschiedene Themen zur Ausgestaltung der Referenzarchitektur parallel erarbeitet werden können (z.B. technische oder Geschäftsmodell-spezifische Themenpunkte).

Die Verwendung von Use Cases wurde bereits in bestehenden Forschungsarbeiten zur Beschreibung von Anforderungen behandelt, deren Ergebnisse für die Entwicklung von BayernCloud Use Cases herangezogen werden. Entsprechend gibt (Cockburn, 2006) in seinen Arbeiten über die Formulierung von Use Cases zentrale Aspekte an, um ihnen eine zielführende Struktur als allgemeines Hilfsmittel zu geben. Die Beschreibung der BayernCloud Use Cases in den folgenden Abschnitten orientiert sich an diesem Aufbau:

- **Name / Bezeichnung des Use Cases** beschrieben aus der Perspektive des zu adressierenden Akteurs.
- **Zweck / Zielsetzung des Use Cases** im Kontext der Anwendung.
- **Akteur / Rolle** des zu adressierenden Anwendungsfalls. Hierbei können mehrere Parteien des DPÖ involviert sein, zumindest jedoch ein Primärakteur, aus dessen Sicht der jeweilige Use Case benannt und ausformuliert wird.
- Der **Umfang / Scope des Use Cases** beinhaltet das zu adressierende System bzw. den betreffenden Bereich des BayernCloud DPÖ.
- Die **Vorbedingungen** umfassen den Status des zugrundeliegenden Systems oder Bereichs des BayernCloud DPÖ, welches dem Use Case zugrunde liegt.

- Die **Szenarien für Erfolg und Misserfolg** beschreiben den Status nach erfolgreichem bzw. erfolglosem Verlauf des Anwendungsfalls.
- Der **Trigger** legt die auslösende Aktion für den entsprechenden Use Case fest.
- Die **Beschreibung** fasst den wesentlichen Ablauf des Use Cases von Trigger bis zur jeweiligen Erreichung der Zielsetzung zusammen. Dies erfolgt durch die Aufzählung der Schritte innerhalb dieses Ablaufs.

Anhand dieser Strukturvorgaben lassen sich die BayernCloud Use Cases für die Verwendung im Spiegel der verschiedenen Sichten gliedern. Auf diese Weise werden die impliziten Anforderungen der jeweiligen Use Cases für die Konzeptionierung des BayernCloud DPÖ berücksichtigt und tragen wesentlich zu dessen Validierung bei. Dementsprechend sind Use Cases ein zentraler Bestandteil für die Erarbeitung der Referenzarchitektur für das zu entwickelnde Konzept eines BayernCloud DPÖ. Zu diesem Zweck zielen die BayernCloud Use Cases darauf ab, die Perspektiven verschiedener Ökosystemteilnehmer abzubilden. Es wird sich dabei auf diejenigen Use Cases konzentriert, die den Kern des DPÖs darstellen und eine domänenübergreifende Allgemeingültigkeit besitzen.

4.2.2. Verwendung von Use Cases im Kontext der Entwicklung der BayernCloud Referenzarchitektur

Die verschiedenen BayernCloud Use Cases werden im Rahmen der Referenzarchitektur aus unterschiedlichen Blickwinkeln untersucht (siehe Abbildung 2). Hierdurch wird gewährleistet, dass die wesentlichen Anforderungen an die Konzeptionierung des DPÖ im Rahmen jeder der vier Sichten des 4+1 Sichtenmodells reflektiert werden. Die Untersuchung der Use Cases findet in jeder der vier Sichten auf eine der Sicht entsprechende Weise statt.

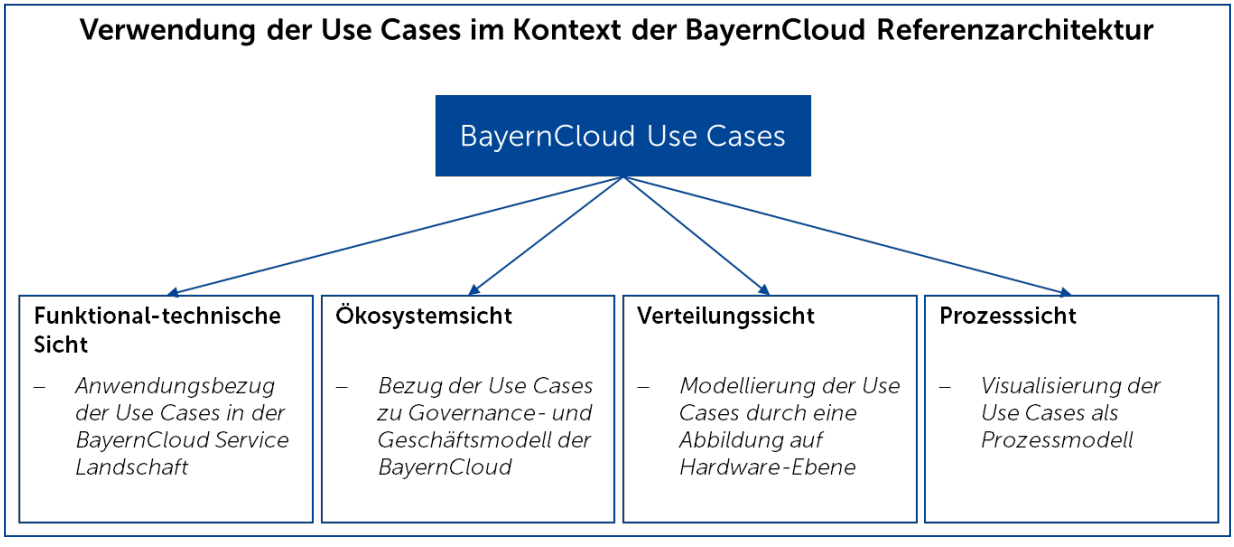

Abbildung 5: Verwendung der BayernCloud Use Cases im Rahmen der Referenzarchitektur

Zunächst werden die BayernCloud Use Cases im Kontext der funktional-technischen Sicht hinsichtlich der Voraussetzungen an die BayernCloud Service Landschaft untersucht. Hierbei soll sichergestellt werden, dass das Spektrum an Basisfunktionalitäten die wesentlichen Anforderungen an das DPÖ abdeckt. Darüber hinaus werden konkrete Angaben gemacht, welche der benötigten Funktionalitäten im Kontext eines einzelnen Use Cases Anwendung finden können. Die Ökosystemsicht setzt die verschiedenen Use Cases

in Bezug zu wichtigen, ökosystemrelevanten Aspekten. Dies bedeutet, dass die Anforderungen jedes Use Case im Kontext der Interaktion verschiedener Akteure im Ökosystem betrachtet werden. Hierunter fällt das Governancemodell, in dem jeder Use Case bezüglich wichtiger Governance-Komponenten aus der Perspektive einer zentralen Plattform des DPÖ untersucht wird, die für den jeweiligen Use Case von Bedeutung sind. Durch die Bezugnahme zum Geschäftsmodell wird festgestellt, inwieweit jeder Use Case für die Geschäftsmodell-relevanten Aspekte des DPÖ von Bedeutung ist.
Die Verteilungssicht modelliert die BayernCloud Use Cases durch eine Abbildung der betreffenden Software- und Servicekomponenten auf Hardware Ressourcen. Sie adressiert insbesondere die Sichtweise von Administratoren und Plattformbetreibern innerhalb des DPÖ und stellt damit das Bindeglied zwischen Implementierungslogik und physischer Infrastruktur dar. Abschließend stellt die Prozesssicht jeden Use Case als Prozessmodell dar, um die Interaktion mit unterschiedlichen Komponenten und Akteuren des DPÖ zu modellieren und übersichtlich darzustellen. Hierdurch können insbesondere auch wechselseitige Beziehungen der verschiedenen Use Cases abgebildet werden, sodass die Prozesssicht die dynamischen Aspekte des Systems verdeutlicht.

In den folgenden Abschnitten werden die BayernCloud Use Cases vorgestellt. Der Aufbau orientiert sich an der in Abschnitt 4.2.1 vorgestellten Struktur nach Cockburn (2002).

4.2.3. Use Case 1: Daten lesen

Name des Use Case	**Daten lesen**	
Use Case Nr.	1	
Zweck/Ziel	Als zentrale Instanz des zu entwickelnden DPÖs, soll eine Plattform es verschiedenen Teilnehmern im Ökosystem ermöglichen, domänenspezifische Daten auszutauschen. Dafür müssen geeignete Datenservices zur Verfügung gestellt werden, welche Schnittstellen zum Auslesen von Daten bereitstellen. Hierbei müssen Kriterien berücksichtigt werden wie etwa die unterschiedlichen möglichen Zugriffsrechte, Lizenzbedingungen, Datenhoheit und Datensicherheit.	
Akteur/Rolle	Drittentwickler, Datenlieferant	
Scope	Plattform des DPÖ	
Vorbedingung	Verfügbarkeit von entsprechend definierten Datenservices mit Datenspeicherlokationen in der BayernCloud Plattform.	
Erfolgsszenario	Erfolgreicher Datenaustausch zwischen unterschiedlichen Parteien.	
Misserfolgs-szenario	Daten können nicht ausgetauscht werden. Die BayernCloud muss den Datenkonsumenten oder Datenprovidern entsprechende Fehlermeldungen zur Verfügung stellen. Die Konsistenz der Daten der jeweiligen Datenservices muss gewährleistet bleiben	
Trigger	Bedarf zur Abfrage domänenspezifischer Daten der BayernCloud insbesondere durch die Nutzung von Drittentwicklern.	
Beschreibung	Schritt 1	Identifizieren der jeweiligen API Schnittstelle des gewünschten Datenservices
	Schritt 2	Abfragen der Daten über API Calls

Tabelle 3: Use Case 1: Daten lesen

4.2.4. Use Case 2: Daten schreiben

Use Case Name	**Daten schreiben**	
Use Case Nr.	2	
Zweck/Ziel	Die BayernCloud soll es verschiedenen Teilnehmern im Ökosystem ermöglichen domänenspezifische Daten auf einer zentralen Plattform auszutauschen. Dafür müssen geeignete Datenservices zur Verfügung gestellt werden, welche Daten in passender Form speichern und verwalten können. Es müssen Kriterien berücksichtigt werden wie etwa die unterschiedlichen möglichen Zugriffsrechte, Lizenzbedingungen, Datenhoheit und Datensicherheit.	
Akteur/Rolle	Drittentwickler, Datenlieferant	
Scope	Plattform des DPÖ	
Vorbedingung	Verfügbarkeit von entsprechend definierten Datenservices mit Datenspeicherlokationen in der BayernCloud Plattform.	
Erfolgsszenario	Erfolgreicher Datenaustausch zwischen unterschiedlichen Parteien.	
Misserfolgs-szenario	Daten können nicht ausgetauscht werden. Die BayernCloud muss den Datenkonsumenten oder Datenprovidern entsprechende Fehlermeldungen zur Verfügung stellen. Die Konsistenz der Daten der jeweiligen Datenservices muss gewährleistet bleiben.	
Trigger	Bedarf zur Einspeisung von Daten seitens verschiedener Teilnehmer des BayernCloud DPÖ.	
Beschreibung	Schritt 1	Identifizieren der jeweiligen API Schnittstelle des gewünschten Datenservices
	Schritt 2	Schreiben der Daten über API Calls

Tabelle 4: Use Case 2: Daten schreiben

4.2.5. Use Case 3: Daten und Datenservices rekombinieren

Use Case Name	**Daten und Datenservices rekombinieren**
Use Case Nr.	3
Zweck/Ziel	Entwickler von externen Dienstleistern und Unternehmen sollen in der Lage sein, anhand der zu instanziierenden Plattform und Service Spezifikationen einfach von aktuellen Möglichkeiten der Daten- und Service-Rekombination zu profitieren. Dazu sollten entsprechend semantisch annotierte Services und technische Möglichkeiten der Rekombination von Schnittstellen für Drittentwickler zur Verfügung stehen.
Akteur/Rolle	Drittentwickler
Scope	Kombinationsschnittstelle
Vorbedingung	Bereits vorhandene Datenservices, welche über die künftige Plattform zur Verfügung gestellt werden.
Erfolgsszenario	Möglichkeit der einfachen und konsolidierten Abfrage von komplexen Datenzusammenhängen.
Misserfolgs-szenario	Fehlermeldung mit Rückmeldung über mögliche Probleme wie etwa fehlerhafte Semantik bei der Abfrage.
Trigger	Notwendigkeit der einfacheren Abfrage unterschiedlicher Datensätze des DPÖ.

Beschreibung	Schritt 1	Definition der gewünschten Rekombinationskriterien und Daten des jeweiligen Entwicklers.
	Schritt 2	Deklarative Eingabe der jeweiligen Anforderungen.
	Schritt 3	Automatisiertes auslesen und verwerten Rückgabewerte.

Tabelle 5: Use Case 3: Daten und Datenservices rekombinieren

4.2.6. Use Case 4: Drittentwickler Service bereitstellen

Use Case Name	**Drittentwickler Service bereitstellen**	
Use Case Nr.	4	
Zweck/Ziel	Drittentwickler sollen in der Lage sein, die instanziierte Infrastruktur zu nutzen, um eigene Services zur Verfügung zu stellen. Dabei soll durch die Plattform eine einfache Möglichkeit gegeben sein, die Services auf der Plattforminfrastruktur zu provisionieren.	
Akteur/Rolle	Drittentwickler	
Scope	BayernCloud Service Delivery API bzw. Dashboard	
Vorbedingung	Der externe Entwickler hat den Onboarding Prozess bereits durchlaufen und ist im Usermanagement der Plattform gelistet. Er erfüllt somit alle notwendigen Bedingungen und Voraussetzungen hinsichtlich der Plattform-Governance.	
Erfolgsszenario	Vollständig über eine Webschnittstelle erreichbarer Service unter den für die jeweilige Situation von der Plattform spezifizierten Service Level Agreement Bedingungen.	
Misserfolgs-szenario	Service wird nicht zur Verfügung gestellt, es werden entsprechende Fehlermeldungen an den Entwickler und Plattformbetreiber kommuniziert. Die restliche Plattform und Servicelandschaft bleibt von der Fehlfunktion unberührt.	
Trigger	Anforderung eines Drittentwicklers, einen Service auf der BayernCloud Plattform zur Verfügung zu stellen.	
Beschreibung	Schritt 1	Der Drittentwickler überführt die notwendige Applikationslogik in ein geeignetes Format für die Bereitstellung in der BayernCloud.
	Schritt 2	Der Drittentwickler nutzt die BayernCloud Service Delivery API oder das dafür vorgesehene Dashboard um einen Service zur Verfügung zu stellen.
	Schritt 3	Der Drittentwickler übergibt die notwendigen Ressourcen in Form von Containern und Parametern wie Ports.
	Schritt 4	Die Plattform stellt den Service automatisiert zur Verfügung und liefert dem Drittentwickler eine geeignete Statusmeldung.

Tabelle 6: Use Case 4: Drittentwickler Service bereitstellen

4.2.7. Use Case 5: Services verschieben

Use Case Name	**Services verschieben**
Use Case Nr.	5

Zweck/Ziel	Entwickler von externen Unternehmen sollen in der Lage sein, Services, welche auf der BayernCloud Infrastruktur zur Verfügung gestellt werden, zwischen unterschiedlichen Infrastruktur-Anbietern zu verschieben. Diese Möglichkeit soll die Betreiberunabhängigkeit des BayernCloud DPÖs gewährleisten.	
Akteur/Rolle	Drittentwickler	
Scope	Plattform Infrastruktur, Plattform Management	
Vorbedingung	Bereits auf der Plattform gehosteter Service des jeweiligen Drittentwicklers.	
Erfolgsszenario	Erreichbarkeit des Services unter der gleichen Domäne auf der neuen Infrastruktur Lokation mit entsprechender Visualisierung im Plattform Management.	
Misserfolgs-szenario	Fehlermeldung des Systems. Wiederherstellung des ursprünglichen Zustandes, um die Verfügbarkeit zu gewährleisten.	
Trigger	Neue Anforderungen an die Servicelokation eines Drittentwicklers.	
Beschreibung	Schritt 1	Authentifizierung des Drittentwicklers im Plattform Management.
	Schritt 2	Auswahl des zu verschiebenden Services.
	Schritt 3	Definition der neuen Lokation.
	Schritt 4	Ausführen der Verschiebung.
	Schritt 5	Verifikation der neuen Information über das Plattform Management.

Tabelle 7: Use Case 5: Services verschieben

4.2.8. Use Case 6: Drittentwickler onboarden

Use Case Name	**Drittentwickler onboarden**	
Use Case Nr.	6	
Zweck/Ziel	Zur kontinuierlichen Entwicklung des DPÖs, soll die Plattform einen einfachen Prozess zum Onboarding dritter Parteien zur Verfügung stellen. Dies ist notwendig, um den rechtlichen- und sicherheits-technischen Anforderungen Rechnung zu tragen. Der Vorgang muss sowohl bei der technischen Architektur berücksichtigt werden als auch Mechanismen zur Plattform-Governance und Sicherheit gewährleisten können.	
Akteur/Rolle	Plattform, Drittentwickler	
Scope	Plattform Management	
Vorbedingung	Neuer Drittentwickler ohne existierenden Zugang zur BayernCloud Plattform.	
Erfolgsszenario	Erfolgreich aufgenommener neuer Entwickler in das User Management System der Plattform.	
Misserfolgs-szenario	Ablehnung des neuen Drittentwicklers unter Angabe von Gründen, technischer sowie nicht-technischer Natur.	
Trigger	Anforderung der Aufnahme von Drittentwicklern in das Plattform-Ökosystem.	
Beschreibung	Schritt 1	Aufrufen des Plattform Managements.
	Schritt 2	Eintragen der von der Plattform spezifizierten Informationen zum onboarden von neuen Drittentwicklern.

	Schritt 3	Teilautomatisierte Evaluierung.
	Schritt 4	Anlegen eines neuen Drittentwicklers im Plattform User Management mit entsprechenden Rechten und Zugangsdaten.
	Schritt 5	Kommunikation der Nutzer spezifischen Zugangsdaten und Plattform Nutzungsbedingungen.

Tabelle 8: Use Case 6: Drittentwickler onboarden

4.2.9. Use Case 7: Services anbieten und nachfragen (Service Marktplatz)

Use Case Name	Services anbieten und nachfragen (Service Marktplatz)	
Use Case Nr.	7	
Zweck/Ziel	Um die verschiedenen Angebote und Services für alle Teilnehmer des DPÖs zur Verfügung zu stellen, sollte die Plattform einen Servicemarktplatz bereitstellen. Diese Funktionalität ermöglicht es, Angebot und Nachfrage von Services, die auf der Plattform bereitgestellt sind, zu managen und entsprechende Interaktionen zwischen Anbietern und Kunden über die Plattform zu fördern.	
Akteur/Rolle	Serviceanbieter, Servicekunde	
Scope	Service Marktplatz	
Vorbedingung	Serviceanbieter, Services und Servicekunde sind onboarded	
Erfolgsszenario	1. Ein Servicekunde kann per Self-Service nach Services suchen, Treffer sehen, auswählen und Beschreibungen lesen sowie einen passenden Service kaufen und sofort nutzen. 2. Ein Serviceanbieter wird benachrichtig und vergütet, entsprechend den Angaben beim onboarden.	
Misserfolgs-szenario	1. Ein existierender Service, der anhand der im Onboarding angegebenen Keywords gefunden werden sollte, wird nicht als Suchergebnis angezeigt. 2. Ein onboardeter Service, der über die App Store angeboten werden soll, kann nicht gebucht werden. 3. Ein gekaufter Service kann nicht genutzt werden. 4. Der Service Anbieter wird nicht vergütet, nachdem ein Service gekauft wurde.	
Trigger	Anbieter wollen anderen Serviceanbietern und weiteren Nutzern der BayernCloud ihre Services bekannt machen und komfortabel anbieten und bereitstellen. Servicekunden wollen Services nutzen und kombinieren, die kompatibel mit der Referenzarchitektur der BayernCloud sind.	
Beschreibung	Step1	Ein Nutzer der BayernCloud geht auf den App Store-Bereich der BayernCloud
	Schritt 2	Der Nutzer sieht die neusten Services, die auf der BayernCloud zum Buchen bereitgestellt wurden
	Schritt 3	Der Nutzer kann anhand von Keywords nach Services suchen
	Schritt 4	Alle treffenden Services werden angezeigt inklusive. Hinterlegter Informationen und Buchungsmodell (einmalig, Nutzung, monatlich, etc.)
	Schritt 5	Der Kunde kann einen Service auswählen und Buchen

	Schritt 6	Der Service wird gebucht und automatisch provisioniert
	Schritt 7	Der Serviceanbieter wird benachrichtig und vergütet
	Schritt 8	Der Kunde bekommt eine Rechnung automatisch per Mail geschickt

Tabelle 9: Use Case 7: Services anbieten und nachfragen (Service Marktplatz)

Die Use Cases der BayernCloud Grundlagenforschung sind so gewählt, dass eine gewisse Basisfunktionalität domänenübergreifend gewährleistet werden kann ohne zu spezifische Anforderung an die Funktionalität zu erfordern. Ein Überblick über den Zusammenhang der verschiedenen Use Cases innerhalb des BayernCloud DPÖs ist in Abbildung 6 dargestellt.

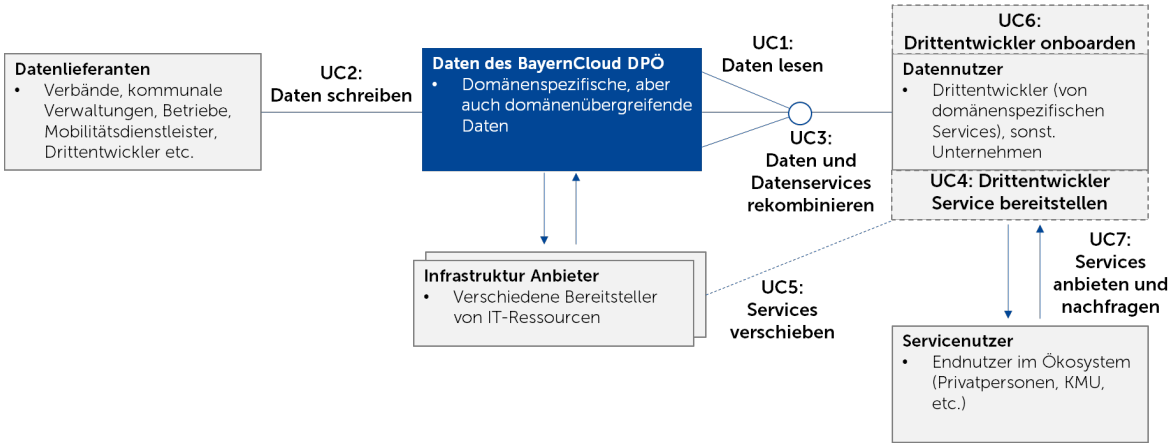

Abbildung 6: Übersicht zum Zusammenhang der BayernCloud Use Cases

Übergeordnet zu den Use Cases gibt es eine Reihe weiterer grundlegender Anforderungen an die Konzeption und Entwicklung des BayernCloud DPÖs. Hierunter fallen grundsätzliche Qualitätsaspekte zur Gewährleistung der allgemeinen Nutzbarkeit wie DSGVO Konformität, digitale Barrierefreiheit und Zertifizierung von möglichen Infrastrukturanbietern. Prozessanforderungen zur Entwicklung des DPÖs beinhalten eine Empfehlung bzgl. einer agilen Vorgehensweise zur flexiblen und effektiven Erarbeitung von Teilkomponenten gekoppelt an ein hohes Maß an Nachvollziehbarkeit der (Teil-)Schritte in diesem Prozess, um den Stand der Umsetzung verstehen und den Architekturfortschritt nachvollziehen zu können. Auf technischer Seite fallen weitere Anforderungen wie etwa Offenheit, Skalierbarkeit oder Plattformunabhängigkeit an, die insbesondere in den technischen Sichten (siehe die Abschnitte 4.3 und 4.5) diskutiert werden.

4.2.10. Auswahlprozess der BayernCloud Use Cases mithilfe von Anforderungen aus der Pilotdomäne Tourismus

Auf Basis der Zielsetzungen des Projektantrags ergibt sich als ein sehr zentrales Arbeitspaket der BayernCloud Grundlagenforschung die „Anforderungsaufnahme für die grundlegenden Funktionalitäten und auf der damit verbundenen Identifizierung der jeweiligen Use Cases“. Die in den vorherigen Abschnitten vorgestellten BayernCloud Use Cases sind das Resultat einer iterativen Verarbeitung von notwendigen Vorgaben und Anforderungen an die Konzeption des DPÖs. Der Fokus bei der Erstellung und Auswahl der Use Cases lag einerseits auf einer domänenübergreifenden Allgemeingültigkeit und andererseits darauf, die essentiell benötigten Kern-Funktionalitäten des DPÖs zu erschließen. Dies bedeutet insbesondere, dass die Auswahl an Use Cases zwar ein möglichst breites Spektrum der

erhobenen Anforderungen und Vorgaben abdecken sollte, jedoch ohne dabei durch eine zu hohe Heterogenität ihre Handhabung für die Anwendung zu erschweren. Entsprechend wurde versucht eine gute Balance zwischen Allgemeingültigkeit und den individuellen Vorgaben zu schaffen.

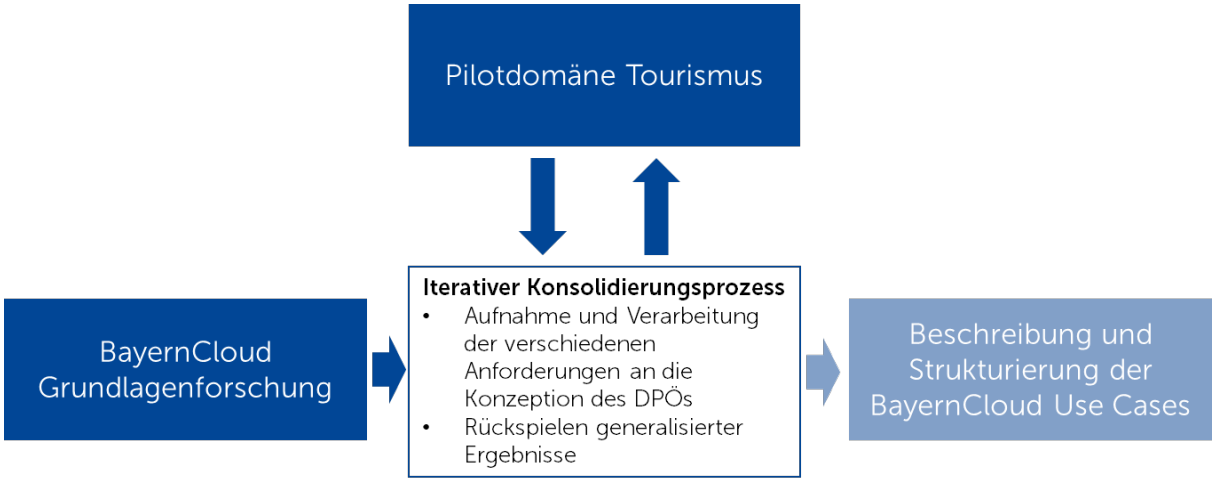

Abbildung 7: Prozess zur Entwicklung der BayernCloud Use Cases

Zur Erhebung der finalen BayernCloud Use Cases wurde ein iterativer Prozess angestoßen, welcher in Abbildung 3 dargestellt ist. Zunächst wurden die bereits dokumentierten Anforderungen aus dem Projektantrag der Grundlagenforschung an das zu konzeptionierende DPÖ als initiale Basis für die weitere Ausarbeitung herausgestellt. Auf Basis dieser Rahmenbedingungen wurden in Zusammenarbeit mit den Projektpartnern aus dem BayernCloud Anwendungsteil domänenspezifische Use Cases in der Pilotdomäne des Tourismus aus der Perspektive verschiedener Steakholder (z.B. Touristen, KMU, Destinationsmanagement) ausformuliert. Die Aufgabe der Grundlagenforschung ist es, die Übertragbarkeit der Anforderungen und Use Cases auf andere Domänen zu gewährleisten und bei Bedarf nach einer entsprechenden Beurteilung anzupassen. Aus diesem Grund wurden die domänenspezifischen Use Cases als inhaltlicher Input aufgenommen und im Rahmen der Grundlagenforschung weiterverarbeitet (siehe nachfolgendes Beispiel zum Use Case „Tagesgast Natur / Kultur"). Hierdurch konnten generalisierte Anforderungen an das zu entwickelnde DPÖ abgeleitet werden, die wiederum über die gesamte Projektlaufzeit insbesondere in den regelmäßig stattfindenden Projektmeetings in die Pilotdomäne des Tourismus zurückgespielt werden konnten. Durch diesen iterativen Prozess konnte ein finaler Satz an domänenübergreifenden BayernCloud Use Cases ausformuliert und für die Entwicklung der Referenzarchitektur herangezogen werden.

Domänenspezifischer Use Case zur Ableitung generischer Anforderungen aus dem Tourismus

Um den Zusammenhang zwischen der Anwendungsdomäne des bayerischen Tourismus und den Anforderungen an das zu entwickelnde DPÖ herauszustellen, wird das folgende Szenario aus dem Anwendungsteil exemplarisch herangezogen:

BayernCloud Anwendungsteil: Use Case "Tagesgast Natur / Kultur"

„Das Ehepaar Paula (60) und Georg (64) verbringen ihren Sonntagabend gemütlich auf dem Sofa und machen sich über ihr nächstes Ausflugsziel Gedanken. Als Paula einen Blick aufs Bücherregal wirft, fällt ihr der Reiseführer fürs Allgäu ins Auge. Sie blättert durch die Seiten

des Reiseführers und Georg ist begeistert von der Idee einen Tagestrip ins Allgäu zu machen. Dementsprechend informiert sich das Ehepaar mit Hilfe einer App über das Wetter der kommenden Woche und einigt sich auf den Mittwoch, da für diesen das beste Wetter vorhergesagt ist. Nachdem sie sich für einen Tag entschieden haben, informieren sie sich per App über die verschiedenen Tagestouren im Allgäu. Nach intensiver Recherche entscheiden sich die beiden für den Wanderweg „Schlösser-Runde" in Füssen, der sowohl für Einsteiger geeignet ist, als auch einen perfekten Blick auf die Königsschlösser Hohenschwangau und Schloss Neuschwanstein bietet. Da Georg nur noch Kurzstrecken mit dem Auto zurücklegt, einigt sich das Ehepaar auf die Anreise mit der Bahn. Ein Bus bringt die beiden zur Haltestelle in der Nähe der Tourist Information Füssen, die den Startpunkt der Wanderroute bildet. Nach erfolgreicher Wanderung wollen die beiden zu Mittag essen und finden dank der App ein kleines Restaurant in der Nähe. Während des Essens informieren sie sich zudem über die Museen entlang des Wanderwegs und entscheiden sich für das „Museum der bayerischen Könige", weil hier keine Reservierung der Tickets notwendig ist und keine Einlasszeiten festgelegt werden. Da die App sehr lange Wartezeiten am Ticket Center anzeigt, beschließt das Paar ihre Eintrittskarten direkt am Museum zu erwerben. Nach der Besichtigung kehren Georg und Paula zurück zur Bushaltestelle und fahren mit dem Bus zum Bahnhof Füssen. Anschließend treten sie mit der Bahn die Heimreise nach Augsburg an."

Der Use Case beinhaltet eine Reihe an Anforderungen an das zu entwickelnde DPÖ auf einer allgemeingültigen und damit domänenübergreifenden Ebene, sowie an die Interaktion der verschiedenen Akteure. Um die Übersichtlichkeit zu gewährleisten, wurden in der Analyse des touristischen Szenarios „Tagesgast Natur / Kultur" in Abbildung 8 bereits die finalen BayernCloud Use Cases zur Beschreibung der benötigten Interaktionen verwendet.

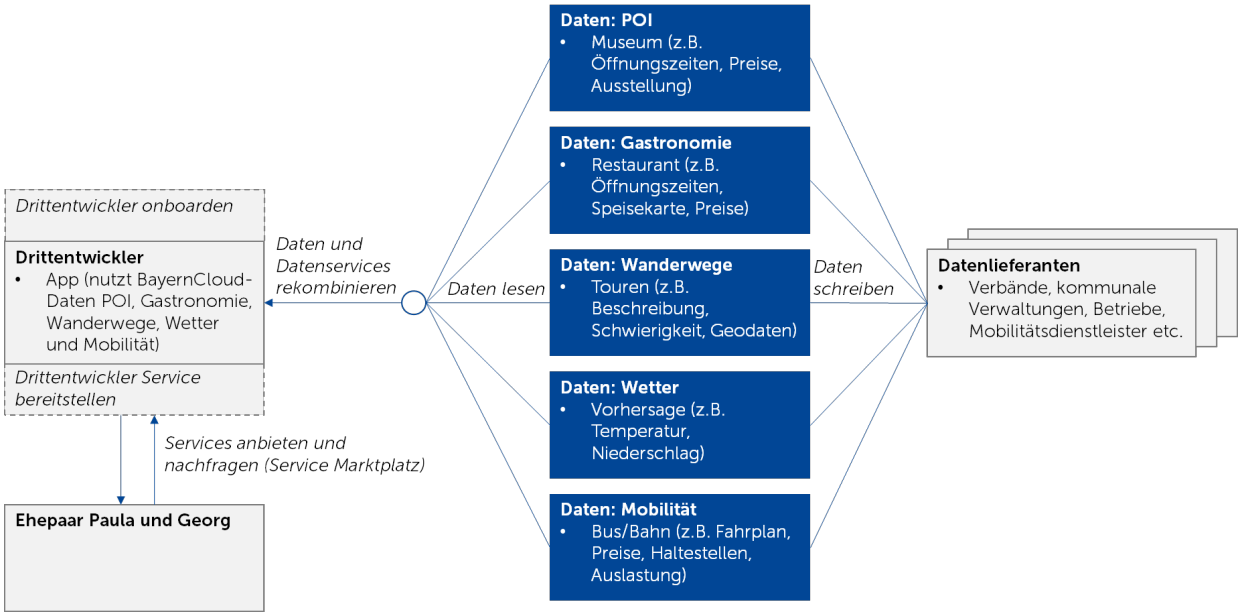

Abbildung 8: Der Use Case "Tagesgast Natur / Kultur" im Spiegel der BayernCloud Grundlagenforschung

Anhand dieser Darstellung lassen sich wesentliche Eigenschaften des zu entwickelnden DPÖs allgemeingültig abstrahieren. Diese werden in Bezug zu den weiteren Anforderungen aus den domänenspezifischen Use Cases auf ihre Verwendung in der BayernCloud

Grundlagenforschung anhand des oben beschriebenen Vorgehens geprüft. In obigem Beispiel zum „Tagesgast Natur / Kultur" ergeben sich etwa die grundlegenden Anforderungen bezüglich der BayernCloud Use Cases *„Daten lesen"* (UC1), *„Daten schreiben"* (UC2), *„Daten und Datenservices rekombinieren"* (UC3), *„Drittentwickler onboarden"* (UC6), *„Drittentwickler Service bereitstellen"* (UC4) und *„Services anbieten und nachfragen (Service Marktplatz)"* (UC7) (vergleiche Abbildung 8).

4.3. Funktional-technische Sicht

4.3.1. Einleitung

Die funktional-technische Sicht basiert auf der logischen Sicht von Kruchten und wurde dahingehend angepasst, dass sie den logischen Aufbau und die Funktionsweise der Referenzarchitektur in abstrakter, möglichst technologieneutraler Weise beschreibt. Dazu wird die gesamte Architektur in einzelne Bausteine, Module und Anforderungen untergliedert. Die funktional-technische Sicht richtet sich daher insbesondere an Systemarchitekten und Entwickler.

Der Kern dieser Sicht bezieht sich auf die Module, die Beschreibung der Funktionalität dieser Module und die Darstellung der Verbindungen der Module zueinander. Diese Module werden wiederum in die beiden Kategorien Basisinfrastruktur, sowie BayernCloud Services eingeteilt, die aus der Gesamtstruktur des Projekts BayernCloud abgeleitet ist.

Der logische Aufbau dieser Sicht folgt dem „Separation of Concerns"-Prinzip, das jedem Modul zudem einen „Concern" zuweist. In Verbindung mit geringer intermodularer Abhängigkeit wird so eine hohe, aber dennoch leicht erweiterbare Modularität erreicht.

4.3.2. Gesamtübersicht Servicelandschaft

Entsprechend der Projektstruktur ist die Servicelandschaft in Basisinfrastruktur und BayernCloud Services geteilt. Die BayernCloud Service Module bilden dabei alle Vorgänge ab, mit denen ein Nutzer Datenservices erstellen und verwalten kann, wohingegen die Basisinfrastrukturmodule die technischen Maßnahmen und Voraussetzungen zum Bereitstellung/Provisioning, Betrieb und Management dieser Services abdeckt. Servicefunktionalitäten für diesen Vorgang umfassen die automatisierte Zertifizierung der Schnittstelle, sowie Editing- und Managementfunktionalitäten.

Eine zentrale Rolle im Kontext verschiedener Use Cases fällt den verschiedenen Datenservices zu, welche durch das generische Datenservice-Modul verkörpert werden. Dieses ermöglicht das vollautomatisch Initialisieren von semantisch annotierten Datenschnittstellen anhand einer zentralen, modularen und domänenspezifischen Ontologie. Um die Pflege einer solchen Ontologie auch für Endnutzer zu ermöglichen definiert das Schema-Management-Modul entsprechende grafische Schnittstellen, um benutzerdefinierten Input in ein standardisiertes Schema übersetzen. Daten, welche durch die so erzeugten Schnittstellen in das System eingebracht werden, werden in einer oder mehreren Datenbanken persistiert. Weiterhin werden Referenzbausteine für spezifische und übergreifende Dashboards spezifiziert, welche der Visualisierung und Analyse solcher Daten dienen – dies umfasst Business-Views, persönliche Views, Deployment-Views sowie Datenviews in jeweils verschiedenen Granularitätsstufen. Weiterhin ist ein Referenzbaustein

für einen Anomaly-Detection-Service zur Überwachung abweichenden Nutzerverhaltens definiert.

Im Gesamtkontext der technischen Module der Referenzarchitektur fällt insbesondere der durch das Modul Ontologie-Service verwalteten, domänenübergreifenden Service-Ontologie eine übergeordnete Rolle zu. Diese Ontologie verkörpert den zentralen Interpretationsrahmen für verschiedene Datenmodelle, welche einerseits über Instanziierungen des Datenservice-Moduls, andererseits für die Komposition verschiedener Services genutzt werden. Sie verkörpert somit eine modulare *Single Source/Point of Truth* für alle in der BayernCloud vorhandenen Datenservices und schafft damit die Grundlage zur Kommunikation von Services innerhalb des BayernCloud Clusters. Die im Modul Schema Management benutzerdefinierten Datenmodelle werden in der Ontologie validiert. Dabei werden Synonyme und Überschneidungen detektiert, aufgelöst und durch eine *semantische Suche* auffindbar gemacht. Das validierte Schema ist dann wiederum Grundlage für den darauf basierenden Datenservice. Ein Dokumentationsservice erfasst und beschreibt diesen angebotenen Datenservice. Durch ein *Service Discovery Modul* wird ein neuer Service, beispielsweise bei seiner Initialisierung, gleichzeitig erfasst. Dadurch wird die Grundlage für die Darstellung und Analyse der Servicelandschaft der BayernCloud in Echtzeit geschaffen. Alle bisher beschriebenen Vorgänge werden darüber hinaus von einem *eventbasierten Messaging System* erfasst, das die Nachvollziehbarkeit der Vorgänge ermöglicht und gleichzeitig Grundlage zur Verbesserung und Abhärtung der Systemlandschaft darstellt. Dieses Messaging System ist Teil der *Monitoring Solution* der BayernCloud, das Echtzeitdaten auf allen Systemschichten erfasst und die Überwachung des Zustandes der BayernCloud ermöglicht. Für das System und seine Services bestehen verschiedene Konfigurationsmöglichkeiten hinsichtlich Organisation, Skalierung und Kommunikation von Datenservices. Das Modul *Database-per-Service* erlaubt jedem Datenservice auf einer eigens dedizierten Datenbankinstanz zu arbeiten. Das Modul *API-Gateway* erlaubt den Zugriff auf eine Servicelandschaft durch eine zentralere Schnittstelle. Die Details und insbesondere der Zerfall in einzelne Datenservice-Instanzen wird dadurch abstrahiert und bleibt als überflüssiges (Implementierungs-)Detail verborgen, wohingegen der Konsum von Datenservices vereinfacht wird. Das Modul *API Composition* stellt, als Basis für die Abfrage über ein Gateway, eine höhere Form der Abstraktion über einzelne Services dar, die eine dynamische Abfrage von beliebigen angebotenen Daten, denen mehrere Services zu Grunde liegen, nach benutzerdefinierten Kriterien erlaubt. Diese Daten können, unter der Voraussetzung der Azyklizität der Anfrage, beliebig verschachtelt sein. Zur Abhärtung des Systems gegen Überlastungen, die Beispielsweise im Rahmen der Komposition entstehen, kommt das Modul *Circuit Breaker* zum Einsatz. Durch den Circuit Breaker werden Überlastungen erkannt und negative Rückkopplungsschleifen unterbunden. Bei akutem Timeout einer Datenbank aufgrund von Überlastung werden durch den Circuit Breaker die dann üblichen Wiederholungen der Anfrage unterbunden, bis die Datenbank wieder funktionsbereit ist, womit eine andauernde Überlastung vermieden wird. Ein weiteres Set von Modulen stellt Management-, Organisations- und Sicherheitsfunktionalitäten zur Verfügung.

Überdies hinaus beschäftigen sich Module der Basisinfrastruktur mit der zu Grunde liegenden Technologie. Dazu zählen vor allem Infrastruktur und Virtualisierungstechnologien sowie Werkzeuge und Methoden die dafür in der aktuellen Softwareentwicklung eingesetzt werden. Grundsätzlich implementiert die BayernCloud eine Architektur, welche auf Microservices und deren Orchestration beruht. Einzelne

Services werden in Form von miteinander interagierenden Containern umgesetzt. Container sind zu einem fundamentalen Bestandteil moderner Plattformen gereift. Allerdings erfordern sie die gesonderte Berücksichtigung von Orchestrierung, Provisionierung, Verwaltung und Monitoring. Entsprechende Komponenten werden im Rahmen der Basisinfrastruktur vorgestellt. Darüber hinaus bietet die BayernCloud die Möglichkeit, Ihre Services Infrastrukturanbieterübergreifend zu organisieren, das dafür notwenigen Multi-Cloud Muster und Deployment Implikationen sind ebenfalls Teil der Betrachtung.

Die im Folgenden beschriebene funktional-technische Sicht basiert auf den Erkenntnissen des ebenfalls von fortiss durchgeführten Forschungsprojekts PRODISYS (Förderkennzeichen 02K16C050). Die dort entwickelten Module wurden im Rahmen der BayernCloud weiterentwickelt und durch weitere, für das digitale Plattform Ökosystem spezifische Module ergänzt. In Einzelfällen kann es hierbei jedoch noch zu inhaltlichen Überschneidungen mit den entwickelten Ergebnissen kommen, welche sich aufgrund der guten Übertragbarkeit etwa im Bereich technischer Komponenten der Basisinfrastruktur bewegen (vergl. Abschnitt 4.3.5).

4.3.3. Kerndienste

Die Kerndienste der BayernCloud-Referenzarchitektur adressieren die maßgebliche Funktionalität, welche zur Umsetzung der im Rahmen von Kapitel 4.2 definierten Use Cases notwendig sind. Dabei lassen sich die entsprechenden Module in insgesamt drei Abschnitte untergliedern:

- Module der *Dateninfrastruktur* adressieren die Realisierung einer (Daten-)Servicelandschaft für den Austausch domänenspezifischer Daten, deren Dokumentation sowie die kombinierte Abfrage von Daten über verschiedene Datendienste hinweg (Use Cases 1-3)
- *Grafische Nutzerschnittstellen* ermöglichen den grafischen Zugriff sowie die Definition der mit der Dateninfrastruktur verbundenen Dienste; zusätzliche Nutzerschnittstellen adressieren zudem das Einbinden, Deployment und Verschieben von Daten- und Drittentwickler-Services sowie deren Inanspruchnahme im Kontext eines App Stores (Use Cases 4-7)
- Der Abschnitt *Sicherheit & Zertifizierung* umfasst zusätzliche, sicherheitsspezifische Module, die für alle zuvor beschriebenen Kerndienste relevant sind.

Dateninfrastruktur

Um eine generische (Daten-)Servicelandschaft für den Austausch domänenspezifischer Daten zu instanziieren kann eine Reihe unterschiedlicher Datendienste (#1) erzeugt werden, welche ein in einer Domänenontologie spezifiziertes Datenschema umsetzen. Konzepte oder Konzeptgruppen der Ontologie werden hierbei jeweils durch einen individuell deploy- und skalierbaren Microservice implementiert – der Zugriff auf die Ontologie wird durch einen Ontologie-Service (#2) gesteuert. Ebenfalls auf Basis der Domänenontologie wird ein Datenkombinationsservice (#3) definiert, welcher die verschiedenen Microservices aufgrund ontologischer Verbindungen in Relation setzt und einen kombinierten Datenzugriff ermöglicht. Zur Dokumentation der Datenservices dient ein generischer Dokumentationsservice (#4).

Datenservice

	#1
Komponente	Datenservice
Zweck/Ziel	Dieses Modul bildet den generisch die Microservices der BayernCloud ab. Es stellt über eine API sogenannte Create/Read/Update/Delete (CRUD) Funktionalitäten zur Verfügung.
Generisch / optional	Generisch
Funktions-beschreibung	Dieses Modul stellt in seiner Basisfunktionalität einen generischen Datenservicebereit. Mittels des durch das Ontologie-Service-Modul verwalteten Schemas wird eine Instanz dieses Moduls instanziiert, dessen Datenmodell dem Schema entspricht. Über eine HTTP Schnittstelle können so Daten ausgetauscht werden. Die Granularität eines einzelnen Datenservices wird anhand der Service-Ontologie des Ontologie-Service-Moduls spezifiziert.
Verbindungen zu anderen Modulen	• Deployment Dashboard • Data Dashboard • Deployment Manager • Ontologie-Service • Database
Quellen und weiterführende Informationen	-

Tabelle 10: Datenservice

Ontologie-Service

	#2
Komponente	**Ontologie-Service**
Zweck/Ziel	Damit Daten verschiedener Microservices miteinander kombiniert werden können, liefert eine zentrale domänenspezifische Ontologie den nötigen Interpretationsrahmen der Schnittstellen und ausgetauschten Daten.
Generisch/optional	Generisch
Funktions-beschreibung	Das Modul Ontologie-Service verwaltet lesende und schreibende Zugriffe auf eine domänenspezifische (Daten-)Ontologie sowie eine domänenübergreifende Service-Ontologie. Die Ontologie definiert eine standardisierte Terminologie wodurch Daten maschinell interpretierbar

	und kombinierbar werden. Die vom Ontologie-Service verwaltete Ontologie bildet somit die Grundlage zur Instanziierung der beschriebenen Dateninfrastruktur. Konkrete Funktionalitäten des Ontologie-Services umfassen: - Lesen und Bearbeiten domänenspezifischer Datenschemata, bspw. auf Basis existierender Ontologien (z.B. schema.org oder DACH-KG im Tourismus) - Lesen und Bearbeiten domänenübergreifender Service-Definitionen, welche die Granularität der Datenservices beschreiben - Lesen und Bearbeiten von Restriktionen (z.B. Pflicht-Attribute), welche dem ontologischen Datenschemata zugrunde gelegt werden
Verbindungen zu anderen Modulen	• Datenservice
Patterns	-
Bestehende Produkte	• Formalismen zur Beschreibung von Ontologien o RDFS o OWL • Schema.org als existierende domänenübergreifende (jedoch spezifizierbare) Ontologie • SHACL als Constraint-Language für RDF-basierte Graphen
Herausforderungen	• Formal korrekte Ontologien zu definieren ist generell herausfordernd und benötigt die Zusammenarbeit von Domänen-Experten und Ontologie-Experten. Methoden zur Entwicklung formal korrekter, domänenspezifischer Ontologien werden unter dem Begriffdes „Ontology Engineerings" beschrieben • Eine weitere Herausforderung ist die Spezifikation einer geeigneten Granularität der zu instanziierenden Datenservices
Ähnliche / Verbundene Konzepte	-
Entscheidungen / Fragen bei der Implementierung	-
Quellen und weiterführende Informationen	-

Tabelle 11: Ontologie-Service

	#3
Komponente	**Datenkombinationsservice**
Zweck/Ziel	Der Datenkombinationsservice erlaubt das Kombinieren und Aggregieren von Daten über Microservice-Grenzen hinweg.
Generisch / optional	Generisch
Funktions-beschreibung	Die Datenkombinationsservice stellt eine deklarative API zur Verfügung, welche die Abfrage microservice-übergreifender Daten ermöglicht. Ein Modell dieser deklarativen Schnittstelle ist durch einen Ontologie-Service gegeben. Für den Nutzer ergibt sich durch das Modul eine Reduktion der Komplexität, während die Performanz dieser Abfragen optimiert wird. Der Datenkombinationsservice kann hierfür auf die Module API Composition und Command Query Responsibility Segregation zurückgreifen, auch eine Kombination zur Erreichung eines Trade-Offs ist hier möglich.
Abhängigkeiten	• Database per Service erzeugt den Bedarf an diesem Punkt • Ontologie-Service
Patterns	• API Composition • Command Query Responsibility Segregation
Bestehende Produkte	-
Herausforderungen	• Je nach Implementierung ergeben sich dieselben Herausforderungen wie bei den Verwendeten Modulen (API Composition, Command Query Responsibility Segregation) • Es gilt zu ermitteln, wie eine Kombination der beiden oben genannten Module aussehen kann
Ähnliche / Verbundene Konzepte	• API Gateway • CQRS
Entscheidungen / Fragen bei der Implementierung	• Welchen Ansatz der Datenkombination wählt man?
Quellen und weiterführende Informationen	-

Tabelle 12: Datenkombinationsservice

	#4
Komponente	**Dokumentationsservice**
Zweck/Ziel	Um die Entwicklung gegen die angebotenen Schnittstellen möglich zu machen, ist eine menschenlesbare Dokumentation der entsprechenden APIs (insb. der Daten-APIs (#1)) notwendig. Hierzu kann ein optionaler Dokumentationsservice zur Verfügung gestellt werden, welcher verteilte APIs an zentraler Stelle beschreibt und anhand der durch den Ontologie-Service (#2) verwalteten Ontologien automatisch generiert werden kann.
Generisch/optional	Optional
Funktions-beschreibung	Entwickler sollen in die Lage versetzt werden, die angebotenen APIs innerhalb der BayernCloud-Plattform mit möglichst geringen Aufwand zu nutzen. Die Beschreibung der APIs dient dabei einerseits zur Beschreibung der zu verwendenden Protokolle und Funktionalitäten als auch der genauen Dokumentation der Datenmodelle. Der Service kann folgende Unterfunktionalitäten anbieten: • Interne und externe Entwickler werden in die Lage versetzt, angebotene APIs schnell aufzufinden • Funktionalitäten der API (z.B. Filter- und Sortieroperationen) können durch interne und externe Entwickler schnell verstanden und effizient genutzt werden • Datenschemata der innerhalb der APIs ausgetauschten Nachrichten werden visualisiert • API-Funktionalität kann interaktiv ausprobiert werden (bspw. durch die Generierung entsprechender Service-Calls)
Verbindungen zu anderen Modulen	• API Gateway
Patterns	-
Bestehende Produkte	• OpenAPI/Swagger • RESTful API Modeling Language (RAML)
Herausforderungen	• (Semi-)automatische Verknüpfung des Dokumentationsservice mit der Entwicklung und Versionierung der beschriebenen APIs • Vermeidung von Inkonsistenzen
Ähnliche / Verbundene Konzepte	• API Gateway • API Composition

Entscheidungen / Fragen bei der Implementierung	• Wie kann die Konsistenz zwischen bestehenden API-Versionen und der Dokumentation sichergestellt werden? • Wie bzw. zu welchem Grad lässt sich die Generierung der Dokumentationen automatisieren?
Quellen und weiterführende Informationen	-

Tabelle 13: Dokumentationsservice

Grafische Nutzerschnittstellen

Grafische Nutzerschnittstellen (bspw. webbasiert) dienen dem einfachen, grafischen Zugriff auf definierte Funktionalitäten der Plattform. Das Schema-Management-Modul (#5) stellt hierbei eine grafische Schnittstelle für lesende und schreibende Zugriffe auf die durch das Ontologie-Service-Modul verwalteten Schemata zur Verfügung. Das Personal Dashboard (#7) ermöglicht authentifizierten Nutzern die Einsicht und Änderung ihrer persönlichen Daten. Im Data Dashboard (#8) lassen sich die in die Dateninfrastruktur-Module eingespeisten Daten visualisieren und bei Bedarf anpassen. Das Business Dashboard (#9) ermöglicht es authentifizierten Nutzern selbst angebotene Dienste zu verwalten bzw. Dienste von Drittentwicklern zu konsumieren. Das Deployment Dashboard (#10) bietet eine grafische Oberfläche zum Management des Deployments, insbesondere von Datenservices.

Schema Management

	#5
Komponente	**Schema Management**
Zweck/Ziel	Das Modul ermöglicht es, anhand einer grafischen Benutzerschnittstelle individuelle Datenmodelle zu definieren
Generisch / optional	Optional
Funktions-beschreibung	Dieses Modul bietet eine grafische Benutzeroberfläche, die es ermöglicht, ohne Vorkenntnisse in einem strukturierten Prozess ein valides Datenmodell zu erstellen, das als Grundlage für darauf basierende Services dient. In einem ersten Schritt werden dabei Felder benannt. Via Dropdown erhält der Benutzer Vorschläge zu möglichen Datentypen und weitere Informationen zum Erstellprozess. Dabei wird auch die Erstellung komplexer Objekte erlaubt. Durch die automatisierte Zuweisung der Felder via JSON-LD wird hierbei ein semantischer Kontext erzeugt. Das aus diesem Prozess entstehende Schema dient als Grundlage für darauf basierende Datenservices.

	Es ergeben sich folgende Unterfunktionen: • Erstellen von Schemata • Auswahl von Feldern • Festlegung von Datentypen • Definition von Relationen • Darstellung bzw. grafische Aufbereitung von Anweisungen und Listen • Abruf von weitergehenden Informationen zu einzelnen Schritten • Standardvorgehen erstell- und veränderbar für bestimmte Fehlergruppen machen • Protokollierung der einzelnen Schritte • Mechanismus zur automatischen Dokumentation nach Benutzung von Handlungsleitfäden/Checklisten
Verbindungen zu anderen Modulen	• Datenservice • Deployment Dashboard • Data Dashboard • Deployment Manager
Quellen und weiterführende Informationen	-

Tabelle 14: Schema Management

Dublettenprüfung

	#6
Komponente	Dublettenprüfung
Zweck/Ziel	Um die Qualität eingespeister Daten (vergl. #1 Datenservice) insbesondere im Fall zeitabhängiger Daten zu gewährleisten, müssen Daten auf übereinstimmende Instanzen geprüft werden können. Sowohl die Notwendigkeit als auch die Umsetzung können jedoch stark von den Eigenschaften des betrachteten Datensatz abhängen.
Generisch / optional	Optional
Funktions-beschreibung	Für die Dublettenprüfung kommen (semi-)automatische Verfahren abhängig von den Eigenschaften des jeweiligen Datensatzes infrage. Diese reichen von redaktionell gepflegten Daten bis hin zu automatisierten

	Verfahren (regelbasiert oder mithilfe von Machine Learning). Auch vordefinierte Anforderungen an die eingespeisten Daten können hier sinnvoll sein.
Verbindungen zu anderen Modulen	• Datenservice
Quellen und weiterführende Informationen	-

Tabelle 15: Dublettenprüfung

Personal Dashboard

	#7
Komponente	**Personal Dashboard**
Zweck/Ziel	Dieses Modul erlaubt die Darstellung und Änderung von Stammdaten für BayernCloud Teilnehmer über eine grafische Benutzerschnittstelle.
Generisch / optional	Optional
Funktions-beschreibung	Dieses Modul bietet eine grafische Benutzeroberfläche für Teilnehmer des BayernCloud Ökosystems zur Einsicht der persönlichen Stammdaten. Diese Stammdaten enthalten sowohl veränderbare Felder als auch nicht durch den Benutzer veränderbare Felder. Die veränderbaren Felder dürfen durch den Nutzer angepasst werden, wohin gegen Änderungen an nicht veränderbaren Feldern durch einen Administrator erfolgen. Dies kann durch den Teilnehmer veranlasst werden. Es ergeben sich folgende Unterfunktionen: • Einsehen von Stammdaten • Änderung von Stammdaten • Veranlassung einer Änderung durch einen Administrator
Verbindungen zu anderen Modulen	• Datenservice • Data Dashboard • Deployment Dashboard • Business Dashboard • Datenbank • Authorization

Quellen und weiterführende Informationen	-

Tabelle 16: Personal Dashboard

Data Dashboard

	#8
Komponente	**Data Dashboard**
Zweck/Ziel	Dieses Modul bietet eine grafische Benutzeroberfläche für Teilnehmer des BayernCloud Ökosystems zur Einsicht und Verwaltung des Datenaustauschs.
Generisch / optional	Optional
Funktions-beschreibung	Das Data Dashboard ermöglicht das Visualisieren vorhandener Daten auf Basis des ontologisch definierten Schemas. Es ergeben sich folgende Unterfunktionen: • Einsehen der Daten • Bearbeiten/Löschen der Daten • Verwaltung von Datenabonnements
Verbindungen zu anderen Modulen	• Schema Management • Ontologie-Service
Quellen und weiterführende Informationen	-

Tabelle 17: Data Dashboard

Business Dashboard

	#9
Komponente	**Business Dashboard**
Zweck/Ziel	Dieses Modul bietet eine grafische Benutzeroberfläche für Teilnehmer des BayernCloud Ökosystems zum Management von Services und dazu gehörigen Geschäftsinformationen und Aktivitäten.
Generisch / optional	Optional

Funktions-beschreibung	Das Business Dashboard bietet eine separate Übersicht für Plattform Nutzer um Services auszuwählen und zu nutzen. Da Services ggf. kostenpflichtig aus einem zentralen Service Register oder Store konsumiert werden, zählt neben dem Deployment Workflow auch die Integration nötiger geschäftsrelevanter Informationen zu den Anforderungen an das Dashboard. Folgende Unterfunktionen sind bei der Implementierung dieser Sicht zu berücksichtigen:
	• Service Auswahl
	• Service Konfiguration
	• Service Provisionierung
	• Service Abrechnungsrelevante Informationen:
	  ○ Kontodaten
	  ○ Rechnungen
	  ○ Ein-/Auszahlungen

Verbindungen zu anderen Modulen	• Datenservice • Datenbank • Monitoring Solution • Logging Solution
Quellen und weiterführende Informationen	-

Tabelle 18: Business Dashboard

Deployment Dashboard

	#10
Komponente	**Deployment Dashboard**
Zweck/Ziel	Dieses Modul bietet eine grafische Benutzeroberfläche für Teilnehmer des BayernCloud Ökosystems zum Management des Deployments von Datenservices.
Generisch / optional	Optional
Funktions-beschreibung	Dieses Modul bündelt sämtliche Funktionalitäten zum Deployment von Microserviceinstanzen. Individuelle Microservices können durch dieses Dashboard konfiguriert werden.

	Dies Betrifft unter anderem die Verfügbarkeit von Microserviceinstanzen, deren Versionierung, das unterliegende Schema und die Anzahl der Instanzen. Es ergeben sich folgende Unterfunktionen: • Darstellung der aktuellen Deploymentinformationen • An-/Abschaltung von Deployments • Festlegung der Anzahl der Instanzen • Rekonfiguration von Umgebungsvariablen
Verbindungen zu anderen Modulen	• Datenservice • Datenbank • Deployment Manager
Quellen und weiterführende Informationen	-

Tabelle 19: Deployment Dashboard

Data Information Dashboard

	#11
Komponente	**Data Information Dashboard**
Zweck/Ziel	Dieses Modul bietet eine grafische Darstellung der verschiedenen Datenschnittstellen welche über die BayernCloud zur Verfügung gestellt werden. Es ist offen verfügbar und soll mögliche Nutzer informieren über die Qualität, Art und den Umfang der konsumierbaren Daten.
Generisch / optional	Optional
Funktions-beschreibung	Das Data Information Dashboard bietet eine separate Übersicht für alle Plattform Interessenten und Nutzer sich über die verfügbaren Daten der BayernCloud zu informieren. Dabei werden für alle Datentypen zusätzliche Metainformationen angegeben, wie: • Aktualität der Daten • Nutzungsbestimmungen und Lizenzrechte • Kategorisierung • Herkunft der Daten • Mögliches Schema oder Datenaustauschformat

Verbindungen zu anderen Modulen	• Datenservice • Data Dashboard
Quellen und weiterführende Informationen	-

Tabelle 20: Data Information Dashboard

Sicherheit & Zertifizierung

Module des Abschnitts Sicherheit & Zertifizierung adressieren die Plattformsicherheit sowie – soweit technisch möglich – das Einhalten von zuvor bestimmten Nutzungsbedingungen der spezifizierten APIs. Dabei beschränken die Module Authentifizierung (#12) und Autorisierung (#13) den Zugriff auf Daten und Services auf definierbare Nutzergruppen. Ein Anomaly-Detection-Tool (#14) erkennt Unregelmäßigkeiten bei der Nutzung verschiedener APIs auf und kann somit mögliche Verstöße gegen Nutzungsbedingungen und/oder unerwünschtes Verhalten aufdecken. Das Zertifizierungstool (#15) automatisiert zudem einen Prüfungsprozess, der darunterliegenden Cloud-Infrastruktur.

Authentication

	#12
Komponente	**Authentication**
Zweck/Ziel	Dieses Modul stellt die Überprüfung der Identität eines Benutzers sicher und bildet somit die Basis für die Zuweisung service-spezifischer Rechte (#11 Authorization).
Generisch / optional	Generisch
Funktions-beschreibung	Das Authentication-Modul überprüft die Identität eines anfragenden Nutzers und stellt somit dessen Authentizität sicher. Zur Implementierung stehen zwei grundliegende Mechanismen zur Verfügung: • Zentrale Authentifizierung: Hierbei existiert ein zentraler Server, der die Authentifizierung eines Benutzers (bspw. anhand von Nutzername/Passwort oder anhand einer Session) übernimmt. • Dezentrale Authentifizierung: Hierbei wird kein zentraler Server benötigt, stattdessen kann die Authentifizierung durch jeden beliebigen Service (bspw. anhand eines Public-Keys) sichergestellt werden. Dies erfolgt beispielsweise mit Hilfe von JSON Web Tokens (JWT). Zur Authentifizierung kommen verschiedene Faktoren in Betracht, welche entweder einzeln (Single-Faktor-Authentifizierung) oder gemeinsam

	(Multi-Faktor-Authentifizierung) genutzt werden können. Mögliche Faktoren sind: • Wissen (z.B. Passwort) • Besitz (z.B. Smartphone) • Inhärenz (z.B. biometrische Attribute)
Verbindungen zu anderen Modulen	• Authorization
Patterns	• Single-Faktor-Authentifizierung (SFA) • Zwei- bzw. Multi-Faktor-Authentifizierung (2FA bzw. MFA)
Bestehende Produkte	• Spring Security • OAuth2.0 /OpenID connect server • Keycloak • Okta
Herausforderungen	• Die Authentifizierung muss service-übergreifend implementiert werden (Single Sign On (SSO)) • Bei zentraler Implementierung: Ein zentraler Authentifizierungs-Service kann ein Bottleneck darstellen • Bei dezentraler Implementierung: Das Sperren bestehender Nutzer-Logins (bspw. anhand einer Blacklist) ist erschwert
Ähnliche / Verbundene Konzepte	• Authorization
Entscheidungen / Fragen bei der Implementierung	• Soll die Authentifizierung zentral oder dezentral implementiert werden? • Wie viele und welche Authentifizierungs-Faktoren sollen berücksichtigt werden? • Ist Blacklisting für bestehende Nutzer (d.h. automatische Logouts) notwendig?
Quellen und weiterführende Informationen	https://www.keycloak.org/ https://jwt.io/

Tabelle 21: Authentication

	#13
Komponente	**Authorization**
Zweck/Ziel	Der Zweck des Authorization-Moduls ist für einen bereits authentifizierten Nutzer (#10) servicespezifische oder -übergreifende Rechte zu prüfen und durchzusetzen.
Generisch / optional	Generisch
Funktions-beschreibung	Das Authorization-Modul verwaltet die Zugriffsrechte von Nutzern oder Services auf die verschiedenen APIs (insb. bzgl. der Datenservices (#1)) der Plattform. Die Autorisierung, d.h. die Zulassung oder Blockierung (im Falle fehlender Rechte) einer Nutzeranfrage kann zentral (bspw. mit Hilfe eines API Gateways) oder dezentral (Services implementieren die Autorisierung selbst) umgesetzt werden. Analog zu den Eigenschaften des Authentifizierungs-Moduls (#10) kann eine zentral umgesetzte Autorisierung ein Bottleneck darstellen, eine dezentrale Implementierung erhöht dafür die Komplexität.
Verbindungen zu anderen Modulen	• Authentication • API Gateway
Patterns	-
Bestehende Produkte	• Spring Security • OAUTH2 • AWS IAM • Cloud IAM
Herausforderungen	• Wie lässt sich eine globale, serviceübergreifende Verwaltung von Nutzerrechten sicherstellen? • Die Autorisierung eines Nutzers muss über Service-Verkettungen hinweg funktionieren
Ähnliche / Verbundene Konzepte	• Service Mesh • API Gateway
Entscheidungen / Fragen bei der Implementierung	• Erfolgt die Implementierung zentral oder dezentral? • Sollen Zugriffsrechte (z.B. Lese- und Schreibrechte) zentral oder dezentral verwaltet werden?

Quellen und weiterführende Informationen	https://oauth.net/2/

Tabelle 22: Authorization

Anomaly Detection Tool

	#14
Komponente	**Anomaly Detection Tool**
Zweck/Ziel	Dieses Modul informiert Datenanbieter und Betreiber der BayernCloud über das Auftreten außerordentlicher Ereignisse.
Generisch / optional	Optional
Funktions-beschreibung	Dieses Modul überwacht verschiedene Datenströme auf der Plattform und Informiert Teilnehmer und Betreiber automatisiert und proaktiv über das Auftreten außerordentlicher Ereignisse, beispielsweise eine unerwünschte Nutzung der zur Verfügung gestellten APIs. Als Datenquellen dienen hierbei beispielsweise (Logs, Eventbroker, Montoringlogs, Tracinglogs, Applikationsmetriken). Durch Subscription auf diese Datenquellen werden Änderungen sofort detektiert und ausgewertet. Im Rahmen von Predictive Analytics kommen zudem Fehlermodelle zur Anwendung, die zur Vorhersage kritischer Ereignisse dienen. Tritt ein außerordentliches Ereignis auf, werden alle involvierten Akteure darüber informiert, indem automatisiert Nachrichten generiert und abgeschickt werden, die alle nötigen Kontextinformationen enthalten. Es ergeben sich folgende Unterfunktionen: • Subscriptionfunktion für Datenquellen. • Abruf von Monitoring-/Tracing-/Loggingdaten • Auswertung von Daten • Predictive Analytics • Erzeugung und Absendung von Warnungen und Berichten • Verschiedene Scopes für Betreiber und Datenanbieter
Verbindungen zu anderen Modulen	• Datenservice • Datenbank • Even-Driven Messaging

	• Monitoring Solution • Data Dashboard
Quellen und weiterführende Informationen	-

Tabelle 23: Anomaly Detection Tool

Zertifizierungstool

	#15
Komponente	**Zertifizierungstool**
Zweck/Ziel	Dieses Modul überprüft die BayernCloud kontinuierlich auf vordefinierte Kriterien.
Generisch / optional	Generisch
Funktions-beschreibung	Dieses Modul überprüft kontinuierlich die Sicherheitsmerkmale von Cloud-Systemen anhand von vordefinierten Kriterienkatalogen. Dies dient zur Unterstützung der Zertifizierung, die nur zu bestimmten Zeitpunkten durchgeführt wird und somit keine Aussage über die Einhaltung der Zertifikatskriterien zwischen den statischen Zertifizierungszeitpunkten getroffen werden kann. Über das Dashboard werden die zu evaluierenden Services und Regeln ausgewählt und die von den Cloud-Providern bereitgestellten APIs zur regelmäßigen Abfrage der Cloud-Ressourcen genutzt um die Ergebnisse mit den erwarteten Ergebnissen zu vergleichen. Das Resultat wird auf dem zugehörigen Dashboard angezeigt. Mögliche Unterfunktionen: • Veranlassung eines Überprüfungsprozesses • Automatisierte Überprüfung • Abruf von Ressourceninformationen • Auswertung von Daten • Bereitstellen von Informationen zu Kriterienkatalogen • Reporting der Resultate des Überprüfungsprozesses
Verbindungen zu anderen Modulen	• Datenservice • Datenbank • Storage

Quellen und weiterführende Informationen	https://github.com/clouditor/clouditor

Tabelle 24: Zertifizierungstool

4.3.4. Kommunikation und Synchronisation

In Microservice-basierten Architekturen werden einzelne Funktionalitäten innerhalb kleiner Softwarekomponenten (Microservices) gekapselt. Eine Anwendung setzt sich daher aus der Kombination verschiedener Microservices zusammen. Unabdingbar ist daher die Umsetzung geeigneter, robuster Kommunikationsparadigmen. Ruft ein Microservice synchron die Funktionalität eines anderen Microservices über dessen API auf, so entspricht dies einer Remote Precedure Invocation. Das entsprechende Modul (#16) innerhalb dieses Abschnitts beschreibt daher an dieser Stelle mögliche Variationspunkte. Da diese direkte Form der Kommunikation zweier Microservices fehlerbehaftet sein kann (etwa im Falle des temporären Ausfalls eines Kommunikationspartners) zeigt das Modul Circuit Breaker (#17) eine geeignete Lösung auf, durch welche sich synchrone Kommunikation robuster gestalten lässt. Ein alternativer Ansatz der Kommunikation setzt auf brokerbasierte oder brokerfreie, asynchrone Kommunikation mit Hilfe sogenannter Events. Mögliche Ansätze werden im Modul Event-Driven Messaging (#18) beschrieben. Sollen zwei oder mehrere Service-APIs miteinander verbunden werden, so kann man hierzu entweder auf Ansätze im Rahmen der API Composition (#19) oder auf Prinzipien der Command Query Responsibility Segregation (#20) zurückgreifen.

Remote Procedure Invocation

	#16
Komponente	**Remote Procedure Invocation (RPI)**
Zweck/Ziel	Das Remote-Procedure-Invocation-Modul beschreibt eine einfache Methode der Kommunikation zwischen zwei (Micro-)Services
Generisch / optional	Optional
Funktions-beschreibung	Microservices stellen ihre Funktionalität über eine zuvor definierte API zur Verfügung. Eine solche API kann sich nach bestimmten Standards richten (z.B. OData, JSON:API) oder auch proprietär sein. Die Kommunikation zwischen zwei Services läuft dann über (synchrone) Aufrufe der service-spezifischen APIs ab. Basierend einen API-Aufruf (Request) führt der die API anbietende Service die gewünschte Funktionalität aus und beantwortet den Aufruf entsprechend (Response). Sicherheitskritische Operationen benötigen zudem eine vorhergehende Authentifizierung bzw. Autorisierung der anfragenden Clients. Ein API-Aufruf wird direkt an die Schnittstelle des anbietenden Services gerichtet, dessen Lokation muss daher bekannt sein (vgl. #22 Service Discovery).

Verbindungen zu anderen Modulen	• Autorisierung • Authentifizierung • Circuit Breaker • (Client/Server Side) Service Discovery
Patterns	-
Bestehende Produkte	• REST • gRPC • Apache Thrift
Herausforderungen	• Services müssen direkt erreichbar sein, eine Service Discovery muss vorhanden sein und ggfs. ein Circuit Breaker, um die Stabilität des Systems zu erhöhen • Für unerreichbare Services kann ein entsprechender Retry-Mechanismus implementiert werden • Komplexe Operationen müssen in einer nicht blockierenden Art implementiert werden (etwa durch Rückgabe einer Referenz-ID, unter der die eigentliche Antwort zu einem späteren Zeitpunkt abgerufen wird)
Ähnliche / Verbundene Konzepte	• Event-Driven Messaging
Entscheidungen / Fragen bei der Implementierung	• Anwendungsfallbezogene Nutzung von RPI und Event-Driven Messaging
Quellen und weiterführende Informationen	• https://microservices.io/patterns/communication-style/rpi.html • https://docs.microsoft.com/de-de/azure/architecture/patterns/retry

Tabelle 25: Remote Procedure Invocation

Circuit Breaker

	#17
Komponente	**Circuit Breaker**
Zweck/Ziel	Synchrone Kommunikation zwischen zwei Services (RPI, #14) ist fehleranfällig, sobald einer der involvierten Services unerreichbar ist. Ein Circuit Breaker dient dazu, unnötige Kommunikation auf ein Minimum zu reduzieren, indem wiederkehrende fehlerhafte Verbindungen unterbunden oder umgeleitet werden.

Generisch / optional	Optional
Funktions-beschreibung	Ein Circuit Breaker fungiert als eine Art „Proxy" für Operationen, welche eine hohe Fehlerwahrscheinlichkeit aufweisen (z.B. Zugriffe auf externe Services). Der Circuit Breaker greift zunächst nicht ins Geschehen ein, zählt jedoch auftretende Fehler. Sobald ein definierter Schwellenwert überschritten wird, löst der Circuit Breaker aus bzw. geht in einen aktiven Zustand über. Konkret kennt ein Circuit Breaker folgende drei Zustände: • Geschlossen: im Normalzustand ist ein Circuit Breaker geschlossenen und damit passiv. Operationen werden so lange durchgeführt, bis ein definierter Fehlerschwellenwert überschritten wird. In diesem Fall geht der Circuit Breaker in den offenen Zustand über. • Offen: Im offenen Zustand antwortet der Circuit Breaker direkt auf eingehende Anfragen mit einer Fehlernachricht, ohne die damit verknüpfte Operation auszuführen. Der Sender bekommt somit ein sofortiges Feedback über den zu erwartenden Misserfolg („fail fast"), während das dem Misserfolg zugrundeliegende Problem aufgrund der ausbleibenden Last behoben werden kann („heal automatic"). Nach Ablauf eines Timers wechselt der Circuit Breaker in den halb-offenen Zustand. • Halb-Offen: Im halb-offenen Zustand führt der Circuit Breaker einen begrenzten Teil der durchzuführenden Operation aus. Werden diese erfolgreich bearbeitet, wechselt der Circuit Breaker in den geschlossenen (Ausgangs-)Zustand. Im Falle des Scheiterns einer Anfrage, wechselt der Circuit Breaker jedoch in den offenen Zustand zurück, um der Problembehebung weiterhin Zeit zu geben.
Verbindungen zu anderen Modulen	• API Gateway • RPI • Container Orchestration
Patterns	-
Bestehende Produkte	• Netflix Hystrix • Istio
Herausforderungen	• Services sollten auf Fehlermeldungen eines offenen Circuit Breakers anwendungsspezifisch reagieren können (z.B. Ausweichen auf einen alternativen Service)

	• Geeignete Schwellwerte sind anwendungsfallspezifisch und müssen ermittelt werden
Ähnliche / Verbundene Konzepte	• Monitoring • Load Balancing • Container Orchestration
Entscheidungen / Fragen bei der Implementierung	• Lohnt sich der durch einen Circuit Breaker verursachte Overhead oder soll auf einen einfacheren Retry-Mechanismus gesetzt werden? • Welche Faktoren sind bei der Definition der Schwellwerte eines Circuit Breakers relevant?
Quellen und weiterführende Informationen	• https://microservices.io/patterns/reliability/circuit-breaker.html • https://docs.microsoft.com/de-de/azure/architecture/patterns/circuit-breaker

Tabelle 26: Circuit Breaker

Event-Driven Messaging

	#18
Komponente	**Event-Driven Messaging (EDM)**
Zweck/Ziel	Das EDM-Modul beschreibt einen Ansatz zur Message-orientierten Kommunikation zwischen zwei oder mehreren (Micro-)Services, welche die Komplexität im Vergleich zu RPI (#14) etwas erhöht, dafür jedoch die Kopplung zweier Services auf ein Minimum reduziert.
Generisch / optional	Optional
Funktions-beschreibung	Bei einer eventgetriebenen Architektur werden Nachrichten eventgetrieben erzeugt und verarbeitet. Diese Nachrichten können entweder von einem Nachrichtenbroker anhand des Publish/Subscribe-Prinzips oder ohne Broker („brokerless") übertragen werden. Die Funktionsweise steht der RPI entgegen, wo sich Services meist unter Kennung ihrer jeweiligen Lokalität (z.B. Hostname) synchron und direkt aufrufen müssen. Bei der Nutzung eines Brokers erfolgt die Kommunikation asynchron, da eine Nachricht zunächst auf diesem zwischengespeichert und anschließend von einem oder mehreren Empfängern abgerufen wird.
Verbindungen zu anderen Modulen	• Authentication • Autorisation

Patterns	• Publish/Subscribe • Message Oriented Middleware (MOM)
Bestehende Produkte	• Apache Kafka • RabbitMQ • HiveMQ • ZeroMQ
Herausforderungen	• Implementierung eines Berechtigungskonzepts und Absicherung des Datenaustauschs • Delivery-Garantien (At-Least-Once vs. At-Most-Once vs. Exactly-Once Semantik) • Performanz des Brokers • Persistenz von Nachrichten
Ähnliche / Verbundene Konzepte	• RPI
Entscheidungen / Fragen bei der Implementierung	• RPI vs. Event-Driven Messaging • Welche Technologie passt am besten zum Anwendungsfall (Abwägung von Funktionalitäten, z.B. Routing von Nachrichten, gegenüber Performanz, Broker-basierte oder broker-less Architektur)? • Sollen Nachrichten nach ihrem Austausch gelöscht oder persistiert werden?
Quellen und weiterführende Informationen	• https://microservices.io/patterns/communication-style/messaging.html • https://docs.microsoft.com/de-de/azure/architecture/patterns/category/messaging

Tabelle 27: Event-Driven Messaging

API Composition

	#19
Komponente	**API Composition**
Zweck/Ziel	Die API Composition bietet eine Methodik an, um Daten, die auf verschiedenen (Micro-)Services verteilt sind, mit einer einzelnen Anfrage abzufragen
Generisch / optional	Optional

Funktions-beschreibung	Die API Composition verknüpft Daten, welche auf verschiedenen Services verteilt sind, indem Anfragen (typischerweise RPI, #14) an mehrere andere APIs gerichtet und deren Antworten anschließend kombiniert werden. Stellt man hierfür eine spezielle Komponente mit einer häufig deklarativen Schnittstelle zur Verfügung (bspw. den Datenkombinationsservice #3), so lässt sich hierdurch die Komplexität service-übergreifender Anfragen verringern. Eine weitere Eigenschaft einer dedizierten API-Composition-Komponente ist die Abstraktion verschiedener Protokolle. Häufig wird eine API-Composition auch innerhalb eines API-Gateways (#24) umgesetzt.
Verbindungen zu anderen Modulen	• Database per Service • Service Registry/Discovery • RPI
Patterns	-
Bestehende Produkte	• GraphQL
Herausforderungen	• API Compositions können die Kopplung zwischen Services erhöhen, da die Änderung einer angebundenen API auch immer in der Composition berücksichtigt werden muss • API Composition kann ineffiziente In-Memory-JOINs bedingen • Standard-Operationen, z.B. Filter, Sortieren, Pagination, können nur an den Nutzer weitergegeben werden, wenn der Original-Endpoint diese unterstützt. Eventuelle Endpoint-Semantik (z.B. HATEOAS zur Beschreibung dieser Funktionalitäten) muss interpretiert werden können • GraphQL: Resolvers müssen manuell entwickelt werden, Schema-Management (möglicher Ausweg: Datenkombinationsservice (#3) und Ontologie-Service (#2))
Ähnliche / Verbundene Konzepte	• API Gateway • CQRS
Entscheidungen / Fragen bei der Implementierung	• Organisiert man API Composition zentral (z.B. #3) oder dezentral (mehrere, spezialisierte API Compositions)? • Wie reagiert man auf die Änderungen von APIs? Gibt es eine zugrundeliegende API-Semantik, welche die Kompatibilität garantiert? Gibt es eine geeignete Versionierung?
Alternative Entscheidungen	• Statt einer API Composition kann auch eine materialisierte View erstellt werden, welche bestimmte, im Voraus definierte Kombinationen vorberechnet (CQRS, #18)

Quellen und weiterführende Informationen	• https://microservices.io/patterns/data/api-composition.html • https://docs.microsoft.com/de-de/dotnet/architecture/microservices/architect-microservice-container-applications/distributed-data-management

Tabelle 28: API Composition

Command Query Responsibility Segregation

	#20
Komponente	**Command Query Responsibility Segregation (CQRS)**
Zweck/Ziel	Command Query Responsibility Segregation implementiert einen alternativen Ansatz, Daten über Servicegrenzen hinweg zu kombinieren, sodass diese mit einer einzelnen Query abgefragt werden können.
Generisch / optional	Optional, stellt eine Architekturentscheidung dar, wenn Database per Service gewählt wurde.
Funktions-beschreibung	Mit Hilfe von CQRS werden Daten, welche aus verschiedenen Microservices stammen, in einer zusätzlichen (read-only) View materialisiert und mit Hilfe von Events synchronisiert. Dies ermöglicht die Trennung schreibender und lesender Zugriffe und erlaubt ein einfaches und vor allem performantes Abfragen der bereits kombinierten Daten. CQRS bietet somit eine skalierbare und performante Möglichkeit, Daten verschiedener Services in kombinierter Weise synchron zu halten.
Verbindungen zu anderen Modulen	• Database per Service erzeugt den Bedarf an diesem Punkt • Event-Driven Messaging wird benötigt, um die materialisierten Views zu synchronisieren
Patterns	• Command Query Responsibility Segregation (CQRS)
Bestehende Produkte	• Axon • Akka • Eventuate • Lagom
Herausforderungen	• Bei Verwendung des CQRS-Ansatzes können zeitweise Inkonsistenzen auftreten (Eventual Consistency) • Daten müssen mit Hilfe geeigneter Events synchron gehalten werden (erhöht Komplexität des Systems) • Gewünschte Datenkombinationen müssen im Voraus festgelegt werden, ein anschließendes Ändern / Einführen neuer Kombinationen ist aufwändig

Ähnliche / Verbundene Konzepte	• API Gateway • API Composition
Entscheidungen / Fragen bei der Implementierung	• Welche Datenkombinationen sollen im Voraus für die materialisierten Views festgelegt werden? • Wie synchronisiert man die Daten zwischen Services und Views in robuster Art und Weise, um mögliche Inkonsistenzen zu vermeiden? • Genügt Eventual Consistency für den vorgesehenen Anwendungsfall?
Alternative Entscheidungen	• API Composition – Anstatt Daten synchron und kombiniert vorzuhalten können diese auch zur Query-Laufzeit kombiniert werden. Dies verringert die Performanz, erhöht jedoch die Flexibilität. Beide Ansätze können auch kombiniert werden
Quellen und weiterführende Informationen	• https://microservices.io/patterns/data/cqrs.html • https://docs.microsoft.com/de-de/dotnet/architecture/microservices/architect-microservice-container-applications/distributed-data-management • https://docs.microsoft.com/de-de/azure/architecture/patterns/materialized-view

Tabelle 29: Command Query Responsibility Segregation

4.3.5. Basisinfrastruktur

Die Basisinfrastruktur beschreibt die technischen Komponenten und Methoden welche eingesetzt werden um eine Microservice basierte Plattformarchitektur zu ermöglichen. Die zuvor beschriebenen Datenservices werden beispielsweise in virtualisieren (Docker) Containern (#21) implementiert. Zum Management und der Provisioning dieser Container dient das Modul Container Orchestration (#22). Die Betrachtung ist dabei zunächst generisch, im später folgenden Kapitel der Verteilungssicht gehen wir auf eine konkrete Variante (Kubernetes) ein, welche im Rahmen von Entwicklung und Pilottests als favorisierte Option identifiziert wurde. Das Orchestrierungssystem übernimmt dabei allerdings nicht vollumfänglich die insgesamt benötigen Aufgaben. Beispielsweise die Verwaltung von Service Policies erfordert die Erweiterung um zusätzliche Tools wie etwa eines Service Meshs (#23). Ein großer Vorteil von Microservices ist ihre Dynamik und Skalierbarkeit. Applikationen können über neue Container schnell skaliert und migriert werden. Dafür wird allerdings ein Service Registry (#24) benötigt, welches Namensauflösung und gegebenenfalls weitere Funktionen in diesem Kontext bietet, um die Erreichbarkeit zu gewährleisten. Von außen sind Services über einen Loadbalancer (#25) erreichbar, der Anfragen unter einer Lastverteilungsstrategie an einzelne Instanzen weiterleitet. Je nach Architekturentscheidung kann es sein, dass auch Funktionen höherer Protokollebenen direkt in die Infrastruktur Module integriert sind. So kann ein Plattform weites API Gateway (#26) eigenständig oder in Zusammenhang mit dem Service Mesh APIs und Schnittstellen

und deren Zugriff verwalten. Neben der Verwaltung spielt auch die native Integration von Service- und Infrastrukturrelevantem Logging (#27) und Monitoring (#28) eine große Rolle.

Container und Orchestrierungssysteme können weitestgehend Plattform und Hersteller Unabhängig sein, ein wichtiges Design Kriterium für die BayernCloud Plattform. Dadurch bietet sich die Möglichkeit Infrastrukturanbieterübergreifend Services zu organisieren, auch auf unterschiedlichen IaaS und Hardwareressourcen, was dem Paradigma einer MultiCloud (#29) entspricht. Um Services in Form von Container Images in solch einer Umgebung provisionieren zu können erfolgt die Integration eines Deployment Managers (#30). Einzelne Services bzw. deren Instanzen können beliebig an das jeweilige Orchestrierungssystem eines unterstützten Anbieters verteilt werden. Der Domain Operator kann so Services im Cluster verwalten ohne sich über die darunterliegende Hardware kümmern zu müssen.

Container

	#21
Komponente	**Container**
Zweck/Ziel	Standardisierung der Schnittstelle zwischen Development und Operations, ermöglicht DevOps-Prozesse. Verringert den Aufwand beim Bereitstellen von Services und Applikationen und ermöglicht einen deklarativen Konfigurationsprozess.
Generisch / optional	Generisch
Funktions-beschreibung	Leichtgewichtige Containervirtualisierung, die auf den Kernelfunktionen LXC, Namespaces, Cgroup und Layered Filesystems aufbaut. Kernfunktionalität ist die Isolation von Ressourcen bei einer gemeinsamen Kernelnutzung. Ein Container ist ein spezieller Typ eines Prozesses, der isoliert von anderen Prozessen ist. Ressourcen dieses Prozesses können nur von diesem benutzt werden.
Verbindungen zu anderen Modulen	-
Patterns	-
Bestehende Produkte	• Docker • LXC • Nspawnd • Rocket

Herausforderungen	• Absicherung der Container • Datenbanken in Docker sind umstritten • Abtrennung hochkritischer Daten
Ähnliche / Verbundene Konzepte	-
Entscheidungen / Fragen bei der Implementierung	Welche Container Technologie soll verwendet werden?
Quellen und weiterführende Informationen	https://www.docker.com/ https://coreos.com/rkt/docs/latest/rkt-vs-other-projects.html

Tabelle 30: Container

Container Orchestration

	#22
Komponente	**Container Orchestration**
Zweck/Ziel	Damit ein Service immer verfügbar ist, kümmert sich die Container Orchestrierung um das Management der Container, auch über mehrere Server und Rechenzentren hinweg.
Generisch/optional	Generisch.

Funktions-beschreibung	Bei der Container Orchestration werden Container über mehrere physische oder virtuelle Maschinen (auch Nodes genannt) hinweg koordiniert. Zu den Hauptaufgaben zählen die Gewährleistung von Verfügbarkeit und Erreichbarkeit nach deklarativ spezifizierten Kriterien.
	Je nach Lösung werden die nachfolgenden Punkte mehr oder weniger stark umgesetzt.
	• Zusammenführen von Ressourcen in einem Master/Worker – Verhältnis.
	  ○ Master Nodes übernehmen die Orchestrierung
	  ○ Worker Nodes führen die Container aus
	• Automatisches Starten/Stoppen von Containern auf den einzelnen Nodes.
	• Deklarative Funktionsbeschreibung
	• Automatische Skalierung
	• Monitoring / Überwachung
	• Managen des Netzwerks von verschiedenen Container über verschiedene Provider / Host / Server hinweg

	• Eine rudimentäre Service Discovery Lösung muss vorhanden sein, wie beispielsweise durch CoreDNS in Kubernetes. • Kommunikation im Cluster – Netzwerkoverlays mit Kommunikationspolicys mit Services, Verschlüsselung, IP-basiert • Fehlerhafte Container werden erkannt und bei Bedarf neu gestartet
Verbindungen zu anderen Modulen	• Container • Load Balancing • Circuit Breaker • Service Registry/Discovery
Patterns	-
Bestehende Produkte	• Kubernetes • Docker Swarm • Apache Mesosphere
Herausforderungen	• Permanente Verfügbarkeit • Lastverteilung • Persistenz
Ähnliche / Verbundene Konzepte	• Load Balancing • Service Discovery • Service Mesh • Multicloud • Container
Entscheidungen / Fragen bei der Implementierung	• Wie wird Service Discovery umgesetzt? • Wie wird das Load Balancing umgesetzt? • Wie sind die Richtlinien für Netzwerkkommunikation zwischen Nodes? • Wie werden Storagelösungen integriert? • Wie erfolgt die Aufgabenverteilung bei der Administration? • Wie können Daten von verschiedenen Clustern/Cloud Providern so ausgetauscht werden, dass von den verschiedenen Orten abstrahiert wird? • Wie können Container aus einem Cluster auf Container aus einem anderen Cluster zugreifen? • Wie greifen Container aufeinander zu?
Quellen und weiterführende Informationen	https://docs.docker.com/samples/library/swarm/ https://kubernetes.io/ http://mesos.apache.org/

Tabelle 31: Container Orchestration

Service Mesh

	#23
Komponente	**Service Mesh**
Zweck/Ziel	Services Meshes ermöglichen die Implementierung von Kommunikationspolicies. Damit Services unter entsprechenden Richtlinien miteinander kommunizieren können, stellt ein Service Mesh alle benötigten Funktionen bereit. Die Umsetzung erfolgt oft über sogenannte Sidecar Container. Es ermöglicht die Abstraktion von Funktionalität und die reduziert den Aufwand innerhalb einzelner zu entwickelnder Services.
Generisch / optional	Optional
Funktions-beschreibung	Ein Service Mesh kann viele in diesem Kapitel beschriebene Module und Funktionalitäten enthalten. Mögliche Bestandteile sind: • Sidecar Proxy • Service Discovery • Load Balancing • Encryption • Authentication • Authorization • Circuit Breaker
Verbindungen zu anderen Modulen	-
Patterns	Service Mesh
Bestehende Produkte	Produkte die für ein Service Mesh benutzt werden können bzw. Sidecar Implementierungen für bestimmte Funktionalitäten eines Service-Mesh haben: • Istio • Consul
Herausforderungen	Komponenten eines Service Mesh sind außerhalb eines jeden Microservices bzw. einer Gruppe von Microservices. Der Traffic zu den Microservices muss durch das Service Mesh geleitet werden. Die Komponenten müssen daher leichtgewichtet sein. Weiter ist die Konfiguration eine Service Meshes mit Aufwand verbunden und kann zusätzliche Komplexität mit sich bringen.
Ähnliche / Verbundene Konzepte	• Load Balancing • Security • Logging

	• Authentification • Authorization • Circuit Breaker • Sidecar Pattern
Entscheidungen / Fragen bei der Implementierung	-
Quellen und weiterführende Informationen	https://istio.io/docs/ https://www.consul.io/docs/index.html

Tabelle 32: Service Mesh

Service Registry / Discovery

	#24
Komponente	**Service Registry / Discovery**
Zweck/Ziel	Damit sich einzelne Services in einem System finden können, müssen diese einander kennen oder über Dritte miteinander kommunizieren.
Generisch / optional	Generisch
Funktions-beschreibung	Bei einer zentralen Registry werden alle Services in einer Datenbank vorgehalten. • Service Registration (An- abmelden, Signalisieren der Verfügbarkeit) • Service Resolution - Finden eines Services bei Service Discovery • Optional - Konstante Health Checks der Verfügbarkeit • Optional - Caching von Service Adressen • Optional - Failover • Optional - Load Balancing
Patterns	
Bestehende Produkte	• Apache Zookeeper • Consul • Etcd • Eureka • „Kubernetes - CoreDNS“

Herausforderungen	• Hohe Verfügbarkeit, da Kernkomponente der Architektur • Load Balancing • Service Discovery über Multi-Cloud hinweg
Ähnliche / Verbundene Konzepte	• DNS • Load Balancing
Entscheidungen / Fragen bei der Implementierung	• Welche Strategie wird für das deployen von Service Discovery Clients verwendet? • Werden einheitliche Technologien bei den Microservices benutzt? • Wieviel Intelligenz soll in einem einzelnen Service sein? • Wie wird die Verfügbarkeit von Services und Service Discovery Clients sichergestellt?
Quellen und weiterführende Informationen	https://kubernetes.io/docs/tasks/administer-cluster/coredns/ https://www.consul.io/discovery.html

Tabelle 33: Service Registry / Discovery

Load Balancing

	#25
Komponente	**Load Balancing**
Zweck/Ziel	Die Load Balancing-Lösung soll die Last auf die Service- und Serverinfrastruktur verteilen und so für eine Stabilität des gesamten verteilten Systems sorgen. Im Umfeld einer Microservicearchitektur dient ein Load Balancing vor allem dazu Last zwischen verschiedenen Serviceinstanzen zu verteilen.
Generisch / optional	Generisch
Funktions-beschreibung	Ein Load Balancer verteilt Anfragen an einen Service auf mehrere Instanzen des Services. Dabei kann der Load Balancer als zentrale Stelle implementiert sein oder dezentral in jedem Service. Dazwischen existieren auch Mischformen, die bspw. das Sidecarprinzip umsetzen. Load Balancing Muster können zum Beispiel sein: • Round Robin: die eingehenden Anfragen werden abwechselnd auf die Serviceinstanzen verteilt. • Least Connections: die eingehende Anfrage wird an diejenige Serviceinstanz weitergeleitet, die aktuell die wenigsten aktiven Verbindungen hat. • Least Response Time: die eingehende Anfrage wird an diejenige Serviceinstanz mit der geringsten Antwortzeit weitergeleitet.

Verbindungen zu anderen Modulen	-
Patterns	-
Bestehende Produkte	• Istio • Nginx • Ribbon • Traefik
Herausforderungen	• Lastverteilung muss verbunden sein mit Container Orchestrierung, sodass bei Bedarf neue Containerinstanzen erstellt bzw. zerstört werden. • Die Last muss korrekt dimensioniert werden.
Ähnliche / Verbundene Konzepte	• Container Orchestration • Service Mesh • Circuit Breaker
Entscheidungen / Fragen bei der Implementierung	• An welchen Stellen sollen sich Load Balancer befinden? • Durch welche Parameter wird Last beurteilt? • Wie ist die Verbindung zur Container Orchestrierung?
Quellen und weiterführende Informationen	https://docs.nginx.com/nginx/admin-guide/load-balancer/http-load-balancer/ https://docs.traefik.io/routing/services/

Tabelle 34: Load Balancing

API Gateway

	#26
Komponente	**API Gateway**
Zweck/Ziel	Das API Gateway stellt die zentrale Schnittstelle unterschiedlicher Plattformservice nach außen dar. Es dient als Einstiegspunkt für verschiedene Clients und stellt spezifische APIs für unterschiedliche Clients bereit. Es abstrahiert das Backend und die einzelnen Services.
Generisch / optional	Optional

Funktions-beschreibung	• Abstrahierung von verschiedenen Protokollen der einzelnen Services • Aggregation von verschiedenen einzelnen Serviceaufrufen • Stellt ggfs. Clientspezifisches API bereit • Verwalten des Lebenszyklus der APIs • Optionales Logging • Optionale Integration von Authentifizierung / Autorisierung
Verbindungen zu anderen Modulen	• Autorisierung • Authentifizierung • Circuit Breaker • Service Discovery • Logging
Patterns	• API Gateway per Client (z.B. Mobile vs. Desktop) • API Composition
Bestehende Produkte	• Kong • Ambassador • Amazon API Gateway • Azure API Management • Google Cloud Endpoints
Herausforderungen	• API Management – wie können verschiedene APIs oder auch Gateways verwaltet werden. • Single Point of Failure – Das API Gateway kann einen kritischen Punkt in der Architektur darstellen, dessen Ausfall abgesichert werden muss.
Ähnliche / Verbundene Konzepte	• API Composition
Entscheidungen / Fragen bei der Implementierung	• Wird alles zentral über das API Gateway laufen? • Wie sichere ich die Verfügbarkeit und die Skalierbarkeit des Gateways ab? • Wie ermöglicht man ein automatisches An- und Abmelden von APIs • Wie geht man mit Veränderungen an den APIs um

Quellen und weiterführende Informationen	https://docs.konghq.com/ https://cloud.google.com/endpoints?hl=de https://azure.microsoft.com/de-de/services/api-management/ https://aws.amazon.com/de/api-gateway/

Tabelle 35: API Gateway

Logging

	#27
Komponente	**Logging Solution**
Zweck/Ziel	Die Logdaten der einzelnen Services sollen gespeichert und für andere Dienste zur Verfügung gestellt werden.
Generisch / optional	Optional
Funktions-beschreibung	Die Kollektion und Aggregation von Logdaten über unterschiedliche Plattformkomponenten und Services muss für die weitere Verarbeitung sichergestellt werden. Plattformweite Loggingschnittstellen, insbesondere auf Basis von Streaming Technologien, sind gängige Konzepte um dies umzusetzen. Ein dedizierter Logging Service kann die Daten dann aufgreifen. Die Verarbeitung besteht im Wesentlichen in einer Reduzierung der Datenmengen durch Filter und Aggregate. Gespeicherte Logs sollen für andere Services abrufbar sein (beispielsweise der Monitoring Solution).
Verbindungen zu anderen Modulen	• Database per Service • Datenservice • Event-Driven-Messaging
Patterns	-
Bestehende Produkte	• InfluxDB • Logstash • Prometheus
Herausforderungen	• Eingrenzung auf relevante Logdaten, ohne wichtige Informationen zu verlieren. • Verbindungen von Logs einer Anfrage über mehrere Services und Technologien hinweg
Ähnliche / Verbundene Konzepte	• Monitoring Solution

Entscheidungen / Fragen bei der Implementierung	• Wie ermögliche ich das Logging zwischen verschiedenen Services und wie visualisiere ich dies? • Wie ist es möglich einen Stacktrace über verschiedene Services zu verfolgen und möglichst wenig Datenrauschen zu erhalten? • Setzt man auf zentrales Logging oder dezentrales Logging?
Quellen und weiterführende Informationen	https://docs.influxdata.com/influxdb/v1.7/ https://prometheus.io/docs/ https://www.elastic.co/guide/en/logstash/7.6/index.html

Tabelle: 36 Logging

Monitoring

	#28
Komponente	**Monitoring Solution**
Zweck/Ziel	Damit der Gesundheitszustand des Plattformökosystems erhalten werden kann, müssen die Zustände der einzelnen Services überwacht werden. Weicht ein Service vom erwarteten Verhalten ab, wird dies von der Monitoring Solution erkannt und gemeldet.
Generisch / optional	Optional

Funktions-beschreibung	Eine Monitoring Solution
	• Verarbeitet die Logdaten aus dem Logging-Dienst
	• Führt Healthchecks durch, sofern dies nicht von der Service Registry gemacht wird
	• Aggregiert die Daten
	• Visualisiert diese ggf. (optional)
	• Gibt Warn-/Fehlermeldungen aus, wenn sich Services anders verhalten, als sie sollten
	  ○ Vergleich gegen definierte Grenzwerte
	  ○ Vergleich mit Modellen
	• Die Meldungen können beispielsweise in einem Dashboard angezeigt werden oder mit einem Nachrichtendienst an das Operations Team geschickt werden

Verbindungen zu anderen Modulen	• Logging Solution • Service Discovery
Patterns	-

Bestehende Produkte	• Grafana • Kibana • Spring Boot Admin • Graphite • Collectd • Zipkin
Herausforderungen	• Erkennung und Betrachtung nur relevanter Metriken und Warnungen • Setzen von geeigneten Grenzwerten, innerhalb derer ein Service als gesund gesehen wird (z.B. Anzahl erfolgreicher Log-In Versuche pro Stunde, durchschnittliche Responsezeiten, ...) • Implementierung geeigneter, robuster Modelle zur Erkennung von Anomalien in den Metriken (Zeitreihendaten)
Ähnliche / Verbundene Konzepte	• Logging
Entscheidungen / Fragen bei der Implementierung	• Wie führe ich Logs verschiedener Services so zusammen, dass diese leicht konsumierbar sind und bei der Fehlersuche helfen? • Wie visualisiere ich das Monitoring?
Quellen und weiterführende Informationen	https://grafana.com/docs/grafana/latest/ https://www.elastic.co/guide/en/kibana/7.6/index.html

Tabelle 37: Monitoring

Multi Cloud

	#29
Komponente	**Multi Cloud**
Zweck/Ziel	Das Modul Multi Cloud ermöglicht es, bei der den Services der BayernCloud zugrundeliegenden Infrastruktur aus verschiedenen Anbietern wählen zu können.
Generisch / optional	Generisch
Funktions-beschreibung	Das Modul stellt ein grundlegendes Designprinzip dar, welches die Infrastrukturangebote verschiedener Anbieter unterstützten soll und für die Plattform nutzbar macht. Auf Basis der integrierten Infrastrukturanbieter

	kann so für jeden Service individuell ausgewählt werden, auf welcher Infrastruktur er läuft. Außerdem wird der Umzug einzelner Services zwischen Anbietern ermöglicht.
Verbindungen zu anderen Modulen	• Authorisation • Authentication • Service Mesh • Container Orchestration • Database per Service • Event-Driven Messaging
Patterns	
Bestehende Produkte	• Amazon Web Services • Google Cloud Platform • Microsoft Azure • Alibaba Cloud • IBM Cloud
Herausforderungen	• Sicherstellung der Abbildung von und Konformität mit verschiedenen technischer Anforderungen und rechtlichen Rahmenbedingungen der Infrastrukturanbieter
Ähnliche / Verbundene Konzepte	• Public Cloud • Private Cloud • Hybrid Cloud
Entscheidungen / Fragen bei der Implementierung	• Welche Anbieter werden in die BayernCloud integriert? • Wie erfolgt die Anbindung beim Infrastrukturanbieter?
Quellen und weiterführende Informationen	https://www.heise.de/tipps-tricks/Multi-Cloud-Was-ist-das-und-was-bringt-das-4264009.html

Tabelle 38: Multi Cloud

Deployment Manager

	#30
Komponente	**Deployment Manager**
Zweck/Ziel	Dieses Modul dient der Instanziierung und dem Lifecycle Management neuer Services auf beliebigen Ressourcen.

Generisch / optional	Generisch
Funktions-beschreibung	Um Services unterschiedlicher Art innerhalb der Plattform zur Verfügung zu stellen müssen je nach Container und Orchestrierungssystem Auswahl unterschiedliche Konfigurationen erstellt werden. Der meist manuelle Prozess erfordert Aufwand der insbesondere die einfache Integration von Services 3ter in der Plattform erschwert. Das Konzept eines Deployment Manager adressiert diese Herausforderung. Es werden Funktionalitäten zur Verfügung gestellt die es ermöglichen über die Aufnahme der wichtigsten Deployment Parameter die Erstellung der Konfiguration und Deployment Details automatisiert zu erstellen, für unterschiedliche Zielsysteme. Die Funktionalität kann programmatisch über eine API konsumiert werden oder durch eine native Integartion in das graphische User Interface der Plattform.
Verbindungen zu anderen Modulen	• Docker • Container Orchestrierung • Deployment Dashboard • Service Registry
Patterns	-
Bestehende Produkte	-
Herausforderungen	Der Deployment Manager muss für die gewählten Orchestrierungs-Ziel Systeme entsprechend angepasst und entwickelt werden.
Ähnliche / Verbundene Konzepte	-
Entscheidungen / Fragen bei der Implementierung	Wie Viele Orchestrierungssysteme oder IaaS Provider müssen unterstützt werden. In wie weit soll eine Integration mit Service Management und Service Monitoring erfolgen?
Verbindungen zu anderen Modulen	• Orchestrierungssystem • Docker • Deployment Dashboard
Quellen und weiterführende Informationen	https://bitbucket.org/amdatulabs/amdatu-kubernetes-deployer https://techbeacon.com/devops/one-year-using-kubernetes-production-lessons-learned

Tabelle 39: Deployment Manager

4.3.6. Persistenz

Microservice-basierte Architekturen ermöglichen die Entwicklung durch individuelle Teams und sind somit in höchstem Maße DevOps-konform. Um Eigenschaften wie individuelle Skalierbarkeit und Deployment zu erhalten wird die Kopplung zwischen einzelnen Microservices möglichst gering gehalten und somit die Autonomie einzelner Services gewahrt. Ein wichtiges Prinzip ist hierbei das sogenannte Database-per-Service-Pattern (#31), wonach Microservices den Zugriff auf ihnen zugrundeliegenden Datenquellen jeweils exklusiv haben. Solche Datenquellen werden durch (häufig externe) Datenbanken (#32) implementiert, eine Software, welche die Datenhaltung beispielsweise auf Basis eines relationalen Schemas oder NoSQL-Technologien ermöglicht. Die eigentliche (physische) Datenhaltung findet in Cloudumgebungen hierbei jedoch nicht innerhalb der Datenbanksoftware, sondern auf cloudbasierten Storage-Modulen (#33) statt.

Database per Service

	#31
Komponente	**Database per Service**
Zweck/Ziel	Um die Entkopplung von Microservices zu gewährleisten und diese unabhängig voneinander deployen und skalieren zu können, sollte die Datenhaltung exklusiv dem jeweiligen Eigentümer zur Verfügung stehen.
Generisch / optional	Generisch
Funktions-beschreibung	Jeder Service verfügt über seine eigene persistente Datenhaltung. Der Zugriff auf die Daten eines Service durch andere Services findet daher ausschließlich über dessen APIs statt. Database-per-Service impliziert keine physische Trennung (d.h. Daten verschiedener Services dürfen innerhalb desselben Datenbankservers liegen), sondern bezieht sich lediglich auf deren logische Trennung. Die konsequente Abkapslung der Daten verschiedener Services ermöglicht darüber hinaus ein unkompliziertes Austauschen der verwendeten Datenhaltungstechnologie (vgl. Datenbank-Modul, #30).
Verbindungen zu anderen Modulen	• API Composition • CQRS • API Gateway
Patterns	-
Bestehende Produkte	-
Herausforderungen	• Wie wird die strikte Trennung der Datenquellen sichergestellt bzw. erzwungen? Bei geteilten Datenbankservern könnte dies problematisch sein

	• Das Database-per-Service-Modul ermöglicht unabhängige Skalierbarkeit und Deployment, erschwert jedoch den serviceübergreifenden Datenzugriff. Dies führt zur Notwendigkeit von API Composition (#17) und CQRS (#18)
Ähnliche / Verbundene Konzepte	-
Entscheidungen / Fragen bei der Implementierung	• Welche Speichertechnologie wird bei welchem Service verwendet? • Ist die Datenhaltung eines Services Bestandteil des Services oder wird diese herausgelöst? • Werden Daten nur logisch (Private-tables-per-service bzw. Schema-per-service) oder auch strikt physisch (Database-server-per-service) voneinander getrennt?
Quellen und weiterführende Informationen	• https://microservices.io/patterns/data/database-per-service.html

Datenbank

	#32
Komponente	**Datenbank**
Zweck/Ziel	Dieses Modul bildet die Persistierung von Daten von Microservices ab.
Generisch / optional	Generisch
Funktions-beschreibung	Dieses Modul bietet eine Persistierungsschicht für Datenservices. Wie die Datenservice unterliegt jede Datenbank einem Schema, das aus der Ontologie des Ontologie-Services stammt. Der Zugriff auf die Datenbank findet lediglich durch die Instanz des Moduls Datenservice statt, dem das selbe Schema zu Grunde liegt. Durch CRUD-Operationen werden Daten erstellt, abgefragt, aktualisiert oder gelöscht. Es ergeben sich folgende Unterfunktionen: • CRUD-Operationen in der jeweiligen DBEngine Syntax. • Extended Logging
Verbindungen zu anderen Modulen	• Datenservice • Storage

Patterns	-
Bestehende Produkte	• Relationale Datenbankmanagementsysteme (z.B. MySQL, PostgreSQL) • NoSQL-Datenbanken (z.B. Neo4j, MongoDB) • NewSQL Datenbanken (z.B. HyPer) • Semantische Datenbanken (z.B. Virtuoso, Ontotext GraphDB)
Quellen und weiterführende Informationen	-

Tabelle 40: Datenbank

Storage

	#33
Komponente	**Storage**
Zweck/Ziel	Das Storage-Modul beschreibt Möglichkeiten, die physische Persistenz von Daten aus ihren ursprünglichen Services auszulagern.
Generisch / optional	Optional
Funktions-beschreibung	Das Storage-Modul bezieht sich auf den physischen Speicher, auf dem beispielsweise eine Datenbank (#30) die ihr zur Verfügung gestellten Daten persistiert. Es beschreibt somit die Möglichkeiten, die physische Persistenz von Daten aus ihren ursprünglichen Services auszulagern. Hierzu kann beispielsweise auf verschiedene Cloud-Speicher (z.B. Object-Storage) zurückgegriffen werden.
Verbindungen zu anderen Modulen	-
Patterns	-
Bestehende Produkte	• AWS S3 Object Storage • Azure Blob Storage • Google Cloud Storage
Herausforderungen	• Technologien sind meist Provider-spezifisch

Ähnliche / Verbundene Konzepte	• Database per Service • Datenbank
Entscheidungen / Fragen bei der Implementierung	• Welche Storage-Technologie eignet sich für den vorgesehenen Use-Case?
Alternative Entscheidungen	• Entscheidung für nur dezentrale Datenhaltung
Quellen und weiterführende Informationen	• https://aws.amazon.com/de/s3/ • https://azure.microsoft.com/de-de/services/storage/blobs/ • https://cloud.google.com/storage/

Tabelle 41: Storage

4.3.7. Bezug zu Use Cases

Wie sich aus den obigen Beschreibungen ergibt sind die bis hier hin beschriebenen funktionalen Module hierarchisch aufgebaut und beziehen sich auf unterschiedliche Abstraktionsebenen – beispielsweise beschreibt das Modul Datenservice eine Sammlung domänenspezifischer APIs und das Database-per-Service-Modul ein jedem Datenservice zugrundeliegendes Architektur-Prinzip. Das Datenbank-Modul beschreibt die logische Verwaltung der eingespeisten Daten und das Storage-Modul beschreibt schließlich deren (physische) Persistenz. Aus diesem Zusammenspiel ergibt sich ein Bezug verschiedener Module zu den in Kapitel 4.2 definierten Use Cases, welche in der folgenden Tabelle abgebildet sind. Dabei weist ein X auf einen direkten Bezug des Moduls auf den Use Case hin, mit einen * markierte Verbindungen sind optional.

Modul	Use Case Nr.						
	1	2	3	4	5	6	7
#1 Datenservice	X	X	X		X		
#2 Ontologie-Service	X	X	X		X		
#3 Datenkombinationsservice			X				
#4 Dokumentationsservice	X	X				X	
#5 Schema Management	X	X	X				
#6 Dublettenprüfung	X	X	X		X		
#7 Personal Dashboard						X	
#8 Data Dashboard				X			X
#9 Business Dashboard				X	X	X	X
#10 Deployment Dashboard				X	X		X
#11 Data Information Dashboard	X	X		X			X
#12 Authentifizierung	X	X	X	X	X	X	X
#13 Autorisierung	X	X	X	X	X	X	X
#14 Anomaly Detection	X	X	X				
#15 Zertifizierungstool	*	*	*	*	*	*	*
#16 Remote Procedure Invocation			X				

#17 Circuit Breaker			*				
#18 Event-Driven Messaging			*				
#19 API Composition			*				
#20 Command Query Responsibility Segregation			*				
#21 Container	X	X	X	X	X	X	X
#22 Container Orchestration	X	X	X	X	X	X	X
#23 Service Mesh	*	*	*	*	*	*	*
#24 Service Registry	X	X	X	X	X	X	X
#25 Load Balancing	X	X	X	X	X	X	X
#26 API Gateway	*	*	*	*	*	*	*
#27 Logging	X	X	X	X	X	X	X
#28 Monitoring	X	X	X	X	X	X	X
#29 Multi Cloud	*	*	*	*	X	*	*
#30 Deployment Manager	*	*	*	*	X	*	*
#31 Database per Service	X	X					
#32 Datenbank	X	X					
#33 Storage	X	X					

Tabelle 42: Übersicht der technischen Module in Bezug auf die BayernCloud Use Cases

4.4. Ökosystemsicht

4.4.1. Einleitung

Die Ökosystemsicht adressiert verschiedene, primär nicht technische Komponenten und Eigenschaften digitaler Plattform Ökosysteme. Hierbei richtet sie sich im Speziellen an mögliche Plattformbetreiber, aber auch (Domänen-)Akteure, die als Teil eines solchen Ökosystems aktiv wären (z.B. Drittentwickler oder Unternehmen). Im Detail behandelt diese Sicht der Referenzarchitektur die Interaktionen und Wertströme zwischen Akteuren eines auf Basis der Referenzarchitektur instanziierten Plattform-Ökosystems. Zur Steuerung des Plattform-Ökosystems wird ein generisches Governancemodell vorgestellt, das einen zentralen Bestandteil zur Steuerung und Wachstum entsprechender Ökosysteme darstellt.

Für die verschiedenen Elemente der Ökosystemperspektive, Governancemodell und Geschäftsmodell werden deren einzelne Elemente vorgestellt und anhand von Use Cases beispielhaft ihre Anwendung illustriert. Somit entsteht ein Leitfaden, wie diese Sicht der Referenzarchitektur für branchenspezifische Instanziierungen umgesetzt werden könnten. Wichtig ist dabei, die entsprechenden domänenspezifischen Besonderheiten zu beachten, um eine optimale Bedienung der unterschiedlichen Akteure zu erzielen.

4.4.2. Governancemodell und Ökosystem

Der Begriff Governance leitet sich vom englischen Government (Regierung) ab. Es existieren verschiedene unscharfe Begriffsdefinitionen. Die österreichische Gesellschaft für Umwelt und Technik versteht unter dem Begriff Governance „allgemein das Steuerungs- bzw. Regelungssystem in einer Gesellschaft. Verschiedene Interessen von privaten und öffentlichen AkteurInnen (Bevölkerungsgruppen, Unternehmen, Politik und Verwaltung) werden über dieses System verhandelt und umgesetzt“ (Österreichische Gesellschaft für Umwelt und Technik).

Im Kontext von digitalen Plattform-Ökosystemen, spielt die Governance der verschiedenen Akteure und ihrer Interaktionen eine zentrale Rolle (Schreieck, Wiesche, & Krcmar). Governance von Softwareplattformen erfordert das Halten der Balance zwischen Kontrolle

durch den Plattformbetreiber und der Freiheiten der einzelnen Entwickler (Tiwana, Konsynski, & Bush, 2010). Die prototypischen Akteure und Stakeholder von digitalen Plattformökosystemen sind in Tabelle 43 aufgeführt und in Kürze beschrieben. Ähnliche Betrachtungsweisen finden sich auch in anderen relevanten Projekten und Referenzarchitekturen, wie etwa GAIA-X[1] oder dem International Data Spaces[2].

Stakeholder	Beschreibung
Plattformbetreiber	Als zentrale Komponente eines digitalen Plattformökosystems steht eine (mehrseitige) Plattform, die verschiedene Stakeholdergruppen miteinander verbindet. Im Kontext der BayernCloud verantwortet ein Domänenoperator die Instanziierung der Referenzarchitektur innerhalb einer bestimmten Domäne, legt Governanceregeln oder mögliche finanzielle Beiträge fest und führt die Implementierung durch oder beauftragt diese. Darüber hinaus ist er für die Koordination innerhalb der Domäne und mit der die Gesamtreferenzarchitektur verantwortenden Instanz als Betreiber verantwortlich.
Drittentwickler / 3rd-Party-Developer	Die Integration externer Entwickler ist ein zentraler Bestandteil künftiger, auf Basis der Referenzarchitektur instanziierter DPÖs - ein in der Praxis zunehmend eingesetztes Mittel im Bereich der digitalen Ökonomie basierend auf dem Ansatz von "Open Innovation". Das dadurch entstehende Ökosystem bietet ein gemeinsames, zentrales Wertversprechen, welches durch optionale Module erweitert wird. Diese Module sind in der Regel Daten, Services und andere meist digitale Dienstleistungen. Dieser zunehmend eingesetzte Mechanismus, dient dazu die Attraktivität einer Plattform für relevante Nutzer- und Interessensgruppen zu maximieren. Er basiert darauf, die Stakeholdergruppen durch geeignete Crowdsourcing- und Crowdinnovationsprozesse in die Weiterentwicklung der Plattform, die Entwicklung und Gestaltung individueller Dienste und in die Erstellung, Strukturierung und Bewertung von benutzergenerierten Inhalten einzubinden.
Daten Provider	Diese Stakeholdergruppe umfasst speziell im BayernCloud Ökosystem zentrale Anbieter von (branchenspezifischen) Daten. Hierunter fallen etwa Kommunal- und Landkreisverwaltungen sowie deren Unternehmen, Vertreter und Angestellte, aber auch öffentliche Einrichtungen. Sie verfügen häufig über umfangreiche Daten-Sammlungen und IT-Systeme. Entsprechend ist ihre Ein- und Anbindung von hoher Relevanz für eine leistungsfähige Instanziierung in Form von DPÖs.

[1] https://www.bmwi.de/Redaktion/DE/Dossier/gaia-x.html
[2] https://www.internationaldataspaces.org/

Endnutzer	Endnutzer sind die jeweiligen Nutzer eines bestimmten Services, Konsumenten von Daten oder anderer Produkt und Dienstleistungen. Die Endnutzer können spezifisch für eine bestimmte Domäne sein. Als Endnutzer können je nach Perspektive Privatpersonen, aber auch Unternehmen oder öffentliche Einrichtungen angesehen werden. Der Begriff des Endnutzers ist somit stets im jeweiligen Kontext zu verstehen und gegebenenfalls weiter zu spezifizieren.

Tabelle 43: Prototypische Akteure und Stakeholder von digitalen Plattformökosystemen

Die Governance von Softwareplattformen definieren Tiwana et al. (2010) darüber, wer innerhalb eines DPÖ welche Entscheidungen trifft. Detaillierter definieren Baars and Jansen (2012) die Governance von Softwareplattformen als die Verfahren und Prozesse mit Hilfe derer ein Unternehmen seine aktuelle oder künftige Position im Software-Ökosystem steuert, verändert oder behauptet. Mit Hilfe der Governance des Software Ökosystems sollen die Teilnehmer der Plattform bei der Erreichung ihrer Ziele unterstützt werden und vorhandene Ressourcen effizienter eingesetzt werden.

Die Produkte und Services von Komplementoren können anhand verschiedener Kriterien wie Materialisierung, Fokus und Individualisierung unterschieden werden. Tabelle 44 Kriterien zur Unterscheidung von komplementären Angeboten in DPÖs fasst dies zusammen.

Komplementor Angebot	Materialisierung		Fokus		Individualisierung	
	Online / Digital	Offline / Analog	Generisch	Domänenspezifisch	Generisch	Individualisiert
Produkt						
Service						

Tabelle 44 Kriterien zur Unterscheidung von komplementären Angeboten in DPÖs

Anhand der Kriterien können die Angebote von Komplementoren beschrieben und kategorisiert werden. Daraus wiederum können sich verschiedene Rollen von Komplementoren ergeben, die der Plattformbetreiber gezielt steuern und ansprechen kann.

Zusammenfassend wird im Kontext dieses Vorhabens unter dem Governancemodell ein Regelwerk verstanden, welches in erster Linie bestimmt, wer in dem Ökosystem teilnehmen darf, welche Abläufe und Prozesse vorgesehen sind, wie geschaffener Mehrwert verteilt und wie mögliche Konflikte gelöst werden.

Abgegrenzt wird der Begriff der Plattformgovernance von der IT-Governance, welche sich mit der Steuerung der unternehmensinternen IT befasst. Sie umfasst die Entscheidungsrechte und Verantwortlichkeiten innerhalb des Unternehmens (Krcmar, 2005). Hierzu existieren Frameworks wie COBIT oder ITIL, die bei der Planung, Umsetzung und Kontrolle von IT-Governance eingesetzt werden.

Kernaufgaben von Governance

Aufbauend auf dem Grundverständnis von Governance beziehungsweise dem des Governancemodells werden nun die Aufgaben und Ziele auf verschiedenen Ebenen eines Governancemodells vorgestellt und abgesteckt.

Mit Bezug auf Tiwana (2014) identifiziert Mumm (2017) drei Kernaufgaben der Governance von Software-Ökosystemen:

1) **Entscheidungsstrukturen**: Die Kommunikation rund um die Entscheidungsfindung und die Verteilung von Entscheidungsmacht innerhalb des Ökosystems
2) **Kontrollmechanismen**: Festlegung und Durchsetzung von Regeln und Mechanismen zur Sicherung eines kohärenten, hochwertigen Angebots
3) **Preissetzung**: Festlegung einer geeigneten Preispolitik, die die Interessen der Plattformseiten berücksichtigt und mit Hilfe von Anreizen versucht, das Wachstum der Plattform anzuregen und zu steigern

Wie in Tabelle 45 dargestellt, können die Governanceinhalte in verschiedene Dimensionen gegliedert werden. Diese umfassen Kontrollmechanismen, Standards, Schnittstellen, Zugang und Rollen, On- und Offboarding, Preissetzung, Berichtsstufen, Trust, Haftung und Dokumentation und Guiding.

Dimension	Aspekt	Beschreibung
Kontroll-mechanismen	Inhaltskontrolle	Bestimmung von Inhalten auf allen Ebenen der Governance und deren jeweilige Kontrolle
	Appropriability Mechanismen	Lizenzierung; Imitationsfähigkeit von Produkten und Services; Marken, Patente, Designs
	Anreize für Komplementoren	Schaffung von Anreizen für Entwickler als künftige Kundengruppe
	Prozesskontrolle von Komplementoren	Vorgabe bestimmter Methoden und Abläufe, um Entwicklungsprozess von Partnern zu steuern
	Prozesskontrolle von B2B Kunden	Vorgabe bestimmter Methoden und Abläufe, um Nutzungsprozesse von Business-Partnern zu steuern
	Informelle Kontrolle	Kontrolle/Steuerung über Werte, die Gemeinschaft bzw. Community und eine gemeinsame Identität; Richtlinien sind nicht bindend
	Sanktionelle Kontrolle	Durchsetzung von Sanktionen zur Kontrolle in Form von konkreten Handlungen wie Strafen und Auflagen
Standards	Kompatibilität und Interoperabilität	Grad und Gestaltung der Kompatibilität und Interoperabilität mit anderen Plattformen, Systemen und Services
	Vorgabe von Standards bezüglich Datenformat	Definition von proprietären (spezifischen) Standards und allgemeiner Standards für Datenformate und Datenhaltung

Schnittstellen	API (Application Programming Interface)	APIs stellen State-of-the-Art Schnittstellen dar um intern oder extern Daten und Services austauschen zu können
	SDK (Software Development Kit)	Ein SDK stellt eine Sammlung von Programmierwerkzeugen und Programmbibliotheken, die zur Entwicklung von Software in Form von Services oder Applikationen dient
Zugang & Rollen	Partnermodell	Festlegung von Rollen und Regeln für Kunden in verschiedenen Segmenten, in Abhängigkeit ihres Verhältnisses zum Ökosystem
	Partner Management	Aktives und zielgerichtetes Management der Partner, Akquise neuer Partner
Onboarding & Offboarding	On- & Offboarding Partner	On- & Offboardingprozess von Partnern
	Onboarding von Teilnehmern	Onboardingprozess von Teilnehmern
	Offboarding von Teilnehmern	Offboardingprozess von Teilnehmern
Preissetzung	Pricing Models	Abrechnungsmodelle und –mechaniken
	Subsidizing	Subventionierung einzelner Marktseiten durch Übernahme ihrer Kosten
	Revenue Sharing	Anteil des Plattformbetreibers am Umsatz/Gewinn; verschiedene Staffelungen möglich
	Payment Options	Zahlungsmöglichkeiten
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	Wege und Möglichkeiten der Kommunikation zwischen Stakeholdern und Stakeholdergruppen
	Kontaktmöglichkeit mit Plattformbetreiber	Möglichkeiten der Kontaktaufnahme Partner, Stakeholder, Kunden und Dritter mit Plattformbetreiber
	Community / Forum / Blog	Ermöglichen von Feedback und Austausch zwischen Stakeholdern und Stakeholdergruppen
Trust	Rating System	Bewertungssysteme für Services, Datensets u.a.
Dokumentation & Guiding	Dokumentation Onboarding	
	Dokumentation Datenformat Standards	Die für Daten definierten Standards müssen offen kommuniziert und klar dokumentiert werden
	API Dokumentation	Strukturierte, umfassende Beschreibung der Nutzungsmöglichkeiten der APIs

	SDK Dokumentation	Strukturierte, umfassende Beschreibung der Nutzungsmöglichkeiten des SDK

Tabelle 45: Dimensionen und Aspekte des Governancemodells als Grundlage für die Evaluierung der Use Cases

Eine genaue Beschreibung der Dimensionen findet sich im Abschlussbericht des Arbeitspakets 2 „Governancemodell". Die Dimensionen und ihre Aspekte dienen als Grundlage für die Evaluierung der Use Cases innerhalb der Referenzarchitektur.

4.4.3. Geschäftsmodell und Ökosystem

Auf Basis der durch diese Referenzarchitektur instanziierte Plattform-Ökosysteme führen zum Entstehen völlig neuer Geschäftsmodelle beziehungsweise der Anpassung bestehender Modelle. Ein Geschäftsmodell sagt aus, welcher Nutzen für Kunden gestiftet wird, wie dieser Nutzen erbracht wird und wie sich ein Unternehmen die Rückflüsse des gestifteten Nutzens in Form von Umsätzen sichert (Schallmo, 2013a, 2013b). Gemäß dieser Definition gibt ein Geschäftsmodell die Umsetzung der Strategie einer Unternehmung als logische Funktionsweise eines Unternehmens wieder und beschreibt insbesondere, auf welche Art Gewinne erwirtschaftet werden.

Digitale Plattformökosysteme stellen in der Regel mehrseitige Märkte dar. Das heißt, dass hier keine typische lineare Anbieter-Kunden-Beziehung herrscht, sondern das vielmehr verschiedene Gruppen von Akteuren miteinander verbunden werden und, meist über eine Plattform, Dienste und Ressourcen untereinander austauschen. Bei diesen mehrseitigen Geschäftsmodellen wird Wert insbesondere auch durch die Interaktion zwischen verschiedenen Akteuren erstellt, weshalb eine detaillierte Betrachtung der individuellen Wertströme hilfreich ist. Auf Basis der BayernCloud Referenzarchitektur können neue Geschäftsmodelle sowohl für das zu instanziierende Plattform-Ökosystem, als auch für die potentiellen Teilnehmer, ermöglicht werden. In diesem Abschnitt ist daher das Vorgehen zur Erstellung von Geschäftsmodellen auf Basis der Referenzarchitektur kurz dargestellt. In Abschnitt 4.4.4 sind die Geschäftsmodelle und Wertströme für spezifische Use Cases dargelegt. Eine detaillierte Übersicht über ein Vorgehen zur Entwicklung von Geschäftsmodellen ist im Detail im Dokument Abschlussbericht Arbeitspaket 4: Geschäfts- und Betreibermodell sowie Transfercenter beschrieben. Das Vorgehen zur Erarbeitung eines Geschäftsmodells beziehungsweise dessen Illustration je Use Case ist in Abbildung 98 dargestellt.

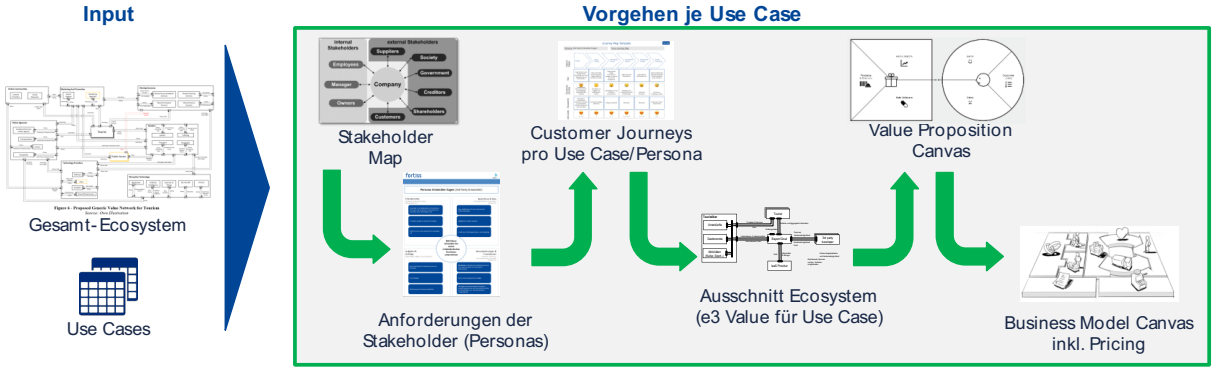

Abbildung 9: Vorgehen zur Erarbeitung eines Geschäftsmodells je Use Case

Wie in Abbildung 98 dargestellt, wird neben der Definition der Use Cases eine detaillierte Analyse der damit verbundenen Wertströme angefertigt. Für das Projekt BayernCloud wurde daher eine Analyse des europäischen Tourismus-Ökosystems durchgeführt. Die Ergebnisse der Analyse können im Abschlussbericht Arbeitspaket 4: Geschäfts- und Betreibermodell sowie Transfercenter gefunden werden und es erfolgt eine Aggergation der Ergebnisse zu einer wissenschaftlichen Veröffenlichung. Dies dient als Input, um in einem ersten Use-Case spezifischen Schritt sogenannte Stakeholder Maps zu erstellen, auf Basis derer Personas relevanter Stakeholder angefertigt werden. Für jede Persona wird eine Customer Journey erstellt. Diese ist Grundlage für eine Übersicht der Wertströme zwischen den involvierten Stakeholdern, welche mit der e3-Value Notation für die individuellen Use Cases dokumentiert werden. Aufbauend werden Value Proposition Canvas und Business Model Canvas pro Use Case erstellt. In einem abschließenden Schritt erfolgt die Ableitung von Use Case übergreifenden Geschäftsmodellen.

Im Folgenden werden die einzelnen Schritte bzw. Methoden des Vorgehens kurz vorgestellt.

Stakeholder Maps

Eine Stakeholder-Map ist eine visuelle oder physische Darstellung der verschiedenen Gruppen, die an einem bestimmten Produkt oder einer bestimmten Dienstleistung beteiligt sind, wie Kunden, Nutzer, Partner, Organisationen, Unternehmen und andere Interessengruppen (Stickdorn & Schneider, 2011). Mit Stakeholder Maps werden die wichtigsten Interessengruppen und ihre Beziehungen für die Geschäftsmodellentwicklung identifiziert. Das Zusammenspiel und die Verbindungen zwischen diesen verschiedenen Stakeholdern werden für die nächsten Schritte analysiert. Die Stakeholder Maps wurden auf Basis der durchgeführten Analyse der Wertströme des europäischen Tourismus-Ökosystems angefertigt.

Personas

Personas sind fiktive Charaktere, die die verschiedenen Nutzertypen der Dienstleistung oder des Produktes Ihres Unternehmens darstellen. Das Erstellen von Personas hilft dabei, die Bedürfnisse und Verhaltensweisen der Kunden zu verstehen (Anforderungen) und hilft somit zielgerichtete Lösungen für die Anliegen der Endnutzer zu entwickeln. Erstellte Personas wurden dann für eine effiziente Zusammenarbeit mit dem technischen Team sowie mit Partnern im Tourismus verwendet. Die Personas wurden auf Basis der durchgeführten Analyse der Wertströme des europäischen Tourismus-Ökosystems angefertigt.

Journey Maps

Customer Journeys visualisieren die verschiedenen Zyklen, die ein Kunde durchläuft, bevor er sich für den Kauf eines Produkts entscheidet. Daneben können Customer Journey genutzt werden, um die verschiedenen Touchpoints des Nutzers mit Systeme und weiteren Akteuren transparent zu visualisieren. Diese Berührungspunkte können durch Marktforschung identifiziert werden. Dazu werden Umfragen, Einzelinterviews, Fokusgruppen, Keyword-Recherchen oder Webanalysen verwendet. Im Projekt BayernCloud wurden die Journey Maps auf Basis der identifizierten Personas entwickelt.

Tabelle 46 zeigt die Zuordnung der Personas innerhalb der BayernCloud Referenzarchitektur zu den Use Cases.

#	Persona	Relevanz für Use Case						
		1	2	3	4	5	6	7
1	Entwickler Eugen	X	X	X	X	X	X	X
2	Entwickler Christoph	X	X	X	X	X	X	X
3	Marketingmitarbeitern Susanne	X					X	X
4	Entwickler Markus	X	X	X	X	X	X	X
5	Beamter Roman	X	X				X	X
6	Beamte Clara	X	X				X	X
7	Web Entwicklerin Sarah	X	X				X	X
8	Wegewart Rudi	X	X					X
9	Busfahrer Holger							X
10	Verwaltungsangestellte Anna						X	X
11	Sachbearbeiterin Lena						X	X
12	Hüttenbetreiber Armin	X	X					X
13	Urlaubsbauernhofbetreibern Johanna	X	X					X
14	Autorin Ariane	X	X				X	X
15	Entwickler Joachim	X	X					X
16	Tourist Fritz							X
17	Tourist Veronika							X
18	Tourist Hannes							X

Tabelle 46: Zuordnung der Personas zu den Use Cases der BayernCloud Referenzarchitektur

Wertströme auf Basis von e3-Value

Um die Werterzeugung innerhalb der Use Cases zu verstehen, müssen die Wertströme der involvierten Akteure im Detail verstanden werden. Auf diese Weise können mehrseitige Geschäftsmodelle konzipiert werden, wobei individuelle Wertversprechen für die unterschiedlichen Akteure entstehen müssen. Die Wertströme können mithilfe der e3-Value Notation modelliert werden. Ein e3-Value Modell veranschaulicht den Austausch von Werten, wie Produkten, Geld oder Dienstleistungen, unter verschiedenen Akteuren. Dieser erfolgt in einem idealen Umfeld und in einem festgelegten Zeitraum, der als Vertragslaufzeit bezeichnet wird. Dabei bieten e3-Value Maps einen Schnittpunkt zwischen technischen und geschäftsmodellfokussierten Anwendungen (Currie, 2004).

Value Proposition Canvas

Ein Value Proposition Canvas (VPC) ist eine graphische Darstellung einer Geschäftsidee, die einem vorgegebenen Muster folgt. Es besteht aus den Produkten und den Services, den gewinnbringenden und den schmerz- oder verlustmindernden Entitäten sowie den allgemeinen Gewinnen, den Schmerzen und den Kundenaufträgen. Dabei hilft die visuelle Veranschaulichung dabei, die Kunden zu verstehen und ein passendes Leistungsversprechen zu erstellen. (Osterwalder, Pigneur, Bernarda, Smith, & Papadakos, 2014) (Siehe **Fehler! Verweisquelle konnte nicht gefunden werden.**)

Business Model Canvas

Die Ergebnisse aus den vorherigen Schritten werden genutzt, um für die individuellen Use Cases Geschäftsmodelle aus Sicht des Plattformbetreibers zu erstellen. Zur Dokumentation

dieser wird das Business Model Canvas genutzt. Das Business Model Canvas (BMC) ist eine Methode, um eine Geschäftsidee graphisch zu modellieren und hat sich inzwischen zum de-facto Standard im Bereich der Geschäftsmodellmodellierung durchgesetzt. Dabei liegt der Fokus darauf, die Anliegen der Kunden zu verstehen. Eine Besonderheit bei dieser Visualisierung der Idee liegt in der großen Flexibilität, die viel Raum zum freien Entwurf verschiedenster möglicher Ideen lässt. Ein weiterer Aspekt, den das BMC auszeichnet, ist die Nachvollziehbarkeit, die durch die knappe, aber präzise Darstellungsweise erzielt wird (Osterwalder & Pigneur, 2010).

Das Business Model Canvas besteht aus verschiedenen Elementen zur Beschreibung von Geschäftsmodellen. Eine zentrale Rolle spielen in mehrseitigen und digitalen Geschäftsmodellen insbesondere die Erlös- bzw. Preismechaniken. Eine Übersicht der häufigsten Formen der Erlös- und Preismechaniken ist in 45 dargestellt.

Erlös- bzw. Preismechanik	Beschreibung
Verkaufserlös	Die klassische Erlösform für transaktionsorientierte Modelle. Sonderformen sind: • Pay-per-Use: Der Kunde zahlt für einzelne Services • Performance-based-payment: Die Bezahlung ist an ein bestimmtes Leistungsniveau gekoppelt
Abonnement-Modell / Flatrate / Subskription	Kunden zahlen periodisch einen festen Preis für eine bestimmte Leistung (häufig bspw. als Software-as-a-Service)
Transaktionsgebühr	Einer der Transaktionspartner zahlt dafür, dass die Transaktion durchgeführt wird
Kontingente	Kunden zahlen einen festen Preis für ein bestimmtes Kontingente, bspw. um 10.000 APIs aufzurufen
Preisfraktionierung bzw. Add-Ons	Die Grundversion des Angebots gibt es zu einem bestimmten, meist sehr geringen Preis, Erweiterungen (Add-Ons) sind kostenpflichtig
Freemium	Freemium lässt sich grundsätzlich ebenfalls der Mechanik der Preisfraktionierung zuordnen. Dabei gibt es eine kostenfreie Grundversion, die in der Regel auch durch Add-Ons erweitert werden kann. Zusätzlich gibt es eine/mehrere kostenpflichtige Premium-Varianten, die gegenüber dem kostenfreien Angebot einen erweiterten Funktionsumfang gewähren.
Dynamisches Pricing	Die Preise werden kontextbezogen generiert (z. B. zeitabhängig, abhängig vom Kundenprofil). Hierzu lässt sich auch pay-per-use zählen, bei dem der Kunde abhängig von der tatsächlichen Nutzung zahlt
Wertbasierte Finanzierung	Nicht der Kunde, sondern ein Werbetreibender zahlt für das Modell
Pay-per-use / pay-as-you-go	Kunden zahlen nur, was sie tatsächlich nutzen (also z.b. pay per call im Falle von APIs)
Pay-per-Link / Pay-per-Data	Kunden zahlen nicht mit monetären Mitteln, sondern durch Weiterempfehlung oder Preisgabe von Daten

Pay-per-improvement	Bezahlung in Abhängigkeit, wie stark der Prozess sich verbessert
Pay-per-saving	Beteiligung des Serviceanbieters an den Einsparungen
Hybride Modelle / Overage Modell	Kombination verschiedener Erlösmechanismen als hybrides Modell. Z. B. ein Abonnement für einen grundsätzliche Zugang, Kontingente für einen festgeschriebenen Leistungsumfang und pay-per-use für alle Leistungen, die diesen Umfang übersteigen

Tabelle 47: Übersicht der häufigsten Formen der Erlös- und Preismechaniken

4.4.4. Bezug zu Use Cases

Dieser Abschnitt evaluiert die Use Cases aus Sicht des Ökosystems. Dafür werden die Use Cases einzeln entlang der Inhalte des der Ökosystemsicht bewertet und aufgezeigt, welche Inhalte für die Umsetzung der jeweiligen Use Cases von Relevanz sind.

Für die Geschäftsmodelle werden je Use Case das entwickelte e3-Value Netzwerk zur Darstellung der Wertströme und das Business Model Canvas dargelegt. Darüberhinaus sind die für den jeweiligen Use Case relevanten Erlöslogiken dargelegt. Die jeweiligen Stakeholder Maps, Personas, Customer Journeys und Value Proposition Canvas sind in Abschlussbericht zu Arbeitspaket 4: Geschäfts- und Betriebsmodell sowie Transfercenter zu finden.

4.4.5.1 Use Case 1: Daten lesen

Use Case 1 beschreibt das Lesen von Daten, die über die Infrastruktur bereitgestellt werden.

<u>Governancemodell:</u>

Dimension	Aspekt	Relevanz für Use Case 1
Kontroll-mechanismen	Inhaltskontrolle	
	Appropriability Mechanismen	x
	Anreize für Komplementoren	x
	Prozesskontrolle von Komplementoren	x
	Prozesskontrolle von B2B Kunden	x
	Informelle Kontrolle	
	Sanktionelle Kontrolle	
Standards	Kompatibilität und Interoperabilität	x
	Vorgabe von Standards bezüglich Datenformat	x
Schnittstellen	API (Application Programming Interface)	x
	SDK (Software Development Kit)	
Zugang & Rollen	Partnermodell	x
	Partner Management	
Onboarding & Offboarding	On- & Offboarding Partner	
	Onboarding von Teilnehmern	

	Offboarding von Teilnehmern	
Preissetzung	Pricing Models	x
	Subsidizing	
	Revenue Sharing	x
	Payment Options	
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	
	Kontaktmöglichkeit mit Plattformbetreiber	x
	Community / Forum / Blog	
Trust	Rating System	x
Dokumentation & Guiding	Dokumentation Onboarding	
	Dokumentation Datenformat Standards	x
	API Dokumentation	x
	SDK Dokumentation	

Tabelle 48: Ökosystemsicht: Evaluation Governancemodell für Use Case 1

Geschäftsmodell:

Wertstromanalyse (e3-Value) für Use Case 1:

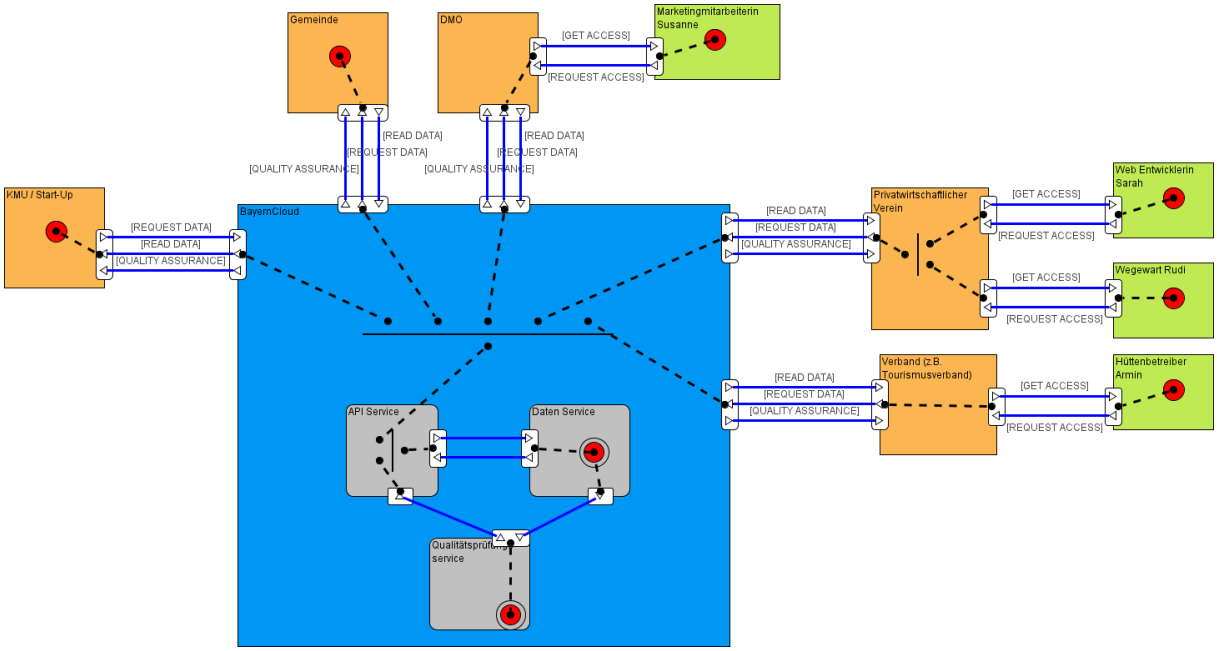

Abbildung 10: Wertstromanalyse für Use Case 1

Business Model Canvas für Use Case 1:

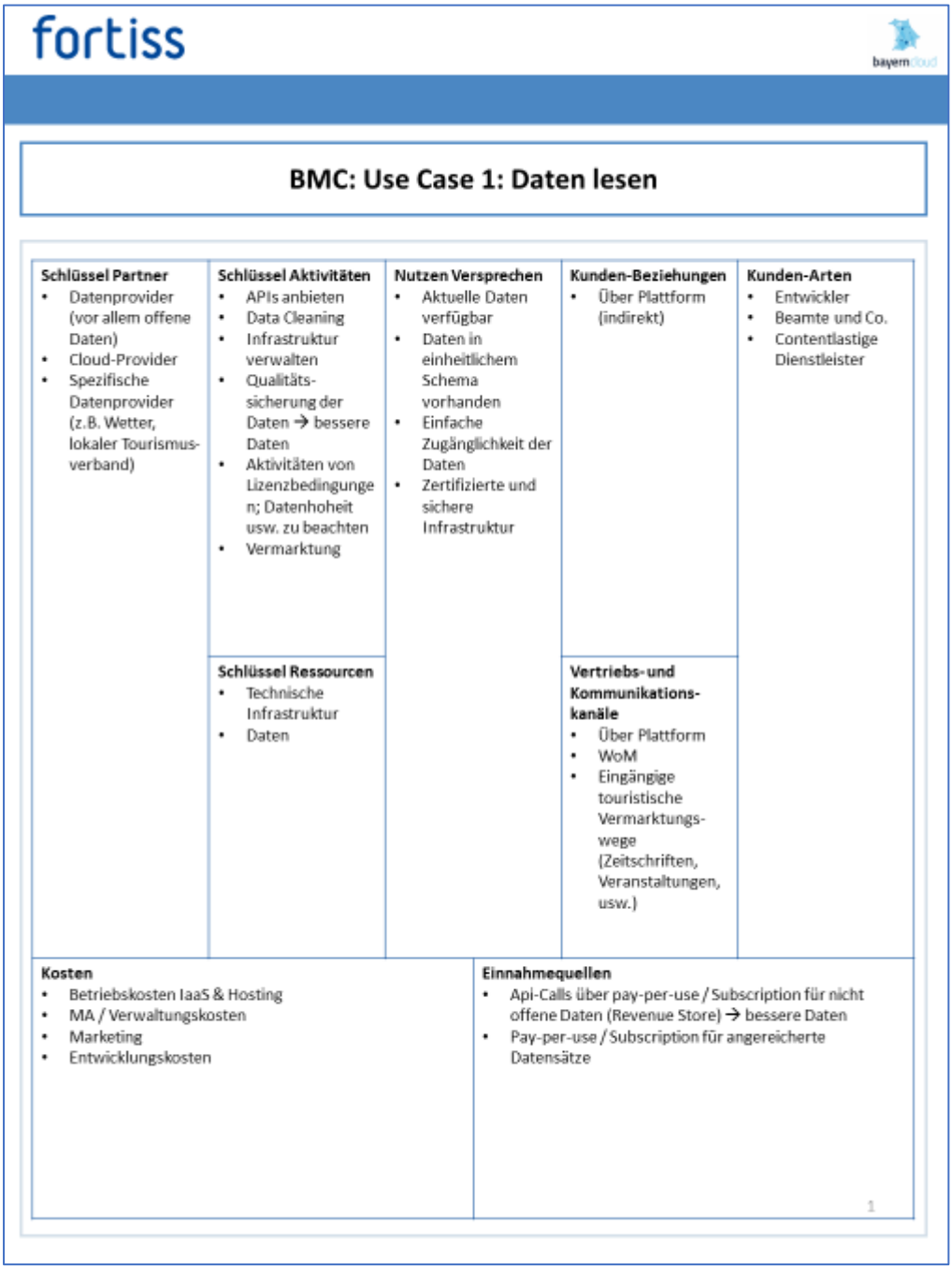

Abbildung 11: Business Model Canvas für Use Case 1

Potentielle Preismechanismen für Use Case 1:

Erlös- bzw. Preismechanik	Relevanz für Use Case
Verkaufserlös	
Abonnement-Modell / Flatrate / Subskription	X
Transaktionsgebühr	X
Kontingente	X
Preisfraktionierung bzw. Add-Ons	

Freemium	
Dynamisches Pricing	
Wertbasierte Finanzierung	
Pay-per-use / pay-as-you-go	X
Pay-per-Link / Pay-per-Data	X
Pay-per-improvement	
Pay-per-saving	
Hybride Modelle / Overage Modell	

Tabelle 49: Evaluation von Preismechanismen für Use Case 1

4.4.5.2 Use Case 2: Daten schreiben

Use Case 2 beschreibt das Schreiben von Daten über die Infrastruktur zur Bereitstellung für Dritte.

<u>Governancemodell:</u>

Dimension	Aspekt	Relevanz für Use Case 2
Kontroll-mechanismen	Inhaltskontrolle	x
	Appropriability Mechanismen	x
	Anreize für Komplementoren	x
	Prozesskontrolle von Komplementoren	x
	Prozesskontrolle von B2B Kunden	x
	Informelle Kontrolle	
	Sanktionelle Kontrolle	x
Standards	Kompatibilität und Interoperabilität	x
	Vorgabe von Standards bezüglich Datenformat	x
Schnittstellen	API (Application Programming Interface)	x
	SDK (Software Development Kit)	
Zugang & Rollen	Partnermodell	x
	Partner Management	
Onboarding & Offboarding	On- & Offboarding Partner	
	Onboarding von Teilnehmern	
	Offboarding von Teilnehmern	
Preissetzung	Pricing Models	x
	Subsidizing	x
	Revenue Sharing	x
	Payment Options	
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	
	Kontaktmöglichkeit mit Plattformbetreiber	x
	Community / Forum / Blog	x
Trust	Rating System	x

Dokumentation & Guiding	Dokumentation Onboarding	
	Dokumentation Datenformat Standards	x
	API Dokumentation	x
	SDK Dokumentation	

Tabelle 50: Ökosystemsicht: Evaluation Governancemodell für Use Case 2

Geschäftsmodell:

Wertstromanalyse (e3-Value) für Use Case 2:

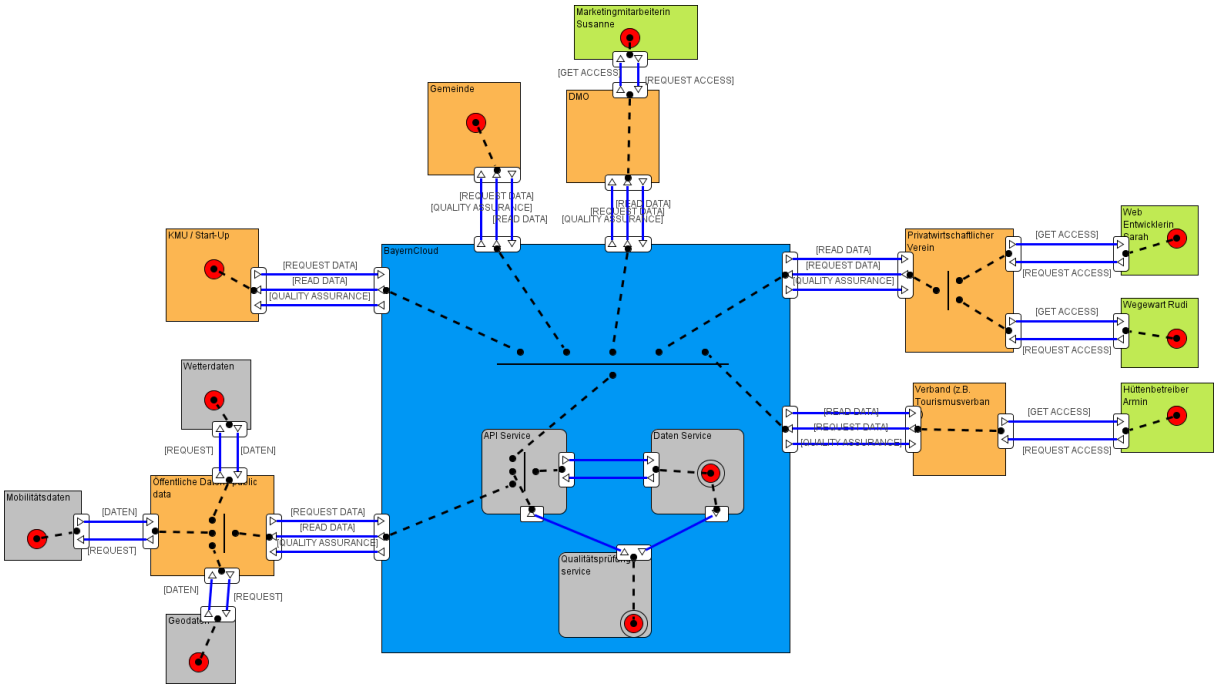

Abbildung 12: Wertstromanalyse für Use Case 2

Business Model Canvas für Use Case 2:

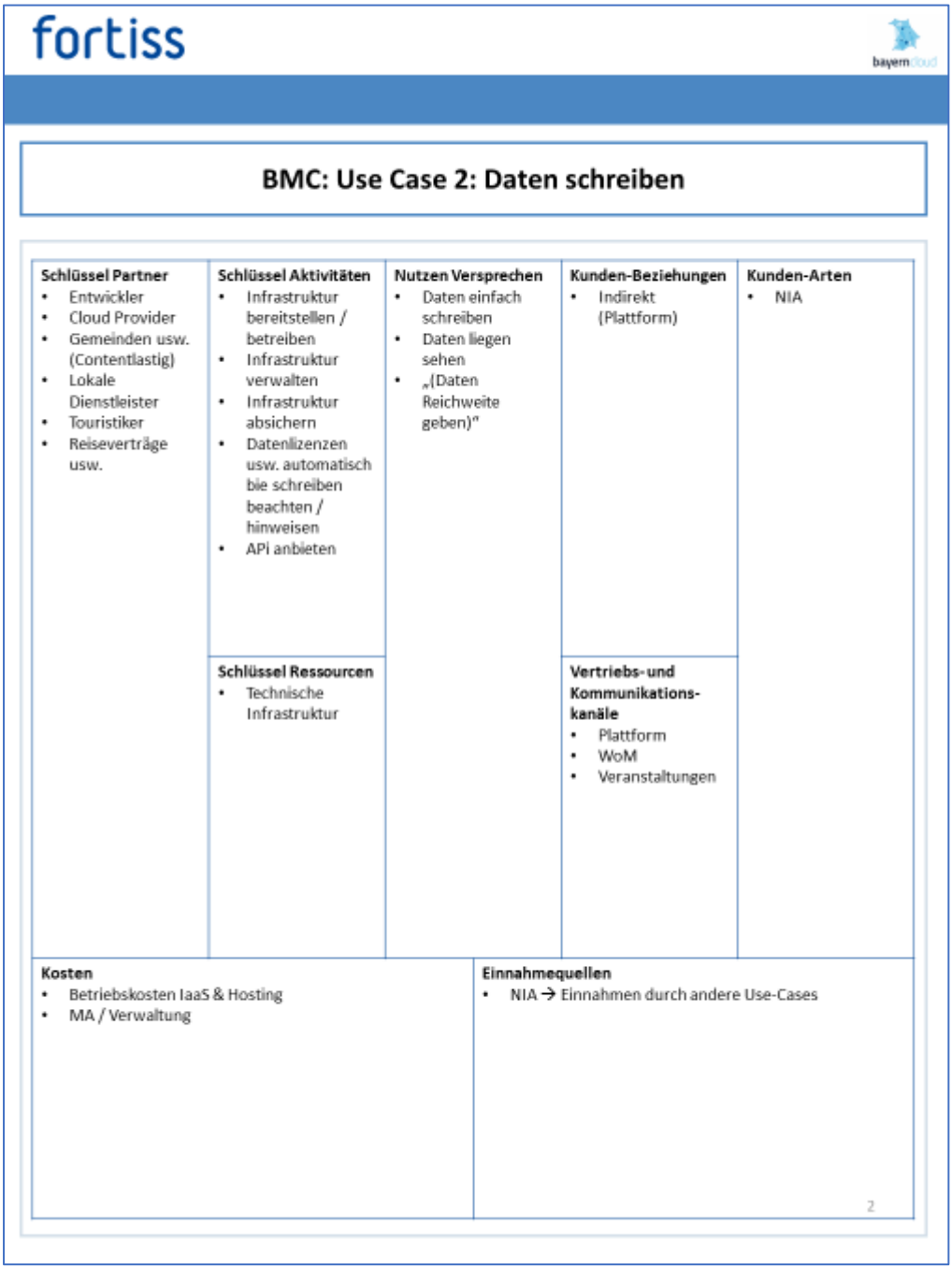

Abbildung 13: Business Model Canvas für Use Case 2

Potentielle Preismechanismen für Use Case 2:

Erlös- bzw. Preismechanik	Relevanz für Use Case
Verkaufserlös	
Abonnement-Modell / Flatrate / Subskription	
Transaktionsgebühr	(x)
Kontingente	
Preisfraktionierung bzw. Add-Ons	

Freemium	
Dynamisches Pricing	
Wertbasierte Finanzierung	
Pay-per-use / pay-as-you-go	
Pay-per-Link / Pay-per-Data	
Pay-per-improvement	
Pay-per-saving	
Hybride Modelle / Overage Modell	

Tabelle 51: Evaluation von Preismechanismen für Use Case 2

4.4.5.3 Use Case 3: Daten und Datenservice Rekombination

Use Case 3 beschreibt die Bereitstellung und Abrufung von kombinierten Daten entsprechend einer zuvor festgelegten Abfrage.

Governancemodell:

Dimension	Aspekt	Relevanz für Use Case 3
Kontroll-mechanismen	Inhaltskontrolle	
	Appropriability Mechanismen	x
	Anreize für Komplementoren	x
	Prozesskontrolle von Komplementoren	x
	Prozesskontrolle von B2B Kunden	x
	Informelle Kontrolle	
	Sanktionelle Kontrolle	x
Standards	Kompatibilität und Interoperabilität	x
	Vorgabe von Standards bezüglich Datenformat	x
Schnittstellen	API (Application Programming Interface)	x
	SDK (Software Development Kit)	x
Zugang & Rollen	Partnermodell	
	Partner Management	
Onboarding & Offboarding	On- & Offboarding Partner	
	Onboarding von Teilnehmern	
	Offboarding von Teilnehmern	
Preissetzung	Pricing Models	x
	Subsidizing	
	Revenue Sharing	x
	Payment Options	
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	
	Kontaktmöglichkeit mit Plattformbetreiber	x
	Community / Forum / Blog	
Trust	Rating System	x

Dokumentation & Guiding	Dokumentation Onboarding	
	Dokumentation Datenformat Standards	x
	API Dokumentation	x
	SDK Dokumentation	x

Tabelle 52: Ökosystemsicht: Evaluation Governancemodell für Use Case 3

Geschäftsmodell:

Wertstromanalyse (e3-Value) für Use Case 3:

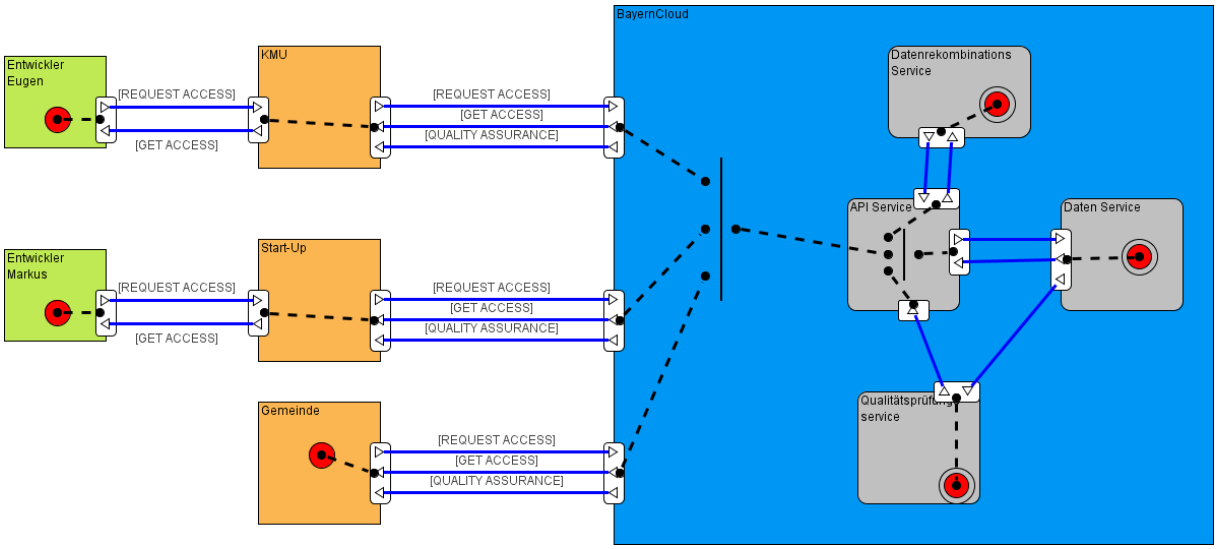

Abbildung 14: Wertstromanalyse für Use Case 3

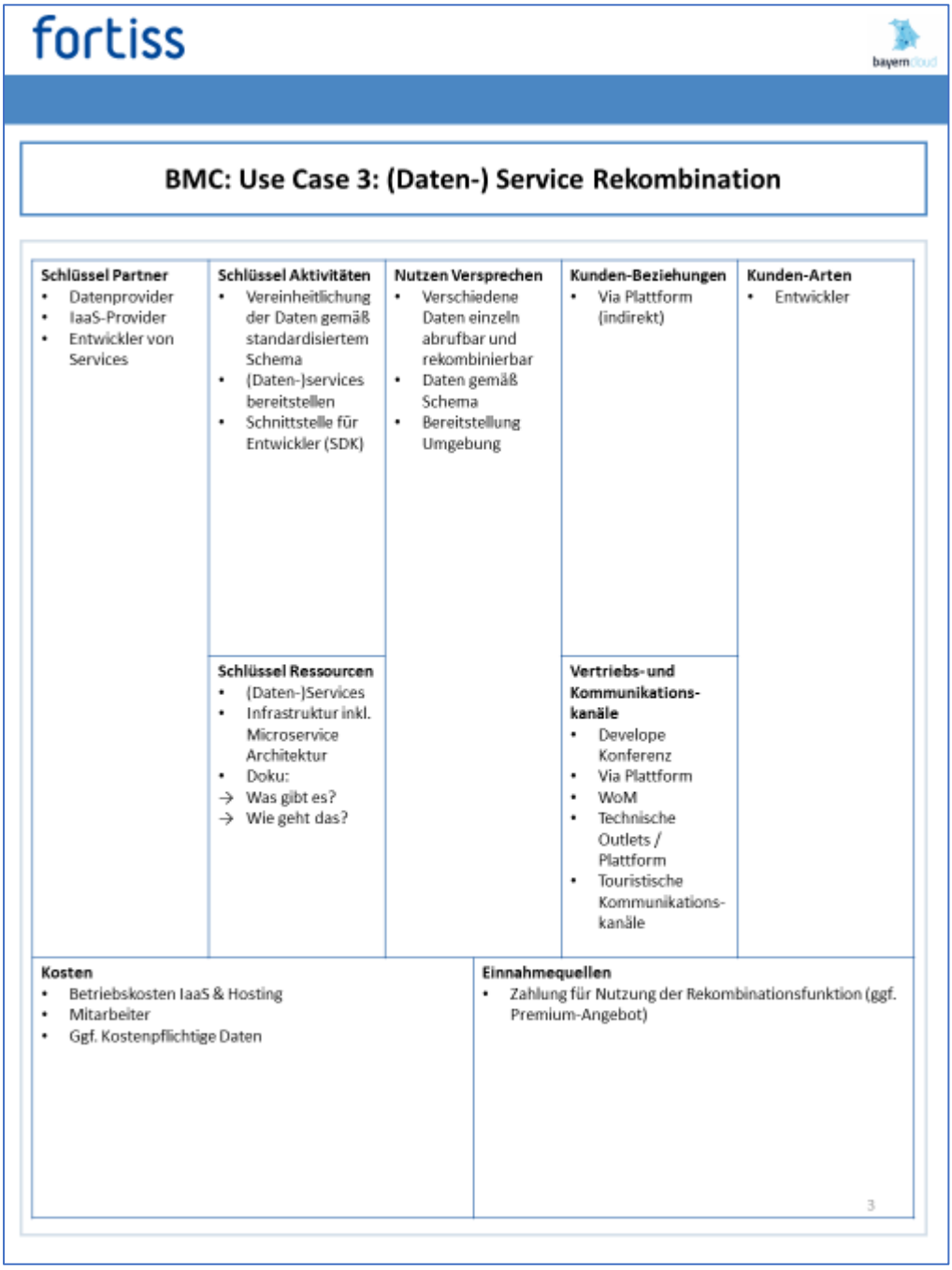

Abbildung 15: Business Model Canvas für Use Case 3

Potentielle Preismechanismen für Use Case 3:

Erlös- bzw. Preismechanik	Relevanz für Use Case
Verkaufserlös	x
Abonnement-Modell / Flatrate / Subskription	x
Transaktionsgebühr	
Kontingente	x
Preisfraktionierung bzw. Add-Ons	
Freemium	

Dynamisches Pricing	
Wertbasierte Finanzierung	
Pay-per-use / pay-as-you-go	
Pay-per-Link / Pay-per-Data	
Pay-per-improvement	
Pay-per-saving	
Hybride Modelle / Overage Modell	

Tabelle 53: Evaluation von Preismechanismen für Use Case 3

1.4.5.1 Use Case 4: Drittentwickler Service bereitstellen

Use Case 4 beschreibt die Bereitstellung von Services durch Drittentwickler.

Governancemodell:

Dimension	Aspekt	Relevanz für Use Case 4
Kontroll-mechanismen	Inhaltskontrolle	x
	Appropriability Mechanismen	x
	Anreize für Komplementoren	x
	Prozesskontrolle von Komplementoren	x
	Prozesskontrolle von B2B Kunden	x
	Informelle Kontrolle	
	Sanktionelle Kontrolle	x
Standards	Kompatibilität und Interoperabilität	x
	Vorgabe von Standards bezüglich Datenformat	x
Schnittstellen	API (Application Programming Interface)	x
	SDK (Software Development Kit)	(x)
Zugang & Rollen	Partnermodell	
	Partner Management	x
Onboarding & Offboarding	On- & Offboarding Partner	
	Onboarding von Teilnehmern	
	Offboarding von Teilnehmern	
Preissetzung	Pricing Models	x
	Subsidizing	
	Revenue Sharing	x
	Payment Options	
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	x
	Kontaktmöglichkeit mit Plattformbetreiber	x
	Community / Forum / Blog	
Trust	Rating System	x
	Dokumentation Onboarding	

Dokumentation & Guiding	Dokumentation Datenformat Standards	x
	API Dokumentation	x
	SDK Dokumentation	(x)

Abbildung 16: Ökosystemsicht: Evaluation Governancemodell für Use Case 4

Geschäftsmodell:

Wertstromanalyse (e3-Value) für Use Case 4:

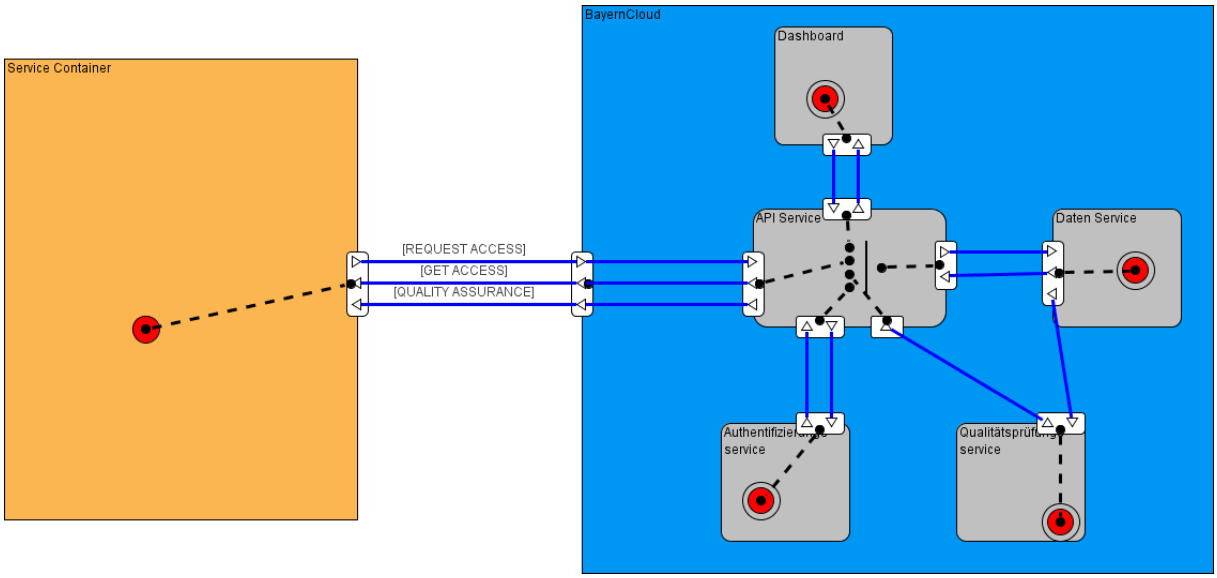

Abbildung 17: Wertstromanalyse für Use Case 4

Business Model Canvas für Use Case 4:

fortiss bayerncloud

BMC: Use Case 4: Drittentwickler Service bereitstellen

Schlüssel Partner	Schlüssel Aktivitäten	Nutzen Versprechen	Kunden-Beziehungen	Kunden-Arten
• Cloud-Provider • Bestehende 3rd Party Entwickler	• APIs anbieten • Infrastruktur verwalten • Vermarktung • Quality Gate für 3rd Party Services	• Zertifizierte und sichere Infrastruktur • Einfaches Deployment • Kostenreduktion	• Über Plattform (indirekt)	• Entwickler
	Schlüssel Ressourcen • Technische Infrastruktur • Daten • Dashboard • (Ggf. SDK)		**Vertriebs- und Kommunikations-kanäle** • Über Plattform • WoM	

Kosten	Einnahmequellen
• Betriebskosten IaaS & Hosting • MA / Verwaltungskosten • Marketing • Entwicklungskosten	• Prozentualer Anteil an Plattform von Third Party gemäß individueller SLAs • Falls per Subscription: keine Fee

4

Abbildung 18: Business Model Canvas für Use Case 4

Potentielle Preismechanismen für Use Case 4:

Erlös- bzw. Preismechanik	Relevanz für Use Case
Verkaufserlös	x
Abonnement-Modell / Flatrate / Subskription	x
Transaktionsgebühr	x
Kontingente	
Preisfraktionierung bzw. Add-Ons	
Freemium	
Dynamisches Pricing	
Wertbasierte Finanzierung	
Pay-per-use / pay-as-you-go	x
Pay-per-Link / Pay-per-Data	
Pay-per-improvement	
Pay-per-saving	
Hybride Modelle / Overage Modell	

Tabelle 54: Evaluation von Preismechanismen für Use Case 4

1.4.5.1 Use Case 5: Services verschieben

Use Case 5 beschreibt das Verschieben von Services

Governancemodell:

Dimension	Aspekt	Relevanz für Use Case 5
Kontroll-mechanismen	Inhaltskontrolle	
	Appropriability Mechanismen	
	Anreize für Komplementoren	
	Prozesskontrolle von Komplementoren	x
	Prozesskontrolle von B2B Kunden	
	Informelle Kontrolle	
	Sanktionelle Kontrolle	
Standards	Kompatibilität und Interoperabilität	
	Vorgabe von Standards bezüglich Datenformat	
Schnittstellen	API (Application Programming Interface)	x
	SDK (Software Development Kit)	
Zugang & Rollen	Partnermodell	
	Partner Management	
Onboarding & Offboarding	On- & Offboarding Partner	
	Onboarding von Teilnehmern	
	Offboarding von Teilnehmern	
Preissetzung	Pricing Models	x
	Subsidizing	

	Revenue Sharing	
	Payment Options	x
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	
	Kontaktmöglichkeit mit Plattformbetreiber	x
	Community / Forum / Blog	
Trust	Rating System	
Dokumentation & Guiding	Dokumentation Onboarding	
	Dokumentation Datenformat Standards	x
	API Dokumentation	x
	SDK Dokumentation	

Abbildung 19: Ökosystemsicht: Evaluation Governancemodell für Use Case 5

Geschäftsmodell:

Für Use Case 5 entfällt die Betrachtung von Wertströmen und des Business Model Canvas. Für den Use Case 5: Service verschieben ergeben sich die relevanten Betrachtungen der Wertströme und des Geschäftsmodells aus den anderen Use cases. Lediglich das Revenue-Modell bedarf einer gesonderten Betrachtung.

Für Use Case 5 bietet sich aus Geschäftsmodellperspektive im Bereich des Erlösmodells insbesondere an, den Service als Teil eines Subscription-Modells, beispielsweise über verschiedener Mitgliedschaftsstufen, abzubilden. So ist es möglich, dass, ggf. kostenfreie, Basisversionen dieses Service nicht ermöglichen, und dieser als sog. Freemium-Service individuell hinzugebeucht werden kann. In weiteren Stufen des Subcription-Modells, welches mehr Rechte zuspricht, könnte dieser Service hingegen inkludiert sein.

Erlös- bzw. Preismechanik	Relevanz für Use Case
Verkaufserlös	x
Abonnement-Modell / Flatrate / Subskription	x
Transaktionsgebühr	x
Kontingente	
Preisfraktionierung bzw. Add-Ons	
Freemium	x
Dynamisches Pricing	
Wertbasierte Finanzierung	
Pay-per-use / pay-as-you-go	x
Pay-per-Link / Pay-per-Data	
Pay-per-improvement	
Pay-per-saving	
Hybride Modelle / Overage Modell	

1.4.5.1 Use Case 6: Drittentwickler onboarden

Use Case 6 beschreibt das Onboarding von Drittentwicklern.

Governancemodell:

Dimension	Aspekt	Relevanz für Use Case 6
Kontroll-mechanismen	Inhaltskontrolle	x
	Appropriability Mechanismen	
	Anreize für Komplementoren	x
	Prozesskontrolle von Komplementoren	x
	Prozesskontrolle von B2B Kunden	
	Informelle Kontrolle	
	Sanktionelle Kontrolle	
Standards	Kompatibilität und Interoperabilität	
	Vorgabe von Standards bezüglich Datenformat	
Schnittstellen	API (Application Programming Interface)	
	SDK (Software Development Kit)	
Zugang & Rollen	Partnermodell	x
	Partner Management	x
Onboarding & Offboarding	On- & Offboarding Partner	x
	Onboarding von Teilnehmern	
	Offboarding von Teilnehmern	
Preissetzung	Pricing Models	
	Subsidizing	x
	Revenue Sharing	
	Payment Options	
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	
	Kontaktmöglichkeit mit Plattformbetreiber	x
	Community / Forum / Blog	
Trust	Rating System	
Dokumentation & Guiding	Dokumentation Onboarding	x
	Dokumentation Datenformat Standards	
	API Dokumentation	
	SDK Dokumentation	

Abbildung 20: Ökosystemsicht: Evaluation Governancemodell für Use Case 6

Die im Governancemodell definierten Dimensionen und Aspekte müssen nun in einen entsprechenden Onboardingprozess gegossen werden. Abbildung 17 zeigt einen standardmäßigen Onboardingprozess eines Drittentwicklers, welche Daten und/oder Services über die Plattform bereitstellen möchte.

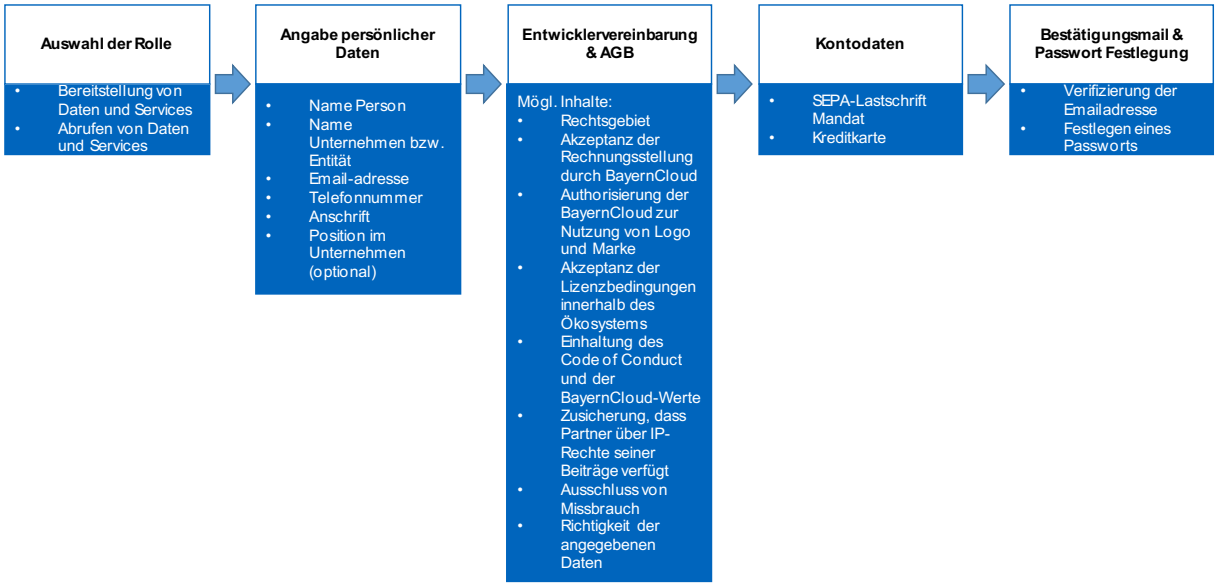

Abbildung 21: Prototypischer Onboardingprozess innerhalb eines DPÖ

<u>Geschäftsmodell:</u>

Für Use Case 6 entfällt die Betrachtung von Wertströmen und des Geschäftsmodells. Für den Use Case 6: Drittentwickler Onboaden ergeben sich keine Geschäftsmodell-relevanten Betrachtungen, die nicht innerhalb der weiteren Use Cases enthalten sind.

1.4.5.1 Use Case 7: Service anbieten und nachfragen (Service Marktplatz)

Use Case 7 beschreibt das Anbieten und die Nachfrage von Services über einen Service Marktplatz

<u>Governancemodell:</u>

Dimension	Aspekt	Relevanz für Use Case 7
Kontroll-mechanismen	Inhaltskontrolle	x
	Appropriability Mechanismen	x
	Anreize für Komplementoren	x
	Prozesskontrolle von Komplementoren	x
	Prozesskontrolle von B2B Kunden	x
	Informelle Kontrolle	x
	Sanktionelle Kontrolle	x
Standards	Kompatibilität und Interoperabilität	x
	Vorgabe von Standards bezüglich Datenformat	x
Schnittstellen	API (Application Programming Interface)	x
	SDK (Software Development Kit)	
Zugang & Rollen	Partnermodell	x
	Partner Management	x
	On- & Offboarding Partner	

Onboarding & Offboarding	Onboarding von Teilnehmern	
	Offboarding von Teilnehmern	
Preissetzung	Pricing Models	x
	Subsidizing	x
	Revenue Sharing	x
	Payment Options	x
Berichtsstufen	Kommunikationsmöglichkeiten zwischen Stakeholdern	x
	Kontaktmöglichkeit mit Plattformbetreiber	x
	Community / Forum / Blog	
Trust	Rating System	x
Dokumentation & Guiding	Dokumentation Onboarding	
	Dokumentation Datenformat Standards	x
	API Dokumentation	x
	SDK Dokumentation	

Abbildung 22: Ökosystemsicht: Evaluation Governancemodell für Use Case 7

Geschäftsmodell:

Wertstromanalyse (e3-Value) für Use Case 7:

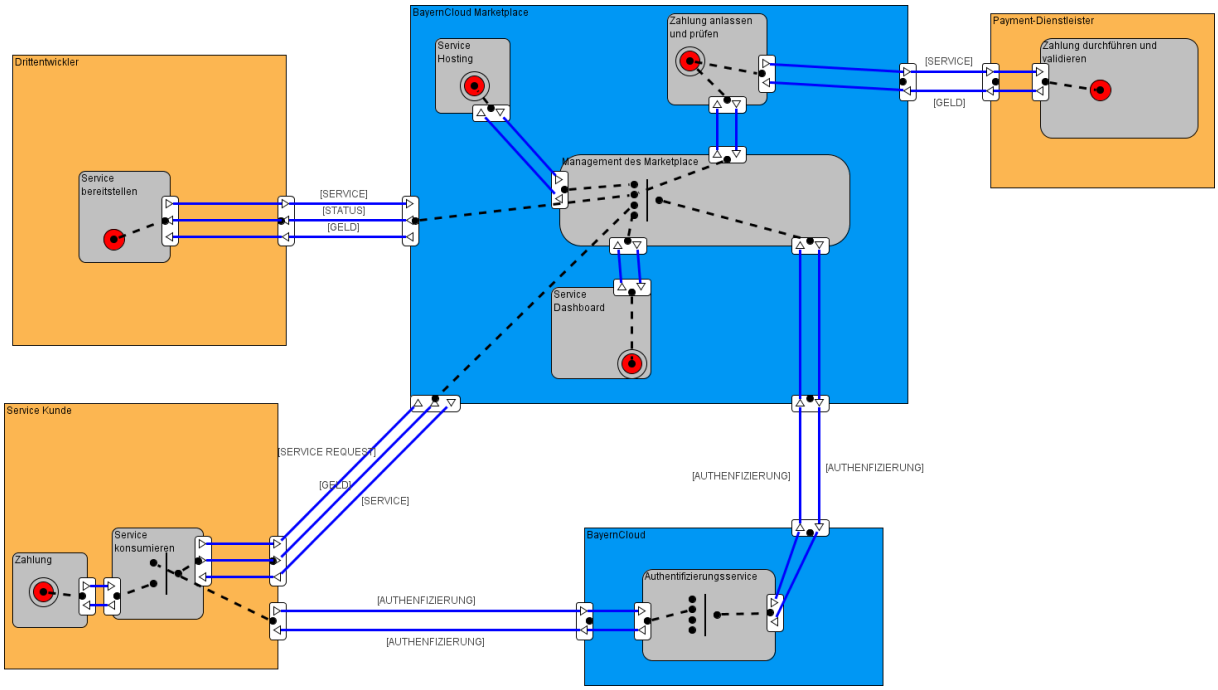

Abbildung 23: Wertstromanalyse für Use Case 7

Business Model Canvas für Use Case 7:

fortiss bayerncloud

BMC: Use Case 7: Services anbieten und nachfragen (Service Marktplatz)

Schlüssel Partner	Schlüssel Aktivitäten	Nutzen Versprechen	Kunden-Beziehungen	Kunden-Arten
• Datenprovider (vor allem offene Daten) • Cloud-Provider • Spezifische Datenprovider (z.B. Wetter, lokaler Tourismus-verband) • Weitere 3rd Party Entwickler • Payment Dienstleister	• APIs anbieten • Data Cleaning • Infrastruktur verwalten • Qualitäts-sicherung der Daten → bessere Daten • Aktivitäten von Lizenzbedingungen; Datenhoheit usw. zu beachten • Vermarktung • Lizenzierung • Support	• Aktuelle Daten verfügbar • Daten in einheitlichem Schema vorhanden • Einfache Zugänglichkeit der Daten • Zertifizierte und sichere Infrastruktur • Marktzugang • Hohe Verfügbarkeit • Unabhängiger Marktplatz	• Über Plattform (indirekt)	• Entwickler • Beamte und Co. • Contentlastige Dienstleister • Verbände • Gemeinden, usw. (Verwaltung)
	Schlüssel Ressourcen • Technische Infrastruktur • Daten • Self Service • Marketplace • Technisches und inhaltliches Informationsmaterial (z.B. Lizenzen)		**Vertriebs- und Kommunikations-kanäle** • Über Plattform • WoM • Eingängige touristische Vermarktungs-wege (Zeitschriften, Veranstaltungen, usw.)	

Kosten	Einnahmequellen
• Betriebskosten IaaS & Hosting • MA / Verwaltungskosten • Marketing • Entwicklungskosten • Revenue Store oder Fee an Payment Dienstleister • Support	• Revenue Sharee gemäß Erlösmodell des individuellen Service • Evtl. als Paket in Membership Fee (einmalige Einnahme aus Membership Fee)

5

Abbildung 24: Business Model Canvas für Use Case 7

Potentielle Preismechanismen für Use Case 7:

Erlös- bzw. Preismechanik	Relevanz für Use Case
Verkaufserlös	X
Abonnement-Modell / Flatrate / Subskription	X
Transaktionsgebühr	X
Kontingente	
Preisfraktionierung bzw. Add-Ons	X
Freemium	X
Dynamisches Pricing	X
Wertbasierte Finanzierung	
Pay-per-use / pay-as-you-go	X
Pay-per-Link / Pay-per-Data	X
Pay-per-improvement	
Pay-per-saving	
Hybride Modelle / Overage Modell	X

Tabelle 55: Evaluation von Preismechanismen für Use Case 7

4.4.5. Zusammenfassung der Ökosystemsicht

Innerhalb der Ökosystemsicht der BayernCloud Referenzarchitektur wurden Governancemechanismen und Geschäftsmodelle je Use Case entwickelt. Zu diesem Zweck wurde jeweils ein Vorgehensmodell vorgestellt, das durch Anwender der Referenzarchitektur genutzt werden kann. Für das zu instanziierende Plattform-Ökosystem auf Basis der Referenzarchitektur ergeben sich für die Ökosystemperspektive übergeordnete Erkenntnisse für das Gesamt-Governancemodell sowie das Gesamt-Geschäftsmodell.

Übergeordnet von spezifischen Use-Cases lassen sich die Informations-und Datenflüsse wie in Abbildung 25 zusammenfassen. Der Konsument von Daten ist in der Regel ein touristischer Dienstleister oder ein Tourist selbst. Somit geht der Informationsfluss in die Richtung der Konsumenten, wobei die Werflüsse heterogen, also in verschieden Richtungen ablaufen. Abhängig der gewählten Erlösmodelle, Governancestrukturen sowie der vorgeschlagene Mitgliedschafts-Struktur zur Nutzung der Dienste ergeben sich an verschiedenen Interaktionspunkten mögliche Erlösquellen.

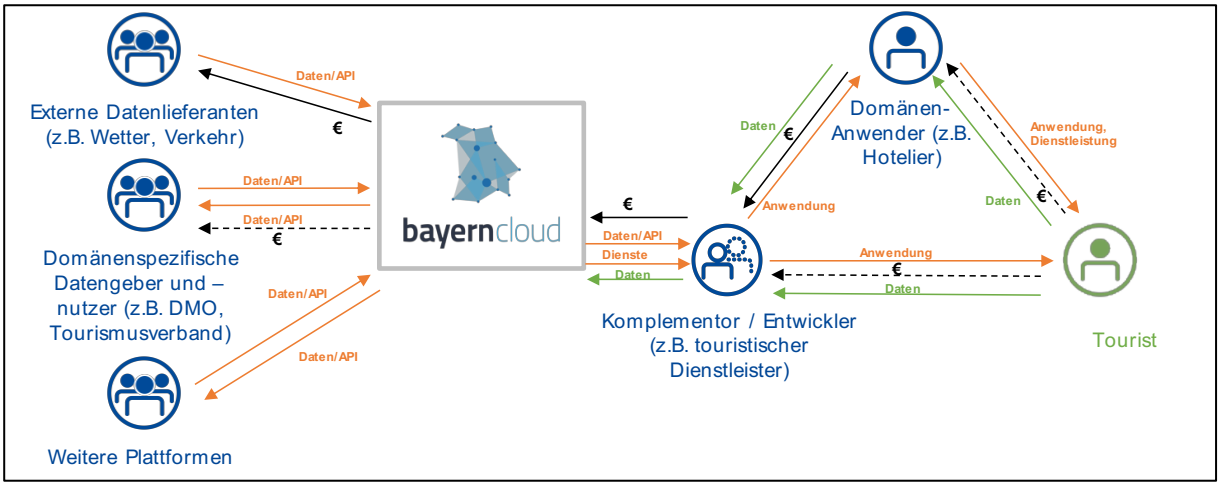

Abbildung 25: Mögliche Wertströme im BayernCloud Ökosystem

Das zentrale Wertversprechen setzt für alle beteiligten Akteure voraus, dass eine durchgängige und qualitativ hochwertige Datenbasis erreicht werden kann. Somit sind zur Skalierung des Geschäftsmodells Partnerschaften mit starkem Domänenbezug notwendig. Die Aspekte und deren Dimensionen zur Steuerung dieser Partner beziehungsweise Komplementoren wird durch das Governancemodell definiert. Es müssen genug Stakeholder angezogen werden, um die entsprechenden Datenbasis zu erreichen und damit das Fundament für eine erfolgreiche Skalierung zu legen. Hierzu liefert das Governancemodell bereits eine Vielzahl von Faktoren, um die Teilnahme Dritter an DPÖs zu stärken. Außerdem sind Partnerschaften wichtig, um die notwendigen Vermarktungswege gezielt zu adressieren. Die Qualität von Beiträgen zu digitalen Ökosystemen ist entscheidend für deren Erfolg und somit Erfolgsfaktor für Plattformgeschäftsmodelle. Die Sicherstellung der Qualität erfolgt primär über das Governancemodell, welches Standards und Kompatibilitäten vorschreibt und Fragen des geistigen Eigentums der Beiträge klärt. Standards und Schnittstellen sind nach neuestem Stand in maximalem Umfang zu dokumentieren und zugänglich zu machen. Daten können beispielsweise über Creative Commons (CC) Lizenzen stufenweise in ihrer Nutzung gesteuert werden. Partner können über Subventionen unterstützt werden, ihre Daten an vorgegebene Standards anzupassen und sie zu motivieren, ihre Daten als Open-Data freizugeben. Zudem kann die Qualität der Beiträge über die Steuerung des grundsätzlichen Zugangs zum Ökosystem sichergestellt werden, während ein vordefiniertes Partnermodell notwendig ist, um die Partner der Plattform zu strukturieren und zielgerichtet steuern zu können. Ein typisches Partnermodell in Verbindung mit einem Partner Management sieht unterschiedliche Voraussetzungen und Vorteile für verschiedene Partner vor. Ein Partner Management wird ergänzt durch Kontrollmechanismen, die positive wie negative Interventionen durch den Plattformbetreiber umfassen müssen.

Der Plattformbetreiber muss geeignete Kommunikationskanäle bereitstellen und deren Nutzung durch die Teilnehmer fördern, um eine Community um die Plattform herum aufzubauen. Insbesondere für die Domäne Tourismus ist Word-of-Mouth ein zu berücksichtigender Kommunikationsweg in der Kundenkommunikation. Daneben ist eine verlässliche, ausfallsichere Infrastruktur notwendig, die idealerweise eine unabhängige Zertifizierung aufweist, um Vertrauen bei den Nutzern zu schaffen. Im Bereich der Erlösmodelle bieten sich insbesondere Kontingente an, die zu unterschiedlicher Nutzung der Infrastruktur und entsprechender Services berechtigen. Individuelle Services können per

direktem-Verkauf oder als Abonnement bezogen werden. Wichtig ist hierbei, dass die Informations- und Wertströme über das instanziierte Plattform-Ökosystem abgebildet werden und nicht an der Plattform vorbei. Der Zugang zu den Daten sollte, zumindest für einen Teil der Daten, im Sinne von Open-Data frei zugänglich sein. Dies ist notwendig um die Skalierung zu erreichen und Nutzern das Testen vorab zu erlauben. Wichtig zu beachten ist, dass auch für eine kostenfreie Nutzung zwanghaft eine Registrierung notwendig ist, um für die Nutzung der jeweiligen Datensätze eine Autorisierung vorzunehmen und ein entsprechendes Monitoring zu ermöglichen.

Auf Basis der Erkenntnisse bezüglich Governancemodell und Geschäftsmodell wird empfohlen, als Gesamtmodell ein mehrstufiges Mitgliedschaftsmodell zu erstellen. Je nach Mitgliedsstufe wird ein regelmäßiger Betrag fällig (bspw. als Mitgliedsbeitrag), der zu bestimmten Kontingenten bspw. auf das Abrufen von Datensätzen, der Nutzung der Infrastruktur oder auch dem Konsum spezifischer Services berechtigt. Bestimmte Dienste und Services sind nur für „höhere" Mitgliedsstufen verfügbar, die entsprechend mehr Berechtigungen enthalten und einen höheren Mitgliedsbeitrag erfordern. Gleichzeitig verstärken sich Mitsprache- und Entscheidungsrechte bei den höheren Mitgliedsstufen entlang der vorgegebenen Strukturen des Partner Managements des Governancemodells. Individuelle Services oder Nutzung von Diensten und Daten über die definierten Kontingente hinaus kann über individuelle Abrechnung erfolgen (pay-per-use).

4.5. Verteilungssicht

Im Folgenden wird die Verteilungssicht der BayernCloud dargestellt. In der Einleitung wird erläutert an wen sich die Sicht richtet und was die Ziele sind. Danach werden Aufgaben und Anforderungen im Kontext der BayernCloud analysiert. Mit Bezug auf moderne Microservicearchitekturen wird insbesondere auf die Rolle von Orchestrierungssystemen dabei eingegangen. Es erfolgt eine Beschreibung der Software Deployments auf generischen Hardwarearchitekturen und Multicloud IaaS Providern unter Berücksichtigung von Storage und Kommunikationsmustern. Abschließend werden mögliche Softwareverteilungen für die einzelnen Use Cases mittels Verteilungsdiagrammen aufgezeigt.

4.5.1. Einleitung

Die Verteilungssicht der RA modelliert die Abbildung von Softwarekomponenten auf Hardware Ressourcen, entsprechend des 4+1 Sichten Modells nach (Kruchten 1995). Neben der Aufteilung einzelner Module werden dabei auch Kommunikationsmuster und Schnittstellen zwischen diesen berücksichtigt. Sie adressiert die Sichtweise von Administratoren und Plattformbetreibern innerhalb des Ökosystems und wird auch als Deployment oder Physische Sicht bezeichnet, da Sie das Bindeglied zwischen Implementierungslogik und physischer Infrastruktur darstellt. Bei der Verteilung der Komponenten, der Auswahl möglicher Verteilungsmechanismen und deren zugrundeliegenden Ressourcen, müssen verschiedene funktionale und nicht funktionale Eigenschaften berücksichtigt wie etwa Skalierbarkeit, Flexibilität, Performanz und Ausfallsicherzeit.

Dies kann durch passende Technologien und Architekturdesign Entscheidungen beim Plattformentwurf gewährleistet werden. Ein wesentlicher Fokus liegt dabei auf der Abstraktion von generischen Server Ressourcen und Möglichkeiten zur effizienten Virtualisierung. Traditionell werden dafür virtuelle Maschinen eingesetzt mit Hypervisorn die es erlauben Software von der physischen Hardware zu entkoppeln. Sie erhöhen die Flexibilität bei der Verteilung von Services oder Service Komponenten auf der Infrastruktur. Darüber hinaus kommen Zuge moderner Softwareentwicklung verstärkt Container basierte Micro Services zum Einsatz, welche die Agilität beim Entwickeln und Verteilen von Komponenten weiter steigern. Daraus folgen weitere Implikationen für die Kommunikation zwischen den Service Einheiten, Skalierung einzelner Komponenten, Datenhaltung sowie für die Lastverteilung und Namensauflösungen. Im Folgenden werden die zentralen Anforderungen an die Verteilung beschrieben, sowie die anhand gestaltungsorientierten Vorgehens entwickelte Verteilung von Softwarekomponenten der Referenz Architektur.

4.5.2. Anforderungen an die Verteilung innerhalb der BayernCloud

Der Architekturentwurf der BayernCloud berücksichtigt die gerade genannten, weitestgehend generischen Anforderungen an Agilität, Verfügbarkeit, Skalierbarkeit. Allerdings müssen für die Zielsetzung des Projekts weitere Kriterien adressiert werden. Wie im bereits im Forschungsantrag gefordert, wird eine überwiegend herstellerunabhängige Plattform konzeptioniert. Ziel es ist, nachhaltig mittelständische Unternehmen bei Digitalisierungsvorhaben und der branchenspezifischen Ökosystembildung zu unterstützen. Dabei wird eine Lösung favorisiert welche es ermöglicht auf dem aktuell stark

wachsenden Markt von IaaS und PaaS Cloud Providern aufzubauen, ohne dabei Hersteller spezifische Lock in Effekte zu erzeugen. Unabhängig davon benötigen die unterschiedlichen Branchen domänenspezifische Services welche von verschiedenen Plattformnutzern zur Verfügung gestellt werden sollen können. Entsprechend folgen hieraus zwei weitere konkretisierte Anforderungen für die Verteilung:

Mutlicloud

Im Zuge des Projekts wurde die Kompatibilität der Verteilung von allen Softwarekomponenten auf verschiedene IaaS-Anbieter überprüft. Damit soll die potentielle Unabhängigkeit von Infrastrukturprovidern gegeben sein. Hierbei liegt der Fokus insbesondere auf der Bewertung von Tools und Möglichkeiten für eine providerübergreifende Umsetzung (Multicloud). Auf diese Weisen können neben der Reduktion von möglichen Lock ins (Oliveira de Carvalho, Trinta, & Vieira) theoretisch auch weitere Vorteile wie eine finanzielle Optimierung von standardisierbaren Serviceangeboten wie Compute Ressourcen erreicht werden. Auch die flexible Integration und Kombination von unterschiedlichen Datenlokationen innerhalb der BayernCloud - ob Server oder Cloud basiert – bleibt dadurch erhalten.

Multitenancy

Die BayernCloud berücksichtigt ein domänenübergreifendes Plattformkonzept mit mehreren voneinander weitestgehend unabhängigen Nutzern. Es werden branchenspezifische Anforderungen in Form unterschiedlicher Daten und Mehrwertservices umgesetzt, welche nicht nur vom Plattformbetreiber, sondern auch von den Nutzern zur Verfügung gestellt werden können sollen. Die Nutzer, beziehungsweise deren domänenspezifische Services müssen logisch getrennt auf Hardware abgebildet werden können. Zwar ist es ein wichtiges Ziel hier flexible Erweiterungen und Rekombinationsmöglichkeiten zu gewährleisten, allerdings wird dies primär durch moderne Kommunikationsmuster und semantische Annotation auf Serviceebene erreicht. Wichtiger ist zunächst für einen zuverlässigen Plattformbetrieb die Segmentierung und Abschottung der Ressourcen und Daten unterschiedlicher Domänen und Basisservices. Als Basisservices werden Plattformweite Funktionalitäten bezeichnet die domänenübergreifend benötigt werden und den grundlegen Plattform betrieb gewährleisten. Die unterschiedlichen Domänen oder Services können als einzelne Plattform „Tenants" betrachtet werden. Die daraus resultierende Anforderung einer Multitenancy kann je nach Compliance Bedingungen eine Vielzahl von Anforderungen zur Folge haben. Für die Verteilung wird dabei darauf geachtet, dass keine Konflikte zwischen den Tenants oder deren Services entstehen und durch die gewählten Tools und Architekturmuster die Integrität der jeweiligen Daten erhalten bleibt.

Resultierend aus den ermittelten Anforderungen werden im Folgenden die Entscheidungen für die Verteilungssicht beschrieben, deren Zusammenhänge erläutert und mittels UML Verteilungsdiagramme veranschaulicht.

4.5.3. Infrastruktur, Virtualisierung und Container

Moderne Softwareentwicklung ist geprägt von dem Wunsch nach Agilität und Flexibilität. Viele der zuvor ermittelten Anforderungen werden davon beeinflusst. Dieser Trend hat auch die zunehmende Adaption von Microservices als das dominante Muster zur Entwicklung verteilter, skalierbarer Services vorangetrieben (Wilhelm Hasselbring and Guido Steinacker, 2017). In der Praxis werden Microservices meist durch Containerbasierte Lösungen umgesetzt und nur indirekt auf Basis von Servern oder Virtuellen Maschinen. Im Rahmen der BayernCloud wird daher primär auf Container gesetzt um Services jeglicher Art zu implementieren. Interessant ist auch die Definition eines Microservices. Idealerweise sollte die Kapselung von Funktionalitäten eines Services möglichst feingranular sein, Datenbankanbindung, Datenauswertung sowie Schnittstellen sollten für jeden Domänenservice oder ggf. für weitere Unterklassifizierungen als getrennte Microservices umgesetzt sein. Daraus resultieren Auswirkung auf die Verteilung der Softwaremodule.

Als erstes erfolgt die Klärung der Softwarebereitstellung und Verteilung in Container basierten Umgebungen. Es können mehrere Microservices in Form von Containern auf einem physikalischen oder virtualisierten Server bereitgestellt werden. Im Folgenden werden die zwei Optionen (virtuell/physisch) weitestgehend äquivalent betrachtet und als Nodes in der Architektur bezeichnet. Container selbst sind zustandslose Instanziierungen von Containerimages, welche die eigentliche Quelle der Service oder Applikationslogik sowie deren Abhängigkeiten darstellen (Jaramillo, Nguyen, & Smart, 2016). Images werden mittels zentraler Registries zur Verfügung gestellt und ermöglichen eine dynamische automatisierte Verteilung. Abbildung 26 veranschaulicht den Zusammenhang zwischen Registries und Containern. Wie hier ersichtlich, ist es möglich verschiedenste Registries als Quelle zu spezifizieren. So wird dediziert entwickelter BayernCloud Code vorwiegend durch ein privates Registry abgerufen welches auf eigenen Nodes implementiert wird. Generischer Code wie etwa für die Nutzung einer generischen Datenbank kann über öffentliche Repositories, z.B. durch DockerHub bezogen werden.

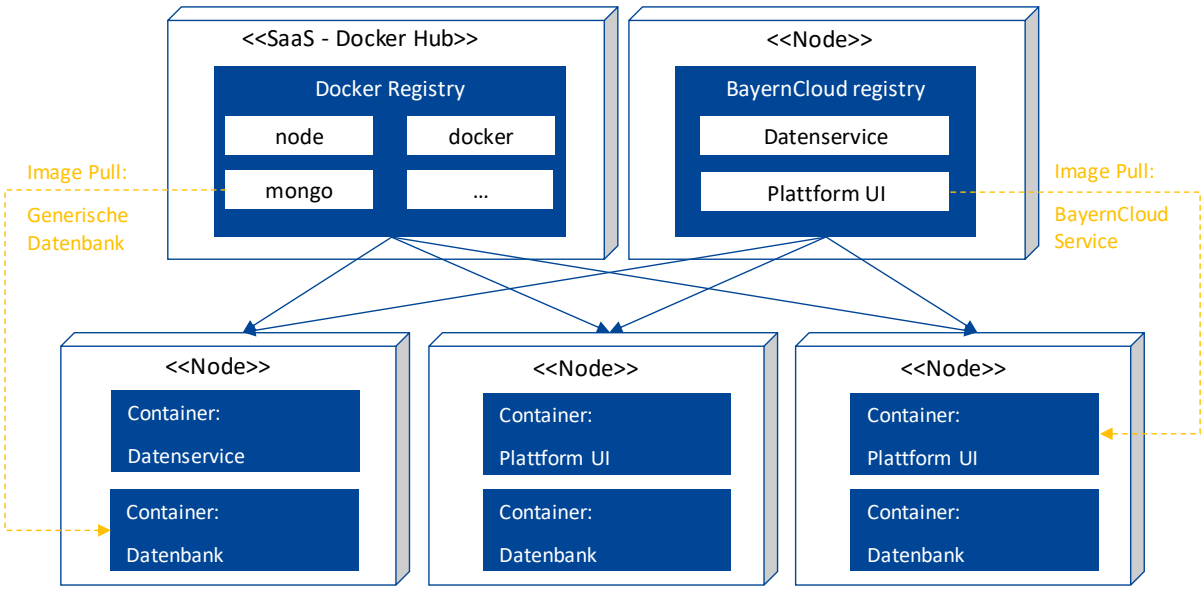

Abbildung 26: Verteilung von Containerimages durch zentrale Registries auf unterschiedlichen Umgebungen

4.5.4. Orchestrierung und Management

Microservices ermöglichen Agilität und helfen bei der Erfüllung zahlreicher weiterer Anforderungen. Doch aus der feineren Granularität resultieren auch Herausforderungen. Container müssen orchestriert und gemanaged werden. Eine händische Prüfung von Verfügbarkeit und Erreichbarkeit der Komponenten ist herausfordernd. Die Lösung dafür sind geeignete Orchestrierungs- und Management-Systeme. Sie übernehmen verschiedene Aufgaben und gewährleisten einen fehlerarmen und stabilen Einsatz von hochskalierbaren Containerumgebungen. Für den Architekturentwurf wird hier im Weiteren auf eine spezifische Orchestrierungslösung eingegangen, das Open Source System Kubernetes.

Kubernetes hat in jüngster Vergangenheit starke Auswirkung auf die flächige Adaption von Container basierten Softwareplattformen genommen und ist in starker Wechselwirkung mit der kontinuierlichen Entwicklung Feldes. Das System ist quelloffen und unter einer Apache Lizenz nutzbar und damit weitestgehend unabhängig von einzelnen Plattformen, was den genannten Multicloud Anforderungen genügt. Die Implementierung des Microservice Architekturen erfolgt damit nach wie vor so generisch wie möglich. Kubernetes besteht aus einer Reihe von Komponenten die auf beliebigen Nodes zur Verfügung gestellt werden können. Für gewöhnlich werden mehrere Nodes in einem Cluster verwaltet und ein oder mehrere Cluster Master Nodes definiert. Dafür werden automatisch auch Clusterspezifische Services deployed (kublet, kubectl, ...). Abbildung 27 zeigt eine mögliche zur Verteilung auf verschiedenen IaaS Providern.

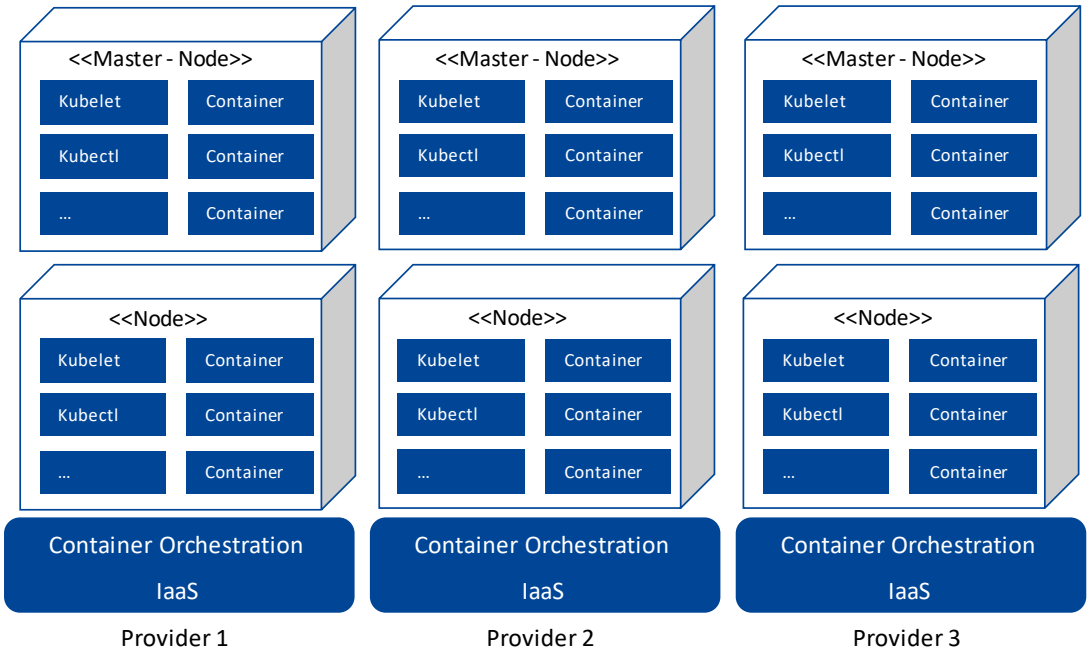

Abbildung 27: Mögliche Verteilung von Kubernetes Nodes und Container auf verschiedene IaaS Provider

Darüber hinaus werden eine Reihe weitere Service Abstraktionslevel definiert um die Anfangs genannten Probleme der Namensauflösung, Skalierung und serviceinternen Lastverteilung zu lösen. Die Implementierung dieser erfolgt durch die deklarative Beschreibung von Verhalten und Verteilungsinformationen in Konfigurationsdateien.

4.5.5. Verteilung von Services

Die Verteilung von Plattform und Service Komponenten erfolgt innerhalb von Kubernetes. Neben der Applikationslogik wird eine Reihe der genannten Konfigurationsdateien benötigt. Diese ermöglichen weitestgehend automatische Deployments und dynamische Zuordnung von Hardwareressourcen in Form von Nodes. Kubernetes definiert eine zusätzliche Abstraktionsebene zwischen Container und Nodes, sogenannte Pods (kubernetes, 2020). Pods erlauben die zusammenhängende Verteilung von funktional sinnvoll koppelbaren Containern. Sie werden meist genutzt um ein Sidecar Muster zu Implementieren. Dadurch können Funktionalitäten von der Applikationslogik abstrahiert werden, wie etwa Applikationsverschlüsselung oder Identitätsbasierte Routing Logik.

Es werden Konfigurationen für Kubernetes Deployments und Services definiert, pro BayernCloud übergreifenden Basisservices aber auch für domänenspezifische Datenservices. Des Weiteren werden Konfigurationsdateien benutzt um sicher Passwörter über definierte Aliases aus dem Cluster internen Key Value Store zu beziehen. Wie im vorherigen Kapitel beschrieben sind Container zustandslos. Daher spezifizieren wir ebenfalls mittels Konfigurationsdateien eine explizite Einbindung externe Datenlokationen. Im Sinne der Multicloud und Multitenancy Anforderungen können hier multiple, an unterschiedlichen und weitestgehend beliebigen Orten befindliche Lösungen genutzt werden. Eine Einbindung von Cloud basierten Storage Diensten ist möglich ebenso wie plattformunabhängige File Server und Storage Netzwerke. Abbildung 28 skizziert den Zusammenhang zwischen Nodes, Pods, Deployments, Services und Storage innerhalb des BayernCloud Plattformkonzepts.

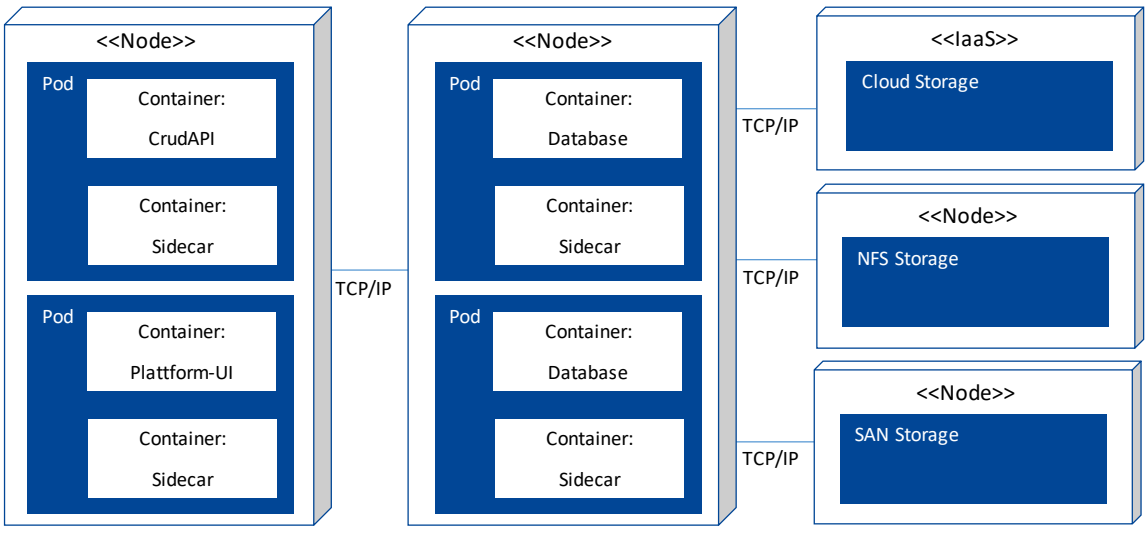

Abbildung 28: Aufteilung von Plattform und Domänenservices auf unterschiedliche Infrastrukturumgebungen

4.5.6. Kommunikation

Zur Dokumentation der Verteilungssicht gehört ebenfalls die Beschreibung der Kommunikation zwischen einzelnen Komponenten im System (Kontio, 2005) wie in der Einleitung erwähnt. Im letzten Kapitel wurde implizit aufgezeigt, dass auch eine Container

bzw. Kubernetes basierte Microservice Implementierung auf normalem TCP/IP basiertem Netzwerkverkehr zwischen Servern basiert. Es werden lediglich Netzwerk Overlays genutzt um Kommunikation zwischen Services und Container entsprechend der Konfiguration einzuschränken. Auf dem traditionellen Netzwerkstack aufbauend werden jedoch unterschiedliche Kommunikationsmuster auf Applikationsebene verwendet. Wir unterscheiden in:

a) synchrone Kommunikation – mittels REST APIs.

b) asynchrone Kommunikation – mittels Message Bus oder Publish/Subscribe Konzepten.

Abbildung 29 veranschaulicht die unterschiedlichen Kommunikationsschnittstellen zwischen Servicekomponenten welche als Microservices auf unterschiedlichen Nodes implementiert wurden.

Konzeptionell erfolgt die Verbindung von außerhalb der Plattform zu den verschiedenen Service Schnittstellen, wie etwa REST APIs, häufig über eine oder wenige Plattformweite zentralisierte Entitäten. Dabei kann es sich um zentrale Loadbalancer, Proxy Server oder ein API-Gateway handeln. Diese können ebenfalls in Form von Containern innerhalb eines Kubernetes Clusters zur Verfügung gestellt werden. Kubernetes stellt in diesem Kontext ein eigenes Service Konzept namens Ingress Controller zur Verfügung. Ingress Controller sind generisch, flexibel und können durch bekannte Proxy Server Implementierung wie beispielsweise Nginx oder ähnliche realisiert werden. Die Konfiguration von Ingress Controllern erfolgt durch Ingress Ressourcen die automatisiert mit der Verteilung der restlichen Konfiguration im Deployment Prozess eingebunden werden können.

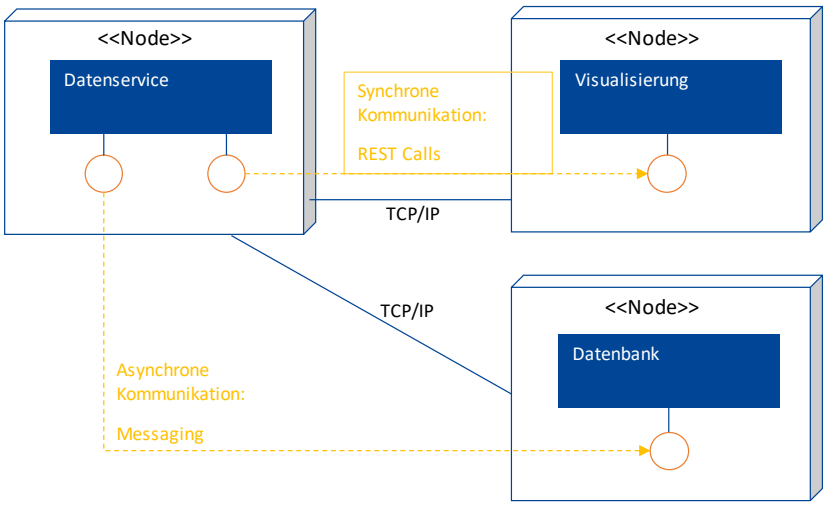

Abbildung 29: Kommunikation mittels REST APIs und asynchroner pub/sub Methoden in Microservice Deployments

4.5.7. Bezug zu Use Cases im Kontext der Architektur

Use Case 1: Daten lesen - Das Auslesen von Daten benötigt entsprechend Datenservices. Im Folgenden wird exemplarisch ein domänenspezifischer Datenservice betrachtet welcher den in diesem Kapitel beschriebenen Microservice Konzepte folgend implementiert wurde.

Die verschiedenen Teilkomponenten des Services können auf beliebiger Infrastruktur instanziiert werden. Bei der Verteilung muss die korrekte Kommunikation zwischen den Komponenten gewährleistet sein und eine Verfügbarkeit der Datenschnittstelle in Form einer API nach außen hin ermöglicht werden. Dabei wird hier als Orchestrierungssystem eine vorhandene Verteilung von Kubernetes auf einer IaaS Plattform oder dedizierten Hardwareumgebung angenommen.

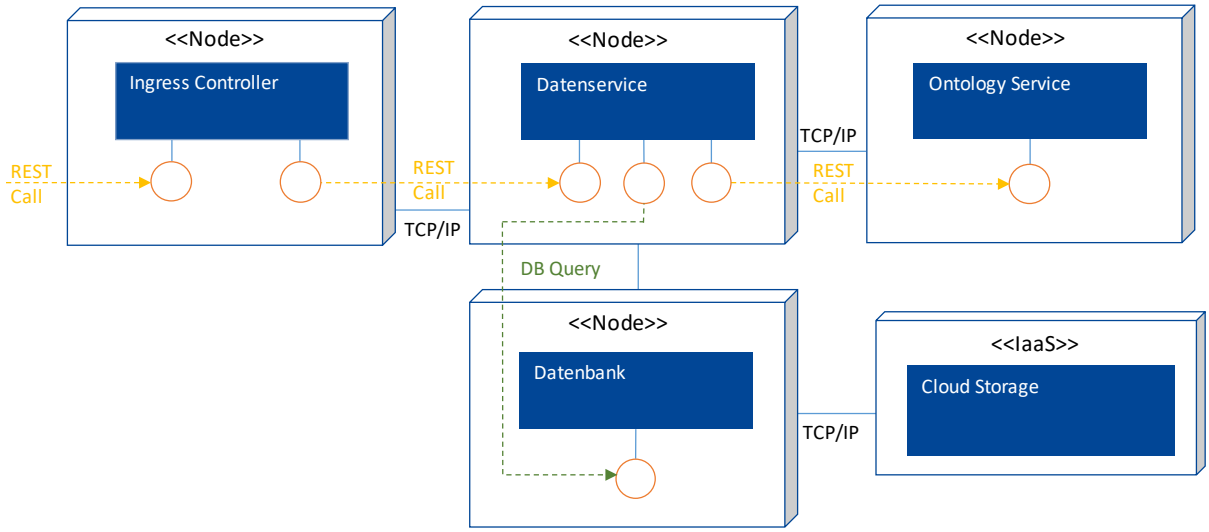

Abbildung 30: Verteilung eines Datenservices für lesende Aktionen von Daten durch REST APIs in Kubernetes basierten Microservice Deployments

Use Case 2: Daten schreiben - Das Schreiben von Daten benötigt ebenfalls entsprechend Datenservices. Verteilung und Kommunikation erfolgt analog zu Use Case 1.

Use Case 3: Daten und Datenservice Rekombination - Das Kombinieren erfordert neben den zu Grunde liegenden Datenservices eine Kompositionsmöglichkeit. Diese wird ebenfalls über einen Service bereitgestellt. Im Folgenden werden exemplarisch alle dafür benötigten Komponenten aufgezeigt. Wie zuvor, können die verschiedenen Teilkomponenten des Services auf beliebiger Infrastruktur instanziiert werden. Als Orchestrierungssystem wird wieder eine vorhandene Verteilung von Kubernetes auf einer IaaS Plattform oder dedizierten Hardwareumgebung angenommen.

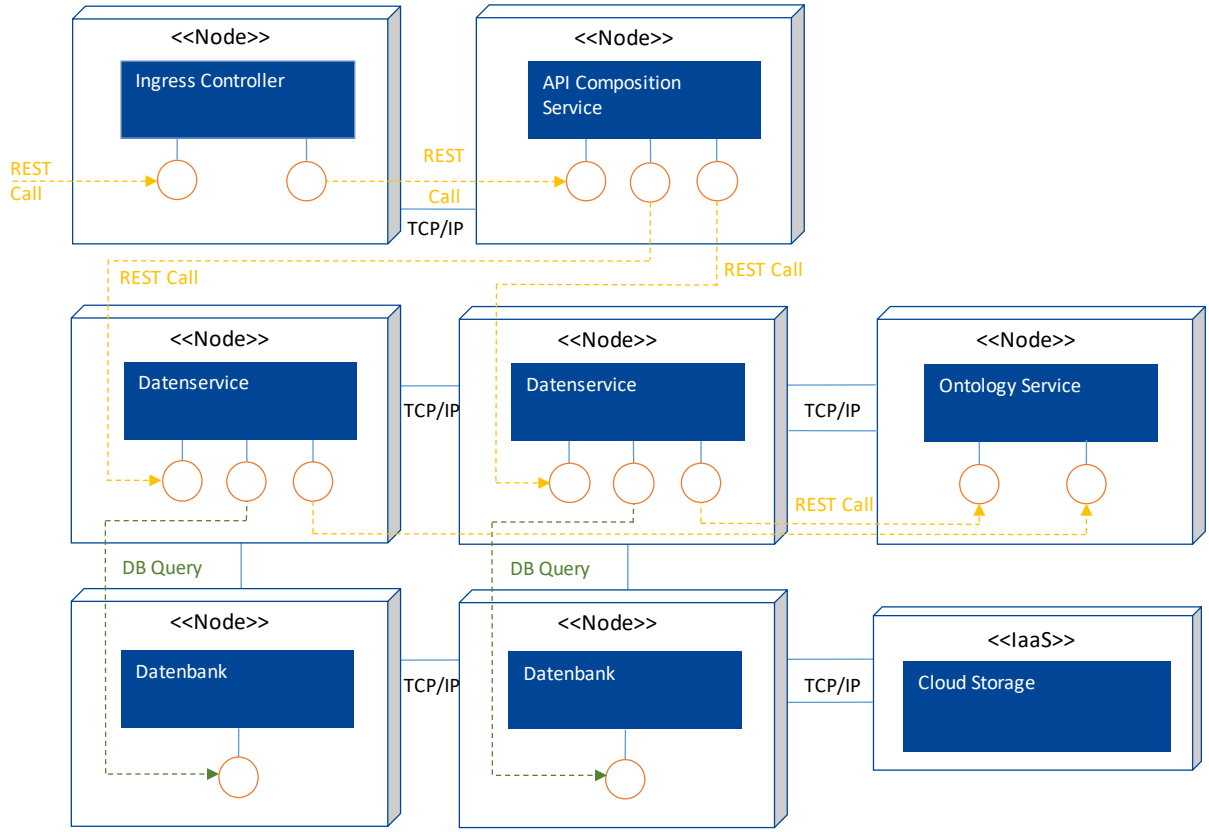

Abbildung 31: Verteilung eines Datenservices für Datenservice Kombination von Daten durch REST APIs in Kubernetes basierten Microservice Deployments

Use Case 4: 3rd Party Service bereitstellen – Die Einbindung externer Services kann auf verschiedene Arten umgesetzt werden. Hier wird exemplarisch der Fall betrachtet, in dem eine dritte Partei diesen direkt auf der Plattform integrieren will. Der Service wird dadurch wie interne Services über das Orchestrierungssystem auf beliebigen Nodes abgebildet und in die Ingress Konfiguration eingebunden.

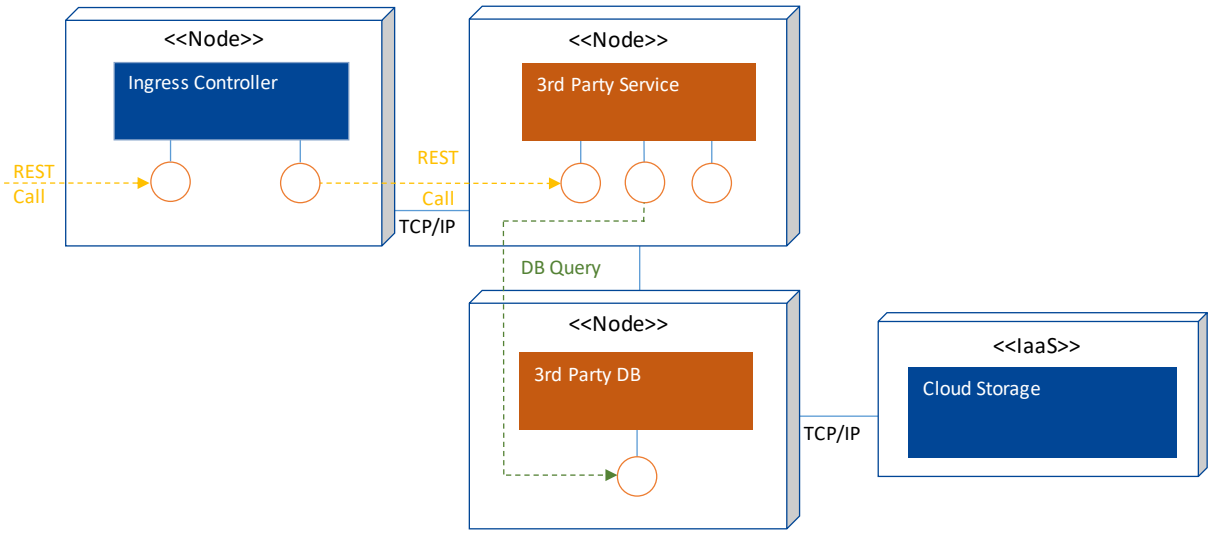

Abbildung 32: Verteilung von 3rd Party Services in Kubernetes basierten Microservice Deployments

Use Case 5: Services verschieben – Die Verschiebung von Services setzt voraus, dass diese bereits mit allen nötigen Services und Container Konfigurationen auf der Plattform verfügbar sind. Das eigentliche Verschieben erfolgt durch neu Instanziierung der zum Service zugeordneten Containern auf neuen Nodes und der Anpassung der Service Konfiguration. Diese kann beispielsweise umgesetzt werden innerhalb eines Clusters durch Nutzung von Flags. Denkbar ist auch die Verschiebung zwischen mehreren Clustern je nach konkreter Implementierung, Abbildung 33 veranschaulicht beide Optionen.

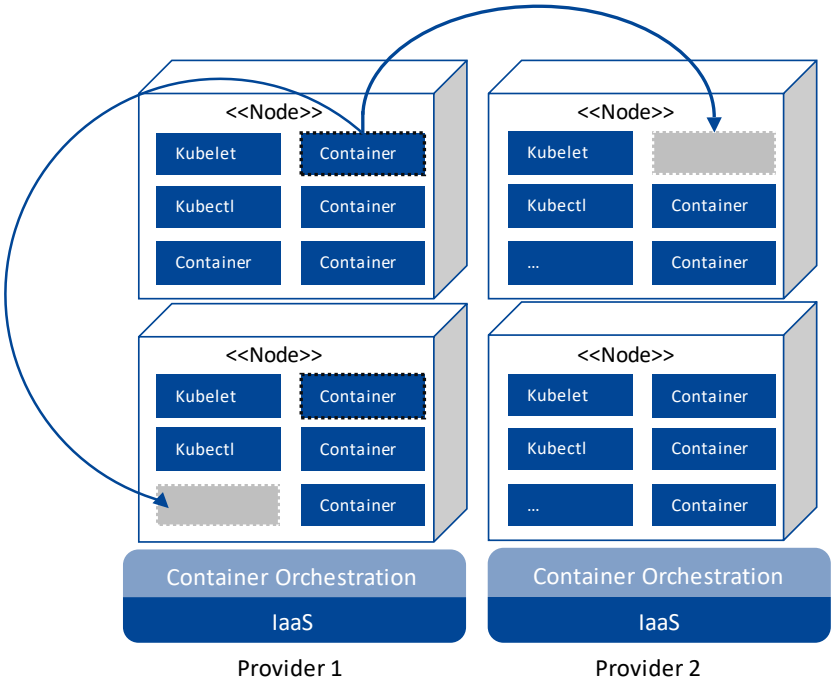

Abbildung 33: Verschieben von Containern zwischen unterschiedlichen Nodes

Use Case 6: Onboarding von 3rd Party Entwicklern – Um neue Entwickler in die Plattform zu integrieren wird das zentrale User Interface benötigt, sowie ein Backend Service zum Onboarden neuer User. Die UI kann über Kubernetes interne Service Abstraktionslevel mit der API Schnittstelle des User Management Service kommunizieren. Wie in der bisherigen Visualisierung aufgezeigt, können auch hier alle Services auf beliebigen Nodes verteilt werden. Das User Management bedient sich einer Plattformglobalen User Datenbank die ebenfalls auf der gleichen Infrastruktur provisioniert wird, ähnlich wie für jeden Datenservice aus den zuvor aufgezeigten Use Cases.

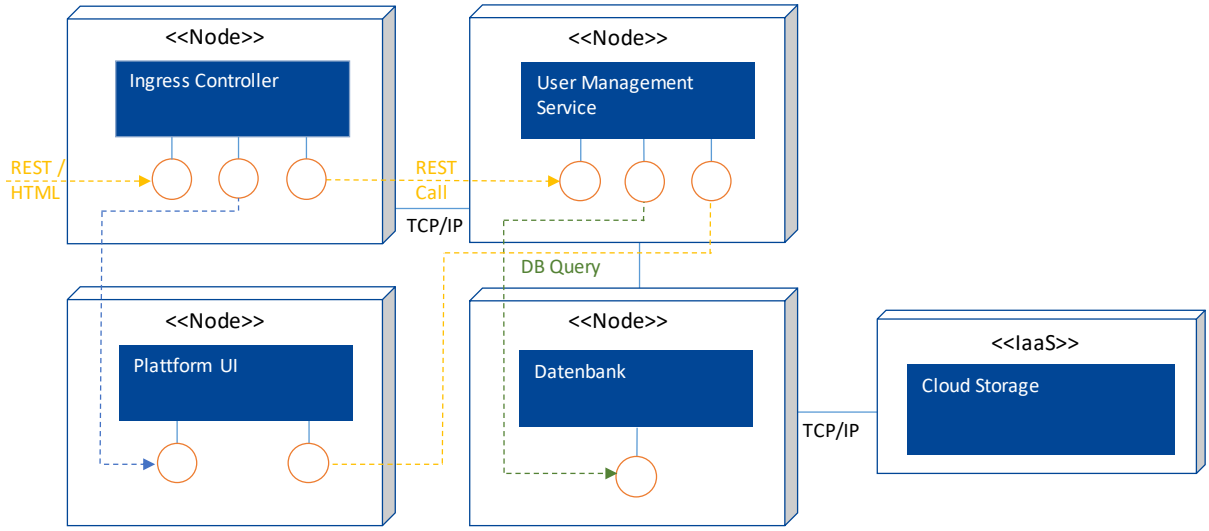

Abbildung 34: Verteilung der für das Onboarding von neuen Entwicklern benötigten Komponenten innerhalb der Kubernetes basierten Infrastruktur

Use Case 7: Service Store – Äquivalent zum Aufbau beim Onboarden neuer User wird das Plattform UI benötigt sowie ein damit interagierender Service. Anstatt eines User Management Service gibt es den Service Marktplatz mit ggf. weiteren Unterfunktionalitäten wie etwa der Suche nach verfügbaren Services. Die Darstellung der UI Service Integration und Verteilung ist weitestgehend gleich zu Use Case 6.

4.5.8. Zusammenfassung der Verteilungssicht

Die Verteilungssicht stellt die Dokumentation der Schnittstelle zwischen Software und Hardware dar. Es wurden Aufgaben und Anforderungen an diese im Kontext der BayernCloud analysiert und es wurde beleuchtet wie die Verteilung durch moderne Microservices beeinflusst wird. Unter dem Einsatz von Container Technologien und Orchestrierungssystemen erfolgte eine Beschreibung der Software Deployments auf generischen Hardwarearchitekturen und Multicloud IaaS Providern. Es wurden Herausforderungen beschrieben und aufgezeigt wie Storage und Kommunikation bei der Verteilung adressiert werden können. Anhand von Verteilungsdiagrammen wurde veranschaulicht wie Zusammenhänge von Software Komponenten auf Hardware abbildbar sind. Im speziellen interessant ist dabei die Verteilung von einzelnen Microservices für die jeweiligen Use Cases aus der Use Case Sicht, entsprechend wurden hier exemplarisch deren Verteilungsdiagramme dokumentiert. In Summe soll damit einem Plattform Administrator oder Operations-Teams die Möglichkeit gegeben werden, die entwickelten Plattform- und Servicekonzepte einfach auf verschiedene Hardware Systeme abzubilden.

4.6. Prozesssicht

4.6.1. Einleitung

In der Prozesssicht werden die dynamischen Aspekte des Systems verdeutlicht. Der Zusammenhang zwischen den Elementen aus der logischen Sicht und deren Zuordnung zu den Services der funktional-technischen Sicht werden beschrieben. Somit bildet die Prozesssicht die Prozesse der BayernCloud Referenzarchitektur ab. Dies sind generische Prozesse, die typischerweise im Betrieb eines digitalen Plattform Ökosystems durchlaufen

werden und basieren auf den in Abschnitt 4.2 skizzierten Use Cases. Für jeden Use Case wird der prozessuale Ablauf beschrieben und dargestellt. Sie richtet sich daher insbesondere an Plattformbetreiber sowie Drittentwickler, die sich primär mit den vorhandenen Prozessen im Falle einer Nutzung auseinandersetzen müssen.

Die dargestellten Prozesse basieren auf den Use Cases und werden mit den Informationen, aus der in Kapitel 4.3 vorgestellten Funktional-technischen Sicht, der in Kapitel 4.4. behandelten Ökosystemsicht und der Verteilungssicht angereichert und in Verbindung gesetzt. Durch diese Verknüpfung geben die Prozesse in ihrer Modellierung ein aggregiertes Gesamtbild der Informationen der BayernCloud Referenzarchitektur wieder. Dies geschieht mithilfe von UML Aktivitätsdiagrammen. Durch diese wird klar erkennbar dargestellt, welche Aktionen der jeweiligen Aktivität auf welcher Entität und durch welche Services durchgeführt wird.

4.6.2. UML Aktivitätsdiagramm

Zur Darstellung der Prozesse wird auf UML-Aktivitätsdiagramme zurückgegriffen[3]. Dabei handelt es sich um ein sogenanntes Verhaltensidagramm der Unified Modeling Language (UML), die eine Modellierungssprache für Software und Systeme darstellt. In UML-Aktivitätsdiagrammen werden die Verbindungen von zentralen Aktionen zur Beschreibung des Ablaufs von Use Cases grafisch dargestellt.

Abbildung 35 stellt ein generisches UML-Aktivitätsdiagramm dar. Der Start des Verhaltensdiagramm erfolgt mit einem Startknoten. Ggf. gibt es existierende Vorbedingungen, die erfüllt sein müssen, damit der Start erfolgen kann. Diese entsprechen den Vorbedingungen der Use Case Beschreibungen aus Abschnitt 4.2. Die zentralen Aktionen sind durch abgerundete Rechteeckte dargestellt und mit weiteren Knoten zu einem logischen Ablauf durch Pfeile verbunden. Entscheidungsknoten werden durch Rauten dargestellt und trennen den logischen Fluss in Abhängigkeit der Entscheidungsvariablen. Neben Entscheidungsknoten gibt es noch sogenannte Parallelisierungs- bzw. Synchronisationsbalken. Wenn verschiedene Aktivitäten parallel ablaufen, werden diese parallel zueinander dargestellt und durch diese Balken getrennt bzw. wieder zusammengeführt. Das Aktivitätsdiagramm endet immer mit mindestens einem End-Knoten, der den Endzustand nach durchlauf der Aktivitäten, die zu diesem Knoten führen, beschreibt. Die Endzustände in den UML-Aktivitätsdiagrammen für die Use Cases 1-7 entsprechen den definierten Erfolgs- bzw. Misserfolgsszenarien aus Abschnitt 4.2.

3 https://www.ibm.com/support/knowledgecenter/de/SS8PJ7_9.6.1/com.ibm.xtools.modeler.doc/topics/twrkactd.html

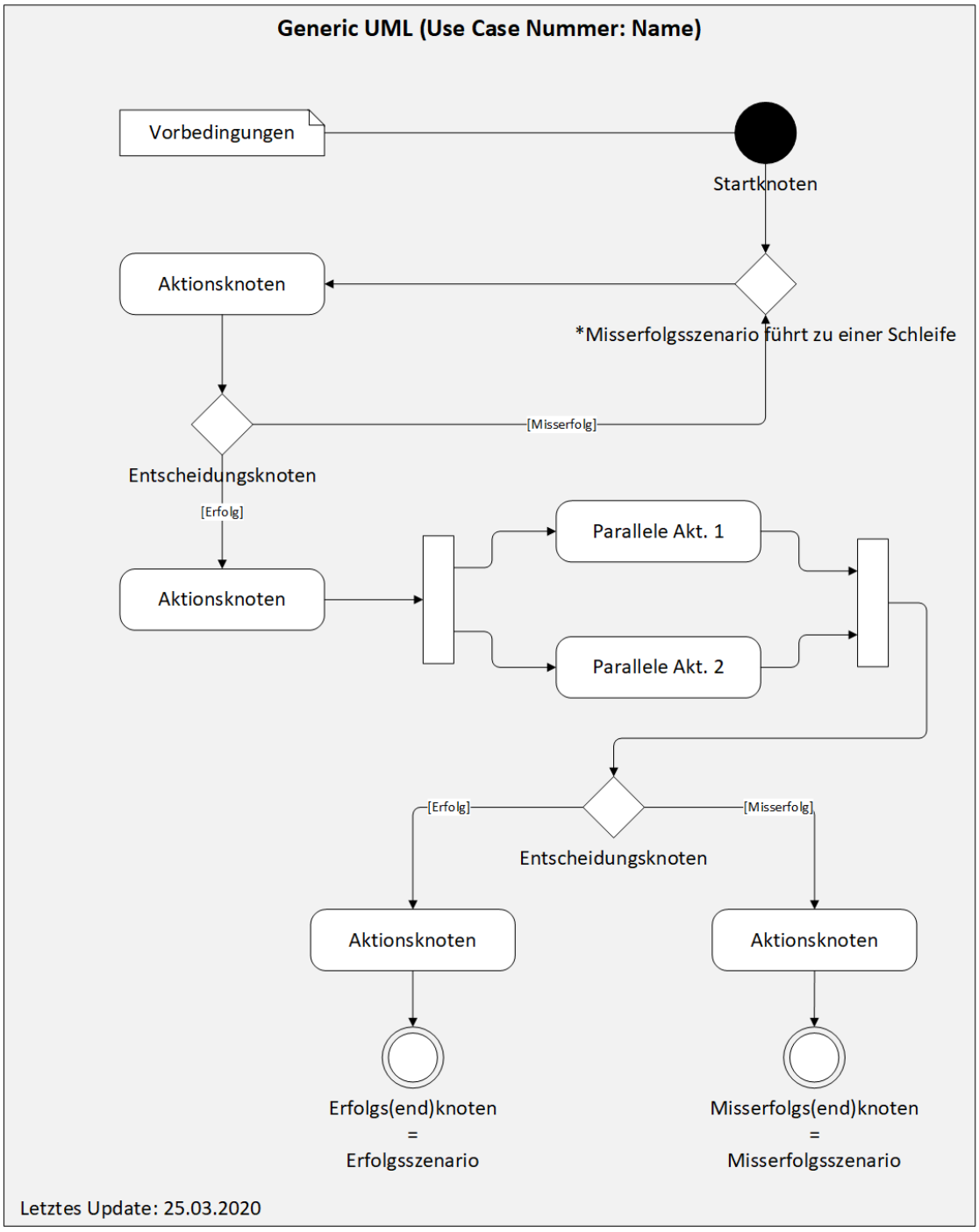

Abbildung 35: Vorlage zur Beschreibung der Prozesssicht anhand von UML Aktivitätsdiagrammen

4.6.3. Prozess zu Use Case 1: Daten lesen

Use Case 1 behandelt das lesen von Daten über die Plattform. Der Prozess stellt den Ablauf des Use Cases aus der Sicht von Anwender, Betreiber, Plattform-Infrastruktur sowie Extern dar. Der Prozess startet durch einen Anwender, der Daten auslesen möchte. Es erfolgt eine Prüfung der Berechtigungen des Nutzers. Ist die Prüfung erfolgreich, erhält der Nutzer die gewünschten Daten. Im Falle eines Misserfolgs erhält der Nutzer, bzw. im Fall von technischen Problemen der Betreiber und ggf. ein externer Datenprovider eine Fehlermeldung durch das System.

Der Prozess wird regelmäßig durchlaufen. Der Ablauf ist in Abbildung 36 dargestellt.

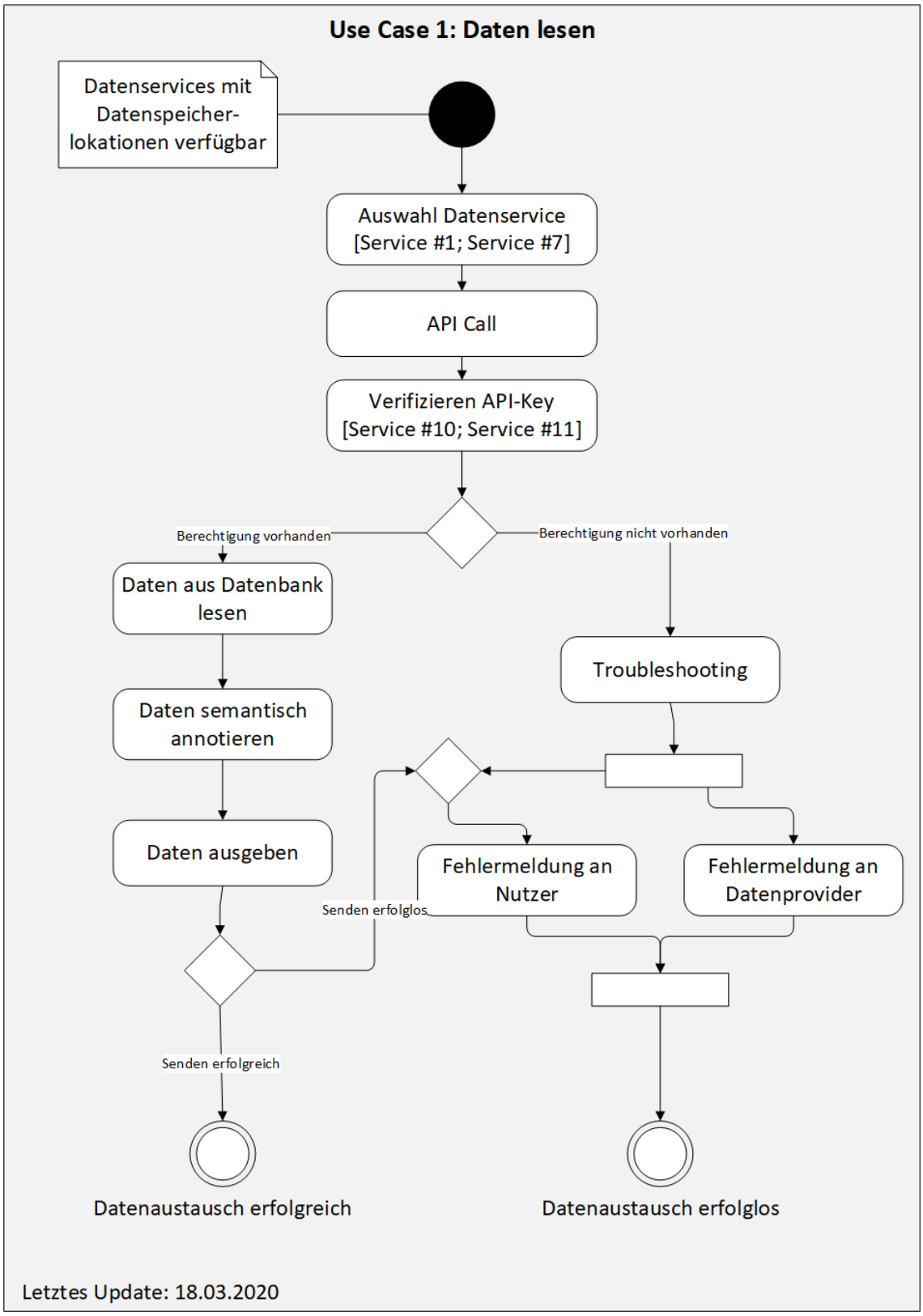

Abbildung 36: Prozess zu Use Case 1 (Daten lesen)

Beschreibung: Als zentrale Instanz des zu entwickelnden DPÖs, soll eine Plattform es verschiedenen Teilnehmern im Ökosystem ermöglichen, domänenspezifische Daten

auszutauschen. Dafür müssen geeignete Datenservices zur Verfügung gestellt werden können, welche Schnittstellen zum Auslesen von Daten bereitstellen. Hierbei müssen Kriterien berücksichtigt werden wie etwa die unterschiedlichen möglichen Zugriffsrechte, Lizenzbedingungen, Datenhoheit und Datensicherheit.

Vorbedingung: Verfügbarkeit von entsprechend definierten Datenservices mit Datenspeicherlokationen

Ablauf:

1. Auswählen der Datenservices
2. Abfragen der Daten über API Calls
3. Hat der Nutzer der entsprechenden Berechtigung?
    a. *Nein*
        i. Fehlermeldung an Nutzer senden
        ii. **ENDE**
    b. *Ja*
        i. Senden der Daten
        ii. War das Senden erfolgreich?
            1. *Nein*
                a. Troubleshooting durchführen
                b. Fehlermeldung an Nutzer und Provider schicken
                c. **ENDE**
            2. *Ja*
                a. **ENDE**

4.6.4. Prozess zu Use Case 2: Daten schreiben

Die BayernCloud soll es verschiedenen Teilnehmern im Ökosystem ermöglichen domänenspezifische Daten auf der Plattform zu speichern. Dafür müssen geeignete Datenservices zur Verfügung gestellt werden können, welche Daten in passender Form speichern und verwalten können. Es müssen Kriterien berücksichtigt werden wie etwa die unterschiedlichen möglichen Zugriffsrechte, Lizenzbedingungen, Datenhoheit und Datensicherheit.

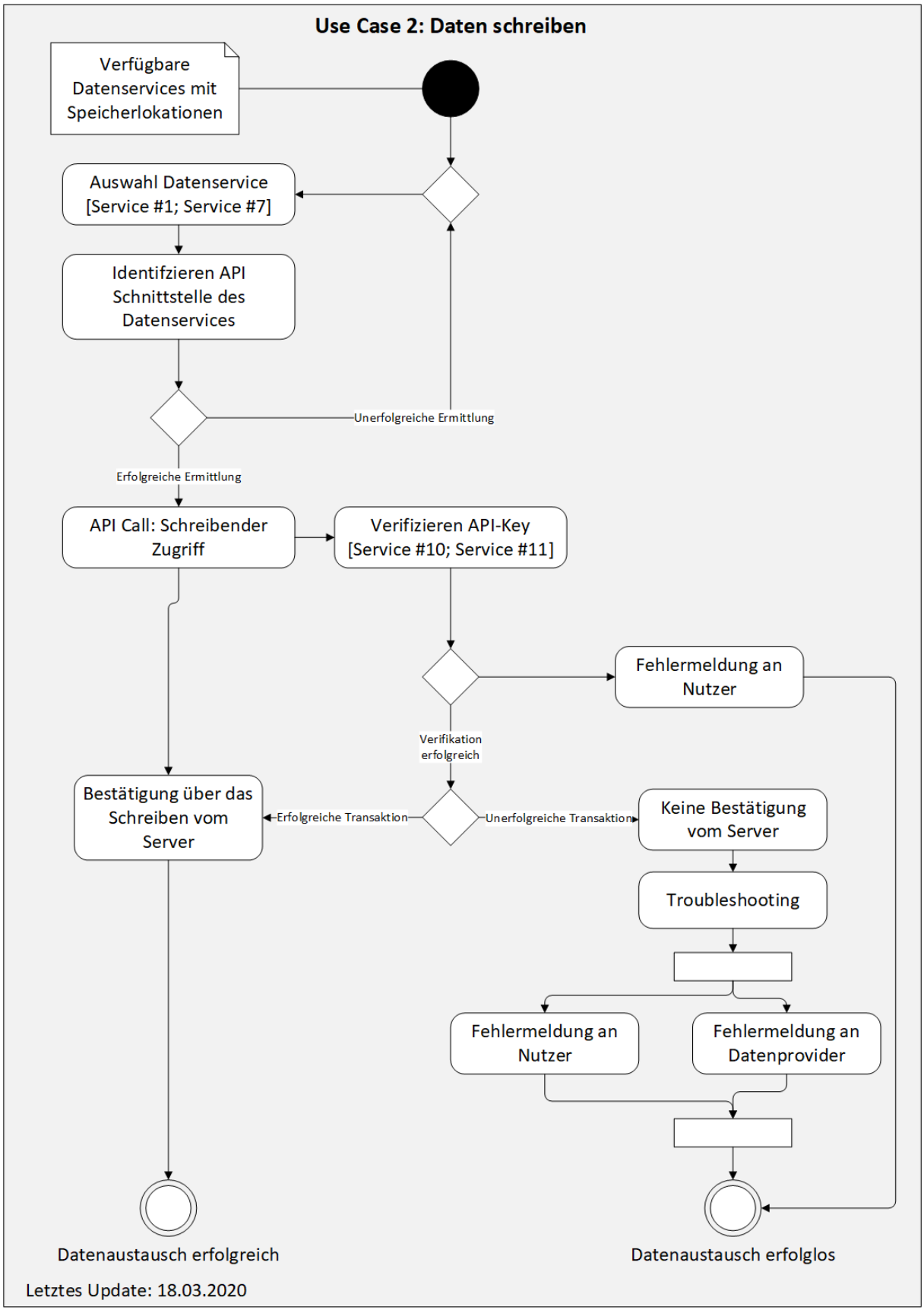

Abbildung 37: Prozess zu Use Case 2 (Daten schreiben)

Beschreibung: Die BayernCloud soll es verschiedenen Teilnehmern im Ökosystem ermöglichen domänenspezifische Daten auf der Plattform zu speichern. Dafür müssen geeignete Datenservices zur Verfügung gestellt werden können, welche Daten in passender Form speichern und verwalten können. Es müssen Kriterien berücksichtigt werden wie etwa die unterschiedlichen möglichen Zugriffsrechte, Lizenzbedingungen, Datenhoheit und Datensicherheit.

Vorbedingung: Verfügbarkeit von entsprechend definierten Datenservices mit Datenspeicherlokationen

Ablauf:

1. Identifizieren der API Schnittstellen der Datenservices
2. War die Ermittlung erfolgreich?
    a. *Nein*
        i. Mit Schritt 1 weitermachen
    b. *Ja*
        i. Senden von einer Schreibanfrage der Daten über API Calls
        ii. War das Senden erfolgreich?
            1. *Nein*
                a. Senden von entsprechender Fehlermeldung an Datenprovider
                b. **ENDE**
            2. *Ja*
                a. Bestätigen über das Schreiben vom Server
                b. **ENDE**

4.6.5. Prozess zu Use Case 3: Daten und Datenservice Rekombination

Entwickler von externen Unternehmen sollen in der Lage sein, anhand der zu instanziierenden Plattform und Service Spezifikationen einfach von aktuellen Möglichkeiten der Daten- und Service-Rekombination zu profitieren. Dazu sollten entsprechend semantisch annotierte Services und technische Möglichkeiten der Rekombination von Schnittstellen für Drittentwickler zur Verfügung stehen.

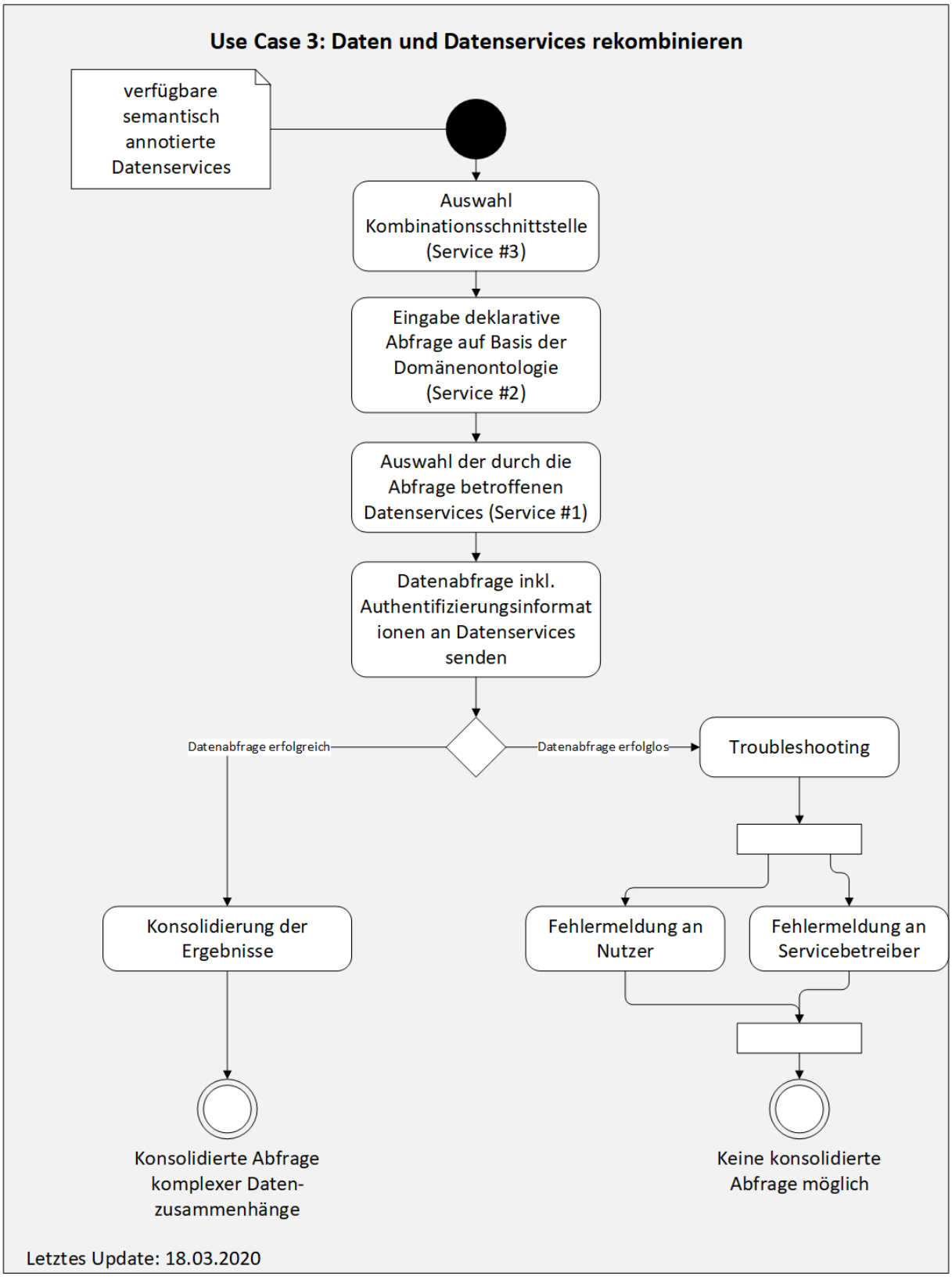

Abbildung 38: Prozess zu Use Case 3 (Daten und Datenservices rekombinieren)

4.6.6. Prozess zu Use Case 4: 3rd Party Service bereitstellen

Drittentwickler sollen in der Lage sein, die instanziierte Infrastruktur zu nutzen, um eigene Services zur Verfügung zu stellen. Dabei soll durch die Plattform eine einfache Möglichkeit gegeben sein, die Services auf der Plattforminfrastruktur zu provisionieren.

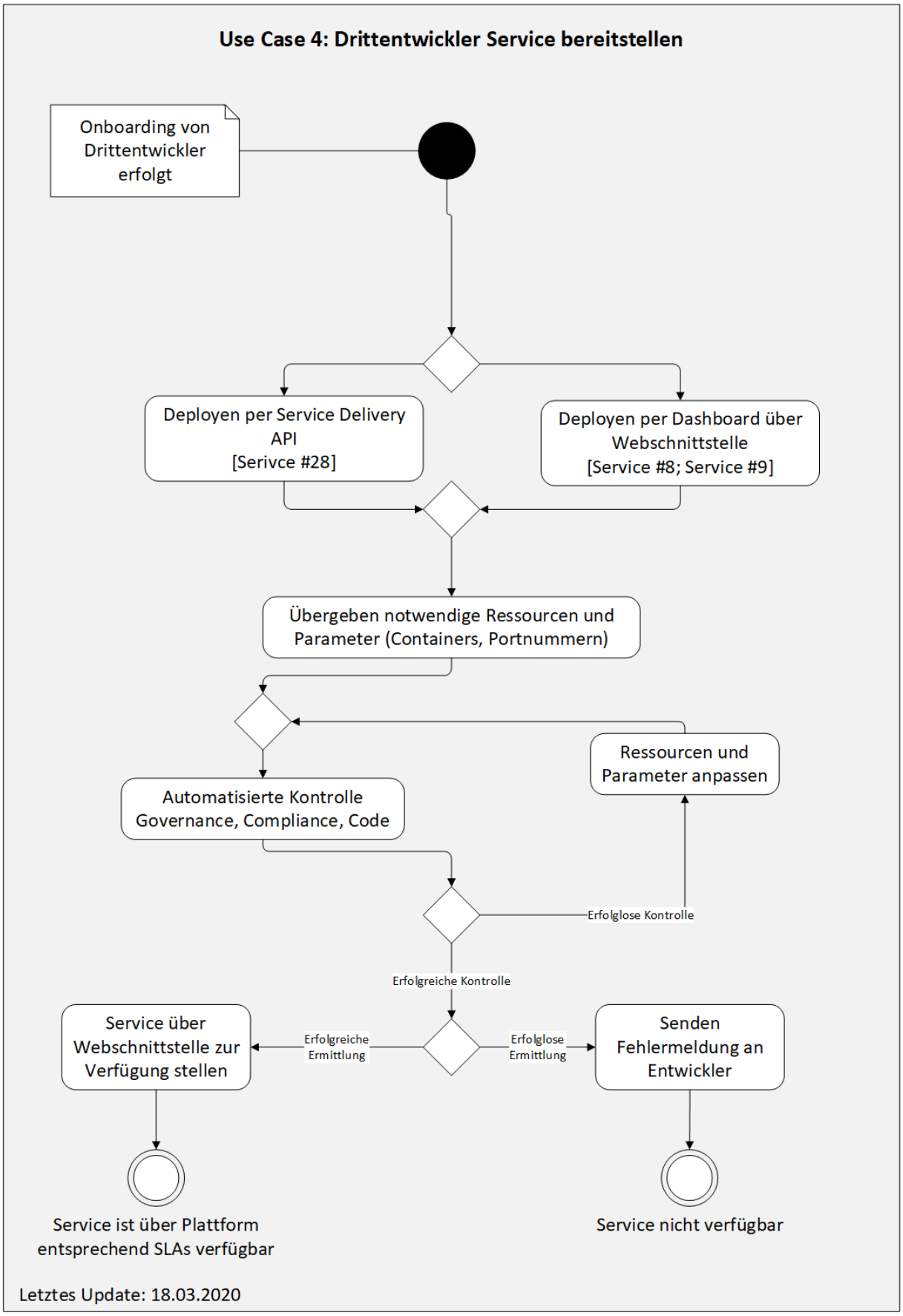

Abbildung 39: Prozess zu Use Case 4 (Drittentwickler Service bereitstellen)

Beschreibung: Drittentwickler sollen in der Lage sein, die instanziierte Infrastruktur zu nutzen, um eigene Services zur Verfügung zu stellen. Dabei soll durch die Plattform eine einfache Möglichkeit gegeben sein, die Services auf der Plattforminfrastruktur zu provisionieren.

Vorbedingung: Der Drittentwickler, der den Onboarding-Prozess durchgelaufen hat.

Ablauf:

1. Überführen der App-Logik in ein geeignetes Format für die Bereitstellung in der BayernCloud
2. Entscheiden über den Deploymentsort
    a. *Deployen per Service Delivery API*
    b. *Deployen per Dashboard über die Webschnittstelle*
3. Übergeben der notwendigen Ressourcen und Parametern (Containers, Portnummern…)
4. War die Ermittlung erfolgreich?
    a. *Nein*
        i. Senden von entsprechender Fehlermeldung an Entwickler
        ii. **ENDE**
    b. *Ja*
        i. Zur Verfügung stellen der Service über Webschnittstelle
        ii. **ENDE**

4.6.7. Prozess zu Use Case 5: Services verschieben

Entwickler von externen Unternehmen sollen in der Lage sein, Services, welche auf der BayernCloud Infrastruktur zur Verfügung gestellt werden, zwischen unterschiedlichen Infrastruktur-Anbietern zu verschieben. Diese Möglichkeit soll die Betreiberunabhängigkeit des BayernCloud DPÖs gewährleisten.

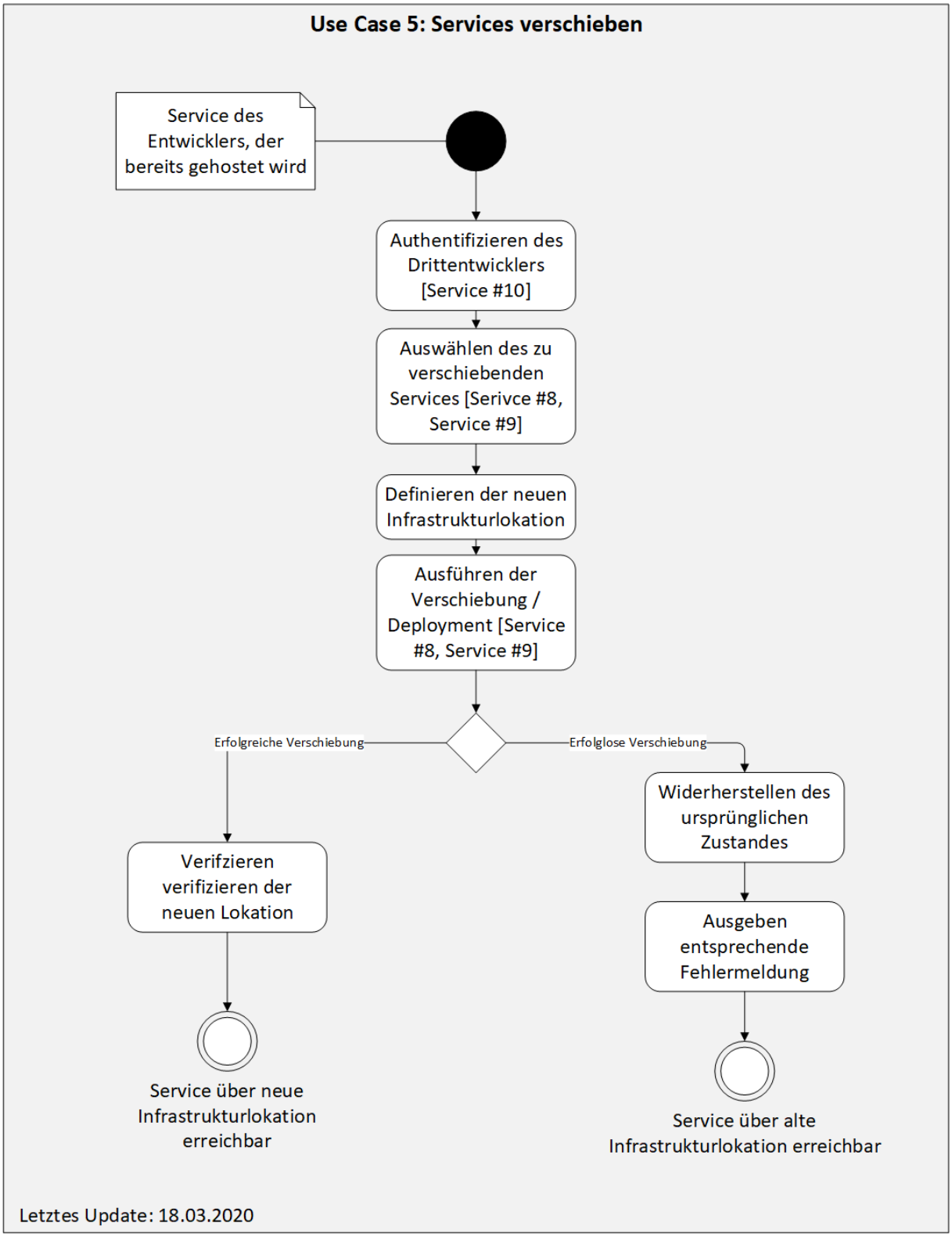

Abbildung 40: Prozess zu Use Case 5 (Services verschieben)

Beschreibung: Entwickler von externen Unternehmen sollen in der Lage sein, Services, welche auf der BayernCloud Infrastruktur zur Verfügung gestellt werden, zwischen unterschiedlichen Infrastruktur-Anbietern zu verschieben. Diese Möglichkeit soll die Betreiberunabhängigkeit des BayernCloud DPÖs gewährleisten.

Vorbedingung: Service des Entwicklers, der bereits gehostet wird

Ablauf:

1. Authentifizieren des Drittentwicklers
2. Auswählen des zu verschiebenden Services
3. Definieren der neunen Infrastrukturlokation
4. Ausführen der Verschiebung
5. War die Verschiebung erfolgreich?
    a. *Nein*
        i. Widerherstellen des ursprünglichen Zustandes
        ii. Ausgeben entsprechende Fehlermeldung
        iii. **ENDE**
    b. *Nein*
        i. Verifizieren der neuen Lokation der Information über das Plattform-Management
        ii. **ENDE**

4.6.8. Prozess zu Use Case 6: Onboarding von 3rd Party Entwicklern

Zur kontinuierlichen Entwicklung des DPÖs, soll die Plattform einen einfachen Prozess zum Onboarding dritter Parteien zur Verfügung stellen. Dies ist notwendig, um den rechtlichen- und sicherheits- technischen Anforderungen Rechnung zu tragen. Der Vorgang muss sowohl bei der technischen Architektur berücksichtigt werden als auch Mechanismen zur Plattform-Governance und Sicherheit gewährleisten können.

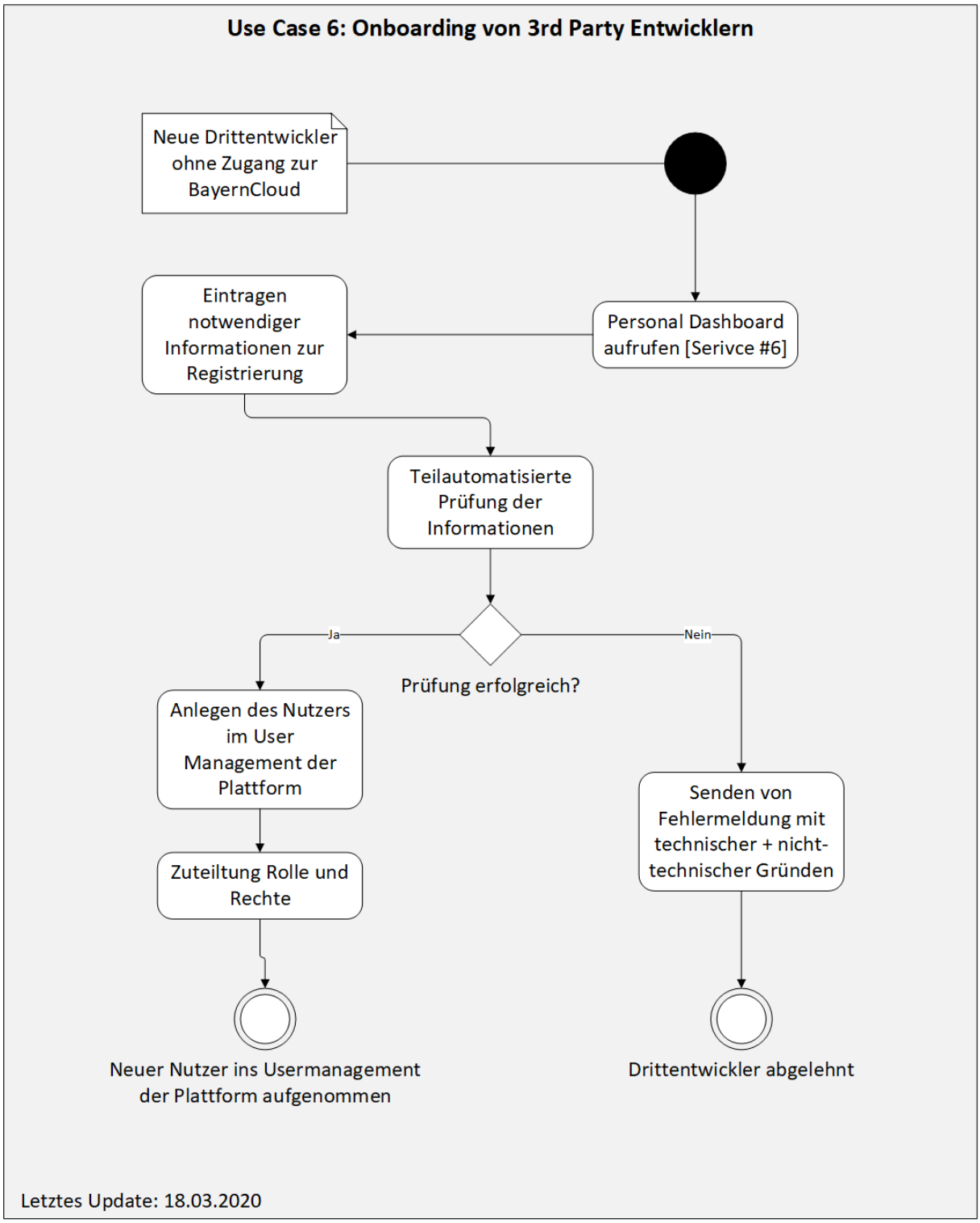

Abbildung 41: Prozess zu Use Case 6 (Drittentwickler onboarden)

Beschreibung: Zur kontinuierlichen Entwicklung des DPÖs, soll die Plattform einen einfachen Prozess zum Onboarding dritter Parteien zur Verfügung stellen. Dies ist notwendig, um den rechtlichen- und sicherheits- technischen Anforderungen Rechnung zu tragen. Der Vorgang muss sowohl bei der technischen Architektur berücksichtigt werden als auch Mechanismen zur Plattform-Governance und Sicherheit gewährleisten können.

Vorbedingung: Neue Drittentwickler ohne Zugang zur BayernCloud

Ablauf:

1. Aufrufen des Plattform-Management
2. Eintragen der von der Plattform benötigten Informationen zur Registrierung
3. Durchführen teilautomatisierte Evaluierung der Informationen
4. War die Evaluierung erfolgreich?
    a. *Nein*
        i. Senden von Fehlermeldung mit technischer + nicht-technischer Gründen
        ii. **ENDE**
    b. *Ja*
        i. Anlegen des Nutzers ins User-Management der Plattform
        ii. Zuteilung der nutzerspezifischen Rollen und Rechte
        iii. **ENDE**

4.6.9. Prozess zu Use Case 7: Service Store

Um die verschiedenen Angebote und Services für alle Teilnehmer des DPÖs zur Verfügung zu stellen, sollte die Plattform einen Servicemarktplatz bereitstellen. Diese Funktionalität ermöglicht es, Angebot und Nachfrage von Services, die auf der Plattform bereitgestellt sind, zu managen und entsprechende Interaktionen zwischen Anbietern und Kunden über die Plattform zu fördern.

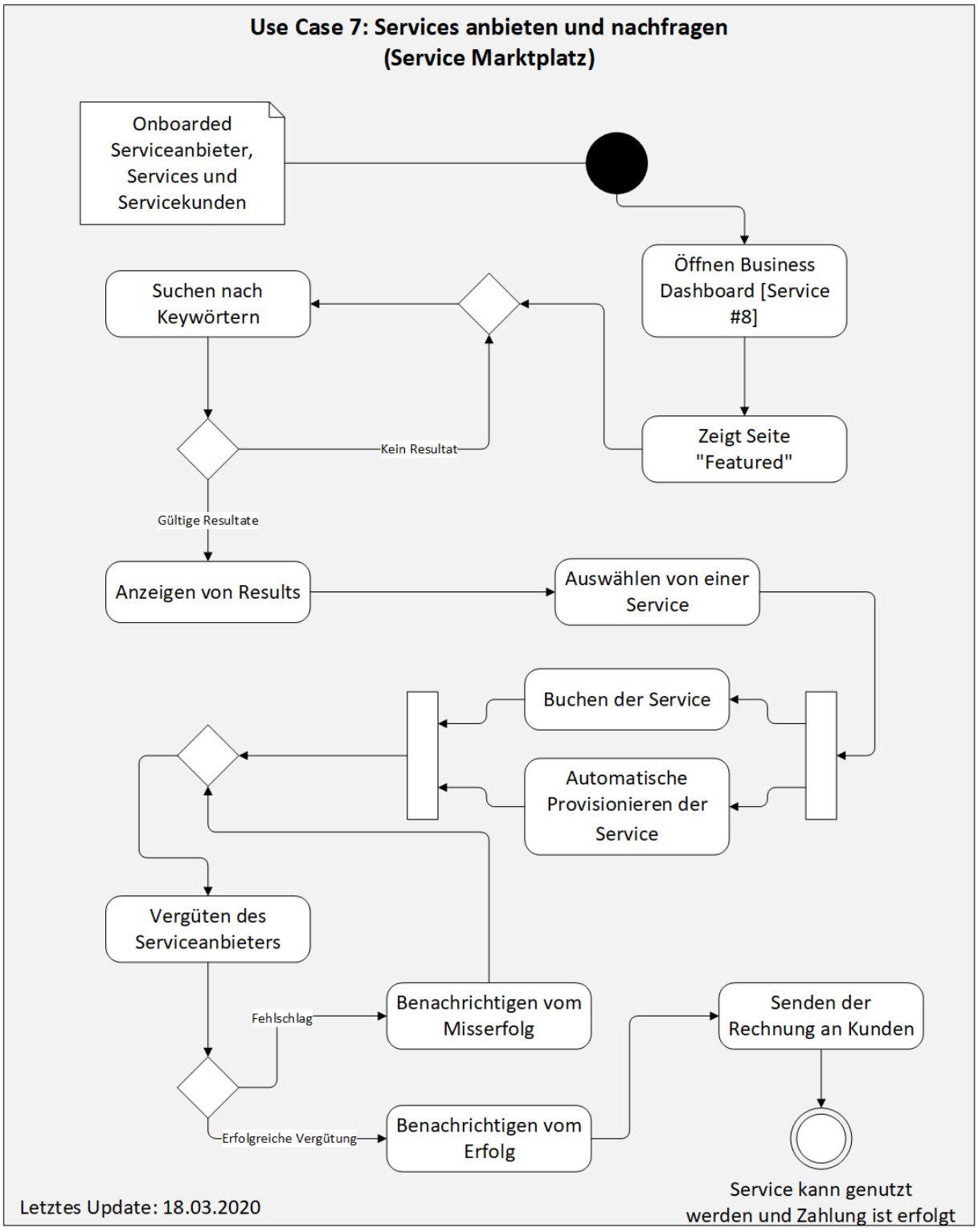

Abbildung 42: Prozess zu Use Case 7 (Services anbieten und nachfragen - Service Marktplatz)

Beschreibung: Um die verschiedenen Angebote und Services für alle Teilnehmer des DPÖs zur Verfügung zu stellen, sollte die Plattform einen Servicemarktplatz bereitstellen. Diese Funktionalität ermöglicht es, Angebot und Nachfrage von Services, die auf der Plattform bereitgestellt sind, zu managen und entsprechende Interaktionen zwischen Anbietern und Kunden über die Plattform zu fördern.

Vorbedingung: Serviceanbieter, Services und Servicekunde sind onboarded

Ablauf:

1. Öffnen vom App-Store-Bereich der BayernCloud
2. Zeigen der „Featured"-Seite
3. Suchen nach Keywörtern
4. Gibt es ein Ergebnis?
    a. *Nein*
        i. Mit Schritt 4 weitermachen
    b. *Ja*
        i. Anzeigen von Ergebnissen
        ii. Auswählen von einem Service
        iii. Buchen UND automatische Provisionieren des Service
        iv. Vergüten des Serviceanbieters
        v. War die Vergütung erfolgreich?
            1. *Nein*
                a. Benachrichtigen vom Misserfolg
                b. **ENDE**
            2. *Ja*
                a. Benachrichtigen vom Erfolg
                b. Senden der Rechnung an Kunden
                c. **ENDE**

4.6.10. Zusammenfassung der Prozesssicht

Die Prozesssicht stellt den Zusammenhang zwischen den Elementen aus der logischen Sicht und deren Zuordnung zu den Services der funktional-technischen Sicht dar. Zu diesem Zweck werden die Use Cases als Prozesse durch UML-Aktivitätsdiagramme beschrieben. Für jeden Use Case wird der Prozess und die zentralen Aktivitäten, die zur Durchführung des Use-Cases notwendig sind, beschrieben. Die verschiedenen Vorbedingungen und Endzustände werden in den Prozessen hierbei explizit dargestellt. Die Prozesssicht adressiert hierbei primär mögliche Plattformbetreiber sowie Drittentwickler, die sich mit den vorhandenen Prozessen auseinandersetzen.

5. Zertifizierungsmodell

Kurzübersicht des Kapitels

Dieses Kapitel fasst die wichtigsten Ergebnisse zur Zertifizierung des Forschungsprojekts BayernCloud zusammen und gibt eine Übersicht der Zertifizierungsmodelle im Cloud-Umfeld. Des Weiteren wird das Zertifizierungsmodell und die entsprechenden Prozesse zur Zertifizierung der BayernCloud Referenzarchitektur vorgestellt. Eine detaillierte Betrachtung erfolgt im gesonderten Bericht „Abschlussbericht Arbeitspaket 5: Zertifizierungsmodell / Dokumentation".

Als Zertifizierung wird ein Verfahren bezeichnet, welches überprüft, dass ein Produkt, Prozess, System oder eine Person definierte Anforderungen erfüllt. Diese Überprüfung wird durch externe, neutrale Prüfer durchgeführt und durch ein Zertifikat, ein Ergebnisdokument der Zertifizierung, festgehalten. Eine Auditierung hingegen kann auch ohne ein solches Ergebnisdokument durchgeführt werden. Häufig werden Auditierungen als internes Audit oder Vorab-Audit vor einer Zertifizierung durch interne Mitarbeiter durchgeführt.

5.1. Dimensionen für die Auswahl von Zertifizierungen

Bei der Auswahl einer Zertifizierung können verschiedene Kriterien herangezogen werden. (Schneider & Sunyaev, 2015) identifizieren die folgenden Dimensionen nach denen Zertifizierungen klassifiziert und verglichen werden können:

Bedeutung des Zertifizierungskriteriums für die Informationssicherheit eines Cloud-Services: Bei dieser Dimension geht es darum, die einzelnen Kriterien einer Zertifizierung gewichten zu können, um die Kriterien und eventuelle Gefährdungen angemessen priorisieren zu können.

Prüfobjekt bei der Auditierung: Diese Dimension beschreibt die Art des Prüfobjekts, z.B.: Prozess, Providereigenschaften, Serviceeigenschaften, Infrastruktur, Softwarearchitektur, Vertrag, Personal.

Eingesetzte Auditierungsmethoden: Folgende Methoden können bei einem Audit eingesetzt werden: Interview, Servicenutzung, technische Prüfung, Assetprüfung, Dokumentationsprüfung, Vor-Ort-Auditierung, kontinuierliche Auditierung.

Angewandte Prüftiefe: Diese Dimension gibt an in welchem Umfang ein Prüfobjekt geprüft wird. Bspw. kann eine Prüfung eine Stichprobe oder die Gesamtheit einer Art von Prüfobjekten zur Auditierung heranziehen. Im Rahmen der BayernCloud ist zu empfehlen die entsprechenden Kriterien so zu wählen, dass auch die Interaktion mit Sub-Cloud-Anbietern und Teilnehmern der Plattform abgedeckt wird.

Messwerte, Messgrößen und weitere quantitative Informationen: Messwerte und Messgrößen beziehen sich auf die konkreten Ausprägungen der Zertifizierungskriterien. Werden z.B. für das Kriterium *Verbindungsverschlüsselung* nicht nur ja/nein Ausprägungen angegeben, sondern konkrete Algorithmen, verbessert dies die Vergleichbarkeit.

Beeinflussbarkeit von Veränderungen seitens des Cloud-Services: Diese Dimension gibt an, bei welcher Art von Veränderung eines Cloud Services die Überprüfung des Kriteriums wiederholt werden muss.

Notwendigkeit einer kontinuierlichen Überprüfung: Die kontinuierliche Überprüfung von Zertifizierungskriterien bei Cloud-Services besonders sinnvoll (siehe auch Abschlussbericht AP5: Zertifizierungsmodell/Dokumentation)

Prüfintervall bei der Auditierung der Einhaltung des Zertifizierungskriteriums: Unabhängig von einer kontinuierlichen Überprüfung von Zertifizierungskriterien gibt diese Dimension an, in welchen Zeitabständen ein vollständiger Audit nötig ist, um ein Zertifikat zu verlängern.

Konsequenzen bei einer Verletzung des Zertifizierungskriteriums: Diese Dimension bezieht sich darauf, inwiefern sich Verletzungen von Zertifizierungskriterien auf das Zertifikat auswirken. Bspw. kann gefordert sein, die Verletzung innerhalb einer Frist zu korrigieren.

Eine Übersicht der relevanten und verfügbaren Zertifizierungsmöglichkeiten, die für das digitale Plattform-Ökosystem in Frage kommen, ist im gesonderten wissenschaftliche Bericht „Abschlussbericht AP5: Zertifizierungsmodell/Dokumentation" zu finden.

Grundsätzlich wird für das DPÖ BayernCloud ein dynamischer Zertifizierungsprozess vorgeschlagen, wie in Abbildung 43 beispielhaft dargestellt, und im Dokument „BayernCloud Arbeitspaket 5: Zertifizierungsmodell/Dokumentation" näher beschrieben. Aktuelle Zertifizierungsmodelle sind für statische IT-Systeme ausgelegt, dabei werden die Zertifizierungen zu bestimmten Zeitpunkten durchgeführt und es wird davon ausgegangen, dass sich das zertifizierte System in der Gültigkeitsperiode nicht dermaßen ändert, so dass eine erneute Zertifizierung nicht mehr zur Erlangung des Zertifikats führt. Abbildung 43 zeigt einen statischen Zertifizierungsprozess. Ausgegebene Zertifikate haben eine festgelegte Gültigkeitsdauer (z.B. 3 Jahre) und werden zu bestimmten Zeitpunkten während der Gültigkeitsdauer durch Überwachungsaudits stichprobenartig überprüft.

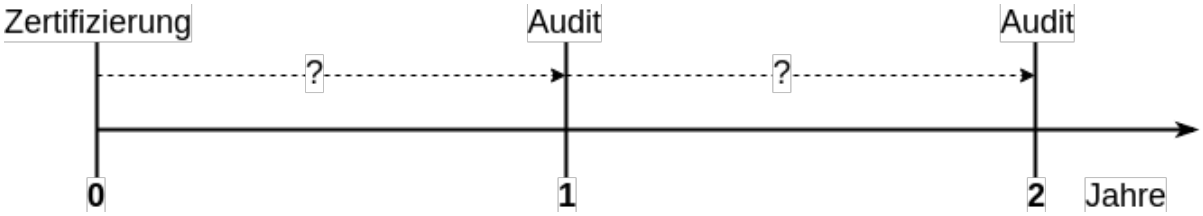

Abbildung 43: Statischer Zertifizierungsporzess

Cloud-Systeme unterliegen jedoch einem steten Wandel durch Veränderungen der Cloud-Services. Gerade Container-basierte Applikationen können schnell skalieren und sich verändern. Somit ist eine statische Zertifizierung nicht mehr ausreichend, da sich das Cloud-System innerhalb der Gültigkeitsperiode oftmals derart ändern kann, dass die Kriterien des Zertifikats nicht mehr erfüllt sind. Änderungen, die zur Nicht-Einhaltung führen können sind nach (Stephanow & Banse, 2018) z.B. Folgende:

- Konfigurationsänderungen des Cloud-Services
- neue Versionen einzelner Komponenten des Cloud-Services
- Migration von Komponenten eines Cloud-Services in ein anderes Rechenzentrum

Aus diesem Grund sind statische Analysen unzureichend für Cloud-Systeme und ein neuer Ansatz zur dynamischen Zertifizierung ist nötig um Cloud-Systeme kontinuierlich im Produktivsystem zu analysieren und somit zertifizieren zu können. Abbildung 44 zeigt einen dynamischen Zertifizierungsprozess, welcher ein Cloud-System fortlaufend überwacht und somit zeitnah die Nicht-Einhaltung von Zertifikatskriterien erkennt.

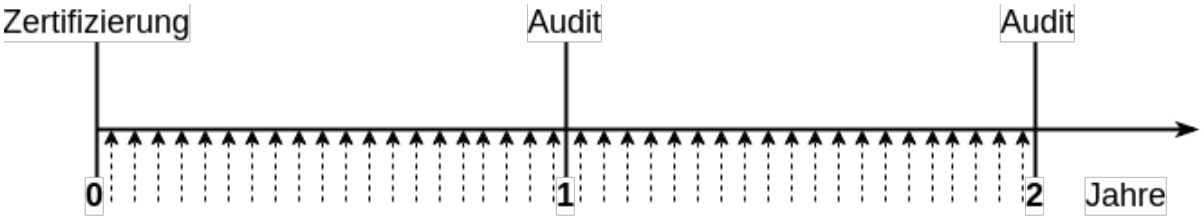

Abbildung 44: Dynamischer Zertfizierungsprozess

Wie in Abbildung 44 gezeigt, wird die Zertifizierung zu festen Zeitpunkten durch einen externen Auditor durchgeführt und durch Zwischenaudits ergänzt (expertenbasiertes Zertifizierungsmodell). Zwischen der Zertifizierung und den Zwischenaudits werden kontinuierlich Überprüfungen durch ein Tool durchgeführt (test-basiertes Zertifizierungsmodell), welches überprüft, ob die BayernCloud auch während der Laufzeit zwischen den Zertifizierungen die definierten Anforderungen erfüllt.

5.2. Zertifizierungsmodelle

Um eine dynamische Zertifizierung durchführen zu können wird ein Zertifizierungsmodell vorausgesetzt, welches die Art und Weise der Informationsgewinnung der Auditinformationen beschreibt. Nach (Stephanow & Banse, 2018) gibt es 4 verschiedene Zertifizierungsmodelle, die im Folgenden beschrieben werden.

Monitoring-basiertes Zertifizierungsmodell

Das Monitoring-basierte Zertifizierungsmodell nutzt existierende Monitoring-Daten als Auditinformationen. Dabei können zwei Arten von Monitoring-Systemen unterschieden werden:

- bereits vorhandene Monitoring-Systeme, die auch für den Betrieb genutzt werden
- Zusätzliche, spezielle Monitoring-Systeme, die nicht zum eigentlichen Betrieb des Cloud-Systems genutzt werden

Für beide Arten existieren bereits einige Monitoringsysteme, wobei Monitoringsysteme, die nicht zum Betrieb des Cloud-Systems selbst benötigt werden, meist aus der Forschung und Entwicklung kommen, wie z.B. (Krotsiani, Spanoudakis, & Mahbub, 2013), (Schiffmann, Sun, Vijayakumar, & Jaeger, 2013).

Test-basiertes Zertifizierungsmodell

Das test-basierte Zertifizierungsmodell nutzt Rückgabewerte eines test-basierten Messverfahrens als Auditinformationen und vergleicht diese mit den erwarteten Werten, um Abweichungen festzustellen. Das test-basierte Zertifizierungsmodell unterscheidet sich von dem monitoring-basierten Zertifizierungsmodell dadurch, dass es aktiv das Cloud-System überprüft wohingegen das monitoring-basierte Zertifizierungsystem passiv agiert.

Auch hier lassen sich zwei Arten von Messverfahren unterscheiden:

- bereits vorhandene Messverfahren, wie z.B. Performanztests
- spezielle test-basierte Messverfahren

Für beide Arten existieren bereits einige Messverfahren. Die Nutzung von bereits vorhandenen Messverfahren für test-basierte Zertifizierungsmodelle wird bspw. von (Pham, Chen, Kalbarczyk, & Iyer, 2011) vorgeschlagen. Spezielle test-basierte Messverfahren kommen aus der Forschung und werden z.B. von (Anisetti, Argostino Ardagna, Damiani, & Gaudezini, 2016) und (Albelooshi, Damiani, & Salah, 2015) vorgeschlagen.

Trusted-Platform-Modul-basiertes Zertifizierungsmodell

Das Trusted-Platform-Modul-basierte Zertifizierungsmodell kann ein TPM (Trusted Platform Module) nutzen um bspw. die Integrität der Plattform zu messen und somit sicherzustellen, dass sich die Cloud-Plattform in einem definierten Zustand befindet. Es lassen sich zwei Arten unterscheiden:

- Trusted Boot: Während des Bootvorgangs werden Messungen der Komponenten durchgeführt bevor sie ausgeführt werden und somit überprüft ob sich die Komponenten in einem definierten Zustand befinden und nicht manipuliert wurden.
- Remote Attestation: Nach dem Bootvorgang wird die gemessene Hash-Kette von einem externen Prüfer überprüft und somit sichergestellt, dass sich das Cloud-System in einem definierten Zustand befindet.

Im Vergleich zu den beiden bereits vorgestellten Zertifizierungsmodellen werden bei dem Trusted-Platform-modul-basierten Zertifizierungsmodell die Auditinformationen nicht auf Basis von Laufzeitinformationen erhalten. Wie Remote Attestation für das Trusted-Platform-Modul-basierte Zertifizierungsmodell genutzt werden kann zeigen (Liu, Lin, & Fang, 2013) und (Xiang, Zhao, Yang, & Wei, 2014).

Experten-basiertes Zertifizierungsmodell

Das experten-basierte Zertifizierungsmodell hebt sich von den bereits vorgestellten Verfahren insofern ab, dass keine technischen Systeme genutzt werden, sondern eine manuelle Überprüfung des Cloud-Systems durch einen Auditor durchgeführt wird. Die Aufgaben der Auditoren beinhalten Folgendes:

- Überprüfung systemrelevanter Dokumentation
- Interview mit Stakeholdern
- Überprüfung von Anforderungen eines Zertifikatkatalogs

Sicherheitszertifizierung

Viele Cloud-Zertifikate legen einen Schwerpunkt auf Sicherheitseigenschaften von Cloud-Diensten, basierend z.B. auf der CCM oder dem C5. Da Cloud-Zertifikate möglichst auf eine Vielzahl von Cloud-Diensten anwendbar sein sollen, beschreiben die Sicherheitsanforderungen bzw. Zertifikatskriterien in der Regel Security-Best-Practices. Dabei handelt es sich um Praktiken, denen unabhängig von der konkreten, Cloud-basierten Applikation gefolgt werden sollte um eingesetzte Cloud-Ressourcen abzusichern.

Eine zentrale Voraussetzung für die sichere Nutzung von Cloud-Ressourcen besteht darin, dass die Nutzer ihre Dienste entsprechend ihrer Sicherheitsanforderungen konfigurieren. Die sichere Konfiguration der Cloud-Ressourcen ist damit maßgeblich für die Erfüllung von Sicherheitsanforderungen eines Zertifikats. Aufgrund der Vielzahl an Einstellungsmöglichkeiten ist die sichere Konfiguration von Cloud-Ressourcen allerdings nicht trivial und erfordert sowohl Know-how als auch Personalressourcen um Cloud-Dienste sicher – insbesondere auch rechtskonform – zu betreiben. Zudem lassen sich einmal vorgenommene Einstellungen durch autorisiertes Personal oft sehr leicht ändern.

Dies führt zu der Situation, dass sich die Sicherheitseigenschaften eines Cloud-Dienstes über die Zeit ändern können. Diese Veränderungen wiederum können dazu führen, dass zuvor erfüllte Sicherheitsanforderungen nicht mehr erfüllt werden und ein ausgestelltes Zertifikat im schlimmsten Fall die Gültigkeit verliert.

Vor diesem Hintergrund ist es folglich entscheidend, die Einhaltung von Zertifizierungskriterien von Cloud-Diensten durch geeignete Werkzeuge automatisiert und kontinuierlich sicherzustellen. Diese Werkzeuge detektieren kontinuierlich Veränderungen von Cloud-Ressourcen während ihres produktiven Betriebs und bewerten die Auswirkungen dieser Veränderungen auf die Erfüllung der Kriterien eines Zertifikates. Eine zentrale Herausforderung an dieser Stelle ist die Interpretation von Zertifikatskriterien mit dem Ziel, ihre technische Abbildung zu definieren und die Anforderungen auf diese Weise automatisiert überprüfbar zu machen.

5.3. Zertifizierung der BayernCloud

Dieses Kapitel befasst sich mit dem Zertifizierungsmodell für die BayernCloud.

Zertifizierungsmodell der BayernCloud

Da existierende Zertifizierungen lediglich das experten-basierte Zertifizierungsmodell verfolgen, ist eine solche Zertifizierung ebenfalls im Rahmen der BayernCloud zu empfehlen. Wie zuvor erläutert, ist es jedoch für dynamische Cloud-Services wie die BayernCloud zu empfehlen, ebenfalls eine dynamische und kontinuierliche Überprüfung möglichst vieler Zertifizierungskriterien durchzuführen. Umgesetzt wird eine solche Überprüfung durch das Tool Clouditor[4]. (siehe auch Abschlussbericht AP5: Zertifizierungsmodell/Dokumentation). Der Clouditor beinhaltet bereits eine Vielzahl von Tests, welche durch die Überprüfung einiger DSGVO-spezifischen Überprüfungen erweitert wird. Diese werden in den folgenden Abschnitten beschrieben.

Kontinuierliche Überprüfung: Clouditor

Clouditor ist ein Tool, das die Überprüfung von Sicherheitsmerkmalen in Cloud-Systemen unterstützt. Ziel ist es, kontinuierlich zu bewerten, ob eine Cloud-basierte Applikation sicher konfiguriert ist und damit Anforderungen, wie z.B. des Cloud Computing Compliance Controls Catalogue (C5) des BSI oder der Cloud Control Matrix (CCM) der Cloud Security Alliance (CSA) entsprechen. Außerdem können auch benutzerdefinierte Anforderungen definiert werden. Hierbei werden Merkmale bei verschiedenen Cloud-Anbietern, wie Amazon Web Services (AWS), Microsoft Azure und OpenStack, unterstützt.

Der Programm-Code wurde in Java entwickelt und seine funktionalen Komponenten bestehen im Wesentlichen aus drei Teilen:

1. Service Discovery: Clouditor nutzt die von den Cloud-Providern bereitgestellten APIs, um sich alle Cloud-Komponenten ausgeben zu lassen, die der Nutzer im Dashboard ausgewählt hat.
2. Regel-Definition: Um Regeln zu definieren, wird die Cloud Compliance Language (CCL) genutzt, die auf einer eigenen Grammatik basiert. Diese erlaubt es, deskriptiv Regeln zu erstellen, wie z.B. "VirtualMachine has dataDiskEncryption == 'Encrypted'", um auszudrücken, dass alle Daten-Disks aller virtuellen Maschinen verschlüsselt sein müssen.

[4] https://clouditor.io

3. Evaluierung: Clouditor nutzt den Parser-Generator ANTLR5, um die Regeln zu interpretieren und vergleicht sie mit den Ergebnissen aus der Service Discovery. Die Ergebnisse werden in einer User-Oberfläche dargestellt.

Die Architektur des Clouditor lässt sich ebenfalls in drei Komponenten einteilen, wie in Abbildung 45: Die Clouditor Toolbox, adaptiert von Stephanow (2018)Abbildung 45 dargestellt:

1. Das Clouditor Dashboard stellt die Benutzeroberfläche dar und erlaubt die Auswahl von zu evaluierenden Services sowie Regeln. Hier kann beispielsweise ausgewählt werden, dass der Speicher-Service S3 Teil der Überprüfung sein soll.
   Außerdem stellt das Dashboard detaillierte Ergebnisse der Evaluierung bereit.
2. Das kontinuierliche Tracking, führt in regelmäßigen Abständen Abfragen von Cloud Ressourcen, bzw. ihrer Konfigurationen, und speichert die Ergebnisse in eine Datenbank. Z.B. können hier Verschlüsselungskonfigurationen des Speicher-Services abgefragt werden.
3. Die kontinuierliche Verifikation vergleicht die Ergebnisse aus dem Tracking gegen erwartete Werte. Das Resultat wird wiederum auf dem Dashboard gezeigt.

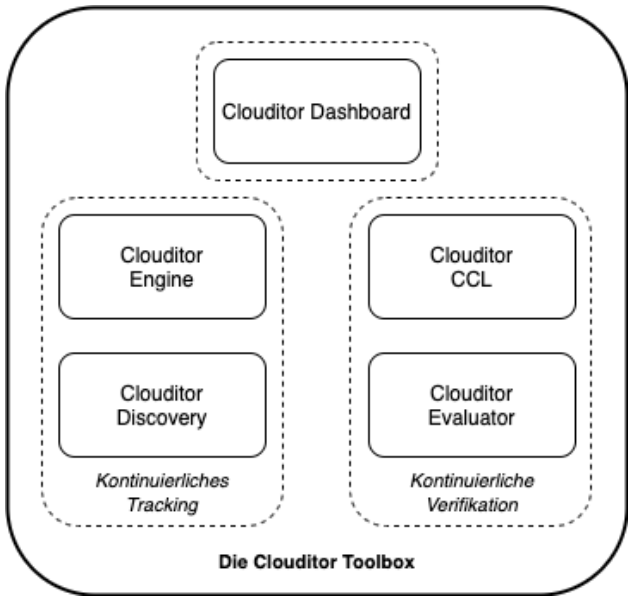

Abbildung 45: Die Clouditor Toolbox, adaptiert von Stephanow (2018)

Clouditor kann in zwei unterschiedlichen Szenarien verwendet werden. Auf der einen Seite können Sicherheits- und Compliance-Verantwortliche das Tool nutzen, um Sicherheitsanforderungen in einem Cloud-System nach angepassten Regeln zu überwachen. Auf der anderen Seite können Auditoren, die das Cloud-System zum Zwecke einer Zertifizierung auditieren, Clouditor nutzen, wenn sie bspw. nicht alle Ressourcen manuell prüfen können.

Zudem kann Clouditor als Docker Image nicht nur lokal, sondern in beliebigen Umgebungen, wie z.B. in einer Cloud-Umgebung, ausgeführt werden.

[5] https://www.antlr.org/

Audit API

Um eine wiederverwendbare Grundlage für die Integration des Clouditor zu schaffen, wurde eine API-Spezifikation erstellt [1]. Diese basiert auf dem OpenAPI-Standard 3.0[6],, der eine Sprachen-unabhängige Beschreibung von REST APIs spezifiziert.

Die API-Spezifikation beschreibt zunächst welche Pfade das System zur Verfügung stellen muss und welche Parameter in Anfragen an diese Pfade enthalten sein müssen. Außerdem wird die Struktur der Antworten, die das System geben muss, definiert. Hierdurch wird die Unabhängigkeit von der Implementierung vom zu testenden System von der Implementierung des testenden Systems sichergestellt (siehe Abbildung 464).

Diese Informationen können zudem genutzt werden, um automatisiert clients und stubs zu generieren.

Grundlage für die API-Spezifikation ist zunächst der Azure Benchmark des Center for Internet Security (CIS) [3]. Dieser beschreibt konkrete Konfigurationen in der Azure Cloud-Umgebung, die aber in der API-Spezifikation soweit generalisiert worden sind, dass sie auch auf die AWS Cloud angewendet werden können. Zudem sind Erfahrungswerte aus Cloud-Audits, die vom Fraunhofer AISEC durchgeführt worden sind, mit in die Spezifikation eingeflossen.

Die API umfasst folgende Cloud-Services: Storage, Identity- and Access Management (IAM), Container Services, Networking, Virtual Machines (VM), Logging, Security Center, Functions.

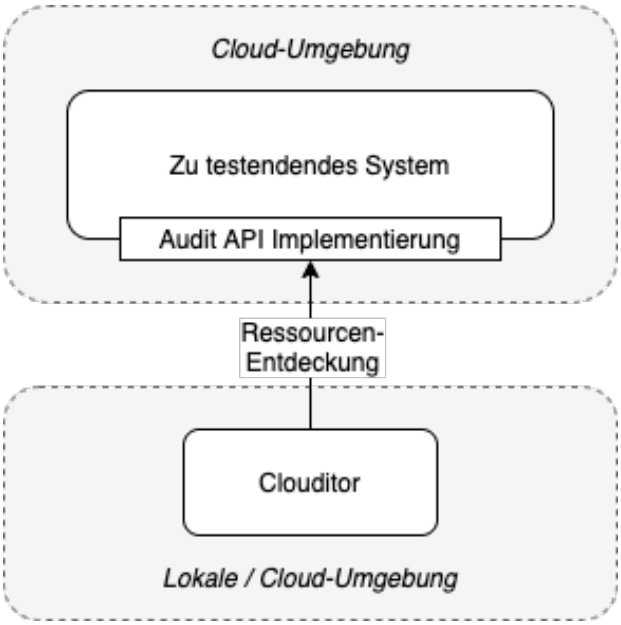

Abbildung 46: Zusammenspiel von Audit API und Clouditor

Eine solche Spezifikation bietet ferner den Vorteil, dass sie leicht erweiterbar ist, sollten sich Zertifizierungs- und individuelle Compliance-Kriterien ändern oder neue hinzukommen.

Des Weiteren werden DSGVO-spezifische Tests für die Überprüfung der BayernCloud vorgeschlagen. Diese wurden bereits im Dokument „BayernCloud Arbeitspaket 5: Zertifizierungsmodell/Dokumentation" vorgeschlagen und in den Clouditor integriert:

[6] https://swagger.io/specification/

- **Data replication geolocation check:** Dieser Test überprüft, ob sich Daten und Datenkopien bzw. Datenreplikationen in der gleichen geografischen Zone befinden (z.B in der EU).
- **Data replication check:** Dieser Test überprüft, ob für jedes replizierte *data bucket* und *data object* das zugehörige *source bucket* bzw. *source data* object existiert.
- **Data validity check:** Dieser Test überprüft, ob die vorhandenen Datenobjekte bezogen auf ihre Aufbewahrungsfristen noch existieren dürfen oder ob diese bereits gelöscht sein müssen.

5.4. Illustration Zertifizierungsmodell anhand der Use Cases

Im Folgenden wird das Zertifizierungsmodell inklusive der beteiligten Elemente und Akteure beschrieben und anhand einzelner Use Cases illustriert.

Elemente und Akteure

Die folgenden Elemente und Akteure sind im Zertifizierungsmodell involviert und in Abbildung 47 dargestellt:

- *System-under-test (SUT)*: Zertifizierungsobjekt ist die BayernCloud bzw. die Services der BayernCloud
- Auditor: Der (externe) Auditor bzw. die (externen) Auditoren auditieren das SUT, führen also die regelmäßige Überprüfung der Zertifizierungskriterien durch und vergeben das Zertifikat.
- Betreiber der BayernCloud
- Clouditor: Mit dem Clouditor wird die kontinuierliche Überprüfung eines Subsets der Zertifizerungskriterien durchgeführt. Zudem findet durch den Clouditor ein automatisiertes *Reporting* statt.

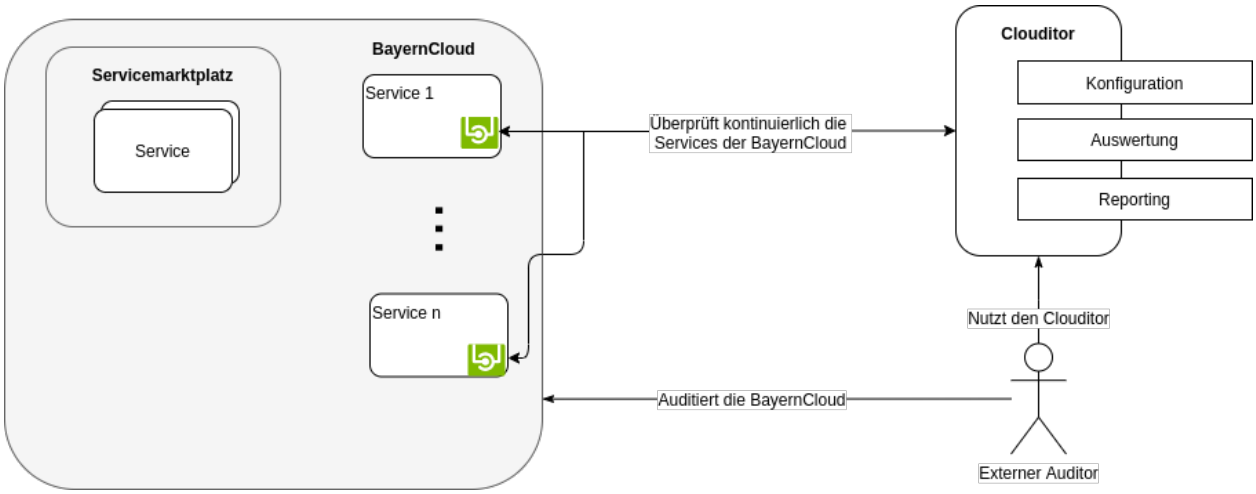

Abbildung 47: Beteiligte Elemente und Akteure des Zertifizierungsmodells der BayernCloud

Beispielhafte Anwendung

Im Folgenden wird ein beispielhaftes Szenario des Zertifizierungsmodells beschrieben.

Kontinuierliche Überprüfung

In diesem Beispiel wird angenommen, dass der Betreiber der BayernCloud den Clouditor einsetzt, um die Konfiguration der Zwei-Faktor-Authentifizierung in der BayernCloud zu prüfen.

0. Voraussetzung ist, dass der Clouditor mit den entsprechenden Berechtigungen ausgestattet ist. Hierzu gehört, dass eine Rolle mit den benötigten Berechtigungen angelegt wurden und credentials für diese Rolle dem Clouditor zur Verfügung gestellt wurden.
1. Um die Zwei-Faktor-Authentifizierung kontinuierlich prüfen zu können, wird in der Service-Übersicht im Clouditor zunächst die *User* Ressource ausgewählt. Folglich werden alle User des Cloud-Systems abgefragt.
2. Im nächsten Schritt wird eine Regel konfiguriert, die ausdrückt, dass jeder User die Zwei-Faktor-Authentifizierung aktiviert haben muss. In AWS kann dies bspw. durch folgenden Ausdruck definiert werden: *User has not empty mfaDevices*.
3. Hiernach wird Clouditor die User-Ressourcen automatisch mit der konfigurierten Regel evaluieren.
4. Angenommen, es existieren User im System, die die Multi-Faktor-Authentifizierung nicht aktiviert haben, wird der Betreiber nun im Dashboard darüber informiert und leitet entsprechende Schritte ein, informiert bspw. die betreffenden Personen.

Bezüge zu den Use Cases

Use Case 1 Daten lesen: Anhand der Zugriffsrechte kann z.B. überprüft werden, ob Zugriffsrechte gesetzt sind und inwieweit bzw. ob diese eingeschränkt sind (vgl. Need-to-know-Prinzip). Anhand der Datensicherheit kann überprüft werden, ob die Datenverschlüsselung aktiviert und eine sichere Datenübertragung durch HTTPS konfiguriert ist.

Use Case 3 Daten- und Datensevice-Rekombination: Teil der Zertifizierungs- und, soweit möglich, kontinuierlichen Prüfkriterien muss sein, dass die Herausgabe von personenbezogenen Daten bei der Rekombination von Daten und Datenservices DSGVO-konform ist. Dies beinhaltet bspw., dass personenbezogene Daten nur entsprechend der Einwilligung der Betroffenen weitergegeben werden.

Use Case 4 Drittentwickler Services bereitstellen: Alle Anwendungs-Container im Service Markplatz die Audit-API integrieren müssen. Dadurch ist gewährleistet, dass die Drittanbieter-Anwendungen ebenfalls in den kontinuierlichen Überprüfungs- und Auditierungsprozess mit einbezogen werden.

Use Case 5 Services verschieben: Obwohl das Zertifizierungsmodell und die entsprechenden Kriterien zunächst betreiberunabhängig sind, kann beim Verschieben von Services sowohl eine erneute experten-basierte Zertifizierung nötig sein, als auch die Rekonfiguration des Tools zur kontinuierlichen Überprüfung. Dies ist bedingt durch die Reichweite einer Zertifizierung, die möglicherweise auf den Cloud-Infrastrukturanbieter, bzw. die APIs, die dieser bereitstellt, beschränkt ist.

Prozess zur kontinuierlichen Zertifizierung

Wie beschrieben, ist eine statische Zertifizierung für sich ständig ändernde Cloud-Systeme nicht mehr ausreichend und es sollte eine kontinuierliche Überprüfung der Zertifizierungskriterien der Bayern Cloud Infrastruktur durchgeführt werden, deren Ablauf in Abbildung 48 dargestellt ist.

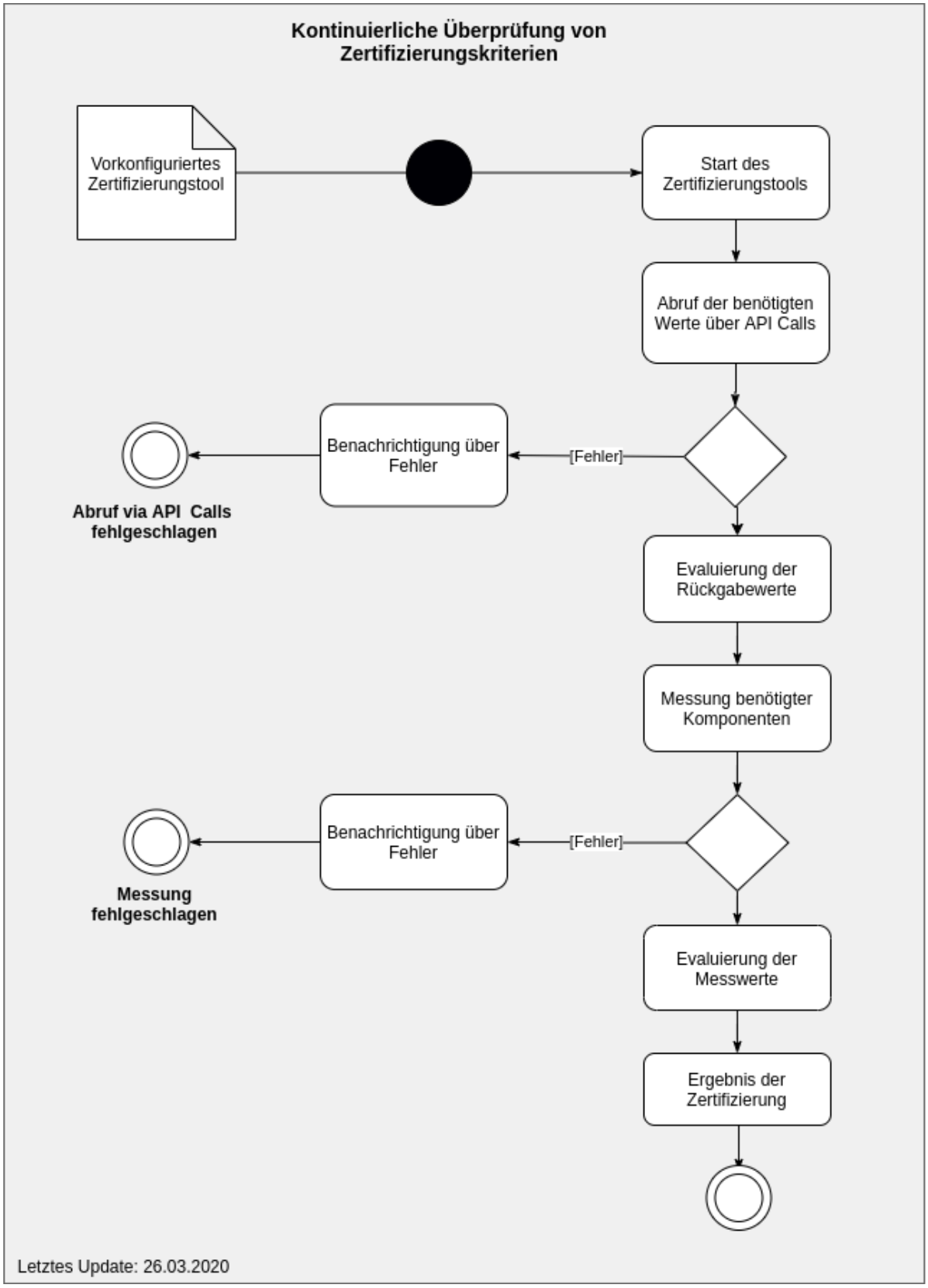

Abbildung 48: Ablauf der kontinuierlichen Überprüfung von Zertifizierungskriterien

Beschreibung: Durch die kontinuierliche Zertifizierung der Plattform soll gewährleistet werden, dass vordefinierte Eigenschaften eingehalten werden. Eigenschaften und

Anforderungen an die Plattform können sich ändern und erfordern daher eine toolgestützte kontinuierliche Überprüfung.

Vorbedingung: Vorkonfiguriertes Zertifizierungstool

Ablauf:

1. Starten des Zertifizierungstools (bspw. via cronjob)
2. Abruf der benötigtren Werte über API Calls
3. Ist ein Fehler aufgetreten?
    a. Ja
        i. Benachrichtigung über Fehler
        ii. **ENDE**
    b. *Nein*
        i. Evaluierung der Rückgabewerte
        ii. Messung benötigter Komponenten
            1. Ist ein Fehler aufgetreten?
                a. Ja
                    i. Benachrichtigung über Fehler
                    ii. **ENDE**
                b. Nein
                    i. Evaluierung der Messwerte
                    ii. Benachrichtigung über Ergebnis der Zertifizierung
                    iii. **ENDE**

6. Evaluation der Referenzarchitektur

Kurzübersicht des Kapitels

Dieses Kapitel umfasst alle Maßnahmen zur Evaluation der BayernCloud Referenzarchitektur. Die durchgeführten Evaluationsschritte dienen dazu, die verschiedenen Arbeitsergebnisse im Austausch mit Projektpartnern und Experten gesondert zu begutachten und durch entsprechendes Feedback anzupassen. Um eine zielführende Nutzung der Forschungsergebnisse zu gewährleisten, wurde für die finale Projektphase ein Konzept zur Validierung der BayernCloud Referenzarchitektur mit internen und externen Stakeholdern aufgesetzt (siehe Abschnitt 6.1). In Abschnitt 6.2 werden zudem weitere Aktivitäten zur Validierung der vorgestellten Konzepte und Wissensbausteine zusammengefasst.

6.1. Validierungskonzept

Abbildung 49 stellt das Validierungskonzept der BayernCloud Referenzarchitektur für den Zeitraum bis 09/2020 über einen Zeitraum von ca. sechs Monaten dar. Dabei erfolgt eine Validierung auf unterschiedlichen Ebenen und in verschiedenen Phasen.

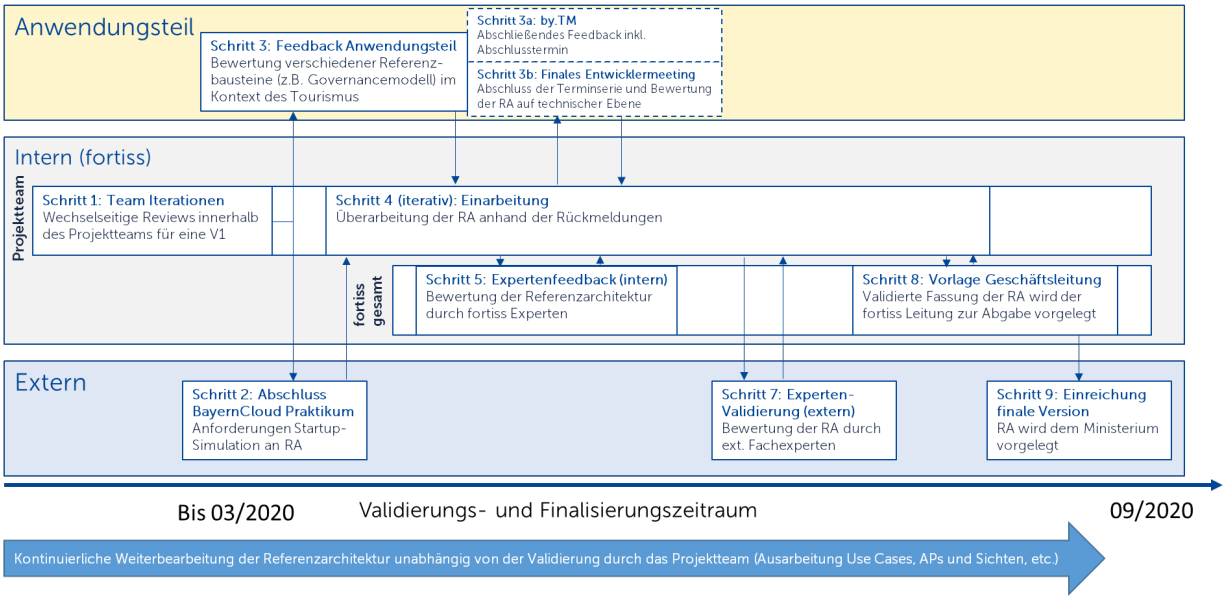

Abbildung 49: Validierungskonzept der BayernCloud Referenzarchitektur

Einerseits liegt der Fokus auf internen Validierungs- und Überarbeitungszyklen (grauer Bereich), andererseits werden durch den Anwendungsteil des Forschungsprojekts in der Pilotdomäne des bayerischen Tourismus (gelber Bereich) sowie durch weitere, unabhängige externe Akteure (blauer Bereich) gesonderte Rückmeldungen und Bewertungen eingeholt. Diese tragen durch eine iterative Einarbeitung zur finalen Version der BayernCloud Referenzarchitektur bei.

Tabelle 56 beschreibt die verschiedenen Phasen des Validierungskonzepts im Detail. Hierbei wird zu jedem Schritt eine kurze Durchführungsbeschreibung mit dem jeweiligen Fokus sowie dem geplanten Zeitraum aufgeführt.

Schritt	Durchführung	Geplanter Zeitraum bzw. Termine
1 – Team Iterationen	Initialer Review-Prozess in Team Iterationen: Zur Erstellung einer V1 der Referenzarchitektur wird das Dokument auf Ausdrucksweise, Rechtschreibung und Schlüssigkeit der verschiedenen Komponenten geprüft. Hierbei werden die einzelnen Abschnitte wechselseitig innerhalb des Teams zum Korrekturlesen vorgelegt. Die Ausdrucksweise und Formalitäten werden auf Basis zuvor abgestimmter Standards begutachtet (z.B. einheitliche Verwendung von Abkürzungen) Fokus: Korrekte, einheitliche Ausdrucksweise; Rechtschreibung; Schlüssigkeit	Bis März 2020
2 – Abschluss BayernCloud Praktikum	In diesem Schritt wird die erste Fassung der Referenzarchitektur durch die Nutzungssimulation mit sieben Teams einer Lehrveranstaltung an der TU München validiert. Hierbei simulieren die einzelnen Teams jeweils ein domänenspezifisches Startup, welches auf Basis der BayernCloud eigene Anwendungen für die Branche bereitstellen soll. Die Erkenntnisse aus diesem Schritt sind insbesondere in Bezug auf die für Anwender wichtigsten Daten und Services relevant. Fokus: Validierung durch Praxis-Simulation	Finale Abgabe Februar 2020, Bewertungszeitraum bis April 2020
3 – Feedback Anwendungsteil	Parallel zu Schritt 2 wird die erste Fassung der Referenzarchitektur bzw. einzelner Referenzbausteine (Governancemodell, Geschäftsmodell, Architektur etc.) dem Anwendungsteil	März 2020 bis Mai 2020 - Bereitstellung RA V1 für Anwendungsteil

	zur Verfügung gestellt (Mitte April 2020). Hierdurch soll eine Validierung der Referenzarchitektur im Kontext der Pilotdomäne (Tourismus) erfolgen und die Vorteile und Eignung der generischen Komponenten bewertet werden. Vorarbeiten hierzu sind bereits über den Austausch über die gesamte Projektlaufzeit erfolgt, sodass in diesem Schritt ein abschließendes Feedback angestrebt wird. Fokus: (Abschließende) Validierung durch Pilot-Domäne (siehe auch 3a und 3b)	ab Ende März 2020
3a – by.TM	Als Teil der abschließenden Validierung durch die Pilotdomäne fällt insbesondere ein abschließender Termin mit der BAYERN TOURISMUS Marketing GmbH (by.TM) an (Ende Mai 2020). Fokus: Validierung durch Tourismus Experten	Mai 2020 - Abschließender Feedbacktermin by.TM Ende Mai 2020
3b – Finales Entwicklermeeting	Als Abschluss der Terminserie wird ein finales „Entwicklermeeting" mit den technischen Partnern des Anwendungsteils zur Bewertung Referenzarchitektur vereinbart. Im Vordergrund hierbei stehen die technischen Komponenten der Referenzarchitektur. Fokus: Validierung durch technische Praxispartner	Mai 2020 - Abschließender Feedbacktermin technische Partner (Outdooractive, Hubermedia)
4 – Einarbeitung	Iterative Einarbeitung der Erkenntnisse: Durch verschiedene Rückmeldungen (fortiss, Anwendungsteil, extern) werden Anpassungen an der Referenzarchitektur vorgenommen. Das BayernCloud Team wird die kontinuierliche Verbesserung des Dokuments dokumentieren.	April 2020 bis August 2020

	Fokus: Anpassungen der Referenzarchitektur und Vorbereitung für finale Einreichung	
5 – Expertenfeedback (intern)	Expertenfeedback fortiss: Die Referenzarchitektur wird auf fortiss-Ebene (außerhalb des Projektteams) zum Review zur Verfügung gestellt. Das Feedback erfolgt durch fortiss-Mitarbeiter, die weitreichende Erfahrung mit Referenzarchitekturen vorweisen können. Fokus: Abgleich mit vorhandener Expertise zum Thema Referenzarchitektur	Mai 2020 - Nach Abschluss der Validierung im Anwendungsteil
7 – Experten-Validierung (extern)	Validierung durch externe Experten: Im Anschluss an die interne Validierung, wird die Referenzarchitektur einer externen Begutachtung durch Experten unterzogen. Diese Beurteilung soll eine unabhängige Bewertung der Referenzarchitektur gewährleisten. Fokus: Validierung für finale Einreichung	Juli 2020
8 – Vorlage Geschäftsleitung	Die auf Basis aller Rückmeldungen konsolidierte Referenzarchitektur wird zusammen mit einer Zusammenfassung der wichtigsten Rückmeldungen im Validierungsprozess an die Geschäftsführung zur Durchsicht übermittelt. Letzte Anmerkungen können vor der finalen Einreichung an das Projektteam gegeben werden. Fokus: Letzte Anpassungen	Anfang September 2020
9 – Einreichung finale Version	Nach Abschluss aller Validierungsschritte, wird die finale Referenzarchitektur an das Ministerium übermittelt. Fokus: Projektabschluss	September 2020

Tabelle 56: Beschreibung der verschiedenen Phasen des Validierungskonzepts

6.2. Kontinuierliche Evaluation der Referenzarchitektur

Während das in Abschnitt 6.1 vorgestellte Konzept insbesondere die Validierung der Referenzarchitektur in der finalen Projektphase abdeckt, werden in diesem Abschnitt sonstige Maßnahmen vorgestellt, die zur Evaluation der Projektergebnisse bereits während der Entwicklung der Referenzarchitektur beitragen. Dementsprechend wurden die verschiedenen Wissensbausteine und Inhalte der Referenzarchitektur im Rahmen ihrer Entwicklung über die gesamte Projektlauzeit durch begleitende Aktivitäten kontinuierlich evaluiert. Hierunter fallen die im Folgenden vorgestellten und kurz zusammengefassten Evaluationsmaßnahmen:

1. **Durchführung von Projektmeetings**
   Um einen stetigen Austausch zu den durchgeführten Aktivitäten und den inhaltlichen Zwischenständen der Referenzarchitektur sowie weiteren Projektergebnissen zu gewährleisten, wurden über die gesamte Laufzeit des Projekts regelmäßige Projektmeetings angesetzt. Dies diente einerseits der gegenseitigen Abstimmung innerhalb der Grundlagenforschung mit dem Projektpartner des Fraunhofer AISEC zur Besprechung von (Teil-)Ergebnissen, der Vereinbarung nächster Schritte und der Planung sonstiger Tätigkeiten. Separat gab es einen regelmäßigen Austausch mit den Projektpartnern aus dem Anwendungsteil, um den Bezug der Grundlagenforschung zum Anwendungsteil und der Zieldomäne sicherzustellen. Dieser Austausch ermöglichte zudem die Planung und Umsetzung einer Reihe von gemeinsamen Aktivitäten und öffentlichkeitswirksamen Veranstaltungen im Tourismus.

2. **Hackathons, Lehrtätigkeiten an der Technischen Universität München sowie wissenschaftliche Veröffentlichungen**
   Um die Rolle von Drittentwicklern, z.B. in Form von touristischen IT-Startups zu simulieren, wurde auf die Möglichkeiten im wissenschaftlich / universitären Umfeld der TU München zurückgegriffen. Einerseits konnte durch die Teilnahme an einem Hackathon (*HackaTUM 2019*) eine Challenge zum Thema „Smart Winter Tourism" gestellt werden, welche neben einer öffentlichen Positionierung der BayernCloud auch die Möglichkeit für die Entwicklung prototypischer Anwendungen in der Domäne ermöglichte. Darüber hinaus wurden durch praxisorientierte Lehrveranstaltungen an der TU München tourismusspezifische Themen bearbeitet. Die hierbei entstandenen Apps und Services der simulierten IT-Startups adressieren verschiedene Probleme oder Potenziale in der Branche und werden in Teilen sogar über die Lehrveranstaltung hinaus weiterentwickelt. Die Simulation verdeutlichte die Relevanz hinsichtlich der Bereitstellung spezifischer Domänendaten innerhalb des BayernCloud DPÖs.

   Neben den Aktivitäten im universitären (Lehr-)Umfeld, wurden im Projektkontext wissenschaftliche Veröffentlichungen zu verschiedenen Themen der Referenzarchitektur erarbeitet. Hierunter fallen etwa eine Analyse des europäischen Tourismus-Ökosystems, eine kontinuierliche Standortvalidierung von Cloudservice-Komponenten sowie weitere Arbeiten in Bereichen wie Plattform-Governance, Ökosystem-Analysen oder digitalen Geschäftsmodellen. Alle Veröffentlichungen unterziehen sich hierbei einer Begutachtung im wissenschaftlichen Umfeld und tragen durch ihre fundierte Ausarbeitung zur Evaluation der Projektinhalte bei.

3. **Durchführung von Entwicklerworkshops**
   Mit dem Fokus auf die technischen Inhalte (wie etwa die technische Architektur, domänenspezifische Datenschemata oder die testweise Bereitstellung von Software-Prototypen) wurden Entwicklerworkshops mit den technischen Anwendungspartnern aus dem Tourismus durchgeführt. Dieser Informationsaustausch ermöglichte einerseits den Wissenstransfer in den Anwendungsteil ebenso wie eine Evaluation der prototypischen Entwicklungen, die zum Testen der konzeptionellen Vorarbeiten erarbeitet wurden.

4. **Interviews in der Pilotdomäne des bayerischen Tourismus**
   Zur Einschätzung der Datenlage und -bestände sowie konkreter Anforderungen in der Pilotdomäne des bayerischen Tourismus wurde eine Interviewreihe mit Stakeholdern aus der Praxis angesetzt. Zu diesem Zweck wurden etwa regionale Verbände, Dienstleister oder Vereine kontaktiert und telefonisch interviewt. Hierdurch wurden bspw. wichtige Datentypen, die Datenlage, die Bereitschaft zum Teilen von Daten ebenso wie technische oder auch rechtliche Aspekte und Herausforderungen in der Domäne beleuchtet und für die weitere Entwicklung transparenter gemacht.

5. **Implementierung**
   Die technischen Konzepte der BayernCloud Referenzarchitektur (vergleiche insbesondere die Abschnitte 4.3 und 4.5) wurden mithilfe prototypischer Implementierungen unter Berücksichtigung von domänenspezifischen Anforderungen testweise evaluiert. Hierunter fallen u.a. prototypisch entwickelte, domänenspezifische Services mit passender Datenarchitektur, eine graphische Interaktionsoberfläche zum Verwalten und Monitoren von domänenspezifischen Plattform-Services oder infrastrukturnahe Komponenten zur Container-Orchestrierung. Weiterhin wurde eine illustrative, domänenspezifische Demo-Anwendung entwickelt und u.a. im Rahmen der fortiss Fachtagung, des HackaTUM der Technischen Universität München sowie der ICIS Konferenz in München einem jeweils hochrangigen Fachpublikum präsentiert. Alle entwickelten Prototypen tragen als „Proof of Concept" zur Evaluation der technischen Ergebnisse bei.

6. **Vergleich mit bestehenden Arbeiten und Austausch mit (Domänen-)Experten**
   Eine Auseinandersetzung mit bestehenden Arbeiten aus Praxis und Wissenschaft ist ein wesentlicher Bestandteil der Erarbeitung von Wissensbausteinen dieser Referenzarchitektur. Dies dient einerseits der methodischen Entwicklung der Referenzarchitektur als Projektergebnis (z.B. durch einen Vergleich mit bestehenden Publikationen wie in Abschnitt 3.2.1 vorgestellt), ebenso wie aller bereitgestellten Inhalte. Entsprechend wurden alle Ergebnisse im Kontext eines wissenschaftlich fundierten Rückblicks, d.h. unter Berücksichtigung des wissenschaftlichen Status Quo erarbeitet und aufbereitet. Auch der Austausch mit Experten aus angrenzenden Themengebieten wurde durch die aktive Teilnahme an Veranstaltungen oder entsprechende Kontaktaufnahmen gefördert (z.B. mit Alpine Bits, DZT, DACH-KG oder die Teilnahme an den Allgäuer Tourismusgesprächen).

7. Zusammenfassung und Ausblick

Durch die Digitalisierung sehen sich Unternehmen aller Größen und in verschiedenen Branchen mit großen Herausforderungen konfrontiert. Die Bereitstellung digitaler Produkte, Services und Angebote ist zu einem wesentlichen Bestandteil heutiger Wertschöpfung geworden. Daten und Informationen werden in einer stetig zunehmenden Menge, Vielfalt und Geschwindigkeit bereitgestellt, ausgetauscht und für immer neue Anwendungsbereiche verwertet. Besonders spürbar werden die Herausforderungen für kleine und mittlere Unternehmen. Diesen fehlen in der Regel die Fähigkeiten und Ressourcen, um selbstständig vollwertige, marktfähige digitale Lösungen zu entwickeln und bereitzustellen. Die Angebote, die sie in stark umkämpften Märkten machen, können von potentiellen Kunden im Wettbewerb nur schwer wahrgenommen und identifiziert werden.

Insbesondere in sehr heterogenen Branchen, wie dem Tourismus in Bayern wirkt sich diese Situation stark aus. Die Heterogenität begründet sich aus der Struktur der Branche aus vielen kleinen und mittleren Betrieben, die etwa Übernachtungen, Aktivitäten oder gastronomische Dienstleistungen anbieten. Diese Betriebe sind in Bayern oft Familienbetriebe, die sich wiederum in lokalen und regionalen Verbänden zusammenschließen. Diese Verbände auf Ebene von Kommunen oder Landkreisen wiederum sind in mehreren größeren bayerischen Tourismusverbänden organisiert. Entsprechend dieser Aufteilung ist der Stand der Digitalisierung der Verbände und Betriebe sehr unterschiedlich. Um dem Tourismus in Bayern, stellvertretend für andere heterogene Branchen wie der Landwirtschaft oder dem Handwerk eine Lösung zur Verfügung zu stellen, die es den Betrieben und/oder den Verbänden ermöglicht digitale Dienste anzubieten und zu konsumieren, wurde das Forschungsvorhaben BayernCloud geschaffen.

Durch eine zentrale, offene digitale Infrastruktur werden gezielt die Bedürfnisse dieser Betriebe und Verbände adressiert. Diese Infrastruktur kann in unterschiedlichen Ausbaustufen geschaffen werden. Basierend auf einer Plattform zum Austausch von offenen Daten im Tourismus können in einem ersten Schritt bereits Anwendungen von Drittanbietern geschaffen werden, die auf eine einheitliche Datenbasis zurückgreifen. Schließlich kann diese Plattform iterativ um zusätzliche Services zur Datenrekombination erweitert werden, um zusätzliche Funktionalitäten abzubilden.

Zum besseren Verständnis für die Anforderungen und Lösungsmöglichkeiten einer solchen Plattform zu erlangen, wurde im Grundlagenprojekt BayernCloud eine Referenzarchitektur erarbeitet. Diese Referenzarchitektur erleichtert zum einen die Entwicklung eines solchen Systems, indem generische und spezifische Anforderungen erfasst und in die Grundkonzepte der Architektur übertragen wurden. Hierdurch ist es für den Anwender möglich, Module für die individuellen Bedürfnisse der Branche auszuwählen und anhand dieser ein System zu implementieren. Zudem trägt die für diese Referenzarchitektur neu konzipierte Ökosystemsicht dazu bei, dass die Referenzarchitektur auch dafür genutzt werden kann, den Entwicklungsprozess und das Anforderungsmanagement unter Einbindung aller Stakeholder durchzuführen und bereits frühzeitig Aspekte der Verstetigung hinsichtlich Geschäfts- und Governancemodell einzubeziehen.

Zur genaueren Erläuterung der BayernCloud Referenzarchitektur wurde diese in diesem Beitrag in mehreren Schritten eingeführt. Zunächst wurde die Notwendigkeit einer zentralen Lösung skizziert und aktuelle Themen im Kontext digitaler Plattformen dargelegt. Anschließend wurde auf Vorarbeiten im Bereich Referenzarchitekturen eingegangen, die

Begrifflichkeit kurz erläutert und die vorliegende Referenzarchitektur abgegrenzt sowie Schnittstellen aufgezeigt. Die Referenzarchitektur basiert auf 4+1 Sichten, welche auf der Schichtenarchitektur von Kruchten aufbauen und so konzipiert sind, dass der generische Charakter einer Referenzarchitektur beibehalten wird. Weiterhin sind diese Sichten so definiert, dass sie als Mittel der Kommunikation innerhalb eines Entwicklungsprozesses dienen können, ohne dass tiefgehendes Detail- oder Notationswissen zum Verständnis nötig ist. Dabei ist jede Sicht auf einen unterschiedlichen Aspekt des zu entwickelnden Systems ausgerichtet, wodurch sie verschiedene Perspektiven ermöglicht. Im Anschluss wurde die BayernCloud Referenzarchitektur mit Hilfe der Sichten dargestellt und detailliert erläutert. Die Kernsicht bildet die Use Case Sicht, welche generische Use Cases beschreibt, die wiederum auf verschiedene Domänen adaptierbar sind. Neben den Use-Cases werden technische Module beschrieben, die als Grundlage einer Implementierung genutzt werden können. Anhand der Module und der Use Cases lassen sich rückverfolgbare Anforderungen ableiten. Die Verteilungssicht der Referenzarchitektur wiederum strukturiert die Abbildung von Softwarekomponenten auf Hardware innerhalb der Architektur. Zudem umfasst die Referenzarchitektur eine Prozesssicht, welche die Module, technischen Komponenten und Use Cases so verbindet, dass konkrete und generische Prozesse abgeleitet werden können. Ergänzt werden die technischen Perspektiven auf die Architektur durch die Ökosystemsicht, die organisatorische Aspekte einer entstehenden Plattform auf generische Weise aufgreift und beschreibt. Dazu gehören Fragen der Rechteverteilung sowie Anleitungen zur Entwicklung von Geschäftsmodellen basierend auf den Use Cases.

Die BayernCloud stellt die Referenzarchitektur für eine digitale Plattform mit Fokus auf den Anforderungen des Bayerischen Mittelstands dar. Anhand der Domäne Tourismus wurde gezeigt, wie der Kern und optionale Module eines solchen übergreifenden Informationssystems ausgestaltet werden können. Entsprechend dient die BayernCloud der Entwicklung und Anforderungsspezifikation einer digitalen Plattform für heterogene Domänen in Bayern zu vereinfachen und zu strukturieren. Die dargestellten Komponenten und Prozesse können als Blaupause genutzt werden, um ein angepasstes System zu erstellen. Die simplifizierte Darstellung von Aspekten des Systems ermöglicht es somit Stakeholdern verschiedenster Domänen gemeinschaftlich an einer derartigen Plattform zu arbeiten, um angepasste Lösungen für Nutzer der Domäne zu erstellen.

Die vorgestellte Referenzarchitektur stellt lediglich einen ersten wichtigen Grundstein für eine Entwicklung von innovativen Lösungen verschiedener Domänen dar. Die gezeigte Architektur ist weitestgehend technologieneutral und besitzt einen hohen Abstraktionsgrad, um die Bedürfnisse unterschiedlicher Einsatzbereiche zu adressieren. Konkrete Lösungen und Standards müssen für die Implementierung definiert und ergänzt werden. So kann die vorgestellte Referenzarchitektur iterativ konkretisiert werden. Als nächsten Schritt empfiehlt sich eine erste Instanziierung beziehungsweise Umsetzung der BayernCloud Referenzarchitektur, die wiederum als Beispiel für andere Domänen dienen kann. Entsprechend kann eine Implementierung der BayernCloud im Tourismus nicht nur für den Tourismus in Bayern vorteilhaft sein, sondern auch für weitere Domänen als Vorlage dienen.

8. Referenzen

References

Abu-Matar, M., & Davies, J. (2017). Data driven reference architecture for smart city ecosystems. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation,* San-Francisco, California, USA.

Albelooshi, B., Damiani, E., & Salah, K. (2015). Experimental Proof: Data Remanence in Cloud VMs. In *8th International Conference on Cloud Computing.* Symposium conducted at the meeting of IEEE, New York, USA.

Angelov, S., & Grefen, P. (2008). An e-contracting reference architecture. *Journal of Systems and Software*, *81*(11), 1816–1844. https://doi.org/10.1016/j.jss.2008.02.023

Anisetti, M., Argostino Ardagna, C., Damiani, E., & Gaudezini, F. (2016). A Certification Framework for Cloud-Based Services. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing.* Symposium conducted at the meeting of Association for Computing Machinery, New York, NY, USA.

Benaboud, R., Maamri, R., & Sahnoun, Z. (2013). Agents and OWL-S Based Semantic Web Service Discovery With User Preference Support. *International Journal of Web & Semantic Technology*, *4*(2), 57–75. https://doi.org/10.5121/ijwest.2013.4206

Berners-Lee, T., Hendler, J., & Lassila, O. (284). THE SEMANTIC WEB. *Scientific American*, *2001*(5), 34–43.

Cheng, H. K., & Liu, Y. (2012). Optimal Software Free Trial Strategy: The Impact of Network Externalities and Consumer Uncertainty. *Information Systems Research*, *23*(2), 488–504. https://doi.org/10.1287/isre.1110.0348

Cockburn, A. (2006). *Writing effective use cases* (16. print). *The Agiel software development series*. Boston: Addison-Wesley.

Currie, W. (2004). *Value creation from e-business models*. Amsterdam, Boston: Elsevier Butterworth Heinemann. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=122004

D'Amato, C. (2020). Machine learning for the semantic web: Lessons learnt and next research directions. *Semantic Web Journal*, *Preprint*, 1–9.

Dapp, M. M., Balta, D., Palmetshofer, W., & Krcmar, H. (2016). Open Data. The Benefits: Das volkswirtschaftliche Potential für Deutschland.

Eisenmann, T. R., Parker, G. G., & van Alstyne, M. W. (2008). Opening Platforms: How, When and Why? *SSRN Electronic Journal.* Advance online publication. https://doi.org/10.2139/ssrn.1264012

Engert, M., Pfaff, M. [Matthias], & Krcmar, H. (2019). Adoption of Software Platforms: Reviewing Influencing Factors and Outlining Future Research. In *Twenty-Third Pacific Asia Conference on Information Systems: PACIS 2019,* Xi'An, China.

Ghazawneh, A., & Henfridsson, O. (2013). Balancing platform control and external contribution in third-party development: The boundary resources model. *Information Systems Journal*, *23*(2), 173–192. https://doi.org/10.1111/j.1365-2575.2012.00406.x

Handschuh, S., & Staab, S. (Eds.) (2003). *Frontiers in artificial intelligence and applications: Vol. 96. Annotation for the semantic web*. Amsterdam: IOS Press.

Handschuh S., S. S. (Ed.) (2003). *Annotation for the semantic web*. Amsterdam: IOS Press.

Hein, A., Schreieck, M., Riasanow, T., Setzke, D. S., Wiesche, M., Böhm, M., & Krcmar, H. (2020). Digital platform ecosystems. *Electronic Markets*, *30*(1), 87–98. https://doi.org/10.1007/s12525-019-00377-4

Immonen, A., Palviainen, M., & Ovaska, E. (2014). Requirements of an Open Data Based Business Ecosystem. *IEEE Access*, *2*, 88–103. https://doi.org/10.1109/ACCESS.2014.2302872

Jaramillo, D., Nguyen, D. V., & Smart, R. (2016). Leveraging microservices architecture by using Docker technology. In *IEEE SoutheastCon 2016,* Norfolk, VA March.

Kiswani, J., Dascalu, S. M., & Harris Jr, F. C. (2018). Cloud-RA: A Reference Architecture for Cloud Based Information Systems. In *ICSOFT 2018*.

Kontio, M. (2005). *Architectural manifesto: Designing software architectures, Part 5: Introducing the 4+1 view model.*

KPMG (2019). Cloud-Monitor 2019 - Interaktiv: Zahlen, Daten, Fakten: Das Präsentationstool zeigt interaktiv die Ergebnisse unserer Cloud Monitor 2019 Umfrage. Retrieved from https://home.kpmg/de/de/home/themen/2019/06/cloud-monitor-2019-interaktiv.html

Krcmar, H., & Leimeister, S. (2010). *CIO-Umfrage 2010 – Selbstverständnis und Themenschwerpunkte der IT-Entscheider im Jahr 2010*. München, Deutschland.

Krcmar, H., Leimeister, J. M., Roßnagel, A., & Sunyaev, A. (2016). *Cloud-Services aus der Geschäftsperspektive*. Wiesbaden: Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-08257-4

Krotsiani, M., Spanoudakis, G., & Mahbub, K. (2013). Incremental certification of cloud services. In *7th International Conference on Emerging Security Information, Systems and Technologies* (pp. 72–80). Rome, Italy.

Kubernetes (2020). Pod Overview: Understanding Pods, Working with Pods, Pod Templates. What's next. Retrieved from https://kubernetes.io/docs/concepts/workloads/pods/

Liu, C., Lin, J., & Fang, B. (2013). T-YUN: Trustworthiness Verification and Audit on the Cloud Providers. *IEICE Transactions on Information and Systems*, *11*(Vol. E96.D.), 2344–2353.

Mell, P. M., & Grance, T. (2011). *The NIST definition of cloud computing*. Gaithersburg, MD. https://doi.org/10.6028/NIST.SP.800-145

Mumm, S. (2017). *Handlungsempfehlungen für die Konzeption, Entwicklung und Etablierung eines plattformbasierten Business Ecosystems auf Basis einer Langzeitintervention bei einem KMU-Softwarehersteller: Dissertation zur Erlangung des Doktorgrades an der Fakultät für Mathematik, Informatik und Naturwissenschaften Fachbereich Informatik der Universität Hamburg*. Hamburg, Germany.

Oliveira de Carvalho, J., Trinta, F., & Vieira, D. PacificClouds: A Flexible MicroServices based Architecture for Interoperability in Multi-Cloud Environments. In *Proceedings ofthe 8th International Conference on Cloud Computing and Services Science (CLOSER 2018)* (pp. 448–455). Funchal, Madeira - Portugal. https://doi.org/10.5220/0006705604480455

Osterwalder, A., & Pigneur, Y. (2010). Business Model Generation. *07376782.* Advance online publication. https://doi.org/10.1523/JNEUROSCI.0307-10.2010

Osterwalder, A., Pigneur, Y., Bernarda, G., Smith, A., & Papadakos, T. (2014). *Value Proposition Design: How to Create Products and Services Customers Want*. Hoboken NJ: John Wiley & Sons, Ltd.

Otto, B., Steinbuß, S., Teuscher, A., & Lohmann, S. (2019). *Reference Architecture Model: Version 3.0*. Dortmund: International Data Spaces Association.

Pham, C., Chen, D., Kalbarczyk, Z., & Iyer, R. (2011). CloudVal: A framework for validation of virtualization environment in cloud infrastructure. In *41st International Conference on Dependable Systems Networks (DSN).* Symposium conducted at the meeting of IEEE/IFIP, Hong Kong, China.

Reichwald, R., & Piller, F. T. (2009). *Interaktive Wertschöpfung: Open Innovation, Individualisierung und neue Formen der Arbeitsteilung* (2., vollständig überarbeitete und erweiterte Auflage). Wiesbaden: Gabler Verlag / GWV Fachverlage GmbH Wiesbaden. https://doi.org/10.1007/978-3-8349-9440-0

Reidt, A. (2019). *Referenzarchitektur eines integrierten Informationssystems zur Unterstützung der Instandhaltung* (Dissertation). Technische Universität München, München.

Reidt, A., Pfaff, M. [M.], & Krcmar, H. (2018). Der Referenzarchitekturbegriff im Wandel der Zeit. *HMD Praxis Der Wirtschaftsinformatik*, *55*(5), 893–906. https://doi.org/10.1365/s40702-018-00448-8

Schallmo, D. R. A. (2013a). Geschäftsmodelle erfolgreich entwickeln und implementieren. *01672681.* Advance online publication. https://doi.org/10.1007/978-3-642-37994-9

Schallmo, D. R. A. (2013b). *Geschäftsmodellinnovation: Grundlagen, bestehende, Ansätze, methodisches Vorgehen und B2B-Geschäftsmodelle*. Wiesbaden: Springer.

Schiffmann, J., Sun, Y., Vijayakumar, H., & Jaeger, T. (2013). Cloud Verifier: Verifiable Auditing Service for IaaS Clouds. In *Proceedings of the 2013 IEEE Ninth World Congress on Services.* Symposium conducted at the meeting of IEEE Computer Society, USA.

Schneider, S., & Sunyaev, A. (2015). *Cloud-Service Zertifizierung: Ein Rahmenwerk und Kriterienkatalog zur Zertifizierung von Cloud-Services*. Berlin: Springer Gabler.

Schreieck, M., & Wiesche, M. (2019). Value Cocreation and Value Capture in Digital Platforms. *Academy of Management Proceedings*, *2019*(1). https://doi.org/10.5465/AMBPP.2019.10450abstract

Schreieck, M., Wiesche, M., & Krcmar, H. (2016). Design and Governance of Platform Ecosystems – Key Concepts and Issues for Future Research. In *Twenty-Fourth European Conference on Information Systems: ECIS 2016,* Istanbul, Turkey.

Sporny, M., Longley, D., Kellogg, G., Lanthaler, M., & Lindström, N. (2014). JSON-LD 1.0: A JSON-based Serialization for Linked Data. *W3C Recommendation*, *16*, 1–33.

Stephanow, P., & Banse, C. (2018). Ansatz der dynamischen Zertifizierung. In H. Krcmar, C. Eckert, A. Roßnagel, A. Sunyaev, & M. Wiesche (Eds.), *Management sicherer Cloud-Services*. Wiesbaden: Springer Fachmedien Wiesbaden.

Stickdorn, M., & Schneider, J. (2011). *This is service design thinking: Basics, tools, cases*. Hoboken, N.J.: Wiley.

Storbacka, K. (2019). Actor engagement, value creation and market innovation. *Industrial Marketing Management*, *80*, 4–10. https://doi.org/10.1016/j.indmarman.2019.04.007

Sunyaev, A., & Schneider, S. (2013). Cloud services certification. *Communications of the ACM*, *56*(2), 33. https://doi.org/10.1145/2408776.2408789

Tiwana, A. (2014). *Platform ecosystems: Aligning architecture, governance, and strategy*. Amsterdam, Waltham, MA: MK.

Vom Brocke, J. (2015). *Referenzmodellierung*. Dissertation (2., unveränderte Auflage). *Advances in information systems and management science: Band 4*.

Wilhelm Hasselbring and Guido Steinacker (2017). Microservice Architectures for Scalability, Agility and Reliability in E-Commerce. In *Proceedings 2017 IEEE International Conference on Software Architecture Workshops*.

Xiang, S., Zhao, B., Yang, A., & Wei, T. (2014). Dynamic measurement protocol in infrastructure as a service. *Tsinghua Science and Technology*. (19), 470–477.

Zuiderwijk, A., Janssen, M., & Davis, C. (2014). Innovation with open data: Essential elements of open data ecosystems. *Information Polity*, *19*(1,2), 17–33. https://doi.org/10.3233/IP-140329

9. Anhang

9.1. BayernCloud Glossar

Das Glossar beinhaltet eine Aufführung wichtiger Begriffe, die meist bereits in den Forschungsanträgen des Grundlagen- und Anwendungsteils genannt werden. Das Glossar dient dazu ein grundsätzliches und allgemeingültiges Verständnis zu bestimmten Begrifflichkeiten und Sachverhalten zu schaffen.

Term	Beschreibung	Bezug zum Antrag
(Gesamt-) Referenzarchitektur	Die (Gesamt-) Referenzarchitektur für ein DPÖ ist das zentrale Ergebnis des BayernCloud Grundlagenteils. Ihre Erstellung erfolgt durch kontinuierliche Integration der Referenzbausteine anhand einer angepassten Version des 4+1 Sichtenmodells nach Kruchten (1995) und Reidt et al. (2019).	
3rd-Party Service	Technische Services mit oder ohne User Interface, die von Drittanbietern über die Infrastruktur einer Instanziierung eines DPÖ angeboten werden. Diese Services laufen innerhalb der technischen Infrastruktur und werden anderen zur kostenpflichtigen und/oder kostenlosen Nutzung freigegeben.	
Drittentwickler / 3rd-Party-Developer	Die Integration externer Entwickler ist ein zentraler Bestandteil künftiger, auf Basis der Referenzarchitektur instanziierter DPÖs - ein in der Praxis zunehmend eingesetztes Mittel im Bereich der digitalen Ökonomie basierend auf dem Ansatz von "Open Innovation". Das dadurch entstehende Ökosystem bietet ein gemeinsames, zentrales Wertversprechen, welches durch optionale Module erweitert wird. Diese Module sind in der Regel Daten, Services und andere meist digitale Dienstleistungen. Dieser zunehmend eingesetzte Mechanismus, dient dazu die Attraktivität einer Plattform für relevante Nutzer- und Interessensgruppen zu maximieren. Er basiert darauf, die Stakeholdergruppen durch	Grundlagenteil Abschnitt 4.1.4

	geeignete Crowdsourcing- und Crowdinnovationsprozesse in die Weiterentwicklung der Plattform, die Entwicklung und Gestaltung individueller Dienste und in die Erstellung, Strukturierung und Bewertung von benutzergenerierten Inhalten einzubinden.	
Basisservice	Unter dem Begriff Basisservice (Antrag: "Kerndienst") werden domänenunabhängige Services zusammengefasst, die über eine vordefinierte Schnittstelle und standardisierte Prozesse zur Verfügung gestellt werden. Dabei handelt es sich um Services, die grundsätzlich für alle Branchen relevant sein können.	Grundlagenteil Abschnitt 4.1.1
Digitales Plattform Ökosystem (DPÖ)	Die Referenzarchitektur bildet die Basis zur Entwicklung eines Digitalen Plattform Ökosystems (DPÖ). Unter einem DPÖ wird eine softwarebasierte (Cloud-)Plattform zusammen mit weiteren Akteuren verstanden, welche die zentrale Plattform zum dynamischen Austausch und Teilen von Diensten, Informationen, Ressourcen und sonstige Artefakte in einheitlicher Art und Weise verwenden. Auf Basis dieser Plattform wird für die Akteure ein gemeinsamer mehrseitiger Markt für (geschäftliche) Aktivitäten gebildet, wodurch ein dynamisches DPÖ entsteht.	Grundlagenteil Abschnitt 1.1
Domäne	Als Domäne wird der Bereich bezeichnet, der die Lösung und den Einsatzbereich einzelner Instanziierungen umfasst. Je nach Problemstellung sind Domänen identisch mit einzelnen Branchen und können synonym verwendet werden.	Grundlagenteil Abschnitt 1.3
Domänenoperator	Im Rahmen der Anwendungsforschung erfolgt eine branchenspezifische Instanziierung der Referenzarchitektur, indem die Referenzbausteine weiter konkretisiert werden und prototypische Basisservices durch	Grundlagenteil Abschnitt 4.1.3

	die Implementierung domänenspezifischer Services ergänzt werden. Der Domänenoperator verantwortet die Instanziierung innerhalb einer bestimmten Domäne, legt Governanceregeln oder mögliche finanzielle Beiträge fest und führt die Implementierung durch oder beauftragt diese. Darüber hinaus ist er für die Koordination innerhalb der Domäne und mit der die Gesamtreferenzarchitektur verantwortenden Instanz verantwortlich.	
Domänenreferenzarchitektur	Eine Domänenreferenzarchitektur umfasst die Untersuchung von Kernaspekten einer Domäne und gibt diese in Form von branchenspezifischen Use Cases, Services und Funktionen wieder. Diese Use Cases werden schließlich mit Hilfe der Grundlagen der Gesamtreferenzarchitektur beschrieben, wodurch eine domänenspezifische Referenzarchitektur entsteht.	Grundlagenteil Abschnitt 1.3; Anwendungsteil Abschnitt 1.1
Endnutzer	*Endnutzer* - generisch beschrieben - sind die jeweiligen Nutzer eines bestimmten Service, Konsumenten von Daten oder anderer Produkt und Dienstleistungen. Die Endnutzer können spezifisch für eine bestimmte Domäne sein. Als Endnutzer können je nach Perspektive Privatpersonen, aber auch Unternehmen oder öffentliche Einrichtungen angesehen werden. Der Begriff des Endnutzers ist somit stets im jeweiligen Kontext zu verstehen und gegebenenfalls weiter zu spezifizieren.	Anwendungsteil Abschnitt 2.1.1
Forschungsvorhaben BayernCloud	Im Rahmen des Forschungsvorhabens BayernCloud werden Konzepte für ein DPÖ ausgerichtet auf die besonderen Anforderungen des bayerischen Mittelstandes entwickelt. Die in diesem Kontext zu entwickelte	Grundlagenteil Abschnitt 1

	Referenzarchitektur bildet dabei Basis für die Erstellung von branchenspezifischen Instanziierungen.	
Kommunale Verwaltungen / Betriebe	Diese Stakeholdergruppe bilden Kommunal- und Landkreisverwaltungen sowie deren Unternehmen, Vertreter und Angestellte, aber auch öffentliche Einrichtungen. Sie verfügen häufig über umfangreiche Daten-Sammlungen und IT-Systeme. Entsprechend ist ihre Ein- und Anbindung von hoher Relevanz für eine leistungsfähige Instanziierung in Form von DPÖs.	Anwendungsteil Abschnitt 2.1.4
Multicloud	Unter Mulitcloud wird im Kontext der konzeptuellen Entwicklung des BayernCloud DPÖ das Ziel des betreiberunabhängigen Systembetriebs der BayernCloud Infrastruktur verstanden. Betreiber sind hier die bekannten Cloud-Provider wie AWS, Microsoft Azure oder Google Cloud von Infrastructure-as-a-Service (IaaS). Eine Multicloud-Lösung sieht demnach vor, Services möglichst einfach zwischen diesen Anbietern wechseln zu können, um einer einseitigen Abhängigkeit zu entgehen.	Grundlagenteil Abschnitt 1.1
Referenzbausteine	Unter *Referenzbausteinen* werden die erforderlichen Teile der Referenzarchitektur verstanden. Es handelt sich um allgemeine, standardisierte Bausteine mit technischen, organisatorischen und juristischen Grundlagen für die Entwicklung eines DPÖ und dessen laufenden Betrieb. Die Bausteine sind durch geeignete Verfahren für einzelne Branchen adaptierbar. Referenzbausteine sind unter anderem Geschäfts- und Governancemodell, die technische Architektur sowie die rechtlichen Rahmenbedingungen.	Projektantrag Grundlagenteil Abschnitt 1.3; Abschnitt 4, Abbildung 1
Sichten der Referenzarchitektur	Die zu erstellende Referenzarchitektur umfasst 4+1 Sichten. Diese dienen der Beschreibung und Dokumentation	

	der Architektur und ihrer Referenzbausteine aus spezifischen Perspektiven. Im Rahmen der BayernCloud Referenzarchitektur handelt es sich um eine angepasste Version des 4+1 Sichtenmodells nach Kruchten (1995), welches als Standard zur Erstellung von Referenzarchitekturen gilt.	
Use Case	Ein Use Case beschreibt eine grundlegende, domänenübergreifende und elementare Anforderung an das später zu instanziierende Digitale Plattform Ökosystem bzw. an die zugrundeliegende Referenzarchitektur. Use Cases dienen als Input zu Ausgestaltung der Referenzarchitektur und werden aus Anforderungen des Projektantrages und aus Anforderungen des Anwendungsteils gewonnen. Zusatz Anwendungsteil Für den Anwendungsteil im Tourismus werden zahlreiche (Domänen) Use Cases im Antrag bereitgestellt (Abschnitt 2.1). Diese werden aus dem Blickwinkel touristischer Nutzergruppen im Detail beschrieben. Die Sammlung "realer" Use Cases im Tourismus und die Ableitung funktionaler und nicht funktionaler Mechanismen für diese Domäne findet in AP2 des Anwendungsteils statt. Sie können als Grundlage für die Use Cases des Grundlagenteils herangezogen werden.	
Verbände	Ziel der BayernCloud ist es, große und kleine Branchen zu verbinden und ihnen den Austausch digitaler Services und Daten zu ermöglichen. Als ***Verbände*** werden die in der Regel als Verband organisierten Teilnehmer einer Domäne verstanden.	Anwendungsteil Abschnitt 2.1.3

Zielsetzung BayernCloud Anwendungsteil	1) Funktionale Domänenreferenzarchitektur 2) Konkretisierung der branchenübegreifenden (Referenz-) Bausteine	Anwendungsteil Abbildung 5
Zielsetzung BayernCloud Grundlagenteil	1) Erstellung und Verifikation einer Referenzarchitektur als Grundlage für die Instanziierung von DPÖs 2) Proof-Of-Concept der BayernCloud auf Basis geeigneter Open-Source Lösungen	Grundlagenteil + zusammengefasst im Anwendungsteil Abbildung 4

Tabelle 57: BayernCloud Glossar

fortiss ist das Landesforschungsinstitut des Freistaats Bayern für softwareintensive Systeme mit Sitz in München. Die WissenschaftlerInnen am Institut arbeiten in Forschungs-, Entwicklungs- und Transferprojekten mit Universitäten und Technologiefirmen in Bayern, Deutschland und Europa zusammen. Schwerpunkte sind die Erforschung modernster Methoden, Techniken und Werkzeuge der Softwareentwicklung, des Systems- & Service-Engineering und deren Anwendung auf kognitive cyber-physische Systeme wie das Internet of Things (IoT).

fortiss ist in der Rechtsform einer gemeinnützigen GmbH organisiert. Gesellschafter sind der Freistaat Bayern (Mehrheitsgesellschafter) und die Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V.

fortiss GmbH
Guerickestraße 25
80805 München
Deutschland
www.fortiss.org
Tel.: +49 89 3603522 0
E-Mail: info@fortiss.org

fortiss

Gefördert durch

Bayerisches Staatsministerium für
Wirtschaft, Landesentwicklung und Energie