Institut für Parallele und Verteilte Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

Erweiterung des Data-Mesh-Konzepts um Datenschutzaspekte

Laura Sophie Schuiki

Studiengang:	Informatik
Prüfer/in:	Prof. Dr.-Ing. Bernhard Mitschang
Betreuer/in:	Christoph Stach, Rebecca Kay Eichler
Beginn am:	3. Mai 2023
Beendet am:	15. November 2023

Kurzfassung

Daten sind einer der begehrtesten Rohstoffe unserer Zeit. Um Informationen aus den Daten zu extrahieren, müssen sie in einer Infrastruktur bereitgestellt werden, die das ermöglicht. Die Verwaltung aller Daten ist eine schwierige Aufgabe für zentrale Datenteams, da die Datenmenge sehr groß ist und das Domänenwissen fehlt, um die Daten fachmännisch zu betreuen. Der Data-Mesh-Ansatz ist ein neuer organisationaler Ansatz für den Umgang mit analytischen Daten. Es handelt sich um einen dezentralen Ansatz, in dem die Betreuung und Aufbereitung der Daten in der Verantwortung der Domäne liegt, die die Daten besitzt. In der Wissenschaft wurde der Privacy-Aspekt im Kontext des Data-Mesh-Ansatz im Vergleich zu anderen Bereichen noch nicht ausreichend berücksichtigt.

Daher wird in dieser Arbeit ein eigenes Privacy-Konzept für die Erfüllung von Privacy-Aspekten im Data-Mesh-Ansatz entwickelt. Es umfasst die Prozesse, die zur Betreibung einer Data-Mesh-Dateninfrastruktur benötigt werden: Die Erhebung sowie Prüfung und Durchsetzung von Privacy-Forderungen und die Erstellung von Datenprodukten. Zudem werden Implementierungsstrategien diskutiert, mit denen eine effektive und effiziente Prüfung und Durchsetzung von Privacy-Forderungen ermöglicht wird. Dafür wird ein Beschreibungsmodell entwickelt, mit dem Privacy-Forderungen effektiv geprüft werden können. Zur Auswahl einer effizienten Technologie zur Durchsetzung von Privacy-Forderungen wird eine prototypische Implementierung umgesetzt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quellcodeverzeichnis

Verzeichnis der Algorithmen

1 Einleitung

Daten sind einer der begehrtesten Rohstoffe unserer Zeit. Allerdings unterscheiden sie sich in einigen zentralen Punkten von herkömmlichen Rohstoffen. Beispielsweise können Daten verlustfrei dupliziert werden und sind nicht verbrauchbar. Das bedeutet, sie können dupliziert und dann von mehreren Nutzer*innen verwendet werden und verlieren durch die Nutzung nicht ihre Qualität. Dadurch sind Daten besonders wertvoll, wenn sie nur wenigen Nutzer*innen zur Verfügung stehen [Sta23a]. Sie sind deshalb eher als ein Vermögenswert der Firma zu betrachten, welche sie besitzt. Häufig erhalten sie ihren Wert durch die Informationen, die in ihnen enthalten sind. Auf deren Basis können beispielsweise Geschäftsprozesse optimiert werden, wodurch der Umsatz gesteigert werden kann [Nee19]. Diese Informationen werden zum Beispiel von Expert*innen mithilfe von Analysen aus den Daten extrahiert. Dies setzt voraus, dass die Daten in einer Infrastruktur bereitgestellt werden, die solche Analysen ermöglicht [Sta23a].

In einem Data Warehouse werde alle ankommenden Daten transformiert, um in ein vordefiniertes, gemeinsames Schema gebracht zu werden [BG13]. Allerdings geht durch die Transformationen und das streng vorgegebene Schema Flexibilität hinsichtlich der Nutzbarkeit der Daten verloren, da sie auf einen Anwedungsfall ausgerichtet werden. Der Data Lake stellt hierzu eine Alternative dar. Die Daten werden bei ihrer Aufnahme lediglich in ihrem Rohformat im Lake abgelegt, wodurch alle Informationen erhalten werden, die sich potentiell in den Daten befinden [RZ19]. Zusätzlich kann eine Strukturierung durch sogenannte Ponds [Inm16] oder Zonen [Sha18] eingeführt werden. Dabei haben sich die Zonen durchgesetzt [GGH+19]. Sie enthalten die Daten in verschiedenen Stufen der Aufbereitung. So können die Nutzer*innen direkt auf aufbereitete Daten zugreifen, welche die Anforderungen ihrer Analyse erfüllen [GGH+20]. Die Betreuung des Data Lakes ist die Aufgabe eines designierten Teams von Datenexperten. Ein Unternehmen besteht allerdings aus vielen verschiedenen Domänen, die jeweils unterschiedliche Anforderungen an die Haltung und Wartung ihrer Daten stellen. Die Verwaltung aller Daten des Unternehmens ist eine schwierige Aufgabe für das Data Lake Team. Zum einen ist die enorme Menge der zu betreuenden Daten ein Problem und zum anderen fehlt das Domänenwissen das benötigt wird, um die Daten fachmännisch zu verarbeiten [MCS22].

Infolgedessen werden Ansätze benötigt, die sich mit diesen Herausforderungen befassen. Der Data Mesh ist ein neuer organisationaler Ansatz für den Umgang mit analytischen Daten sowohl innerhalb einer Organisation, als auch über ihre Grenzen hinweg [Deh22]. Es handelt sich um einen dezentralen Ansatz in dem kein zentrales Team zur Datenverwaltung existiert. Stattdessen liegt die Betreuung und Aufbereitung in der Verantwortung der Domäne, welche die Daten besitzt. Dadurch steht bei der Aufbereitung sämtliches Fachwissen zur Verfügung und die Menge der zu betreuenden Daten ist geringer. Bei Bedarf können Domänen die Daten anderer Domänen anfragen und bekommen diese dann zur Verwendung bereitgestellt [Deh22]. Die Daten sollen aber nicht erst auf Anfrage einer anderen Domäne herausgesucht und vorbereitet werden. Sie sollten als fertiges Datenprodukt vorliegen, das darauf ausgelegt ist, dass es von anderen Domänen verwendet wird. Die DAUTNIVS-Eigenschaften von Dehghani [Deh22] beschreiben, welche Eigenschaften ein qualitativ

hochwertiges Datenprodukt besitzen sollte. Dabei steht jeder Buchstabe für eine zu erfüllende Eigenschaft. Dazu gehören auch Sicherheitsaspekte die besagen, dass eine Balance zwischen der Domänenverantwortung und einem zentralen Sicherheitsstandard gefunden werden muss. Privacy gehört auch zu den DAUTNIVS-Eigenschaften: Das heißt der Zugriff muss so geregelt werden, dass keine personenbezogenen Daten an unberechtigte Dritte weitergegeben werden [Deh22].

Dies ist beispielsweise im Gesundheitssektor besonders relevant, da dort fast ausschließlich mit vertraulichen, personenbezogenen Daten gearbeitet wird. Die jüngsten Entwicklungen zeigen, dass die Einhaltung von Datenschutzbestimmungen ein zentrales Thema für einen Großteil aller Einrichtungen und Firmen ist [HWW23]. Die Umsetzung des Privacy-by-Design-Konzepts, also Datenschutz durch Technikgestaltung und durch datenschutzfreundliche Voreinstellungen[1], steht besonders seit dem Inkrafttreten der Datenschutz-Grundverordnung (DSGVO) [EU16] im Mittelpunkt. Die Einhaltung und Durchsetzung der gesetzlich vorgegebenen sowie der individuellen Privacy Forderungen der Betroffenen muss von vornherein bei der Entwicklung von Datenverarbeitungsprozessen berücksichtigt werden.

1.1 Problemstellung und Lösungsansatz

Während sich aktuelle Forschungsarbeiten mit den Herausforderungen der Data Mesh Privacy beschäftigen, mangelt es an konkreten Umsetzungsvorschlägen zur Lösung dieser Herausforderungen. Deshalb ist das Ziel dieser Arbeit ein Konzept für den bedarfsgerechten Schutz der Datenprodukte in einem Data Mesh zu entwickeln und dabei den Fokus auf personenbezogene Daten zu legen. Dabei muss die Nutzbarkeit der Datenprodukte bestmöglich gewahrt bleiben.

Zu diesem Zweck wird zunächst eine Analyse hinsichtlich der bestehenden Datenschutzkonzepte für Data Meshes durchgeführt. Sie dient als Grundlage für den Entwurf eines eigenen Konzepts für die Durchsetzung von Privacy im Data-Mesh-Ansatz, das zu beachtende Datenschutzanforderungen mittels Annotation an die Datenprodukte bindet. Für eine effektive Umsetzung der Datenschutzanforderungen wird eine Modellierung ausgearbeitet. Weiter werden Techniken zur Umsetzung der Datenschutzanforderungen hinsichtlich einer effiziente Durchsetzung der Datenschutzanforderungen diskutiert. Das erarbeitete Konzept wird als Proof of Concept prototypisch umgesetzt. Zuletzt werden die Ergebnisse der Arbeit hinsichtlich ihrer Eignung zum Einsatz in einer Data-Mesh-Plattform sowohl unter theoretischen als auch praktischen Gesichtspunkten evaluiert.

1.2 Aufbau der Arbeit

Zuerst wird in Kapitel 2 ein Anwendungsfall erläutert, der ein Krankenhaus zum Thema hat. Er umfasst mehrere unterschiedliche Szenarien und wird als Laufbeispiel verwendet. Es folgt die Einführung in die Grundlagen in Kapitel 4. Darin werden zunächst die Konzepte und Prinzipien des Data Mesh erläutert. Außerdem werden Filter zur Umsetzung und Metriken zur Messung von Privacy vorgestellt. Im Anschluss werden in Kapitel 3 die Anforderungen an die Lösung definiert und in Kapitel 5 die verwandten Arbeiten diskutiert. Dabei werden verschiedene Aspekte der Sicherstellung

[1] siehe Art. 25 DSGVO [EU16]

von Privacy betrachtet. Es wird geprüft. inwieweit sich deren üblichen Umsetzungen von einer Umsetzung im Data Mesh Kontext unterscheidet. Das entwickelte Konzept wird in Kapitel 6 erläutert. Zunächst werden die abstrakten Prozesse der Erhebung und Umsetzung von Privacy Forderungen erläutert, wobei auf mehrere Aspekte besonders detailliert eingegangen wird. In Kapitel 7 werden anschließend Implementierungsstrategien für die Beschreibung von Privacy-Forderungen sowie die Anwendung von Privacy-Filtern besprochen. Anschließend werden die Vorgehensweise bei der prototypische Umsetzung sowie die Ergebnisse der durchgeführten Messungen in Kapitel 8 vorgestellt. In Kapitel 9 werden die Ergebnisse der Arbeit anhand der Anforderungen (siehe Kapitel 3) evaluiert. Hierfür wird geprüft ob und wie die theoretischen und praktischen Ergebnisse der Arbeit (siehe Kapitel 6 bis 8) die gestellten Anforderungen erfüllen. Außerdem werden die Ergebnisse mit den verwandten Arbeiten zu Data Mesh verglichen. Abgeschlossen wird die Arbeit durch die Zusammenfassung in Kapitel 10.

2 Anwendungsfall: Krankenhaus

In diesem Kapitel wird der Anwendungsfall eingeführt, der für die gesamte Arbeit als Laufbeispiel dient. Er steht stellvertretend für alle Anwendungsfälle, in denen ein besonderer Fokus auf den Datenschutz gelegt werden muss. Dadurch können die Konzepte, die in der Arbeit entwickelt werden, auf andere Data-Mesh-Anwendungsfälle übertragen werden. Zunächst wird in Abschnitt 2.1 der übergeordnete Anwendungsfall eines großen Krankenhauses erläutert. Anschließend werden in Abschnitt 2.2 drei Szenarien vorgestellt, welche innerhalb des Anwendungsfalls angesiedelt sind.

2.1 Übergeordneter Anwendungsfall

Um die Anforderungen des Anwendungsfalls besser erläutern zu können, wird zunächst die organisationale Struktur des Krankenhauses beschrieben. Abbildung 2.1 veranschaulicht die Struktur des Krankenhauses. Sie ist an die Struktur von realen Krankenhäusern angelehnt [Bor20; Rob23; Uni23b].

Komplexität der Domänen: Ein Krankenhaus enthält unterschiedliche Fachabteilungen wie z.B. die Notaufnahme, die Radiologie oder die Chirurgie. Diese verschiedenen Domänen werden unter der übergeordneten Domäne Fachabteilungen (oben rechts) zusammengefasst. In der Domäne der Stationen (oben links) sind die Subdomänen der Pflegefachkräfte sowie die Pflegedienstleitung

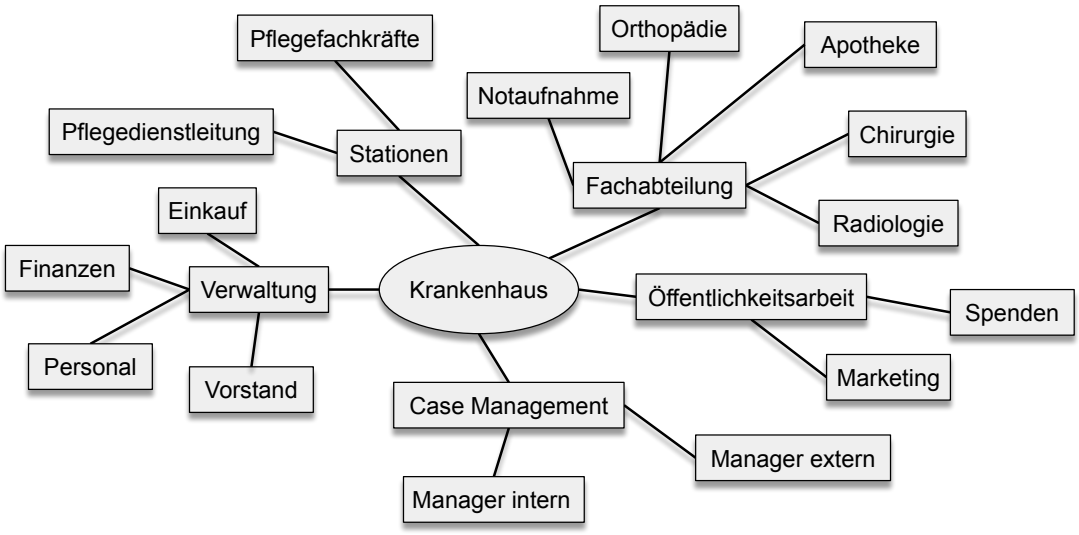

Abbildung 2.1: Organisationale Struktur eines Krankenhauses.

enthalten. Das Case Management (unten) enthält Manager, die sich entweder um interne oder externe Fälle kümmern. Die Verwaltung (links) umfasst den Einkauf, die Personalabteilung und die Finanzabteilung des Krankenhauses. In der Domäne der Öffentlichkeitsarbeit (rechts) sind einerseits das Marketing und andererseits die Spenden untergebracht. Abbildung 2.1 stellt nur einen Ausschnitt der organisationalen Struktur dar und soll nur die Argumentation dieser Arbeit unterstützen. In der Realität können die Domänen und Subdomänen in weitere Subdomänen unterteilt werden. Beispielweise kümmert sich ein Teil der Finanzabteilung ausschließlich um die Finanzierung der Angestellten, während ein anderer Teil die Leistungsabrechnung bei der Versicherung durchführt. Es ist ersichtlich, dass die organisationale Struktur des Krankenhauses sehr komplex ist und aus vielen sehr unterschiedlichen Domänen besteht. Aufgrund ihrer Unterschiede haben die Domänen sehr unterschiedliche Anforderungen an die Aufbereitung und -wartung ihrer Daten. Zudem ist für die optimale Betreuung der Daten domänenspezifisches Fachwissen notwendig. Eine geeignete Dateninfrastruktur muss es deshalb möglich machen, dass domänenspezifisches Fachwissen bei der Datenaufbereitung und -wartung angewendet werden kann.

Große Datenmenge: Die Domänen erzeugen Daten, die in der Dateninfrastruktur des Krankenhauses aufbewahrt werden. Ihre Daten sind wertvoll, da man daraus viel nützliche Informationen und Wissen generieren kann. Dies setzt voraus, dass ein sinnvolles System für das Management der Daten besteht [DSSK19]. Es gibt viele verschiedene Szenarien, in denen Daten produziert werden. Zum Beispiel wird bei der Aufnahme einer Patient*in eine Patientenakte angelegt. Außerdem dokumentieren die Fachabteilungen alle Leistungen, welche die Patient*in von den verschiedenen Domänen in Anspruch nimmt, wie zum Beispiel das Durchführen einer Magnetresonanztomographie (MRT) in der Radiologie. Ein durchschnittliches Krankenhaus hat rund 9000 stationäre Patient*innen pro Jahr, für welche die genannten Daten gespeichert werden. Hinzu kommen nicht stationäre Patienten, deren Behandlungen ebenfalls dokumentiert werden [Sta23c]. Auch nicht-medizinische Daten werden produziert. Zum Beispiel speichert die Verwaltung für jede Mitarbeiter*in eine Personalakte. Im Jahr 2021 waren in ganz Deutschland rund 1,3 Millionen Menschen in Krankenhäusern angestellt. Das sind im Durchschnitt rund 7000 Personalakten pro Krankenhaus [Sta23b]. Hinzu kommen regelmäßig auftretende Szenarien, in denen Daten produziert werden. Die Finanzabteilung erarbeitet ihre Finanzberichte regelmäßig und die Inventur wird in allen Abteilungen zu festen, wiederkehrenden Terminen durchgeführt. Derartige Vorgänge tragen zu einer stetig wachsenden, großen Datenmenge bei, die in der Dateninfrastruktur gespeichert wird. Die Dateninfrastruktur muss daher mit großen Mengen an Daten zurechtkommen.

Heterogenität: Wie im vorigen Absatz beschrieben, produzieren die Domänen des Krankenhauses die verschiedene Daten. Beispielsweise werden pro Patient*in eine Patientenakte angelegt sowie alle in Anspruch genommenen Leistungen dokumentiert. Dabei müssen sowohl textuelle Daten festgehalten werden, als auch die Ergebnisse von bildgebenden Untersuchungen wie zum Beispiel Röntgenaufnahmen. Zusätzlich gibt es Daten, die unabhängig von den Patient*innen produziert werden. Daten wie Finanzberichte, Personalakten und Inventuren von Lagerbeständen werden ebenfalls in der Dateninfrastruktur festgehalten. Diese Beispiele zeigen, dass das Krankenhaus heterogene Daten produziert [DSSK19]. Daher muss eine geeignete Dateninfrastruktur die Möglichkeit bieten, heterogene Daten in verschiedenen Formaten zu speichern.

Datenaustausch: Es gibt Szenarien in denen eine Domäne die Daten von einer oder mehreren anderen Domänen verwenden möchte. Die Mitarbeiter*innen des Case Managements verwenden beispielsweise die Daten der verschiedenen Fachabteilungen, um den Aufenthalt der Patient*in zu planen. Da die Planungen für jede Patient*in durchgeführt werden, fragt das Case Management die

Daten der Fachabteilungen regelmäßig an. Die Fachabteilungen müssen ihre Daten jedes Mal für die Übergabe vorbereiten. Zunächst werden alle relevanten Daten der Behandlung der Patient*in zusammengetragen. Anschließend werden beispielsweise alle Untersuchungsergebnisse in einem einzigen Bericht zusammengefasst, um die Verständlichkeit zu erhöhen. Wird die gleiche Anfrage regelmäßig gestellt, kann der Vorbereitungsprozess bereits definiert sein und muss nur noch durchgeführt werden. Ist die Anfrage jedoch einzigartig, wird der Vorbereitungsprozess langwieriger, da er zunächst entworfen werden muss und erst dann durchgeführt werden kann. Wenn die Nutzer*innen beim Austausch von Daten in der verwendeten Dateninfrastruktur durch Technologien unterstützt werden, können sich die Wartezeiten reduzieren, da die Vorbereitungen schneller erledigt werden können. Zum einen ist der Austausch von Daten wichtig, um Patient*innen eine optimale Behandlung zu ermöglichen [DSSK19]. Zum anderen gibt es administrative Prozesse, die zum Beispiel von der Verwaltung durchgeführt werden, in denen Daten von mehreren Domänen gebraucht werden. Die Digitalisierung von Arbeits- und Verwaltungsabläufen wird im Gesundheitswesen zunehmend umgesetzt. Diese Entwicklung fördert den Datenaustausch der Gesundheitsdienstleister [KHJ18]. Aus diesem Grund muss beim Design einer passenden Dateninfrastruktur der effektive und effiziente Austausch von Daten bedacht werden.

Privacy: Die jüngsten Entwicklungen zeigen, dass die Einhaltung von Datenschutzbestimmungen ein zentrales Thema für einen Großteil aller Einrichtungen und Firmen ist [HWW23]. Ein Großteil der Daten, die in diesem Anwendungsfall produziert werden, sind vertrauliche, personenbezogene Gesundheitsdaten. Daher müssen die gesetzlichen Bestimmungen zum Datenschutz eingehalten werden. Beispielsweise dürfen gemäß der Zweckbindung der DSGVO nur Daten erhoben werden, die einem im Voraus definierten Zweck dienen[1]. Auch die Domänen des Krankenhauses können Privacy-Forderungen stellen. Beispielsweise gewährt die Finanzabteilung den anderen Domänen nur eingeschränkten Zugriff auf die Gehaltsdaten der Mitarbeiter*innen. Zusätzlich haben die Betroffenen individuelle Privacy-Forderungen . Bei der Behandlung einer Patient*in werden neben einer Kurzbeschreibung (z.B. Leistung: „Diagnose, anschließend Schienung“) zusätzlich eine detaillierte Beschreibung der Befunde verfasst (z.B. Details: „Knochenbruch im linken kleinen Finger. Anbringung einer Schiene.“). Diese kann vertrauliche Informationen beinhalten, welche die Patient*in mit so wenigen Domänen bzw. Mitarbeiter*innen wie möglich teilen möchte. Während die Kurzbeschreibung für alle Domänen zur Verfügung stehen muss, ist ein Zugriff auf die detaillierte Beschreibung nicht zwingend notwendig. In diesem Fall können Betroffene beispielsweise fordern, dass die Details der Diagnose nur von medizinischem Personal eingesehen werden können. Es ist ersichtlich, dass eine Vielzahl von Privacy-Forderungen realisiert werden müssen, bevor der Zugriff auf einen Datensatz gewährt werden kann. Wann immer personenbezogene Gesundheitsdaten verwendet werden, ist die Privacy offensichtlich ein zentrales Thema. Deshalb muss eine Dateninfrastruktur, in der Gesundheitsdaten aufbewahrt und verarbeitet werden, ein Berechtigungssystem mit einschließen [ABK18]. Dieses muss die verschiedenen Arten von Privacy-Forderungen abbilden, speichern und umsetzen können.

[1] siehe Art. 5(1) b) DSGVO [EU16]

2.2 Drei Szenarien

In diesem Abschnitt werden drei konkrete Szenarien beschrieben, die sich innerhalb des Krankenhauses ereignen. Sie stellen konkrete Geschäftsprozesse dar, in denen eine Domäne die Daten von einer oder mehreren anderen Domänen anfragt und verwendet. Für jedes Szenario wird beschrieben, welche Aufgabe von wem und mit wessen Daten erfüllt werden soll. Außerdem wird erläutert, welche Privacy-Forderungen jeweils relevant sind.

Szenario 1: Optimierung der Dienstpläne einer Station

Wenn eine Patient*in über Nacht im Krankenhaus bleiben muss, dann übernachtet sie auf einer Station. Auf einer der Stationen des gegebenen Anwendungsfalls herrscht häufig Unterbesetzung, wodurch das Pflegepersonal überfordert ist. Die Pflegedienstleitung stellt fest, dass die Einteilung der Pflegekräfte unabhängig von der Auslastung der Station erfolgt, sodass jeden Tag die gleiche Anzahl Pflegekräfte bei der Arbeit sind. Es wird also nicht darauf eingegangen, wie lange Patient*innen voraussichtlich auf der Station bleiben werden oder wie intensiv sie betreut werden müssen, um die Pflegekräfte entsprechend einzuteilen. So kommt es vor, dass die Pflegekräfte an einem Tag zu viel Arbeit haben und an einem anderen zu wenig. Um diesem Problem entgegen zu wirken, möchte die Pflegedienstleitung der Station einen Weg finden vorauszusagen, wie hoch die Auslastung in den kommenden Tagen sein wird, um die Pflegekräfte entsprechend einteilen zu können. Die Optimierung von Geschäftsprozessen mithilfe von Daten wird hauptsächlich in der Prozesskontrolle angewandt. Diese Strategie kann jedoch auch im Gesundheitssektor zu positiven Ergebnissen führen [BAW+12].

Die Pflegedienstleitung der Station fragt die Daten vergangener Behandlungen bei der ihr zugeordneten Fachabteilung an. Dieser Datensatz enthält Informationen über die Leistungen, die Patient*innen erhalten haben, beispielsweise dass eine Operation durchgeführt wurde (siehe Tabelle 2.1). In den domäneneigenen Aufenthaltsdaten, die die Station selbst über ihre Patient*innen gesammelt hat, sind Informationen über die Dauer des Aufenthalts sowie die nötige Betreuung enthalten (siehe Tabelle 2.2). Die Pflegedienstleitung reichert die Behandlungsdaten mit den Aufenthaltsdaten an. Hierzu werden für jede Patient*in die Behandlungsdaten mit den Aufenthaltsdaten zusammengebracht. Auf Grundlage dieses neuen Datensatzes kann festgestellt werden, welche Merkmale in den Behandlungsdaten der Patient*innen darauf hindeuten, dass sie sich lange auf der Station aufhalten werden oder dass sie intensiv betreut werden müssen.

Name	VersichertenNr	Alter	Abteilung	Leistung	Details
...	...	...	...	...	...
Erika Mustermann	5678	37	Chirurgie	Appendektomie	Operation nach Diagnose Blinddarmenzündung angeordnet.
...	...	...	...	...	...

Tabelle 2.1: Beispiel für Behandlungsdaten in der Chirurgie.

Name	VersichertenNr	Alter	Abteilung	Leistung	Details
...	...	...	...	...	...
Erika Mustermann	5678	37	Station Chirurgie	Aufenthalt	Beobachtung nach Operation, intensive Betreuung angeordnet.
...	...	...	...	...	...

Tabelle 2.2: Beispiel für Aufenthaltsdaten der Station der Chirurgie.

Gemäß Artikel 5, Absatz 1, Buchstabe b der DSGVO (Zweckbindung) dürfen nur Daten erhoben werden, die einem im Voraus definierten Zweck dienen. Bei diesem Szenario handelt es sich um eine Analyse zur Optimierung der internen Prozesse. Zum Zeitpunkt der Erhebung der Daten ist nicht vorauszusehen, für welche Analysen die Daten in der Zukunft verwendet werden können. Daher müssen die Betroffenen vor der Datenerhebung informiert werden, dass diese auch zu Analysezwecken verwendet werden könnten. Dabei handelt es sich nicht um eine Verwendung, die für die erfolgreiche Abwicklung der Behandlung einer Patient*in notwendig ist. Deshalb muss die Patient*in explizit zustimmen, dass ihre Daten für Analysezwecke verwendet werden dürfen. Tut eine Patient*in dies nicht, dürfen ihre Daten in solchen Szenarien nicht verwendet werden. Außerdem können Patient*innen andere Privacy-Forderungen haben, die einzelne Teile ihrer Daten betreffen. Beispielsweise muss das Attribut, das die Details ihrer Diagnose enthält, entfernt werden, bevor die Behandlungsdaten an die Pflegedienstleitung übermittelt werden. Eine andere Möglichkeit ist, dass ein Teil der Patient*innen fordern, dass ihre Daten vor einer solchen Verarbeitung zunächst anonymisiert werden. Dies kann erreicht werden, indem der Datensatz mit den Daten aller Patient*innen anonymisiert wird. Zusätzlich kann eine zweite Version des Datensatzes erstellt werden, der unveränderte Daten von den Patient*innen enthält, die keine Anonymisierung fordern.

Die Dateninfrastruktur des Krankenhauses muss mehrere Dinge leisten, damit dieses Szenario umgesetzt werden kann. Es muss möglich sein, Datensätze einer anderen Domäne anzufragen und sie mit domäneneigenen Daten anzureichern. Somit wird daraus ein neuer domäneneigener Datensatz. Außerdem muss es möglich sein sowohl gesetzliche Privacy-Forderungen als auch die Privacy-Forderungen der Patient*innen abzubilden und umzusetzen.

Szenario 2: Anpassung der Einkäufe auf Basis von Lagerbeständen

Die Einkaufsabteilung ist dafür zuständig, regelmäßig Produkte wie Medikamente, Verbandsmaterial oder Schreibwaren zu bestellen, die in den unterschiedlichen Domänen des Krankenhauses benötigt werden. In diesem Szenario haben die Mitarbeiter*innen der Einkaufsabteilung festgestellt, dass die eingekauften Mengen häufig nicht mit dem Bedarf der jeweiligen Abteilung übereinstimmen. Teilweise wird wesentlich mehr eingekauft, als bis zur nächsten Lieferung verbraucht wird und an anderen Stellen ist der Vorrat bereits Tage vor der nächsten Lieferung aufgebraucht. Die Mitarbeiter*innen vermuten, dass das daran liegt, dass die Produkte gleichmäßig an die verschiedenen Domänen verteilt werden, anstatt die Menge an den jeweiligen Bedarf anzupassen. Deshalb soll analysiert werden, wie hoch der Verbrauch in den verschiedenen Domänen tatsächlich ist, um anschließend sowohl die Bestellmengen als auch die Verteilung daran anzupassen.

Hierfür fragt die Einkaufsabteilung Daten über die Lagerbestände von allen Domänen an. Beispielsweise werden in regelmäßigen Abständen Inventuren in den Domänen durchgeführt. Die Mitarbeiter*innen der Einkaufsabteilung wollen alle Datensätze miteinander kombinieren und so einen neuen Datensatz erstellen. Die Daten können verwendet werden, um zu berechnen, wie sich die Menge an Verbandsmaterial, Schreibwaren und ähnliche Produkte seit der letzten Inventur in den jeweiligen Domänen verändert hat.

Allerdings können die Domänen Privacy-Forderungen hinsichtlich der Verwendung ihrer Daten stellen. Ein Beispiel hierfür ist die Krankenhausapotheke, die Medikamente lagert und sie bei Bedarf an die Fachabteilungen aushändigt [Kai30; Ste11]. Die Apotheke des Anwendungsfalls vermerkt jede Medikamentenbestellung in ihrer Dateninfrastruktur. Wie in Tabelle 2.3 dargestellt, wird die Mitarbeiter*in, die die Bestellung aufgibt, ihre Domäne und der Verwendungszweck dokumentiert. Der Verwendungszweck enthält beispielsweise einen Verweis auf die Patient*in, der das Medikament verabreicht wird, und soll ausschließlich innerhalb der Apotheke verwendet werden. Beispielweise kann damit das Risiko des Diebstahls mithilfe von falschen Angaben minimiert werden. Denn wenn sehr viele oder nicht zueinander passende Medikamente für dieselbe Patient*in bestellt werden, kann das ein Grund für eine Nachforschung sein. Die Apotheke hat deshalb die Privacy-Forderung, dass das Attribut „Verwendungszweck“ nicht von anderen Domänen eingesehen werden kann. Dabei handelt es sich nicht um die individuelle Forderung eines einzelnen Betroffenen, sondern eine Forderung, welche die ganze Domäne umfasst.

Um dieses Szenario umsetzen zu können, müssen in der Dateninfrastruktur des Krankenhauses die Privacy-Forderungen von Domänen berücksichtigt werden. Diese gelten im Gegensatz zu Privacy-Forderungen von Betroffenen für alle Daten einer Domäne und müssen für alle Daten umgesetzt werden. Daher muss es möglich sein, sie von Privacy-Forderungen von Betroffenen zu unterscheiden.

Präparat	Mitarbeiter*in	Domäne	Verwendungszweck
...	...	...	...
Ibuprofen 600mg	98765	Station Chirurgie	Schmerzlinderung bei Patient*in 5678 nach Operation.
...	...	...	....

Tabelle 2.3: Beispiel für Daten über aufgenommene Bestellungen.

Szenario 3: Leistungsabrechnung bei den Versicherungen

Ärzte leiten ihre Leistungsabrechnungen an Kassenärztliche bzw. Kassenzahnärztliche Vereinigungen weiter, diese übermitteln die Daten anschließend quartalsweise an die zuständigen Versicherungen. Im Gegensatz dazu kommunizieren Krankenhäuser direkt mit den Versicherungen. Deshalb muss die Finanzabteilung regelmäßig, zum Beispiel einmal pro Quartal, Abrechnungsunterlagen erstellen, welche alle in diesem Zeitraum erbrachten Leistungen enthalten [Bund23].

Die Leistungsdaten einer Fachabteilung gehören immer dieser Abteilung. Deshalb fragt die Finanzabteilung die Daten über alle abzurechnenden Leistungen des jeweiligen Quartals bei allen anderen Domänen an. Sie erhalten je einen Datensatz von jeder Fachabteilung, der die Leistungen aller Patient*innen für das Quartal enthält. Zum Beispiel kommt der Patient Max Mustermann mit Schmerzen in der linken Hand in die Notaufnahme. Dort wird ein Bruch im kleinen Finger diagnostiziert und eine Schiene angelegt. Später geht er zur Beratung bezüglich der Nachbehandlung zum Case Management und wird dort über eine mögliche Physiotherapie informiert. Die daraus resultierenden Daten sind in Tabelle 2.4 und Tabelle 2.5 dargestellt. Die Finanzabteilung erkennt anhand der Stichwörter im Feld „Leistung“ welche Leistungen erfolgt sind und kombiniert das mit der Liste der Preise aller angebotenen Leistungen. Da an jede Versicherung nur ein Dokument übermittelt werden soll, fusioniert die Finanzabteilung alle erhaltenen Datensätze und reichert sie mit den Preisdaten an. Danach werden die Leistungen nach der Versicherung der Patient*innen aufgeteilt, sodass für jede Versicherung ein Datensatz vorliegt. Anschließend stellt die Finanzabteilung die Unterlagen ordnungsgemäß zusammen und übermittelt sie schließlich an die Versicherung.

Name	VersichertenNr	Alter	Abteilung	Leistung	Details
...	...	...	...	...	...
Max Mustermann	1234	35	Notaufnahme	Diagnose, anschließende Schienung	Knochenbruch im linken kleinen Finger. Anbringung einer Schiene.
...	...	...	...	...	...

Tabelle 2.4: Beispiel für Leistungsdaten in der Notaufnahme.

Name	VersichertenNr	Alter	Abteilung	Leistung	Details
...	...	...	...	...	...
Max Mustermann	1234	35	Case Management	Beratung Nachbehandlung	Wurde über Möglichkeit der Physiotherapie informiert.
...	...	...	...	...	...

Tabelle 2.5: Beispiel für Leistungsdaten des Case Managements.

Bei diesem Szenario handelt es sich um einen Prozess, der durchgeführt werden muss, damit Patient*innen behandelt werden können. Deshalb ist es Patient*innen nicht möglich, die Verwendung ihrer Daten generell zu verweigern. Auch gibt es für Krankenhäuser, im Gegensatz zu Ärzten, keine Richtlinien, welche Daten an die Versicherung übermittelt werden dürfen [Bund23]. Für gewöhnlich würden die Daten einer erbrachten Leistung komplett an die Finanzabteilung übermittelt werden, obwohl nur die Attribute „VersichertenNr“ und „Leistung“ benötigt werden, um die Abrechnung durchzuführen.

Daher ist es möglich, dass Betroffene Privacy-Forderungen für dieses Szenario haben. Beispielsweise möchte Max Mustermann geheim halten, dass er sich den kleinen Finger gebrochen hat. Deshalb fordert er, dass das Attribut „Details“ verborgen wird, bevor die Daten von nicht-medizinischen Abteilungen eingesehen wird. Da die Finanzabteilung keine medizinische Domäne ist, muss das Attribut entfernt werden, bevor sie Zugriff auf die Daten erhält. Es ist zudem möglich, dass eine Domäne eine Privacy-Forderung hat die für ihren ganzen Datensatz gilt. Beispielsweise hat die Krankenhausapotheke die Privacy-Forderung , dass das Attribut „Verwendungszweck“ nicht von anderen Domänen eingesehen werden kann (siehe Tabelle 2.3).

Um dieses Szenario umsetzen zu können, muss die Dateninfrastruktur die Möglichkeit bieten, Datensätze aus verschiedenen Domänen zu fusionieren. Dadurch entsteht ein neuer Datensatz, der zu der anfragenden Domäne gehört. Die Privacy-Forderungen, die für die ursprünglichen Datensätze gelten, müssen zu dem neuen Datensatz zugeordnet werden können und weiterhin umgesetzt werden.

Gewonnene Erkenntnisse: Das Krankenhaus hat eine komplexe Domänenstruktur, in der oft Daten zwischen den einzelnen Domänen ausgetauscht werden. Deshalb muss der effektive und effiziente Austausch von Daten bedacht werden. Dabei müssen viele, heterogene und Privacy-problematische Daten gehandhabt werden. Die vorgestellten Szenarien unterscheiden sich dabei folgendermaßen: In Szenario 1 wird der Datensatz einer anderen Domäne angefragt, angereichert und zu einem domäneneigenen Datensatz gemacht. Dabei müssen gesetzliche Privacy-Forderungen und Privacy-Forderungen der Patient*innen beachtet werden. In Szenario 2 werden mehrere Datensätze von anderen Domänen angefragt. Neu ist hierbei, dass Privacy-Forderungen von Domänen beachtet werden müssen. In Szenario 3 werden mehrere Datensätze von anderen Domänen angefragt und anschließend fusioniert, wobei ein neuer Datensatz entsteht. Die Fusion von Datensätzen und die Übernahme aller geltenden Privacy-Forderungen muss hierfür möglich sein. Im folgenden Kapitel werden auf Grundlage der Szenarien die Anforderungen an die Dateninfrastruktur des Krankenhauses definiert.

3 Anforderungen

Im Folgenden werden die Anforderungen an die Dateninfrastruktur des Krankenhauses (siehe Kapitel 2 Gewonnene Erkenntnisse) vorgestellt. Sie beschreiben was gegeben sein muss, damit eine Dateninfrastruktur im vorgestellten Anwendungsfall verwendet werden kann. Die Anforderungen sind in zwei Gruppen unterteilt. Die erste Gruppe **A1-A4** beschreibt, welche Anforderungen für die Privacy-Forderungen gelten. Die zweite Gruppe **A5-A7** beschreibt, was gelten muss, damit die Lösung in der Praxis nutzbar ist.

A1: Alle Privacy-Forderungen werden abgebildet.

In den Szenarien des Anwendungsfalls kommen gesetzlichen Privacy-Forderungen, Privacy-Forderungen von Domänen und Privacy-Forderungen von individuellen Betroffenen vor. Alle drei Arten von Privacy-Forderungen müssen beachtet werden. Dafür müssen sie zunächst alle abgebildet werden. Die gesetzlichen Privacy-Forderungen werden beispielsweise von der DSGVO [EU16] vorgegeben. Sie sind unabhängig von den vorhandenen Daten, müssen aber hin und wieder aktualisiert werden, wenn sich die gesetzlichen Bestimmungen ändern. Anders verhält es sich bei den Privacy-Forderungen der Domänen und Betroffenen. Wenn neue Daten über Betroffene in einer Domäne erhoben werden, können Domäne und die Betroffenen das Bedürfnis haben, ihre Privacy-Forderungen entsprechend anzupassen. Entweder indem sie eine neue Privacy-Forderung definieren oder indem sie ihre bestehenden Privacy-Forderungen ändern. Deshalb ist die Abbildung der Privacy-Forderungen Teil der Datenerhebung.

A2: Die Privacy-Forderungen werden gespeichert.

Nachdem die Privacy-Forderungen erfasst sind, werden sie nicht direkt auf die Daten angewendet, da die Rohdaten verfügbar bleiben sollen. Die Privacy-Forderungen werden erst verwendet wenn die Daten weiter verarbeitet werden. Außerdem müssen die Privacy-Forderungen anschließend weiterhin verfügbar sein. Sie müssen erneut geprüft werden, wenn die Daten mit anderen Domänen ausgetauscht werden. Und sie können auch für Daten gelten, die in der Zukunft erhoben werden. Dies gilt für alle gesetzlichen Privacy-Forderungen. Aber auch Privacy-Forderungen von Domänen oder Betroffenen können für mehrere oder sogar alle Datensätze gelten, die zu einer Domäne gehören bzw. Daten einer Betroffenen enthalten. Deshalb muss später auf die Privacy-Forderungen zugegriffen werden können. Um das möglich zu machen, müssen sie gespeichert werden. Dabei müssen alle Verantwortlichen Zugriff auf die Privacy-Forderungen haben. Außerdem darf es nicht möglich sein sie zu manipulieren, außer sie werden auf Wunsch ihres Verfassers verändert.

A3: Die Privacy-Forderungen können den Daten, für die sie gelten, zugeordnet werden.

Es werden gesetzliche Privacy-Forderungen, Privacy-Forderungen von Domänen und Privacy-Forderungen von individuellen Betroffenen erfasst. Die Privacy-Forderungen haben verschieden Wirkungsbereiche. Während gesetzliche Privacy-Forderungen für alle Daten gelten, wirken sich Privacy-Forderungen von Domänen und Betroffenen nur auf Datensätze aus, die zu einer Domäne gehören bzw. Daten einer Betroffenen enthalten. Bevor geprüft werden kann, ob ein Datensatz die Privacy-Forderungen erfüllt, muss ersichtlich sein welche Privacy-Forderungen für diesen Datensatz gelten. Ist diese Information nicht vorhanden, könnten zu viele oder die falschen Privacy-Forderungen geprüft werden. Im schlimmsten Fall werden dadurch Gesetze nicht beachtet oder sensitive Informationen an unberechtigte Dritte weitergegeben. Daher ist es essentiell, dass zugeordnet werden kann, für welche Datensätze eine Privacy-Forderung gilt. Aus diesem Grund muss diese Information Teil der gespeicherten Privacy-Forderungen sein. So können sie später den Daten zugeordnet werden.

A4: Die Einhaltung der Privacy-Forderungen, die für einen Datensatz gelten, muss durch den/die Verantwortlichen für den Datensatz sichergestellt werden.

Die DSGVO schreibt vor, dass die Durchsetzung des Datenschutzes für einen Datensatz in der Zuständigkeit des/der Verantwortlichen des Datensatzes liegt. Demnach ist der/die Verantwortliche auch für die Einhaltung der Privacy-Forderungen verantwortlich. Deshalb muss der/die Verantwortliche die Privacy-Forderungen prüfen und gegebenenfalls umsetzen. Bevor ein Datensatz für einen bestimmten Zweck verwendet wird, muss geprüft werden, ob die Privacy-Forderungen eingehalten werden. Dafür muss der Verantwortliche auf die Privacy-Forderungen zugreifen, die für den Datensatz gelten. Anschließend muss er/sie prüfen, ob sie im gegebenen Verwendungszweck erfüllt sind. Falls nicht, muss der/die Verantwortliche in der Lage sein Schritte einzuleiten die sicherstellen, dass der Datensatz die Privacy-Forderungen erfüllt.

A5: Das Privacy-Konzept muss möglichst unabhängig von der vorhandenen Infrastruktur umsetzbar sein.

Angenommen, eine Organisation ohne Dateninfrastruktur implementiert einen neue Dateninfrastruktur. Sie kann für jede Komponente der neuen Dateninfrastruktur die Technologie verwenden, die am besten passt, ohne dass zusätzliche Kosten entstehen. Hat eine Organisation bereits eine funktionierende Infrastruktur, ist das schwieriger. In diesem Fall ist es von Vorteil, wenn Teile der bestehenden Struktur weiter in der neuen Dateninfrastruktur verwendet werden können. Dadurch können Kosten und Zeit gespart werden, die für die Anschaffung und Integration neuer Technologien aufgewendet werden müssen. Deshalb sollten zur Umsetzung des Privacy-Konzepts keine spezifischen Technologien notwendig sein. Dafür muss auch die Privacy-Konzept unabhängig von spezifischen Technologien definiert werden. Beispielsweise wird vorausgesetzt, dass die Privacy-Forderungen so gespeichert werden, dass alle Verantwortlichen auf sie zugreifen können. Es wird jedoch nicht spezifiziert, mit welcher Technologie der Speicherort umgesetzt wird. Es ist ersichtlich, dass die Anforderungen an die Implementierung der einzelne Komponenten der neuen Dateninfrastruktur vorgegeben sein müssen. Die Entscheidung welche Technologie dafür verwendet wird, bleibt dabei den Betreibern überlassen.

A6: Das Privacy-Konzept muss die Verwendung von Standardtechnologien ermöglichen.

Das Krankenhaus des Anwendungsfalls (siehe Kapitel 2) hat bereits in Teilen eine Dateninfrastruktur. Die Nutzer*innen der bisherigen Dateninfrastruktur sind erfahren im Umgang mit den Schnittstellen. Sie haben präferierte Technologien die sie gerne für ihre jeweiligen Szenarien verwenden. Würde die Organisation diese Technologien beim Wechsel zur neuen Dateninfrastruktur gegen eine neue Technologie austauschen, würde das Nachteile mit sich bringen. Die Nutzer*innen müssten den Umgang mit den neuen Technologien lernen, was Zeit in Anspruch nimmt und frustrierend sein kann. Aus diesem Grund ist es von Vorteil Standardtechnologien zu verwenden, die beispielsweise aufgrund ihrer Effektivität und Verständlichkeit weit verbreitet sind und von vielen Fachleuten verwendet werden. Das Privacy-Konzept soll die Verwendung von Standardtechnologien zur Implementierung ermöglichen. Dafür muss auch das Privacy-Konzept unabhängig von bestimmten Technologien definiert werden. Stattdessen sollen lediglich die Anforderungen an die Technologien beschrieben werden.

A7: Das Privacy-Konzept muss Skalierbarkeit für große Datenmengen ermöglichen.

In dem vorgestellten Anwendungsfall (siehe Kapitel 2) wird mit großen Datenmengen gearbeitet und häufig Daten zwischen den Domänen ausgetauscht. Die Datensätze müssen für den Austausch vorbereitet werden. Der dadurch entstehende Zeitaufwand sollte die Arbeit der Verantwortlichen und Nutzer*innen nicht behindern. Beispielsweise fragt eine Nutzer*in einen Datensatz an, den die Verantwortliche zuerst aufbereiten muss. Die Umsetzung der Privacy-Forderungen dauert jedoch aufgrund der Datenmenge viele Tage. Das bedeutet eine lange Wartezeit für die Nutzer*in und die Verantwortliche muss sich stetig um die Umsetzung kümmern. Deshalb muss die Dateninfrastruktur in der Lage sein mit großen Datenmengen umzugehen. Es muss die Möglichkeit geben das Privacy-Konzept so umzusetzen, dass die Skalierbarkeit für große Datenmengen gegeben ist. Zu diesem Zweck können Technologien und Werkzeuge eingesetzt werden, die für den Umgang mit großen Datenmengen geeignet sind.

Im folgenden Kapitel werden die Hintergründe und das Konzept eines neuen organisationalen Ansatz für den Umgang mit analytischen Daten erläutert. Dieser sogenannte Data-Mesh-Ansatz ist ein Kandidat für die Dateninfrastruktur des Krankenhauses im vorgestellten Anwendungsfall. Zudem werden Techniken zur Umsetzung von Privacy-Forderungen sowie zur Messung der Privacy beschrieben.

4 Grundlagen

In diesem Kapitel werden die Grundlagen zu verschiedenen Aspekten dieser Arbeit erläutert. Sie werden in den folgenden Kapiteln benötigt. In Abschnitt 4.1 werden die Hintergründe und das Konzept des Data-Mesh-Ansatzes erläutert. Anschließend werden in Abschnitt 4.2 Filter zur Umsetzung von Privacy-Forderungen und Metriken zur Messung der Privacy vorgestellt.

4.1 Data Mesh

Der Data Mesh ist ein neuer organisationaler Ansatz für den Umgang mit analytischen Daten sowohl innerhalb als auch über die Grenzen einer Organisation hinweg. Es handelt sich um einen dezentralen Ansatz für die Verwaltung, den Austausch und den Zugriff auf Daten in großem Maßstab [Deh22]. Der Data-Mesh-Ansatz beruht auf vier Prinzipien, die in Abschnitt 4.1 erläutert werden. Er adressiert die Herausforderungen in den bisherigen zentralisierten Systemen, speziell die der Systeme Data Warehouse und des Data Lake [Bal23b]. Diese werden in Abschnitt 4.1 erklärt.

Herausforderungen, die der Data Mesh adressiert

Das in Abschnitt 2.1 vorgestellte Krankenhaus möchte den Austausch von Daten zwischen mehreren Domänen ermöglichen. Dafür muss es seine Daten in einer Infrastruktur aufbewahren, die Nutzer*innen Analysen der Daten ermöglicht. Eine etablierte Möglichkeit ist das Data Warehouse [BG13]. Dessen Grundprinzip ist, alle ankommenden Daten zu transformieren, um sie in eine vordefiniertes, gemeinsamen Schema zu bringen. Anschließend können Nutzer*innen auf den Daten Analysen durchführen. Dadurch werden nur die vordefinierten Anwendungsfälle unterstützt, während Anwendungsfälle, die in der Zukunft noch auftreten könnten, außen vor sind. Wenn eine neue Datenquelle hinzugefügt wird oder sich das Schema einer bestehenden Quelle verändert, müssen die Transformationen neu definiert werden. Das Durchführen der Transformationen stellt zudem einen großen Rechenaufwand dar. Diese Nachteile zeigen sich besonders in Anwendungsfällen, in denen viele heterogene, sich häufig verändernde Datenquellen eine enorme Menge an Daten produzieren.

Der Data Lake wurde als Alternative hierzu entwickelt [Dix10]. Die Daten werden bei ihrer Aufnahme in ihrem Rohformat im Lake abgelegt [RZ19]. Dadurch werden alle Informationen erhalten die sich potentiell in den Daten befinden. Bevor eine Nutzer*in Daten für ihre Analyse verwenden kann, muss sie diese im Lake finden und aufbereiten, sodass die Anforderungen ihrer Analyse erfüllt sind. Um diese schwierige Aufgabe zu erleichtern, kann der Data Lake, in sogenannten Zonen, intern strukturiert werden. Zusätzlich zu dem Bereich in dem die Rohdaten aufbewahrt werden, können Zonen ergänzt werden die Daten in verschiedenen Stufen der Aufbereitung enthalten. So können die Nutzer*innen direkt auf aufbereitete Daten zugreifen [GGH+20]. Allerdings überfordert

diese Aufgabe auch die Betreiber*innen des Data Lakes. Daher gibt es Ansätze, die es ermöglichen, Domänenwissen in einfacher Weise als Verarbeitungsanweisungen zu definieren [SBE+22b] und mit denen sich nützliche Verarbeitungsschritte für gegebene Aufgaben auffinden lassen [SES23]. Dennoch erweist sich bei einem zentralistischem Ansatz wie dem Data Lake die enorme Menge der zu betreuenden Daten als Problem [MCS22].

Data-Mesh-Prinzipien

Die Grundidee des Data Mesh ist, die Herausforderungen, die mit zentralisierten Systemen wie dem Data Warehouse und dem Date Lake anfallen, mithilfe von Dezentralisierung zu umgehen [Bal23b]. Dafür hat der Data Mesh vier Prinzipien, die im Folgenden erläutert werden.

Domänenverantwortung: Anstatt dass ein zentrales Team für die Betreuung aller Daten zuständig ist, wird die Verantwortung für die Daten nach der organisationalen Struktur eingeteilt. Das bedeutet, dass jede Domäne für die Daten in ihrem Besitz zuständig ist [Deh22]. Dies ist von Vorteil, da diese Domäne das Fachwissen hat, welches für die optimale Wartung und Aufbereitung der Daten notwendig ist. Indem die Daten direkt an ihrer Quelle aufbereitet werden, wird die Qualität der Daten bereits zu diesem Zeitpunkt sichergestellt. Dadurch sind die Daten, die von der Domäne zur Verfügung gestellt werden, direkt brauchbar und konsumierbar. Es bedeutet auch, dass jede Domäne ihre Daten in einer beliebigen Infrastruktur speichern kann. Bei Bedarf können Domänen die Daten anderer Domänen anfragen und bekommen diese zur Verwendung bereitgestellt [Deh22].

Daten als Produkt: Das zweite Prinzip bezieht sich auf die Art wie die Daten im Data Mesh aufbereitet und zur Verfügung gestellt werden. Sie werden den Nutzer*innen als sogenannte Datenprodukte zur Verfügung gestellt [Bal23a]. Die Daten, die domänenübergreifend benötigt werden, werden als Datenprodukt aufbereitet. Das bedeutet, dass die Daten explizit für den Zugriff und die Verwendung in anderen Domänen vorbereitet wurden. Es ist möglich, dass auf Anfrage einer anderen Domäne ein neues Datenprodukt erstellt wird. Da die zukünftigen Anwendungsfälle für die Daten aber im Allgemeinen unbekannt sind, sollten möglichst keine Annahmen über die zukünftige Nutzung der Daten getroffen werden. Stattdessen sollten sich die Datenprodukte möglichst an den Rohdaten orientieren, damit sie flexibel einsetzbar sind. Die Domäne muss entscheiden, welche Datensätze zu Datenprodukten aufbereitet werden sollen. Jedes Datenprodukt einer Domäne hat einen Verantwortlichen, einen sogenannten Data Owner , der für das Datenprodukt verantwortlich ist und sich darum kümmert, dass das Datenprodukt beispielsweise die geforderten Qualitätsstandards und gesetzliche Vorschriften einhält [Deh22]. Ein qualitativ hochwertiges Datenprodukt soll dabei eine Reihe an Anforderungen erfüllen, die sogenannten DAUTNIVS-Eigenschaften [Deh22]. Sie sind in Abbildung 4.1 dargestellt. Die Eigenschaften wurden so gewählt, dass sie eine gute Benutzererfahrung fördern. Jeder Buchstabe entspricht einer Eigenschaft. **D**iscoverable bedeutet, dass ein Datenprodukt auffindbar sein muss, zum Beispiel durch einen Datenkatalog. **A**ddressable beschreibt, dass die Datenprodukte zu jeder Zeit mithilfe einer einzigartigen Adresse adressierbar sein müssen. Außerdem müssen die Daten **U**nderstandable sein. Das bedeutet Nutzer*innen müssen den Inhalt und die Präsentation der Daten nachvollziehen können. Damit ein Datenprodukt für wichtige Analysen verwendet werden kann, muss es **T**rustworthy also vertrauenswürdig sein. Auf Datenprodukte mit der Eigenschaft **N**atively Accessible können die Nutzer*innen in ihrer präferierten Sprache bzw. mit ihren präferierten Werkzeugen zugreifen. Dadurch soll den Nutzer*innen die Arbeit mit den Datenprodukten erleichtert werden. **I**nteroperable bedeutet, dass die Datenprodukte aus unterschiedlichen Domänen ohne Schwierigkeiten miteinander verbunden werden können.

Die Eigenschaft **V**aluable hilft bei der Entscheidung, ob ein bestimmtes Datenprodukt erstellt werden sollte oder nicht. Nur wenn ein Datenprodukt für sich allein einen Wert hat bzw. ein Wert damit generiert werden kann, ist es lukrativ es zu erstellen und zu warten. Datenprodukte die keinen Wert mehr besitzen, sollten nicht weiter betreut werden. Die letzte Eigenschaft **S**ecure beinhaltet die Sicherheitsaspekte der Datenprodukte. Sie besagt, dass eine Balance zwischen der Domänenverantwortung und einem zentralen Sicherheitsstandard gefunden werden muss. Während die Zugriffskontrolle zentral für den ganzen Data Mesh festgelegt werden kann, muss die Umsetzung lokal bei den Datenprodukten vorgenommen werden, damit das Datenprodukt eine eigenständige Einheit ist. Dabei handelt es sich allerdings nur um eine generische Beschreibung der Privacy im Data Mesh und die Bitte, die Privacy nicht außer Acht zu lassen. Allerdings fehlen konkrete Hinweise bezüglich der Umsetzung im Data Mesh [Deh22].

Self-Service-Datenplattform: Wenn ein Datenprodukt bereit ist konsumiert zu werden, wird es über eine zentrale Plattform zur Verfügung gestellt. Diese soll den Data Owner der Datenprodukte entlasten, indem sie die Komplexität seiner Aufgaben vereinfacht. Hierzu gehören die Erstellung der Datenprodukte, die Verwaltung der benötigten Infrastruktur und die Wartung der Datenprodukte. Über die Plattform können außerdem Nutzer*innen auf die Datenprodukte der Domänen zugreifen. Dadurch können die Datenprodukte über die Grenzen der Infrastruktur der Domäne und sogar über die Organisation hinaus geteilt werden. Dafür müssen die verschiedenen Infrastrukturen integriert werden um eine möglichst reibungslose Zusammenarbeit zu erreichen [Deh22].

Federated Data Governance: Für Aspekte, die den ganzen Data Mesh betreffen, werden globale Regeln festgelegt. Darunter fallen beispielsweise auch globale Privacy-Richtlinien oder Richtlinien für die Zugriffskontrolle. Diese werden von einem föderalen, domänenübergreifenden Team festgelegt [BKK+23a]. Trotzdem haben die Domänen weiterhin lokal die Verantwortung. Die Data Owner der Datenprodukte müssen neben den anderen Managementaufgaben auch die Umsetzung der globalen Richtlinien sicherstellen [GKD+23]. Die Menge der globalen Regeln sollte sich auf ein Minimum beschränken, um das Prinzip der Domänenverantwortung nicht zu untergraben [Deh22].

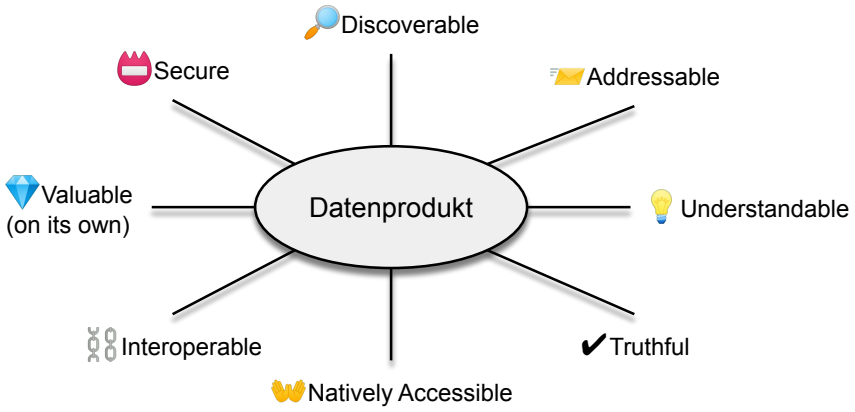

Abbildung 4.1: Die DAUTNIVS-Prinzipien, die von Datenprodukten umgesetzt werden sollen.

Aus den gewonnenen Erkenntnissen von Kapitel 2 ist ersichtlich, dass sich der Data-Mesh-Ansatz für den vorgestellten Anwendungsfall eignet. Die komplexe organisationale Struktur des Krankenhauses umfasst unterschiedliche Domänen mit verschiedenen Anforderungen an die Datenhaltung. Daher ist die Übergabe der Verantwortung an die Domänen ein sinnvoller Schritt. Der Data-Mesh-Ansatz enthält keine Vorgaben hinsichtlich der Infrastrukturen, die die Domänen zum Speichern ihrer Daten verwenden. Dadurch kann eine große heterogene Datenmenge beliebig in den Infrastrukturen aufbewahrt werden, die die Domänen präferieren. Der Datenaustausch zwischen Domänen ist ein wichtiger Aspekt der vorgestellten Szenarien. Genau diese Arbeitsweise unterstützt das Konzept der Datenprodukte. Da hauptsächlich mit persönlichen Gesundheitsdaten gearbeitet wird, ist die Privacy eine zentrale Anforderung des Anwendungsfalls. Sie ist in den geforderten Eigenschaften eines Datenprodukts enthalten. Allerdings wird nicht spezifiziert, wie die Einhaltung von Privacy-Forderungen im Data-Mesh-Ansatz erfolgt. Die bisherige Literatur stellt nicht klar, inwiefern unterschiedliche Aspekte der Privacy sich auf den Data-Mesh-Ansatz auswirken und wie sie im Rahmen der vier Prizipien erfüllt werden können. Deshalb wird die Erfüllung der Privacy in Data Meshes in dieser Arbeit genauer betrachtet.

4.2 Privacy

Besonders seit dem Inkrafttreten der DSGVO [EU16] haben Unternehmen und Organisationen mehr Verantwortung hinsichtlich des Schutzes der Daten, die sie über ihre Kund*innen sammeln. Deshalb ist die Umsetzung des Privacy-by-Design-Konzepts, also Datenschutz durch Technikgestaltung und durch datenschutzfreundliche Voreinstellungen, ein zentrales Thema. Das bedeutet, dass der Datenschutz bereits bei der Entwicklung von Lösungen zur Datenverarbeitung berücksichtigt werden muss. Es müssen sowohl die gesetzlich vorgegebenen, als auch die individuellen Privacy-Forderungen der Betroffenen eingehalten werden. Die Betroffenen werden im Folgenden als Data Subjects bezeichnet. Eine Möglichkeit zur Einhaltung der Privacy-Forderungen ist die Datenvermeidung. Außerdem müssen bereits vorhandene Daten vor unberechtigtem Zugriff Dritter geschützt werden. Dafür können Techniken angewendet werden, die die Daten verändern, sogenannte Privacy-Filter [SGB+22]. Beispielweise entfernen sie Teile der Daten oder wandeln sie ab, sodass sie ungenauer werden. Die Effektivität der Filter und das Risiko, das ein Datensatz mit sich bringt, kann mithilfe von Privacy-Metriken gemessen werden. In den folgenden beiden Abschnitten werden Privacy-Filter und Privacy-Metriken vorgestellt, die zu diesem Zweck angewendet werden können.

Privacy-Filter

In diesem Abschnitt werden vier Privacy-Filter in ihren Grundzügen vorgestellt. Dabei handelt es sich um eine kleine Auswahl der großen Menge an existierenden Privacy-Filtern. Sie können verwendet werden, um Daten zu verbergen bzw. zu verändern. Mit ihrer Hilfe können die Privacy-Forderungen aus Kapitel 2 umgesetzt werden. Außerdem wird das Risiko, das ein Datensatz für die Privacy eines Data Subjects darstellt, gesenkt. Zunächst werden vertikale und horizontale Filter vorgestellt und anschließend die Generalisierung und Aggregation.

Vertikale & horizontale Filter

Mithilfe von vertikalen Filtern können einzelne Attribute bzw. Spalten ganz aus einem Datensatz entfernt werden. In der relationalen Algebra sind sie im Projektionsoperator definiert [Cod72]. Die deklarativen Sprachen wie SQL setzen sie zum Beispiel als SELECT Operator um [Dat89]. Beispielsweise soll ein Datensatz, der die Behandlungsdaten (siehe Abschnitt 2.2) von vielen Patient*innen enthält, pseudonymisiert werden. Hierfür müssen zunächst direkt identifizierenden Attribute, zum Beispiel der Name, entfernt werden. Dafür kann ein vertikaler Filter eingesetzt werden, indem alle Attribute außer „Name“ ausgewählt werden. Dadurch wird „Name“ herausgefiltert. Anschließend wird eine ID eingeführt, mit der die Daten, die von einer Patient*in stammen, einander zugeordnet werden können.

Ein horizontaler Filter wird verwendet, um bestimmte Tupel bzw. Reihen eines Datensatzes auszuwählen. In der relationalen Algebra ist er als Selektion, auch Restriktion genannt, definiert [Cod72]. In SQL wird er als WHERE Operator umgesetzt [Dat89]. Damit können alle Reihen selektiert werden, die eine bestimmte Bedingung erfüllen. Beispielsweise sollen aus dem oben genannten Datensatz die Tupel aller Patienten herausgesucht werden, die mindestens 35 Jahre alt sind. Dafür muss ein vertikaler Filter angewendet werden, der alle Tupel selektiert, deren Wert des Attributs „Alter“ ≥ 35 ist.

Um ein Attribut aus einem Teil des Datensatzes zu entfernen, können vertikale und horizontale Filter kombiniert werden. Beispielsweise sollen die „Details“ der Behandlungen entfernt werden, bei denen eine „Diagnose“ Teil des Attributs „Leistung“ ist. Hierfür werden zunächst mithilfe eines horizontalen Filters die Tupel selektiert, deren „Leistung“ eine „Diagnose“ enthält. Anschließend werden mit einer Projektion alle Attribute außer „Details“ ausgewählt. Es ist ersichtlich, dass vertikale und horizontale Filter eingesetzt werden können, um spezifische Daten aus einem Datensatz zu entfernen. So können sensitive Daten entfernt werden, um eine Privacy-Forderung zu erfüllen.

Generalisierung

Vertikale und horizontale Filter haben den Vorteil, dass sie sich auf alle (semi-)strukturierten Daten anwenden lassen [Sta15]. Allerdings können die Auswirkungen auf die Datenqualität gravierend sein- Deshalb sind dedizierte Filter, die auf einen bestimmten Datentyp spezialisiert sind, insbesondere beim Umgang mit Gesundheitsdaten vorzuziehen [SBE+20; SSM23]. Die Generalisierung ist ein solcher dedizierter Filter. Anstatt ein Attribut ganz oder teilweise zu entfernen, werden die Werte des Attributs weniger präzise gemacht [ERM15]. Beispielsweise soll der pseudonymisierte Datensatz mit Behandlungsdaten (siehe Abschnitt 2.2) weiter bearbeitet werden. Es gibt eine Privacy-Forderung die besagt, dass es nicht möglich sein soll das „Alter“ der Patient*innen genau abzulesen. Anstatt das Attribut komplett zu entfernen und damit alle Informationen zu verlieren, kann es generalisiert werden. Dafür werden Intervalle festgelegt, denen das Alter aller Patient*innen zugeordnet werden kann (0-20, 21-40, 41-60, etc.). Anschließend werden in jedem Tupel der bisherige Wert von „Alter“ durch das zugeordnete Intervall ersetzt. Dadurch ist die Privacy-Forderung erfüllt. Trotzdem können noch Informationen über die Altersgruppe der Patient*innen gewonnen werden.

Aggregation

Eine weitere Option um das Risiko von sensitiven Daten zu reduzieren, ist die Aggregation. Diese fasst einzelne Werte mithilfe einer Operation, beispielsweise der Summe, zu einem Wert zusammen. Dieser Filter eignet sich für Szenarien, in denen die gesuchte Information als aggregierter Wert dargestellt werden kann. Anstelle des eigentlichen Datensatzes wird daher nur eine aggregierte Version der Daten weiter- und preisgegeben [ANTK23]. Beispielsweise verbietet es die Privacy-Forderung der Apotheke (siehe Abschnitt 2.2), dass der „Verwendungszweck“ in den originalen Bestelldaten eingesehen wird. Eine Möglichkeit wäre, den Datensatz so zu bearbeiten, dass alle „Verwendungszwecke“ verborgen oder entfernt wird. Da die Einkaufsabteilung jedoch nur an der Anzahl der bestellten Medikamente interessiert ist, gibt es eine andere Lösung. Anstatt den kompletten Datensatz zu übergeben, kann der Data Owner des Datensatzes nur die gewünschten Informationen übergeben. Dafür werden die Bestellungen für die einzelnen Medikamente aggregiert und summiert, so dass eine Liste mit der Anzahl der verbrauchten Einheiten pro Medikament entsteht. Es ist ersichtlich, dass die Aggregation verwendet werden kann, um Privacy-Forderungen zu erfüllen und darüber hinaus die Preisgabe von Informationen stark zu reduzieren [KS21].

Privacy-Metriken

In diesem Abschnitt werden drei Privacy-Metriken in ihren Grundzügen vorgestellt. Dabei handelt es sich um eine kleine Auswahl der großen Menge an existierenden Privacy-Metriken. Mit ihrer Hilfe können verschiedene Aspekte der Privacy eines Datensatzes gemessen werden. Dadurch kann die Effektivität von Privacy-Filtern und das Risiko das ein Datensatz mit sich bringt, beurteilt werden. Die Erfüllung der Privacy-Forderungen durch die Privacy-Filter kann so überprüft und sichergestellt werden.

Verhältnis unvollständiger Daten zur Gesamtmenge

Ein Indikator dafür, wie viel Risiko ein Datensatz für die Privacy bedeutet, ist seine Vollständigkeit. Die Vollständigkeit ist eine Möglichkeit, um die Menge an Informationen in einem Datensatz zu messen. Ein Datensatz mit hoher Vollständigkeit enthält potentiell mehr Informationen als der gleiche Datensatz aus dem Teile entfernt wurden [PLW02]. Das bedeutet jedoch auch, dass er ein höheres Risiko für die Privacy darstellt. Beispielsweise können aus einem tabellarischen Datensatz einzelne Werte oder ganze Tupel und Attribute entfernt werden. Dadurch wird die Menge der ableitbaren Informationen reduziert, wodurch sich die Privacy erhöht. Zum Beispiel soll die Privacy des Datensatzes mit den Behandlungsdaten (siehe Abschnitt 2.2) geprüft werden. Es werden die Einträge einiger Patient*innen komplett entfernt und zusätzlich die Werte des Attributs „Details“ für einige andere Patient*innen gelöscht. Dadurch wird das Verhältnis der unvollständigen Daten zur Gesamtmenge größer. Auch die Privacy steigt, da nun weniger Informationen in dem Datensatz enthalten sind. Es ist ersichtlich, dass das Verhältnis der unvollständigen Daten zur Gesamtmenge eine geeignete Metrik zur Abschätzung der Privacy ist.

k-Anonymity

Die k-Anonymity ist eine geläufige Metrik, die bewertet, wie einfach einzelne Individuen in einem Datensatz identifiziert werden können [Sam01]. Hierfür werden die Attribute eines Datensatzes als identifizierende und sensitive Attribute benannt. Mithilfe der identifizierenden Attribute kann ein einzelnes Individuum identifiziert werden. Die sensitiven Attribute sollen den Individuen nicht zugeordnet werden können [Sam01]. Ein Datensatz enthält die Behandlungsdaten (siehe Abschnitt 2.2) von vielen Patient*innen. Hier ist das sensitive Attribut die „Details“ und alle anderen Attribute sind identifizierende Attribute. Um die k-Anonymity zu berechnen, wird der Datensatz anhand der identifizierenden Attribute in Äquivalenzklassen unterteilt. Innerhalb der daraus resultierenden Klassen sind alle identifizierenden Attribute identisch, nur die sensitiven Attribute unterscheiden sich. Sobald eine Klasse die Einträge mehrerer Patienten mit unterschiedlichen „Details“ enthält, kann nicht herausgefunden werden, welche zu der gesuchten Patient*in gehören. Damit die Äquivalenzklassen die Einträge mehrerer Patient*innen enthalten können, müssen vorab die direkt identifizierenden Attribute, wie die „Versichertennummer“ oder der „Name“, entfernt werden. Ein Datensatz ist k-anonym, wenn die Größe aller Äquivalenzklassen mindestens k beträgt [Sam01].

l-Diversity

Bei der l-Diversity handelt es sich um eine Erweiterung der k-Anonymity [MKGV07]. Sie adressiert eine Einschränkung der k-Anonymity. Angenommen in einer Äquivalenzklasse der k-Anonymity sind alle Vorkommen der sensitiven Attribute gleich. In diesem Fall kann der Wert der sensitiven Attribute für die gesuchte Person, unabhängig von der Größe der Äquivalenzklasse, ermittelt werden [MKGV07]. Ein Datensatz enthält die Behandlungsdaten (siehe Abschnitt 2.2) von vielen Patient*innen. Das sensitive Attribut „Details“ enthält in einer Äquivalenzklasse für alle Patient*innen den Wert „Impfung COVID-19“. Der Eintrag der gesuchten Patient*in, die sich in dieser Äquivalenzklasse befinden muss, kann nicht genau identifiziert werden. Trotzdem ist klar, dass sie eine Impfung gegen COVID-19 erhalten hat. Um dieses Problem zu beheben, verwendet die l-Diversity eine andere Bedingung als die k-Anonymity. Ein Datensatz ist l-divers, wenn jede Äquivalenzklasse mindestens l unterschiedliche Werte für ein sensitives Attribut enthält [MKGV07].

Gewonnene Erkenntnisse: Aus Abschnitt 4.1 ist ersichtlich, dass sich der Data-Mesh-Ansatz für den vorgestellten Anwendungsfall eignet. Die komplexe organisationale Struktur des Krankenhauses eignet sich für die Übergabe der Datenverantwortung an die Domänen. Jede Domäne kann die für ihre Anforderungen passenden Infrastrukturen verwenden, wodurch eine große, heterogene Datenmenge so aufbewahrt werden kann, wie die Domänen es für richtig halten. Zudem unterstützt der Data-Mesh-Ansatz nativ den Datenaustausch zwischen Domänen, welcher für die Szenarien des Krankenhauses essentiell ist. Allerdings stellt die bisherige Literatur nicht klar, inwiefern sich unterschiedliche Aspekte der Privacy auf den Data-Mesh-Ansatz auswirken und wie sie erfüllt werden können. Insbesondere der Umgang und die Erfüllung von Privacy-Forderungen sind ein

wichtiger Aspekt der Anforderungen des Anwendungsfalls (siehe Kapitel 3). Da diese Aspekte bisher nicht ausreichend berücksichtigt wurden, gilt es sie zu adressieren. Im folgenden Kapitel werden deshalb hierzu die verwandten Arbeiten untersucht.

5 Verwandte Arbeiten

In diesem Kapitel werden die verwandten Arbeiten zu verschiedenen Aspekten der Privacy untersucht. Um die Privacy von Nutzer*innen durchsetzen zu können, werden zunächst deren Privacy-Forderungen benötigt. Deshalb wird als Erstes die Erhebung von Privacy-Forderungen untersucht. Dabei werden besonders Arbeiten betrachtet, die sich damit beschäftigen, inwieweit Nutzer*innen bei der Erhebung ihrer eigenen Privacy-Forderungen unterstützt werden können. Anschließend werden bestehende Ansätze zur Beschreibung von Privacy-Forderungen untersucht. Eine Beschreibung ist notwendig um die Privacy-Forderungen abbilden und speichern zu können. Danach werden Techniken zur Prüfung und Durchsetzung von Privacy-Forderungen untersucht. Abschließend wird geprüft in welchem Maß sich diese Aspekte im Kontext des Data-Mesh-Ansatz auswirken und wie sie umgesetzt werden.

Erhebung von Privacy-Forderungen

Bei der Entwicklung von Software-Systemen müssen Privacy-Forderungen für den Datenschutz erhoben werden. Deng et al. [DWS+11] stellen ein Framework zur Unterstützung der Erhebung von Privacy-Forderungen für Software-Systeme vor. Privacy-Risiken werden mit den Schritten des Datenflussdiagramms des Systems assoziiert. Privacy-Risiken, die für das System relevant sind, werden zu Misuse-Cases zusammengefasst, für die Privacy-Forderungen erhoben werden. Für die Privacy-Forderungen werden anschließend angemessene Privacy-Enhancing-Techniques ausgewählt. Dabei werden alle Schritte manuell durchgeführt.

Um den manuellen Aufwand zu verringern, präsentieren Miyazaki et al. [MMZ08] eine computergestützte Technik zur Erhebung von Datenschutzanforderungen für Software-Systeme. Sie soll Software-Ingenieur*innen und deren Stakeholder bei der Erhebung von Privacy-Forderungen unterstützen. Dafür stellt sie einen Fragebogen zur Verfügung, den die Beteiligten ausfüllen. Auf dessen Grundlage sucht das Tool nach passenden Privacy-Forderungen in seiner Datenbank und empfiehlt sie den Software-Ingenieur*innen.

Die Nutzer*innen von Software-Systemen gehören zu den Stakeholdern, unter anderem bezüglich des Datenschutzes. Chhetri und Motti [CM19] führen deshalb eine Analyse von Online-Reviews zu Smart-Home-Geräten durch. Sie finden heraus, dass Nutze*innen sich klare Aussagen von Herstellern hinsichtlich der Richtlinien zur Datensammlung, Datenteilung und zum Datenschutz wünschen. Zudem möchten die Nutzer*innen Kontrolle über die Daten, die über sie gesammelt werden. Sie möchten sie selbst verwalten und löschen können. Nutzer*innen haben offensichtlich Interesse am Schutz ihrer Privacy und möchten ihre Daten aktiv schützen.

Hierfür müssen Nutzer*innen informiert werden, wenn ihre Daten verwendet werden. Murmann et al. [MRF19] untersuchen, inwieweit sich Benachrichtigungen eignen, um Nutzer*innen über Datennutzung zu informieren. Zum einen ermitteln sie, ob Nutzer*innen generell Benachrichtigungen

zur Datennutzung erhalten möchten. Außerdem untersuchen sie, wie die Möglichkeit, eingreifen zu können, die Bereitschaft benachrichtigt zu werden verändert. Die Mehrheit der Nutzer*innen möchte informiert werden, wünscht sich aber mehr Details in den Benachrichtigungen. Zudem wissen Nutzer*innen nicht wie sie eingreifen können, sie wünschen sich Orientierungshilfen, die sie dabei unterstützen, ihre Privacy-Rechte wahrzunehmen.

Eine Möglichkeit Nutzer*innen zu unterstützen ist, ihnen passende Privacy-Forderungen zu empfehlen. Mehrpouyan et al. [MAP17] untersuchen, ob Persönlichkeitsmerkmale für die automatische Erhebung von Privacy-Forderungen verwendet werden können. Sie kommen zu dem Ergebnis, dass sich die Vorhersagen von Machine-Learning-Algorithmen zu passenden Privacy-Forderungen durch den Input von Persönlichkeitsmerkmalen signifikant verbessern. Daraus ziehen sie den Schluss, dass es möglich ist, die Empfehlung von Privacy-Forderungen zu automatisieren. Dadurch können Nutzer*innen bei der Erhebung ihrer Privacy-Forderungen unterstützt werden.

Stach and Steimle [SS19] stellen hierfür EPICUREAN vor. Dabei handelt es sich um einen empfehlungsbasierten Ansatz zur Erhebung von Datenschutzanforderungen. Damit werden Nutzer*innen auf Basis ihrer Privacy-Forderungen mithilfe einer Wissensbasis weitere passende Privacy-Forderungen vorgeschlagen. Zunächst wird diese Wissensbasis mit der Hilfe von Expert*innen aufgebaut, die sensitive Muster in den Daten definieren und sie mit Schlüsselwörtern und Privacy-Forderungen verknüpfen. Nutzer*innen definieren ihre Privacy-Forderungen in natürlicher Sprache. Anschließend werden die Schlüsselwörter extrahiert und verwendet, um in der Wissensbasis die damit assoziierten Muster zu finden. Die damit verknüpften Privacy-Forderungen werden den Nutzer*innen vorgeschlagen. EPICUREAN kann verwendet werden, um Nutzer*innen bei der Erhebung ihrer Privacy-Forderungen automatisiert zu unterstützen.

Die Erhebung von Privacy-Forderungen ist ein großes und komplexes Problem, das in zahlreichen wissenschaftlichen Arbeiten behandelt wird. Eine Auswahl dieser Arbeiten wurde hier diskutiert. Ansätze wie EPICUREAN können dazu verwendet werden, die Erhebung der Privacy-Forderungen semi-automatisch zu unterstützen.

Beschreibung von Privacy-Forderungen

Die Beschreibung der Privacy-Forderungen ist notwendig um sie abbilden und speichern zu können. Eichler et al. [EGG+21] beschäftigen sich mit dem Metadaten-Management in Data Lakes. Sie präsentieren die Ziele, Lösungsansätze und Probleme einer realen Firma. Um Self-Service für Daten zu realisieren wird Datentransparenz benötigt. Dafür ist eine gute Dokumentation der Daten notwendig. Deshalb muss unter anderem ein gemeinsames Metamodell entstehen, mit dem alle Daten beschrieben werden. Da Privacy-Forderungen Teil der Metadaten eines Datensatzes sind, müsste für sie ebenfalls ein allgemeingültiges Beschreibungsmodell erstellt werden.

Eine Möglichkeit ist, ein solches Modell mithilfe von Experten auszuarbeiten. Stach et al. [SDM+18] stellen den Zugriffskontrollmechanismus PATRON vor. Nutzer*innen beschreiben ihre Privacy-Forderungen in natürlicher Sprache. Diese werden dann von Expert*innen formalisiert. Dafür werden Daten in sensitive und nicht-sensitive Attribute eingeteilt. Nutzer*innen benötigen also kein Domänenwissen, die Expert*innen nutzen ihr Wissen zur manuellen Formalisierung und stellen sicher, dass die Privacy-Forderungen erfüllt werden.

Alshugran und Dichter [AD14] stellen eine Methode zur Erhebung von Privacy-Forderungen auf Grundlage des Health Insurance Portability and Accountability Act (HIPAA) vor. Sie soll helfen den Gesetzestext zu interpretieren und Privacy-Forderungen auf dessen Grundlage zu formulieren. Dafür werden relevante Paragraphen aufbereitet und verdeutlicht. Anschließend werden organisationale und geschäftliche Privacy-Forderungen hinzugefügt. Aus dieser Menge werden Kontextelemente extrahiert und klassifiziert, die für eine Context-Based-Access-Control (CBAC) verwendet werden können. Die Privacy-Forderungen werden manuell anhand der Gesetzestexte erhoben und über Kontextelemente beschrieben. Dieser Ansatz kann in allen Anwendungsfällen angewendet werden, in denen die Privacy-Forderungen für eine CBAC abgebildet werden sollen.

He und Antón [HA03] präsentieren ihr Framework zur Erhebung von Privacy-Forderungen im Role Engineering. Die Privacy-Forderungen bestehen aus den Kontexten und Pflichten von Entitäten der Roll-Based-Access-Control (RBAC). Das Ergebnis des Erhebungsprozesses ist eine Menge an Rollen und eine Menge an Erlaubnissen, die den Rollen zugeordnet sind. Diese Menge stellt die Privacy-Forderungen dar. Während dieser Beschreibungsansatz unabhängig von der Art der Privacy-Forderungen verwendet werden kann, ist er auf Anwendungsfälle im Role Engineering beschränkt.

Mouratidis et al. [MIKG13] präsentieren ein Framework zur Auswahl eines Cloud Providers auf der Grundlage von Security- und Privacy-Forderungen . Sie definieren eine Beschreibungssprache für Privacy-Forderungen . Ein „Constraint“ ist eine Menge von Beschränkungen die sich auf einen „Actor“ beziehen. Dabei handelt es sich entweder um einen „Security-Constraint“ oder einen „Privacy-Constraint“. Die Möglichkeiten zur Erfüllung der „Constraints“ werden mit verschiedenen Elementen beschrieben. Diese Beschreibungsmöglichkeit für Privacy-Forderungen ist für einen speziellen Anwendungsfall entwickelt worden. Es kann jedoch auch in anderen Anwendungsfällen verwendet werden.

In den bisher vorgestellten Ansätzen wurden die Privacy-Forderungen von Experten erhoben und modelliert. Stach und Mitschang [SM19] präsentieren ACCESSORS, ein datenzentriertes Berechtigungsmodell für das IoT, mit dem Nutzer*innen die Modellierung ihrer Privacy-Forderungen selbst durchführen können. Die Modellierung erfolgt indem eine Applikation einen Zugriffszweck definiert. Dieser wird mit einem Datentyp verbunden, der die angefragte Information enthält. Der Datentyp ist wiederum mit Datenquellen verbunden. Nutzer*innen können Einschränkungen für jede Assoziation aus Applikation und Datentyp definieren. Sie können dabei Unterstützung durch Empfehlungen erhalten, beispielsweise von EPICUREAN [SS19]. Dieser Ansatz funktioniert unabhängig von den zugreifenden Applikationen, da sich die Privacy-Forderungen auf den Datentyp beziehen. Allerdings ist er auf den IoT Kontext beschränkt.

Damit Nutzer*innen ihre Privacy-Forderungen verwalten können, benötigen sie eine Schnittstelle. Hechler et al.[HWW23] sagen, dass die Einhaltung von Datenschutzbestimmungen ein sehr wichtiges Thema ist bzw. wird. Sie erklären, dass diese Prozesse mithilfe von künstlicher Intelligenz (KI), besonders Natural Language Processing (NLP), gelöst werden kann. In ihrer Arbeit zeigen sie eine grafische Oberfläche, in welcher die Datenqualiät aufgeschlüsselt wird. Die Privacy-Forderungen von Nutzer*inne können in ähnlicher Weise in einem Tool dargestellt werden. Darüber könnten die Nutzer*innen ihre Privacy-Forderungen intuitiv verwalten.

Es ist ersichtlich, dass es bereits verschiedene Ansätze zur Beschreibung von Privacy-Forderungen gibt. Die Mehrheit der Beschreibungsansätze ist für konkrete Anwendungsfälle konzipiert. Bisher fehlen Ansätze, die die Beschreibung von Privacy-Forderungen für verteilte komplexe Dateninfrastrukturen, wie beispielsweise Data Meshes, umsetzen.

Prüfung und Durchsetzung von Privacy-Forderungen

Nachdem die Privacy-Forderungen erhoben und abgebildet sind, müssen sie für die jeweiligen Daten geprüft und durchgesetzt werden.

McSherry [McS09] stellen ihre Plattform PINQ vor. Dabei handelt es sich um eine Privacy verstärkende Schicht zwischen Analysten und der Analyse-Engine. Sie verhält sich wie das Interface der darunterliegenden Engine, indem sie Anfragen der Analyst*innen an die Engine weiterleitet. Sie wendet Differential Privacy auf die Ergebnisse der Anfragen an, bevor sie an die Analyst*innen weitergeleitet werden. So können Anfragen auf originalen Daten erfolgen und trotzdem werden Privacy-Forderungen durchgesetzt. Allerdings ist es nicht möglich, dass einzelne Data Subjects individuelle Privacy-Forderungen stellen. Alle Privacy-Forderungen werden auf alle Daten angewendet.

Es ist jedoch ein zentaler Anwendungsfall, dass Nutzer*innen bzw. Data Subjects eigene Privacy-Forderungen stellen können. Dafür stellen Stach et al. [SBE+22a] die maßgeschneiderte Datenaufbereitungszone BARENTS vor. Sie vereinfacht den Aufbereitungsprozess für nicht-IT-Expert*innen. Nutzer*innen werden bei der Definition von Verarbeitungsschritten unterstützt. Die Verarbeitungsschritte werden mithilfe einer Ontologie beschrieben. Es werden die zu bearbeitenden Daten, die Transformationen sowie der Speicherort für das Ergebnis ausgewählt. Zur Umsetzung der Transformationen lädt BARENTS die Daten in ein *pandas* Data Frame, worauf die Transformationen ausgeführt werden. So kann BARENTS zur Umsetzung von Privacy-Forderungen verwendet werden.

Allerdings müssen Nutzer*innen die Verarbeitungsschritte manuell auswählen. Mithilfe des Zugriffskontrollmechanismus PATRON von Stach et al. [SDM+18] kann auch die Auswahl automatisiert werden. PATRON formalisiert Privacy-Forderungen und setzt sie, mithilfe von Privacy-Filtern automatisch um. Es wählt aus mehreren Möglichkeiten zur Umsetzung diejenige aus, die die höchste Datenqualität ergibt.

Stach et al. [SGM20] stellen weiter den Privacy by Design Ansatz DISPEL für das IoT vor. Nutzer*innen können Privacy-Forderungen erstellen, mit denen sie definieren, welche Applikationen auf welche Daten zugreifen dürfen. Dafür wird das ACCESSORS Berechtigungsmodell [SM19] verwendet. Mithilfe von Plugins setzt DISPEL die Privacy-Forderungen direkt an den Datenquellen um. Dafür stellt jede Datenquelle Plugins zur Verfügung, mit denen ihre Datentypen bearbeitet werden können. DISPEL bietet also die Möglichkeit, Privacy-Forderungen unabhängig von zugreifenden Applikationen zu definieren. Allerdings beschränkt es sich auf den IoT Kontext.

Palanisamy et al. [PDTR18] präsentieren ebenfalls einen Ansatz zur Umsetzung von Privacy-Forderungen im IoT. Die Geräte der Nutzer*innen zeichnen einen kontinuierliche Abfolge von Ereignissen auf. Nutzer*innen definieren Abfolgen von Ereignissen, sogenannte Private-Patterns, die verschleiert werden sollen. Die Pattern Based Access Control (PBAC) überarbeitet die Daten,

bevor sie über Middleware an Konsumenten geht. Sie ordnet die Ergeignisse so um, dass alle Private-Patterns verschleiert sind und die Nutzbarkeit der Daten möglichst hoch ist. Dieser Ansatz ist speziell für Anwendungsfälle in denen Complex Event Processing zum Einsatz kommt.

PUPPIES ist ein Ansatz von He et al. [HLK+16] zur Erhaltung der Privacy bei der Freigabe von Bildern. Bevor eine Sender*in ein Bild auf einer Bild-Sharing-Plattform hochlädt, werden Bereiche, die Privacy relevant sind, verschleiert. Dadurch hat die Bild-Sharing-Plattform keinen Zugriff auf diese Bereiche des Bilds. Eine Empfänger*in kann die Verschleierung mithilfe eines Schlüssels, der über einen sicheren Kanal übertragen wird, wieder rückgängig machen. Hierbei handelt es sich um einen Ansatz speziell zur Erfüllung von individuellen Privacy-Forderungen in Bilddaten.

Conti et al. [CNC11] stellen ihr System CRêPE vor, das kontextbezogene Privacy-Richtlinien für Android-Applikationen durchsetzt. Nutzer*innen definieren Kontexte sowie die damit assoziierten Richtlinien. Wenn ein Kontext aktiviert wird, werden dessen Richtlinien für alle laufenden Applikationen durchgesetzt. Wenn eine Applikation eine Berechtigung anfragt, werden die Richtlinien für diese Anfrage durchgesetzt. Mit CRêPE können individuelle Privacy-Forderungen durchgesetzt werden. Allerdings ist der Ansatz auf Android-Applikationen beschränkt.

Ähnlich verhält es sich mit dem Ansatz von Backes et al. [BGH+13]. Es handelt sich um einen Ansatz zum Management von Berechtigungen für Android-Applikationen. Sie verwenden vorgefertigte Security- und Privacy-Richtlinien, die direkt in den Programmcode der Applikationen integriert werden. Nutzer*innen können den verschiedenen Applikationen Berechtigungen erteilen bzw. sie widerrufen und konfigurieren. Allerdings können sie keine neuen, individuellen Privacy-Forderungen erstellen.

Es ist ersichtlich, dass eine Vielzahl von wissenschaftlichen Arbeiten hinsichtlich der Prüfung und Durchsetzung von Privacy-Forderungen existiert. Die Eignung der vorgestellten Ansätze für den Einsatz in Anwendungsfällen unterscheidet sich. Einige sind nur in spezifischen Kontexten verwendbar, beispielsweise im IoT-Kontext [SGM20], für Android-Anwendungen [BGH+13; CNC11] oder für Datenströme [PDTR18]. Für jeden Anwendungsfall muss abgewogen werden, welche Umsetzung sich am besten eignet.

Umsetzung von Privacy in Data Mesh

In den vorangegangen Abschnitten wurden wissenschaftliche Arbeiten vorgestellt, die sich mit der Erhebung, Beschreibung und Durchsetzung von Privacy-Forderungen beschäftigen. In diesem Abschnitt werden verwandte Arbeiten betrachtet, die sich damit beschäftigen wie sich Privacy-Aspekte im Data-Mesh-Ansatz auswirken und wie sie erfüllt werden können.

Zunächst werden Arbeiten vorgestellt, die sich mit dem Konzept des Data-Mesh-Ansatz beschäftigen. Dehghani [Deh22] hat mehrere Arbeiten zu Data Mesh geschrieben und in einem Buch zusammengefasst. Dort werden Aspekte des Data Mesh im Detail beschrieben. Unter anderem die vier Prinzipien, unter welchen Umständen ein Data Mesh eingesetzt wird und wie ein Datenprodukt implementiert wird. Sie sagt, dass die Privacy unter die Secure-Eigenschaft von Datenprodukten fällt. Das domänenübergreifende Governance-Team ist dafür zuständig, die Privacy-Richtlinien festzulegen, da diese den ganzen Mesh betreffen. Die Richtlinien werden allerdings direkt am Datenprodukt vom Data Owner umgesetzt, da das Datenprodukt eine eigenständige Einheit sein

soll. Deshalb müssen die Richtlinien mit den Datenprodukten, für die sie gelten, verknüpft werden. Dehghani legt wert auf Privacy-Aspekte, beschreibt jedoch nicht, wie diese Aspekte umgesetzt werden sollen.

Machado et al. [MCS21] stellen ein Domänenmodell und eine konzeptionelle Architektur für den Data Mesh vor. Die Architektur enthält einen Security-Mechanismus, der für den ganzen Mesh gilt. Er umfasst Autorisierung sowie Authentifizierung von Nutzer*innen. Allerdings lassen sie die Privacy außer Acht. Diese kommt in den Modellen nicht vor. In [MCS22] beschreiben Machado et al., welche Technologien verwendet werden können, um die Konzepte aus [MCS21] umzusetzen. Auch dabei gehen sie nicht auf die Privacy ein. Es existieren also Konzepte und Umsetzungsvorschläge, die die Zugriffskontrolle beinhalten, aber die Privacy nicht berücksichtigen.

In einer Studie hat die PricewaterhouseCoopers GmbH [Pri22] herausgefunden, dass Data Owner die Zugriffsbeschränkungen für ihre Datenprodukte festlegen und umsetzen. Die Richtlinien, die für den ganzen Mesh gelten, werden von einem Governance Officer festgelegt. Die Menge dieser globalen Richtlinien wird minimal gehalten. Sie werden automatisch von der Datenplattform geprüft, bevor ein Datenprodukt verfügbar gemacht wird. Diese Arbeit steht im Kontrast zu Dehghanis Aussagen, dass alle Richtlinien lokal am Datenprodukt umgesetzt werden sollen. PricewaterhouseCoopers beschreiben außerdem, wie IKEA den Data Mesh umsetzt. Sie erwähnen allerdings nicht, wie die Privacy sichergestellt wird oder welche Privacy-Prozesse vorhanden sind. Es handelt sich also um ein Praxisbeispiel ohne Beschreibung von Privacy-Prozessen.

Goedegebuure et al. [GKD+23] beschäftigen sich ebenfalls mit Data Meshes in der Praxis. Sie präsentieren einen Gray Literature Review zum Thema Data Mesh. Dort fassen sie die Ergebnisse der Literatur zu verschiedenen Aspekten des Data Mesh zusammen. Dabei gehen sie an mehreren Stellen auf mit der Privacy verwandte Aspekte ein. Die Global-Governance soll die Standards für Security und Privacy festlegen. Die Local-Governance soll die Regeln für die Zugriffskontrolle der Datenprodukte festlegen, da die Domänen das meiste Fachwissen besitzen. Umgesetzt werden soll die Zugriffskontrolle von der Self-Service Datenplattform. Sie erklären, dass es sowohl an Methodiken als auch an Werkzeugen fehlt, um die Data-Mesh-Prinzipien in der Realität umzusetzen. Die Gray Literature ist sich einig, welche Rollen bzw. Module für die Einhaltung von Privacy zuständig sind. Jedoch gibt es keine Information darüber, mit welchen Prozessen die Privacy-Regelungen erfüllt werden oder wie Privacy-Regelungen modelliert werden können.

Bode et al. [BKK+23a] stellen das Ergebnis von Interviews mit 15 Expert*innen, die mit Data Meshes gearbeitet haben, vor. Die Arbeit beschreibt die bedeutendsten Herausforderungen und die Best-Practices beim Wechsel zum Data-Mesh-Ansatz . Die Expert*innen betonen die Wichtigkeit einer zentralen domänenübergreifenden Steuerungseinheit, die die strategische Planung übernimmt. Die Mitglieder der Steuerungseinheit sollten unter anderem auch Governance-Regeln für die Themen Rund um Privacy festlegen. Es wird nicht genau darauf eingegangen welche Prozesse verwendet werden sollten, um die Privacy-Regelungen zu erfüllen.

Einige Arbeiten befassen sich mit der Data-Mesh-Lösung einer spezifischen Firma. Joshi et al. [JPR21] beschäftigen sich mit der Governance-Strategie für Data Meshes. Sie beschreiben die Strategie auf konzeptioneller Ebene und gehen dabei zum Beispiel auf die Beschreibung der Datenqualität und andere Metadaten ein. Weiter präsentieren sie eine mögliche Umsetzung im Rahmen der Pilotstudie der Saxo Bank. Allerdings wird nicht erwähnt, inwiefern sie die Privacy handhaben. Es handelt sich um ein konkretes Praxisbeispiel, aber es fehlt die Beschreibung von Privacy-Prozessen.

Lei et al. [LPS+22] beschreiben, wie der Data-Mesh-Ansatz als Dateninfrastruktur bei Netflix umgesetzt wird. Netflix verschiebt immer mehr Anwendungsfälle in den Data Mesh. Dort werden die Anwendungsfälle als Pipelines umgesetzt, die Daten aus einer Quelle nehmen, transformieren und anschließend an ihrem Bestimmungsort ablegen. Sie gehen nicht darauf ein, wie der Privacy-Prozess gestaltet ist oder wie er in ihrem Data Mesh umgesetzt wird.

Die Arbeit von Chee [Che21] beschreibt, wie HelloFresh begonnen hat, ihre Dateninfrastruktur in einen Data Mesh zu transformieren. Neben Teams für die Datenprodukte haben sie ein Team für die Dateplattform sowie ein Governance-Team. Allerdings wird auch hier nicht erwähnt, wie die Privacy gehandhabt wird.

Gewonnene Erkenntnisse: Die Literaturrecherche hat gezeigt, dass Privacy-Aspekte im Kontext des Data-Mesh-Ansatzes im Vergleich zu anderen Bereichen (z.B. das IoT) noch nicht ausreichend berücksichtigt wurden. Beispielsweise existieren keine Konzepte für den Prozess der Durchsetzung von Privacy-Forderungen, wenn der Zugriff auf ein Datenprodukt angefragt wird. Für die Erhebung von Privacy-Forderungen gibt es Lösungen, die unabhängig vom Anwendungsfall eingesetzt werden können. Während es generische Beschreibungsmodelle für Privacy-Forderungen gibt, fehlen solche, die speziell auf den Data-Mesh-Ansatz angepasst sind. Lösungen zur Durchsetzung von Privacy-Forderungen sind zahlreich vorhanden. Allerdings sind diese meist für einen spezifischen Anwendungsfall konzipiert, weshalb Anpassungen erforderlich sind, bevor sie im Data-Mesh-Ansatz verwendet werden können. Im folgenden Kapitel wird daher ein eigenes Konzept für die Erfüllung der Privacy-Aspekte im Data-Mesh-Ansatz vorgestellt, das speziell auf die Anforderungen des Data-Mesh-Ansatzes ausgerichtet ist. Dabei werden insbesondere die Aspekte adressiert, die bisher noch nicht ausreichend berücksichtigt wurden.

6 Konzept

In diesem Kapitel wird ein eigenes Konzept für die Erfüllung von Privacy-Aspekten im Data-Mesh-Ansatz vorgestellt. Es bringt die Privacy-Aspekte konzeptionell in den Data-Mesh-Ansatz ein, sodass das Privacy-by-Design-Konzept erfüllt ist. Im Laufe des Kapitels werden die Teile des Konzepts nacheinander erklärt. Zunächst wird der Prozess der Erhebung der Privacy-Forderungen beschrieben (siehe Abschnitt 6.1). Die Privacy-Forderungen eines Data Subjects werden zusammen mit seinen Daten erhoben. Danach folgt der Prozess der Zugriffsanfrage auf ein Datenprodukt (siehe Abschnitt 6.2). Hierbei fragt eine Domäne den Zugriff auf das Datenprodukt einer anderen Domäne an. Anschließend wird der Prozess bei der Erstellung eines neuen Datenprodukts durch die anfragende Domäne vorgestellt (siehe Abschnitt 6.3). Dabei gibt es verschiedene Punkte, die beachtet werden müssen, beispielsweise der Zeitpunkt der Übergabe der Ownership. Abschließend wird der Prozess bei der Fusion mehrerer Datenprodukte beschrieben (siehe Abschnitt 6.4). Dies ist ein Spezialfall der Erstellung eines neuen Datenprodukts, in dem mehrere bestehende Datenprodukte verwendet werden, um ein neues Datenprodukt zu erstellen.

6.1 Erhebung der Privacy-Forderungen

In diesem Abschnitt wird der Prozess der Erhebung von Privacy-Forderungen erklärt. Im vorgestellten Anwendungsfall (siehe Kapitel 2) kommen verschiedene Arten von Privacy-Forderungen vor. Es gibt Privacy-Forderungen, die gesetzlich vorgegeben sind und damit für alle Datenprodukte des Krankenhauses gelten. Außerdem gibt es Privacy-Forderungen, die von den Domänen festgelegt werden und für alle Datensätze dieser Domäne gelten. Auch im Data-Mesh-Ansatz sind Privacy-Richtlinien vorgesehen, die für Domänen oder sogar den ganzen Mesh gelten [Deh22]. In [CM19] hat sich gezeigt, dass Nutzer*innen bzw. Data Subjects ein großes Interesse an ihrem Mitspracherecht zur Verwendung ihrer Daten haben (siehe Kapitel 5). Deshalb müssen auch die Privacy-Forderungen der Data Subjects erhoben werden. Es wird in diesem Kontext davon ausgegangen, dass Data Subjects die Erhebung ihrer Daten nur erlauben, wenn sie der Verarbeitung im Data Mesh grundsätzlich zustimmen. Um die Zahl der Privacy-Forderungen und Überprüfungen zu reduzieren, werden Privacy-Forderungen nur modelliert, wenn es eine Einschränkung gibt. Dadurch stellen die Privacy-Forderungen eine Blacklist dar. Widerspricht ein Data Subject der Verarbeitung der Daten generell, findet die Erhebung gar nicht statt.

Die gesetzlichen Privacy-Forderungen können von einem zentralen Governance-Team modelliert werden, beispielsweise mit der Methode von Alshugran und Dichter [AD14]. Ebenso können die Privacy-Forderungen der Domänen von domäneninternen Governance Teams definiert werden, beispielsweise mithilfe der Technik von Miyazaki et al. [MMZ08]. Da sich Gesetze und Richtlinien innerhalb einer Organisation oder Domäne eher selten ändern, müssen die jeweiligen Privacy-Forderungen selten aktualisiert werden. Deshalb kann diese Aufgabe von den zuständigen

Governance Teams übernommen werden. Im Vergleich dazu findet die Erhebung neuer Privacy-Forderungen von Data Subjects deutlich häufiger statt. Zum einen kann ein Data Subject spontan Änderungen an seinen Privacy-Forderungen vornehmen. Zum anderen passt ein Data Subject seine Privacy-Forderungen bei Bedarf an, wenn neue Daten des Data Subjects erhoben werden. Aus diesem Grund ist es sinnvoll, die Erhebung der Privacy-Forderungen von Data Subjects an die Erhebung ihrer Daten zu koppeln. Eine spontane Änderung der Privacy-Forderungen kann als erneute Erhebung der Daten realisiert werden. Für den bestehenden Datensatz gelten die zuvor definierten Privacy-Forderungen und für den neu erhobenen Datensatz gelten die geänderten Privacy-Forderungen.

Der Erhebungsprozess ist in Abbildung 6.1 dargestellt. Er beginnt damit, dass die Intention besteht, Daten eines Data Subjects zu erheben **(1)**. Das Data Subject hat daraufhin die Möglichkeit, Privacy-Forderungen zu definieren **(2)**. Dies geschieht informell, beispielsweise in natürlicher Sprache, da Data Subjects oft nicht das benötigte Fachwissen besitzen, um eine formale Beschreibung zu erstellen. Anschließend werden die Privacy-Forderungen formalisiert, damit sie später besser verarbeitet und analysiert werden können **(3)**. Die Formalisierung von informellen Privacy-Forderungen ist allerdings out of scope für diese Arbeit. Zur Lösung dieses großen und komplexen Problems sei daher auf existierende Lösungen wie [AD14] oder EPICUREAN [SS19] verwiesen, die im Rahmen der verwandten Arbeiten in Kapitel 5 diskutiert wurden. Nachdem sie formalisiert wurden, werden die Privacy-Forderungen gespeichert **(4)**. Sie werden getrennt von den Daten an einer zentralen Stelle gespeichert, wo sie nicht unrechtmäßig verändert werden können. Zudem müssen alle Domänen Zugriff auf die Privacy-Forderungen haben, da sie von den Data Owner überprüft werden. Dafür kann zum Beispiel die Blockchain-Technologie verwendet werden [DCL23a; DCL23b]. Die erhobenen Daten werden in der Infrastruktur der Domäne gespeichert **(5)**. Die Forderungen des Data Subjects sind über den Hashwert des entsprechenden Blocks mit dem Datensatz verbunden. Zuletzt wird der Datensatz verwendet, um ein Datenprodukt zu erstellen **(6)**. Das Datenprodukt enthält einen oder mehrere Datensätze oder Teile davon. Für Datensätze bzw. Teile von Datensätzen können Privacy-Forderungen gelten. Beispielsweise sind die Daten eines bestimmten Data Subjects Teil eines Datensatzes. Das Data Subject hat Privacy-Forderungen, die für seine Daten, also diesen Teil des Datensatzes, gelten. Ist ein Datensatz bzw. ein Teil eines Datensatzes in dem Datenprodukt

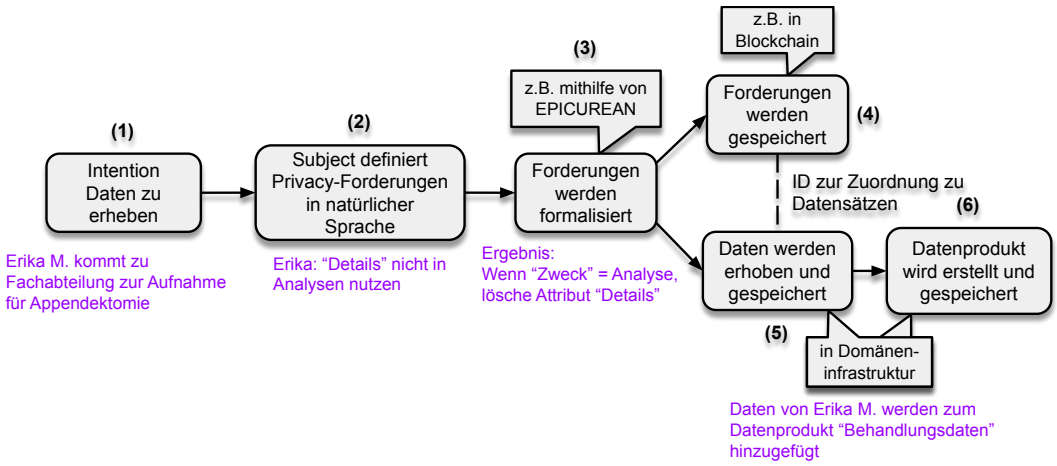

Abbildung 6.1: Prozess der Datenerhebung inklusive der Erhebung von Privacy-Forderungen des Data Subjects.

enthalten, gelten diese Privacy-Forderungen weiterhin. Deshalb werden alle drei Arten von Privacy-Forderungen über die Hashwerte mit dem Datenprodukt verknüpft. Das Datenprodukt wird ebenfalls in der Infrastruktur der Domäne gespeichert und einem Data Owner aus der Domäne zugeordnet.

In Abbildung 6.1 ist ein beispielhafter Ablauf mithilfe von lila Annotationen dargestellt. Er stammt aus dem Szenario der Optimierung der Dienstpläne einer Station in Abschnitt 2.2. Erika Mustermann kommt zur Aufnahme für ihre Appendektomie in die Chirurgie, wo ihre Daten erhoben werden **(1)**. Dort definiert sie ihre Privacy-Forderungen **(2)**. Sie möchte nicht, dass die Details ihrer Behandlung zu Analysezwecken verwendet werden. Anschließend wird ihre Privacy-Forderung formalisiert **(3)**. Das Ergebnis ist: **„**Wenn 'Zweck' != 'Analyse', lösche Attribut 'Details'**“**. Die erhobenen Daten und die Privacy-Forderung werden gespeichert. Sie können über den Hashwert des Blocks der Privacy-Forderung verknüpft werden. Später wird der Datensatz von Erika Mustermann zusammen mit den Datensätzen anderer Patienten für das Datenprodukt „Behandlungsdaten“ verwendet. Dabei werden alle Privacy-Forderungen der Datensätze mit dem Datenprodukt verknüpft.

Im nächsten Abschnitt wird der Prozess der Zugriffsanfrage einer anderen Domäne auf ein Datenprodukt beschrieben.

6.2 Zugriffsanfrage auf ein Datenprodukt

In diesem Abschnitt wird der Prozess der Zugriffsanfrage einer anderen Domäne auf ein Datenprodukt beschrieben. Im vorgestellten Anwendungsfall ist der Austausch von Daten zwischen den Domänen essentiell für alle drei Szenarien (siehe Kapitel 2). Der Austausch von Daten in Form von Datenprodukten ist eine zentrale Charakteristik des Data-Mesh-Ansatz. Hierfür müssen Domänen die Datenprodukte anderer Domänen anfragen. Wenn sie Zugriff erhalten, können sie die Daten anschließend für ihre eigenen Analysen verwenden. Keine der bisherigen Arbeiten zum Data-Mesh-Ansatz geht darauf ein, wie dabei die Privacy der Data Subjects aufrecht erhalten wird.

Der Prozess der Zugriffsanfrage ist in Abbildung 6.2 dargestellt. Er beginnt damit, dass eine Domäne den Zugriff auf ein Datenprodukt einer anderen Domäne anfragt **(1)**. Zunächst müssen die Zugriffsrechte allgemein festgestellt werden **(2)**. In Fällen, in denen die übergeordnete Zugriffskontrolle den Zugriff auf die Daten nicht zulässt, müssen die Privacy-Forderungen nicht geprüft werden, da der Zugriff direkt verweigert wird **(3)**. Falls das allgemeine Zugriffsrecht besteht, prüft der Data Owner ob Privacy-Forderungen für das Datenprodukt gelten **(4)**. Da Data Owner die Verantwortung für ihre Datenprodukte tragen, liegt nahe, dass sie auch für die Durchsetzung der Privacy-Forderungen verantwortlich sein sollten. Gibt es keine Privacy-Forderungen, muss nichts weiter geprüft werden. Gibt es Privacy-Forderungen, prüft der Data-Owner, welche das sind und ob diese erfüllt sind oder nicht **(5) + (6)**. Um eine effektive Suche und Prüfung der Privacy-Forderungen zu ermöglichen, müssen sie angemessen modelliert sein. Ein Beschreibungsmodell für die Privacy-Forderungen wird im folgenden Kapitel vorgestellt. Sind die Privacy-Forderungen nicht erfüllt, bereitet der Data Owner die Daten mithilfe von Privacy-Filtern auf **(7)**. Dafür können Lösungen aus den verwandten Arbeiten verwendet werden. Eine Möglichkeit ist die Datenaufbereitungszone BARENTS [SBE+22a]. Sie unterstützt die Data Owner bei der Definition von Verarbeitungsschritten für die Daten. Wie BARENTS effizient für diesen Anwendungsfall umgesetzt werden kann, wird im folgenden Kapitel diskutiert. Gibt es zwei Privacy-Forderungen die den gleichen Aspekt unterschiedlich stark einschränken wollen, wird die stärkere Einschränkung gewählt. Beispielsweise soll ein Attribut laut der Privacy-Forderung einer Domäne in einem bestimmten Kontext generalisiert

werden, während es laut der Privacy-Forderung eines Data Subjects in diesem Kontext ganz entfernt werden soll. In diesem Fall gilt die strengere Privacy-Forderung und das Attribut wird entfernt. Nachdem alle Privacy-Forderungen erfüllt sind, wird der anfragenden Domäne der Zugriff auf die aufbereitete Version des Datenprodukts gestattet **(8)**. Dieses enthält weiterhin die Referenz auf alle Privacy-Forderungen, die für die in ihm enthaltenen Daten gelten.

In Abbildung 6.2 ist ein beispielhafter Ablauf mithilfe von lila Annotationen dargestellt. Dieser stammt aus dem Szenario der Optimierung der Dienstpläne einer Station in Abschnitt 2.2. Der Prozess beginnt damit, dass die Station bei ihrer Fachabteilung die Behandlungsdaten anfragt **(1)**. Nachdem die allgemeinen Zugriffsrechte bestätigt sind **(2)**, wird geprüft, ob für das Datenprodukt Privacy-Forderungen gelten **(4)**. Das Datenprodukt in diesem Beispiel umfasst die Behandlungsdaten von vielen Patient*innen. Es werden die folgenden beiden Privacy-Forderungen gefunden: Die erste besagt, dass die Daten der Patient*in gar nicht für Analysezwecke verwendet werden dürfen. Die zweite besagt, dass lediglich das Attribut der „Details“ der Behandlung der Patient*in nicht für Analysezwecke verwendet werden dürfen **(5)**. Einige der Patient*innen haben die erste und einige andere die zweite Privacy-Forderung. Der Rest der Patient*innen haben keine Privacy-Forderungen an dieses Datenprodukt. Die Station, die das Datenprodukt anfragt, möchte die Daten für ihren Analyseanwendungsfall nutzen. Würde der Data Owner das Datenprodukt einfach so an die Station übergeben, wären die beiden Privacy-Forderungen nicht erfüllt, da sie sich auf Analysezwecke beziehen **(6)**. Deshalb muss der Data Owner die Daten so aufbereiten, dass die Privacy-Forderungen erfüllt sind **(7)**. Um die erste Privacy-Forderungen zu erfüllen, löscht er die Einträge zu diesen

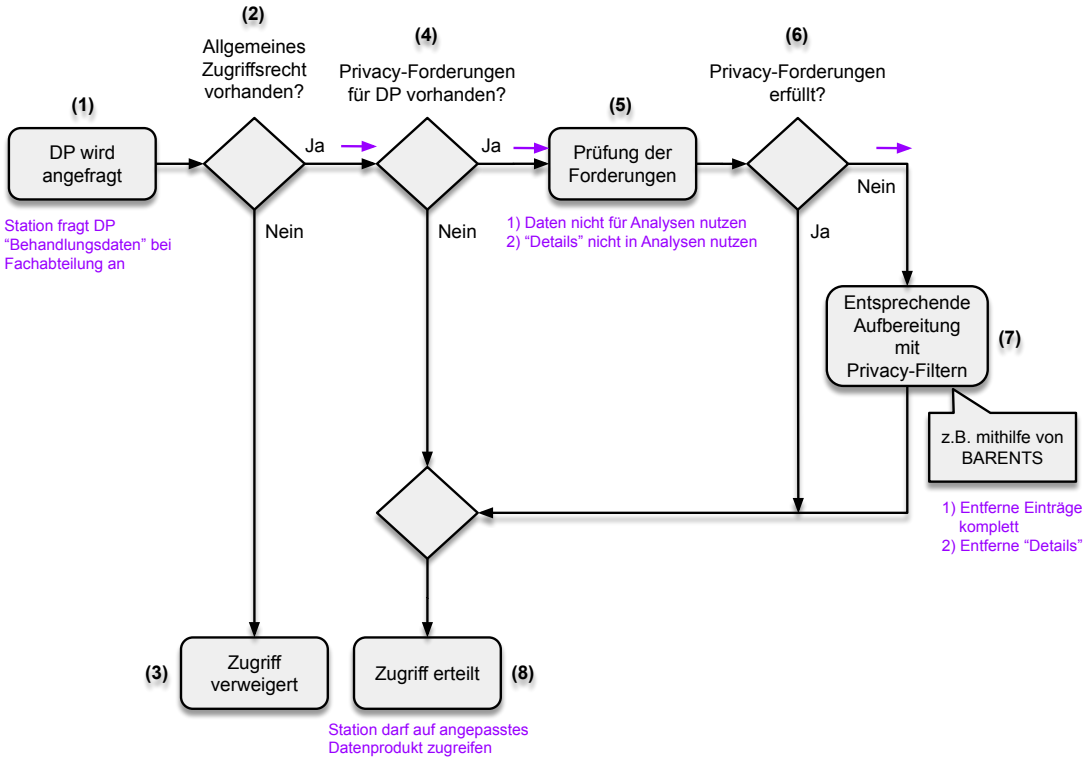

Abbildung 6.2: Prozess der Zugriffsanfrage auf ein Datenprodukt einer anderen Domäne.

Patient*innen komplett aus dem Datensatz. Um die zweite Privacy-Forderungen zu erfüllen, entfernt er das Attribut „Details“ aus den entsprechenden Einträgen. Da nun alle Privacy-Forderungen erfüllt sind, erteilt der Data Owner der Station Zugriff auf die aufbereitet Version des Datenprodukts **(8)**.

Im nächsten Abschnitt wird der Prozess der Erstellung eines neuen Datenprodukts beschrieben.

6.3 Erstellung eines neuen Datenprodukts

In diesem Abschnitt wird der Prozess der Erstellung eines neuen Datenprodukts erklärt. Wenn eine Domäne Zugriff auf das Datenprodukt einer anderen Domäne erhält, gibt es zwei Möglichkeiten wofür es verwendet werden kann. Die Domäne kann die Daten für eine Analyse verwenden und die gewonnen Informationen beispielsweise für eine Geschäftsentscheidung nutzen. Die andere Möglichkeit ist beispielsweise Teil des Szenarios der Optimierung der Dienstpläne einer Station in Abschnitt 2.2. Nachdem die Station die Daten der Fachabteilung erhalten hat, kann sie sie anreichern und anschließend ein neues Datenprodukt erstellen. Dieses kann die Station selbst verwenden oder anderen Domänen zur Verfügung stellen. Solche Produkte aus Produkten sind im Data-Mesh-Ansatz vorgesehen, da so nutzbare Kombinationen von Daten direkt verfügbar gemacht werden. Dadurch wird der Austausch von Daten zwischen den Domänen gefördert [Deh22]. Dabei muss die Erfüllung der Privacy-Forderungen sichergestellt werden, da Kombinationen von Daten entstehen können, die die Privacy-Forderungen verletzen. Beispielsweise wird ein Attribut aufgrund einer Privacy-Forderung aus einem Datensatz entfernt. Der Datensatz wird anschließend mit einem anderen Datensatz kombiniert, welcher dieses Attribut ebenfalls enthält. Dadurch ist die Privacy-Forderung nicht mehr erfüllt und müsste erneut durchgesetzt werden, indem das Attribut wieder entfernt wird. Deshalb ist es relevant, diesen Prozess zu definieren.

Der Prozess der Erstellung ist in Abbildung 6.3 dargestellt. Initial hat die anfragenden Domäne den Zugriff auf das Datenprodukt erhalten **(1)**. Das geschieht am Ende des Prozesses der Zugriffsanfrage. Im nächsten Schritt analysiert und verarbeitet sie die Daten **(2)**. Dabei reichert sie die Daten des Datenprodukts mit domäneneigenen Daten an. Anschließend soll aus den angereicherten Daten ein neues Datenprodukt entstehen **(3)**. Dafür wird zunächst geprüft, ob für die angereicherten Daten Privacy-Forderungen vorliegen **(4)**. Dabei werden die Privacy-Forderungen aller ursprünglichen Daten miteinbezogen, denn durch die Verarbeitung der Daten kann es passieren, dass die Daten so angereichtert werden, dass die Privacy-Forderungen nicht mehr erfüllt sind. Die Privacy-Forderungen der domäneninternen Daten werden von einem Verantwortlichen der Domäne geprüft. Für die Privacy-Forderungen des bestehenden Datenprodukts ist der Data Owner verantwortlich, da er immer noch die Verantwortung besitzt. Die Prüfung und Durchsetzung werden so durchgeführt wie beim Prozess der Zugriffsanfrage in Abschnitt 6.2. Der Data Owner überprüft, welche Privacy-Forderungen vorhanden sind **(5)**. Anschließend prüft er, ob alle Privacy-Forderungen erfüllt sind **(6)**. Die Durchsetzung der nicht erfüllten Privacy-Forderungen könnte entweder durch den Data Owner des bestehenden Datenprodukts oder durch die Domäne, die das Datenprodukt erstellen möchte, erfolgen **(7)**. Für den Data Owner spricht, dass er sich bereits mit den Privacy-Forderungen auseinandergesetzt hat, er hat daher viel Wissen über die Aspekte, die durchgesetzt werden müssen. Außerdem ist die Privacy höher, wenn der Data Owner beispielsweise Informationen, welche die Domäne nicht erhalten soll, aus dem Datenprodukt entfernt, anstatt der Domäne diese Informationen zu zeigen. Für die Domäne spricht, dass dort das Fachwissen über das erstellte Datenprodukt besteht. Da sie die Struktur des Datenprodukts kennen, fällt es den zuständigen

Mitarbeitern leichter, die Privacy-Forderungen umzusetzen. Der Data Owner hingegen müsste sich in das Datenprodukt einarbeiten, um die Privacy-Forderungen durchzusetzen. Allerdings muss er sich bereits mit dem neuen Datenprodukt beschäftigen, um die Erfüllung der Privacy-Forderungen zu prüfen. Dadurch muss er sich nicht komplett neu einarbeiten, um die nicht erfüllten Privacy-Forderungen durchzusetzen. Aus diesen Gründen erfolgt die Durchsetzung der nicht erfüllten Privacy-Forderungen im neuen Datenprodukt in diesem Konzept durch den Data-Owner. Es liegt in seinem eigenen Interesse, die Durchsetzung gründlich durchzuführen, da die Daten des Datenprodukts verwendet werden, für das er Verantwortung trägt. Für die Konfliktlösung bei der Durchsetzung gilt das gleiche Prinzip wie bei der Zugriffsanfrage auf Datenprodukte (siehe Abschnitt 6.2). Wenn zwei unterschiedlich strenge Privacy-Forderungen für ein Datum gelten, wird die strengere durchgesetzt. Sind alle Privacy-Forderungen erfüllt, kann das neue Datenprodukt freigegeben werden **(8)**. Die Verantwortung wird dabei an die anfragende Domäne übergeben. Ab diesem Punkt ist ein Data Owner aus dieser Domäne für die Betreuung des neuen Datenprodukts zuständig.

In Abbildung 6.3 ist ein beispielhafter Ablauf mithilfe von lila Annotationen dargestellt. Dieser stammt aus dem Szenario der Optimierung der Dienstpläne einer Station in Abschnitt 2.2. Zunächst erhält die Station das Zugriffsrecht für das vorbereitete Datenprodukt „Behandlungsdaten “ der Fachabteilung **(1)**. Sie reichert es mit ihren domäneneigenen Daten „Aufenthaltsdaten “ an **(2)**.

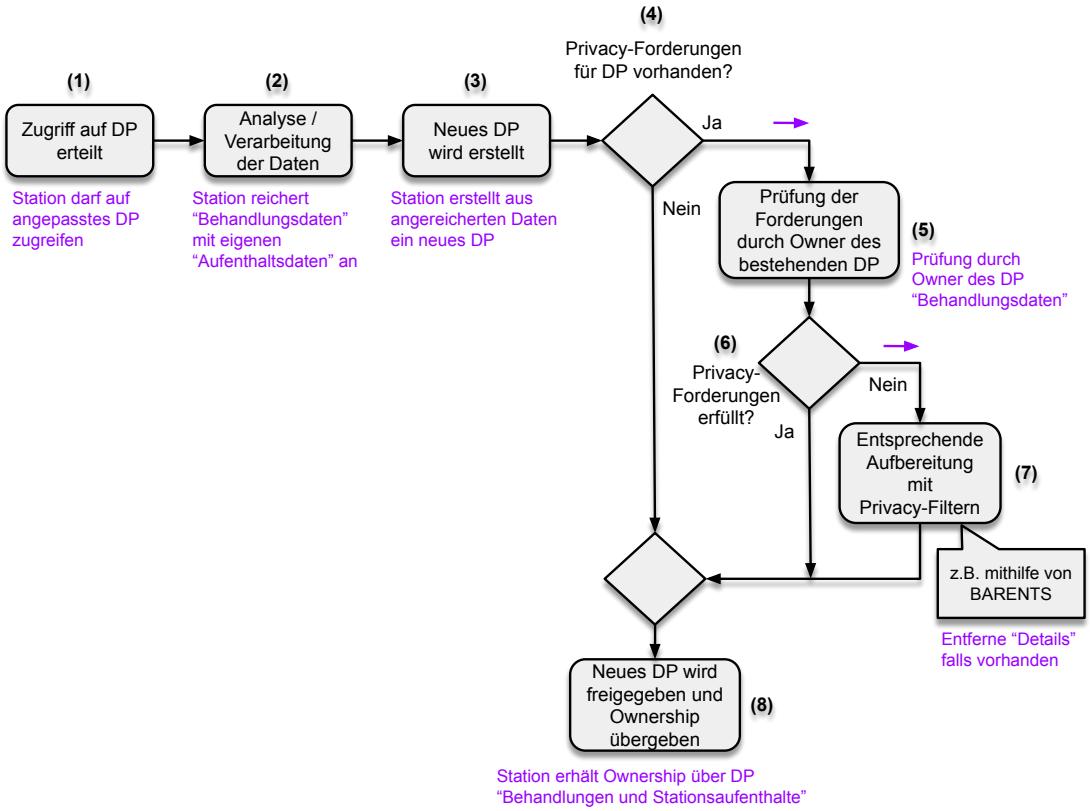

Abbildung 6.3: Prozess der Erstellung eines neuen Datenprodukts inklusive Prüfung und Durchsetzung der Privacy-Forderungen.

Anschließend möchte sie aus der Kombination ein neues Datenprodukt erstellen **(3)**. Deshalb wird zunächst geprüft, ob Privacy-Forderungen für die Daten vorliegen **(4)** und welche das sind **(5)**. Anschließend wird geprüft, ob sie erfüllt sind **(6)**. Für die „Behandlungsdaten “ wird die Prüfung von dem Data Owner der Fachabteilung durchgeführt. Er überprüft dabei die Privacy-Forderung, die nach der Vorbereitung in Abschnitt 6.2 noch vorhanden ist. Sie besagt, dass lediglich das Attribut der „Details“ der Behandlung der Patient*in nicht für Analysezwecke verwendet werden darf. In den „Aufenthaltsdaten “ können ebenfalls „Details“ der Behandlung vermerkt sein. Diese würden dann mit den restlichen „Behandlungsdaten “ verknüpft werden, wodurch die Privacy-Forderungen nicht mehr erfüllt wäre. Deshalb muss der Data Owner das Attribut „Details“ in diesem Fall wieder entfernen **(7)**. Anschließend erhält die Station die Verantwortung für das neue Datenprodukt „Behandlungen und Stationsaufenthalte “**(8)**.

Eine Domäne kann ein angefragtes Datenprodukt nicht nur mit domäneneigenen Daten anreichern, sondern auch mit Datenprodukten anderer Domänen kombinieren. Dies stellt einen Spezialfall der Erstellung eines neuen Datenprodukts dar. Dieser wird im nächsten Abschnitt genauer betrachtet.

6.4 Fusion mehrerer Datenprodukte

Im letzten Abschnitt wurde der Prozess zur Erstellung eines neuen Datenprodukts erklärt. Ein Spezialfall hiervon ist die Fusion mehrerer Datenprodukte. Dabei handelt es sich um die Erstellung eines neuen Datenprodukts, bei der mehrere bestehende Datenprodukte verwendet werden. Dieser Fall tritt im Szenario der Leistungsabrechnung bei den Versicherungen auf, da dort die Datenprodukte mehrerer Domänen angefragt werden (siehe Abschnitt 2.2). Da der Prozess dadurch anders abläuft, wird er gesondert in diesem Abschnitt beschrieben.

In Abbildung 6.4 ist der allgemeine Ablauf dargestellt. Zu Beginn erhält die Domäne Zugriff auf mehrere Datenprodukte, die verarbeitet werden **(1) + (2)**. Dabei sind Bestandteile von mehreren bestehende Datenprodukten in den resultierenden Datensatz eingeflossen. Aus diesem Datensatz soll ein neues Datenprodukt erstellt werden **(3)**. Hierfür muss zunächst geprüft werden, ob und welche Privacy-Forderungen für die bestehenden Datenprodukte gelten **(4) + (5)** und ob diese weiterhin erfüllt sind **(6)**. Da es in diesem Fall mehrere bestehende Datenprodukte gibt, müssen die Data Owner jeweils für ihre eigenes Datenprodukt alle Privacy-Forderungen überprüfen. Es ist nicht ausreichend, dass einer der Data Owner seine Privacy-Forderungen überprüft. Die Privacy-Forderungen aller bestehenden Datenprodukte, die in das neue Datenprodukt einfließen, müssen vom jeweiligen Data Owner überprüft werden. Sind die Privacy-Forderungen nicht erfüllt, muss der Data Owner des jeweiligen bestehenden Datenprodukts, diese Privacy-Forderungen durchsetzen **(7)**. Da mehrere Data Owner eine Prüfung durchführen ist es wichtig, eventuelle Änderungen zu koordinieren um Versionskonflikte zu vermeiden. Eine Möglichkeit ist, dass die Data Owner das neue Datenprodukt nacheinander überprüfen und anpassen. Es könnte aber auch ein Versionsverwaltungssystem verwendet werden, um Änderungen zu koordinieren und unter Umständen rückgängig machen zu können. Für die Konfliktlösung bei der Durchsetzung gilt das gleiche Prinzip wie bei der Zugriffsanfrage auf Datenprodukte (siehe Abschnitt 6.2). Wenn zwei unterschiedlich strenge Privacy-Forderungen für ein Datum gelten, wird die strengere durchgesetzt. Anschließend kann das neue Datenprodukt freigegeben werden und die Verantwortung wird an einen Data Owner der Domäne übergeben **(8)**.

In Abbildung 6.5 ist ein beispielhafter Ablauf der Fusion mehrerer Datenprodukte am Szenario der Leistungsabrechnung bei den Versicherungen dargestellt. Die Finanzabteilung fragt die Datenprodukte der Leistungsdaten für das erste Quartal bei den anderen Domänen an **(1)**. Beispielsweise werden die Datenprodukte der Notaufnahme und des Case Managements angefragt. Diese bereiten ihre Datenprodukte entsprechend vor **(2)**. Dabei gehen sie vor wie im Prozess der Zugriffsanfrage auf ein Datenprodukt beschrieben (siehe Abbildung 6.2). Das Ergebnis ist eine aufbereitete Kopie des Datenprodukts, auf welche die Finanzdomäne Zugriff erhält. Die Finanzdomäne verarbeitet die Datensätze und fusioniert sie zu einem Datenprodukt, welches die gesammelten Leistungsdaten aller Domänen enthält **(3)**. Der Entwurf dieses Datenprodukt muss durch alle Data Owner geprüft werden **(4)**. Beispielsweise stellt der Data Owner des Datenprodukts des Case Managements sicher, dass die Privacy-Forderungen seines Datenprodukts im neuen Datenprodukt weiterhin eingehalten werden. Sind alle Privacy-Forderungen durchgesetzt, wird das Datenprodukt aller Leistungen des ersten Quartals freigegeben und die Finanzabteilung erhält die Verantwortung für das Datenprodukt.

Warum es nicht ausreicht, dass lediglich ein Data Owner eines der bestehenden Datenprodukte die Privacy-Forderungen überprüft, wird im Folgenden an einem Beispiel erläutert. Angenommen das Datenprodukt mit den Leistungen einer Station enthält die Information, dass eine Patient*in am 01. Juli ein Medikament erhalten hat. Die Patient*in kommuniziert die Privacy-Forderung, dass der Name des Medikaments verborgen wird um zu vermeiden, dass daraus durch nicht-medizinisches Personal ihre Krankheit abgeleitet wird. Die Finanzabteilung fragt außerdem die Leistungen der

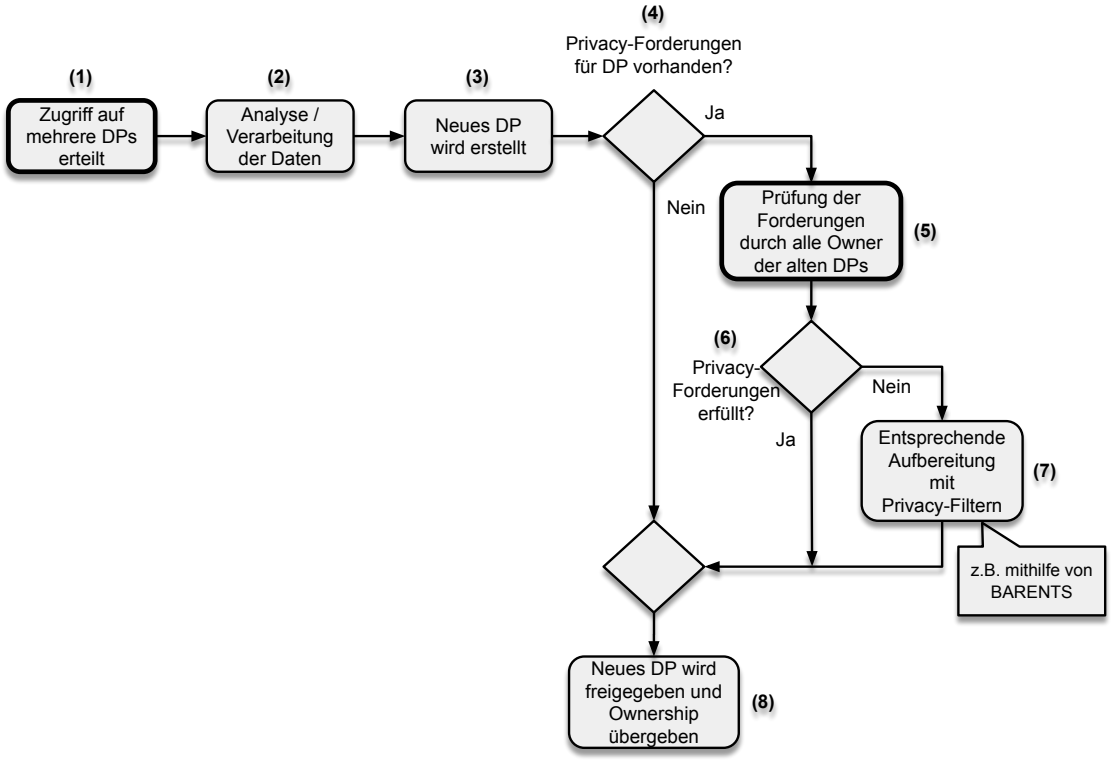

Abbildung 6.4: Prozess der Erstellung eines neuen Datenprodukts aus mehreren bestehenden Datenprodukten inklusive Prüfung und Durchsetzung der Privacy-Forderungen.

Krankenhausapotheke an. Diese enthalten zwar keine Patientendaten, die an die Versicherungen weitergeleitet werden, aber die Finanzabteilung benötigt sie dennoch, um intern eine komplette Leistungsübersicht zu erstellen. In den Leistungsdaten der Apotheke befindet sich ein Eintrag über den Verbrauch einer Dosis des Medikaments am 01. Juli. Dies ist der einzige Eintrag für diesen Tag. Die Finanzabteilung fusioniert nun die Datenprodukte, die sie von den Abteilungen erhalten hat. Darin sind jetzt sowohl die Information über eine Medikamenteneinnahme der Patient*in als auch der Eintrag über den Verbrauch des Medikaments (mit Name des Medikaments) enthalten. Da sich beide Einträge auf den 01. Juli beziehen und das Medikament das Einzige ist, welches an diesem Tag verbraucht wurde, kann das Personal der Finanzabteilung daraus schließen, dass diese Patient*in genau dieses Medikament eingenommen hat. Das wollte die Patient*in mit ihrer Privacy-Forderungen jedoch verhindern. Wenn lediglich der Data Owner der Apotheke die Privacy-Forderungen prüfen würde, welche für sein Datenprodukt gelten, würde die Verletzung der Privacy-Forderung nicht bemerkt werden. Nur der Data Owner der Station kann diese Verletzung finden. Aus diesem Grund müssen alle Data Owner die Privacy-Forderungen ihrer Datenprodukte prüfen, bevor das neue Datenprodukt freigegeben werden kann. Der Data Owner der Station setzt die nicht erfüllten Privacy-Forderung durch. Beispielsweise könnte er das Datum der Einnahme des Medikaments durch eine Zeitspanne von mehreren Tagen ersetzen. Anschließend kann das Datenprodukt freigegeben werden.

Gewonnene Erkenntnisse: In diesem Kapitel wurden selbst entwickelte Konzepte vorgestellt, welche die im Anwendungsfall geforderten Privacy-Aspekte in den Data-Mesh-Ansatz einbringen. Es wurde jeweils ein Konzept für den Prozess der Erhebung der Privacy-Forderungen sowie für den Prozess der Prüfung und Durchsetzung der Privacy-Forderungen bei Zugriffsanfragen vorgestellt. Desweiteren wurden Konzepte für den Prozess der Erstellung eines neuen Datenprodukts aus einem bestehenden Datenprodukt und für den Prozess der Erstellung eines neuen Datenprodukts aus

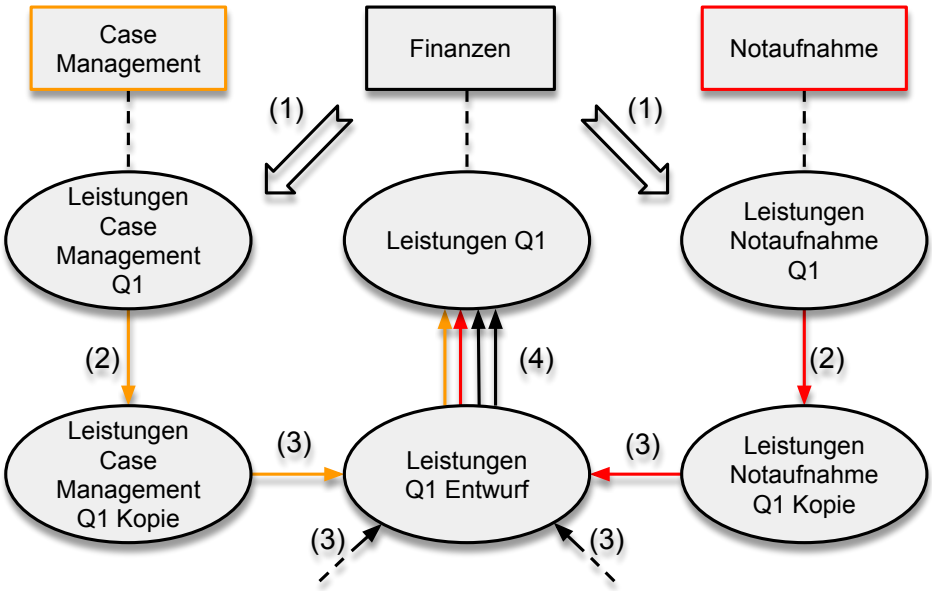

Abbildung 6.5: Beispiel für den Ablauf der Erstellung eines neuen Datenprodukts aus mehreren bestehenden Datenprodukten.

mehreren bestehenden Datenprodukten präsentiert. Die Privacy-Vorgänge sind nach dem Privacy-by-Design-Konzept von vornherein in die Prozesse integriert. Beispielsweise findet während der Datenerhebung auch die Erhebung der Privacy-Forderungen statt und bei Zugriffsanfragen auf Datenprodukte ist die Prüfung der Privacy-Forderungen mitbedacht. Im folgenden Kapitel werden Implementierungsstrategien für zwei Aspekte des Konzepts diskutiert. Damit Data Owner der Datenprodukte die Privacy-Forderungen effektiv prüfen und durchsetzen können, müssen die Privacy-Forderungen geeignet abgebildet werden. Daher wird in Abschnitt 7.1 ein eigenes Modell für die Beschreibung der Privacy-Forderungen vorgestellt. Um die Privacy-Forderungen effizient umsetzen zu können, benötigen Data Owner Unterstützung durch Technologien. In Abschnitt 7.2 wird diskutiert, wie BARENTS [SBE+22a] Data Owner bei der effizienten Anwendung von Privacy-Filtern unterstützen kann.

7 Implementierungsstrategien

In diesem Kapitel werden Implementierungsstrategien für zwei Aspekte des zuvor vorgestellten Konzepts (siehe Kapitel 6) präsentiert. Zuerst wird in Abschnitt 7.1 ein Beschreibungsmodell für Privacy-Forderungen vorgestellt. Eine geeignete Modellierung der Privacy-Forderungen ist notwendig, um die Privacy-Forderungen zu beschreiben und sie festzuhalten. Außerdem ermöglicht es dem Data-Owner, die Privacy-Forderungen effektiv durchzusetzen. Beispielsweise erleichtert es dem Data-Owner, alle relevanten Privacy-Forderungen eines Data Subjects zu finden. Der zweite Aspekt ist die Umsetzung der Privacy-Forderungen mithilfe von BARENTS [SBE+22a]. BARENTS ist ein Werkzeug das Data Owner bei der Umsetzung von Privacy-Forderungen unterstützen kann. Die Umsetzung muss effizient erfolgen, da mit großen Datenmengen gearbeitet wird und die Wartezeiten auf das Ergebnis möglichst kurz gehalten werden sollen. Daher wird in Abschnitt 7.2 diskutiert, inwieweit BARENTS eine effiziente Umsetzung ermöglicht.

7.1 Beschreibung der Privacy-Forderungen

In diesem Abschnitt wird das Beschreibungsmodell für die Privacy-Forderungen vorgestellt. Das Modell muss folgende Anforderungen erfüllen:

- Data Owner müssen alle zu einem Datenprodukt zugehörigen Privacy-Forderungen finden können. Dies ist notwendig damit die Data Owner die Durchsetzung der Privacy-Forderungen effektiv prüfen können (siehe Kapitel 3).
- Das Modell muss eine Referenz auf den Ersteller der Privacy-Forderung enthalten, damit dieser auffindbar ist. Dadurch kann der Ersteller beispielsweise bei Unklarheiten auf Seiten der Data Owner kontaktiert werden. Auch wenn ein Data Subject Änderungen an seinen Privacy-Forderungen vornehmen möchte, wird die Referenz verwendet, um die entsprechenden Privacy-Forderungen zu identifizieren.
- Das Modell sollte sowohl die formalisierte als auch die originale, informelle Version der Privacy-Forderungen enthalten. Die Privacy-Forderungen werden bei der Erhebung formalisiert, um die Verarbeitung zu erleichtern. Die informelle Version kann bei Unklarheiten zu Rate gezogen werden und neue Methoden der Formalisierung können auf sie angewendet werden.

Die erste Anforderung ist Teil der in Kapitel 3 definierten Anforderungen. Hat der Data Owner die relevanten Privacy-Forderungen für das Datenprodukt gefunden, muss er diese prüfen und durchsetzen. Dabei können Unklarheiten bezüglich des Inhalts und der Bedeutung der Privacy-Forderungen auf Seiten des Data-Owners auftreten. Um zu vermeiden, dass die Informationen über eine Privacy-Forderung von verschiedenen Stellen zusammengetragen werden müssen, sollen sie

alle in ihrer Beschreibung enthalten sein. Daher müssen auch alle Anhaltspunkte und Anlaufstellen zur Klärung von Unklarheiten Teil der Beschreibung der Privacy-Forderung sein. Dafür wurden die zweite und dritte Anforderung an das Beschreibungsmodell der Privacy-Forderungen definiert.

In Abbildung 7.1 ist das generische Beschreibungsmodell einer Privacy-Forderung dargestellt. Eine Instanz des Modells beschreibt eine Privacy-Forderung. Das erste Element der ersten Unterebene umfasst die „**Metadaten**“ der Privacy-Forderung. Darin enthalten sind die einzigartige „**ID**“ der Privacy-Forderung, sowie der „**Zeitstempel**“ des Zeitpunkts ihrer Erstellung. In diesem Ansatz werden drei Arten von Privacy-Forderungen modelliert und zwar gesetzlichen Privacy-Forderungen, Privacy-Forderungen von Domänen und Privacy-Forderungen von Data-Subjects. Die Art einer Privacy-Forderung wird im Element „**Ursprung**“ abgebildet. Dort können die Kategorien „DSGVO“, „Domäne“ oder „Subject“ eingetragen werden, um den jeweiligen Ursprung der Privacy-Forderung zu dokumentieren. Um die Verbindung zum Ersteller der Privacy-Forderung herstellen zu können, wird im Element „**Subject / Domänen ID**“ gegebenenfalls die ID des Data Subjects oder der Domäne eingetragen, welche die Privacy-Forderung erstellt hat. Im Falle von gesetzlichen Privacy-Forderungen ist dieses Element nicht vorhanden, da gesetzliche Privacy-Forderungen von einem zentralen Governance-Team modelliert werden.
Diese ID macht es dem Data Owner möglich, alle Privacy-Forderungen die für ein Datenprodukt gelten, zu identifizieren. Dabei kann er folgendermaßen vorgehen: Da sie für alle Datenprodukte gelten, ruft er zunächst alle gesetzlichen Privacy-Forderungen auf. Anschließend kann er alle Privacy-Forderungen aufrufen, deren Beschreibung die ID der Domäne enthalte,n der das Datenprodukt gehört. Diese gelten für das ganze Datenprodukt. Zuletzt ruft er alle Privacy-Forderungen auf, welche die ID der Data Subjects enthalten, deren Daten im Datenprodukt enthalten sind. Diese gelten jeweils nur für die Daten des Data-Subjects, das sie erstellt hat. Sollten mehrere Versionen derselben Forderung existieren, hilft der Zeitstempel bei der zeitlichen Einordnung der Versionen. So kann die neueste Version geprüft werden und trotzdem bleiben die Informationen, die in der Historie enthalten sind, erhalten. Das letzte Element der „**Metadaten**“ ist das optionale Element „**Tags**“, das Schlüsselwörter der Privacy-Forderung enthält. Diese Schlüsselwörter können verwendet werden, um weitere relevante Privacy-Forderungen mithilfe eines Recommender-Ansatzes wie EPICUREAN [SS19] identifizieren zu können.

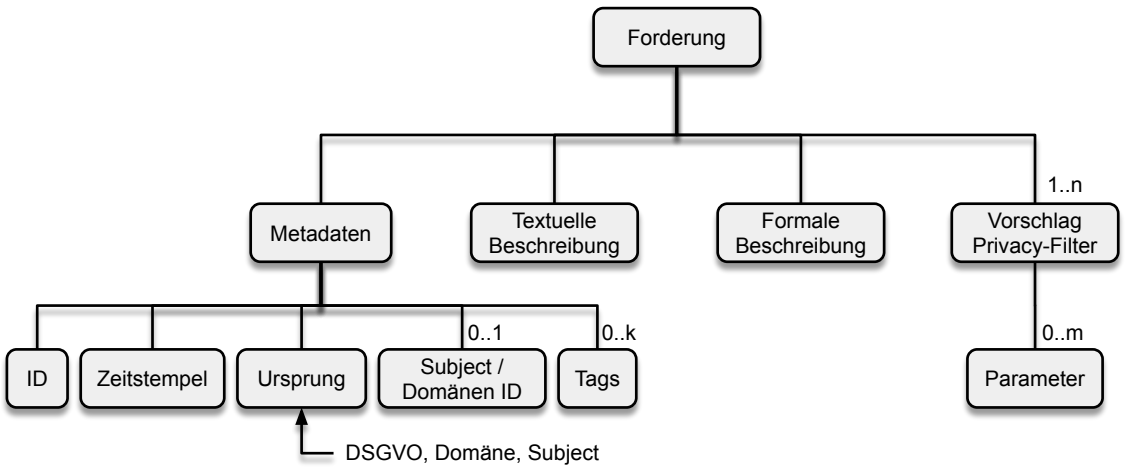

Abbildung 7.1: Generisches Beschreibungsmodell einer Privacy-Forderung.

Das zweite Element der ersten Unterebene enthält die „**textuelle Beschreibung**“ der Privacy-Forderung. Das ist beispielsweise der Gesetzestext, der die Grundlage für eine gesetzliche Privacy-Forderung darstellt oder der Text, in dem ein Data Subject beschreibt, wie und von wem bestimmte Daten verwendet werden dürfen. Indem diese Beschreibungen in das Modell aufgenommen werden, bleibt die originale, informelle Version erhalten und kann später zu Rate gezogen werden.
Das dritte Element enthält die „**formale Beschreibung**“ der Privacy-Forderung. Dabei handelt es sich um eine formalisierte Version der „**textuelle Beschreibung**“ der Privacy-Forderung. Zum Beispiel würde „Der Verwendungszweck darf nur innerhalb der Krankenhausapotheke eingesehen werden.“ zu „Wenn Domäne =! 'Apotheke' dann lösche Attribut 'Verwendungszweck'“. Die Syntax der „**formale Beschreibung**“ kann von den Betreibern des Data-Mesh an ihre Bedürfnisse angepasst werden.
Das vierte und letzte Element „**Vorschlag Privacy-Filter**“ enthält einen oder mehrere Vorschläge für Privacy-Filter, die zur Umsetzung der Privacy-Forderung verwendet werden können. Durch diese Vorschläge kann der Data Owner bei der Umsetzung der Privacy-Forderungen unterstützt werden. Diese Empfehlung kann durch Experten oder durch Werkzeuge wie beispielsweise PATRON [SDM+18] erfolgen. Optional können zu einem Filter „**Parameter**“ angegeben werden, beispielsweise der gewünschte k Parameter der k-Anonymity.

In Abbildung 7.2 ist eine Instanz des Beschreibungsmodells für eine gesetzliche Privacy-Forderung dargestellt. Es handelt sich um eine Privacy-Forderung, die in allen drei Szenarien des vorgestellten Anwendungsfalls auftreten kann (siehe Kapitel 2). Im Element der „Metadaten“ sind von links nach rechts die „ID“ der Privacy-Forderung sowie der „Zeitstempel“ eingetragen. Außerdem ist für den „Ursprung“ die „DSGVO“ eingetragen. Da es sich um eine gesetzliche Privacy-Forderung handelt, wird sie vom zentralen Governance-Team erhoben. Deshalb ist kein expliziter Ersteller vorhanden und das Element „Subject / Domänen ID“ wird weggelassen. Im Element „Tags“ sind die zwei Schlüsselwörter „Zweck“ und „Datenspeicherung“ eingetragen. Diese wurden aus der „informellen Beschreibung“ extrahiert. Hierzu können Techniken des Natural Language Processings [SSS+19], wie beispielsweise der RAKE Algorithmus [RECC10], zum Einsatz kommen. Die „informelle Beschreibung“ enthält den orginalen Gesetzestext aus Artikel 5, Absatz 1, Buchstabe b der DSGVO (Zweckbindung). Daraus wird die „formelle Beschreibung“ generiert, die besagt, dass ein Tupel gelöscht werden muss, wenn der aktuelle Verwendungszweck nicht in den vereinbarten Zwecken

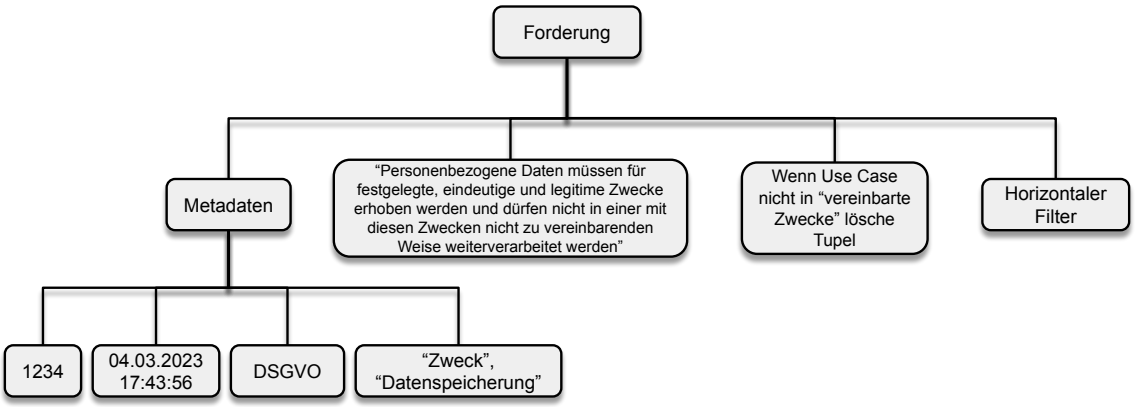

Abbildung 7.2: Instanz des Beschreibungsmodells für eine gesetzliche Privacy-Forderung.

enthalten ist. Im Element „Vorschlag Privacy-Filter“ ist ein „horizontaler Filter“ eingetragen (siehe Abschnitt 4.2), da damit ganze Tupel gelöscht werden können, die nicht für den gegebenen Verwendungszweck genutzt werden dürfen. Da der ein „horizontaler Filter“ keine „Parameter“ benötigt, hat das Element keinen Eintrag. Weil es sich bei der beschriebenen Privacy-Forderung um eine gesetzliche Privacy-Forderung handelt, gilt sie für alle Datenprodukte die sich im Data Mesh des Krankenhauses befinden.

In Abbildung 7.3 ist eine Instanz des Beschreibungsmodells für die Privacy-Forderung einer Domäne dargestellt. Es handelt sich um die Privacy-Forderung der Krankenhausapotheke aus dem Szenario der Anpassung der Einkäufe (siehe Abschnitt 2.2). In den „Metadaten“ sind die „ID“ der Privacy-Forderung, der „Zeitstempel“ und der „Ursprung“ eingetragen. Da es sich beim „Ursprung“ um eine Domäne handelt, ist die „Domänen ID“ der Apotheke „D-Apotheke“ angegeben. Als „Tags“ sind die Schlüsselwörter „Verwendungszweck“, „Attribut“ und „Domäne“ angegeben. Die „informelle Beschreibung“ sagt aus, dass das Attribut „Verwendungszweck“ nicht von anderen Domänen eingesehen werden darf. Die „formale Beschreibung“ definiert, dass das Attribut „Verwendungszweck“ gelöscht werden muss, wenn es sich um eine andere Domäne als die Apotheke handelt. Als möglicher Filter wird ein „vertikaler Filter“ vorgeschlagen (siehe Abschnitt 4.2). Damit kann das Attribut „Verwendungszweck“ aus dem Datensatz entfernt werden. Da der „vertikale Filter“ keine „Parameter“ benötigt, hat das Element keinen Eintrag. Bei der beschriebenen Privacy-Forderung handelt es sich um die Privacy-Forderung einer Domäne. Deshalb gilt sie für alle Datenprodukte, für die die Domäne verantwortlich ist.

In Abbildung 7.4 ist eine Instanz des Beschreibungsmodells für die Privacy-Forderung eines Data Subjects dargestellt. Es handelt sich um die Privacy-Forderung eines Data Subjects aus dem Szenario der Optimierung der Dienstpläne einer Station (siehe Abschnitt 2.2). Im Element der „Metadaten“ sind die „ID“ der Privacy-Forderung, der „Zeitstempel“ und der „Ursprung“ eingetragen. Da es sich beim „Ursprung“ um ein Data Subject handelt, ist zusätzlich die „Subject ID“ des Data Subjects „S-85739“ eingetragen. Als „Tags“ sind die Schlüsselwörter „Diagnose“ und „Anonymisierung“ eingetragen. Diese wurden aus der „informellen Beschreibung“ extrahiert. Sie besagt, dass das Attribut „Diagnose“ des Data Subjects nicht zu ihm zuordenbar sein soll, wenn seine Daten für einen anderen Zweck als den Behandlungsprozess verwendet werden. Dabei wird davon ausgegangen, dass

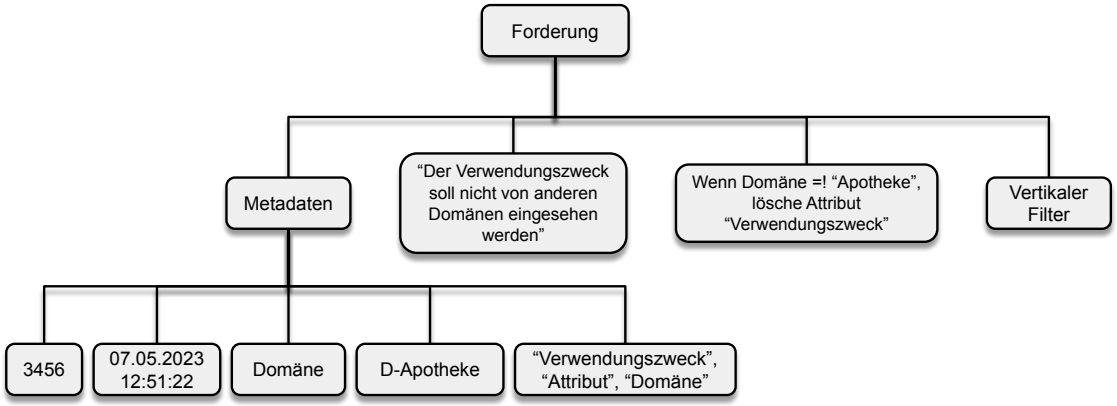

Abbildung 7.3: Instanz des Beschreibungsmodells für die Privacy-Forderung einer Domäne aus Szenario 2.

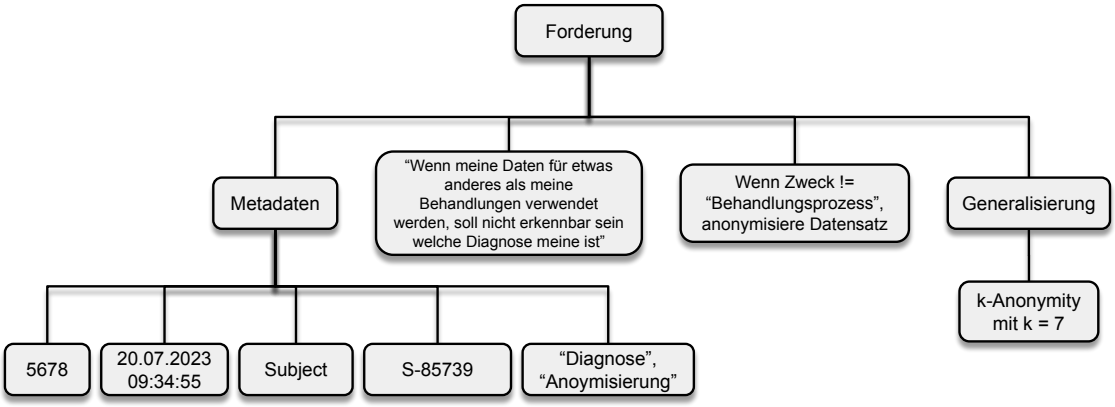

Abbildung 7.4: Instanz des Beschreibungsmodells für die Privacy-Forderung eines Data Subjects aus Szenario 1.

die Daten in diesen Fällen Teil von einem Datensatz sind, der die Daten von vielen Data Subjects enthält. In der „formalen Beschreibung“ ist diese Privacy-Forderung folgendermaßen beschrieben: Wenn der Zweck nicht der „Behandlungsprozess“ des Data Subjects ist, muss der Datensatz, in dem die Daten enthalten sind, anonymisiert werden. Hierfür wird die Generalisierung als „Vorschlag Privacy-Filter“ angegeben (siehe Abschnitt 4.2). Als „Parameter“ wird die k-Anonymity Metrik mit $k = 7$ angegeben (siehe Abschnitt 4.2). Damit wird angegeben, dass der Datensatz so weit anonymisiert werden soll, dass das Data Subject in eine Gruppe von sieben Data Subjects landet, in der sich nur das Attribut „Diagnose“ der Data Subjects unterscheidet. So kann nicht zugeordnet werden, welche „Diagnose“ zu welchem Data Subject gehört. Die Privacy-Forderung gilt nur für die Daten des Data-Subjects, zu denen sie zugeordnet ist. Diese können Teil jeglicher Datenprodukte im Data Mesh des Krankenhauses sein.

In den Abbildung 7.1 bis 7.4 wurde eine UML-artige Darstellung für das Beschreibungsmodell der Privacy-Forderungen verwendet. Solche Darstellungen eignen sich gut, um das Beschreibungsmodell zwischen Menschen zu kommunizieren. Für eine maschinelle Verarbeitung und Speicherung muss es jedoch in eine interne Darstellung überführt werden. Hierfür eignen sich Datenformate wie JSON [Bas15] und XML [BE99]. Sie besitzen die notwendige Flexibilität, um alle Aspekte zu beschreiben und bieten gleichzeitig effiziente Verarbeitungsmöglichkeiten. Zudem sind sie im Data-Mesh-Umfeld bereits für die Metadatenbeschreibung weit verbreitet [DCL23c]. Quellcode 7.1 zeigt das JSON-Schema, welches den Aufbau des Beschreibungsmodells festlegt. Es entspricht dem generischen Modell in Abbildung 7.1. Die Hierarchie der UML-Elemente ist mithilfe von Verschachtelung der JSON-Elemente umgesetzt. Durch das Schlüsselwort „**required** “ wird festgelegt, welche Elemente enthalten sein müssen. Das JSON-Dokument in yz zeigt die Instanz des Beschreibungsmodells für eine Privacy-Forderung eines Data Subjects wie in Abbildung 7.4 dargestellt.

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "title": "Beschreibungsmodell Privacy-Forderung",
  "description": "Beschreibt eine Privacy-Forderung",
  "type": "object",
  "required": ["Metadaten", "Textuelle Beschreibung", "Formale Beschreibung"],
  "properties": {
    "Metadaten": {
      "description": "Metadaten der Privacy-Forderung.",
      "type": "object",
      "required": ["ID", "Zeitstempel", "Ursprung"],
      "properties": {
        "ID": {
          "description": "ID der Privacy-Forderung.",
          "type": "string",
        },
        "Zeitstempel": {
          "description": "Zeitpunkt der Erhebung der Privacy-Forderung.",
          "type": "string",
          "format": "date-time",
        },
        "Ursprung": {
          "description": "Ursprung der Privacy-Forderung",
          "enum": ["DSGVO", "Domaene", "Subject"],
        },
        "Subject / Domaenen ID": {
          "description": "ID Data Subject/Domaene zu der Privacy-Forderung gehoert.",
          "type": "string",
        },
        "Tags": {
          "description": "Tags zur Beschreibung der Privacy-Forderung.",
          "type": "array",
          "items": {
            "type": "string",
          }
        }
      }
    },
    "Textuelle Beschreibung": {
      "description": "Urspruengliche Beschreibung auf der Privacy-Forderung basiert.",
      "type": "string",
    },
    "Formale Beschreibung": {
      "description": "Formalisierte Version der textuellen Beschreibung",
      "type": "string",
    },
```

```
    "Vorschlaege Privacy-Filter": {
      "description": "Privacy-Filter die zur Umsetzung verwendet werden koennen.",
      "type": "array",
      "items": {
        "description": "Privacy-Filter der auf Daten angewendet werden kann.",
        "type": "object",
        "properties": {
          "Name": {
            "description": "Name des Privacy-Filters.",
            "type": "string",
          },
          "Parameter": {
            "description": "Parameter des Privacy-Filters.",
            "type": "string",
          }
        }
      }
    }
  }
}
```

Quellcode 7.1: JSON-Schema des Beschreibungsmodells der Privacy-Forderungen.

```
{
  "Metadaten": {
    "ID": "5678",
    "Zeitstempel": "2023-07-20T09:34:55.511Z",
    "Ursprung": "Subject",
    "Subject/Domaenen ID": "S-85739",
    "Tags": ["Diagnose", "Anonymisierung"]
  },
  "Textuelle Beschreibung": "Wenn meine Daten fuer etwas anderes als...",
  "Formale Beschreibung": "Wenn Zweck != 'Behandlungsprozess', anonymisiere...",
  "Vorschlaege Privacy-Filter": [
    {
      "Name": "Generalisierung",
      "Parameter": "k-Anonymity mit k=7"
    }
  ]
}
```

Quellcode 7.2: JSON-Dokument der Instanz des Beschreibungsmodells der Privacy-Forderung eines Data Subjects.

Das vorgestellte Beschreibungsmodell ermöglicht es dem Data-Owner, alle zu einem Datenprodukt zugehörigen Privacy-Forderungen zu finden. Zunächst ruft der Data Owner alle gesetzlichen Privacy-Forderungen und alle Privacy-Forderungen der Domäne, der das Datenprodukt gehört, auf. Anschließend ruft er alle Privacy-Forderungen von Data Subjects auf, deren „IDs“ in den Daten des Datenprodukts referenziert werden. Das Modell enthält außerdem eine Referenz auf den Ersteller des Datenprodukts. Im Fall von gesetzlichen Privacy-Forderungen ist das das zentrale Governance-Team. Bei Privacy-Forderungen von Domänen und Data Subjects ist jeweils die „Domänen ID“ bzw. die „Subject ID“ angegeben. Damit kann der Ersteller identifiziert werden. Um die originale Version der Privacy-Forderung zu erhalten, ist sowohl die „informelle Beschreibung“ als auch die „formale Beschreibung“ der Privacy-Forderung angegeben. Damit erfüllt das Beschreibungsmodell alle Anforderungen und unterstützt den Data Owner bei der effektiven Durchsetzung der Privacy-Forderungen.

7.2 Anwendung von Privacy-Filtern mit BARENTS

In diesem Abschnitt wird die effiziente Anwendung von Privacy-Filtern auf Daten betrachtet. Die Data Owner müssen zahlreiche und teils sehr komplexe Privacy-Forderungen mittels Privacy-Filtern umsetzen. Da sie unter Umständen keine IT-Experten sind, benötigen sie dabei Unterstützung. Hierfür kann die maßgeschneiderte Datenaufbereitungszone BARENTS [SBE+22a] verwendet werden. Im Data Mesh erlaubt sie nicht-IT-Experten, die Privacy-Filter zu beschreiben, die auf einen Datensatz angewendet werden sollen. So kann ein Data Owner die Privacy-Filter beschreiben, mit denen Privacy-Forderungen umgesetzt werden. Die beschriebenen Privacy-Filter werden anschließend automatisch auf die Datensätze angewendet.
BARENTS wurde in Python, mithilfe der *pandas*[1] Bibliothek umgesetzt. *Pandas* ist ein weit verbreiteter Standard zum Arbeiten mit Daten, skaliert aber schlecht für größere Datenmengen, da die Berechnungen im Arbeitsspeicher durchgeführt werden. Ist ein Datensatz zu groß für den Arbeitsspeicher oder werden durch eine Operation Kopien erzeugt, müssen Teile in den Hauptspeicher ausgelagert werden, wodurch sich die Laufzeit erheblich verlängert [Yıl23]. Der effiziente Umgang mit großen Datenmengen ist allerdings eine Anforderung an den Data Mesh-Ansatz, da in Organisationen und Firmen häufig mit großen Datenmengen gearbeitet wird. Auf diese Datenmengen werden zur Durchsetzung von Privacy-Forderungen die Privacy-Filter angewendet. Somit muss das unterstützende Werkzeug gut für große Datenmengen skalieren.

Aus diesem Grund werden in diesem Abschnitt alternative Bibliotheken zur Datenverarbeitung betrachtet, die das gleiche oder ein ähnliches API wie *pandas* besitzen und für große Datenmengen skalieren. Somit können sie *pandas* in BARENTS problemlos für die Anwendung von Privacy-Filtern ersetzen. Die Bibliotheken sind anhand ihrer Technik in drei Kategorien eingeteilt: In der erstes Kategorie befindet sich *pandas 2.0* [Pan23b], das ein Update für *pandas* darstellt. Die zweite Kategorie enthält *Grizzly* [HKS21] und *AFrame* [SC19] bzw. *PolyFrame* [SC21], welche die mit den Privacy-Filtern verbundenen Berechnungen in ein Datenbankmanagementsystem (DBMS) auslagern. Die dritte Kategorie enthält *MODIN* [PMX+20], *Koalas* [Dat20] und *Polars* [Pol23].

[1] siehe *pandas* Dokumentation [Pan23a]

Diese Bibliotheken verwenden Partitionierung und Parallelisierung, um die Berechnungen zu beschleunigen. Die Skalierungstechniken der Bibliotheken werden in den folgenden Abschnitten jeweils in ihren Grundzügen erklärt.

Update für Pandas: Pandas 2.0

Pandas 2.0 [Pan23b] ist die neue Version von *pandas*, die die Performance-Probleme adressiert. Es wurden zahlreiche Änderungen eingeführt, von denen im Folgenden drei beschrieben werden. Die erste Änderung ist, dass *pandas 2.0 PyArrow* im Backend unterstützt. Dadurch kann *Apache Arrows* Spaltenspeicherformat für Operationen verwendet werden [Par23]. Bisher wurde ausschließlich *NumPy* verwendet. *PyArrow* bietet eine bessere Perfomance für einige Operationen, beispielsweise das Lesen und Schreiben von Daten aus dem Speicher [Dat23]. Eine weitere Änderung ist die Anwendung des Copy-on-Write-Prinzips zur Speicheroptimierung. Dabei handelt es ich um einen lazy Kopiermechanismus, durch den das Kopieren von Daten verzögert wird. Das Kopieren erfolgt erst, wenn die Daten in einer Operation verwendet werden. Dadurch kann die Performance verbessert und Platz im Arbeitsspeicher gespart werden [Par23]. Weiter kann der *pandas* Index nun alle NumPy numeric Datentypen verwenden. Zuvor waren lediglich die int64, uint64 und float64 Datentypen möglich. Jetzt können insbesondere Datentypen mit kleinerer bit-Anzahl verwendet werden. Damit können Operationen nun Indizes mit kleineren bit-Größen verwenden, wodurch die Performance der Operationen verbesser wird [Par23].

Eignung für BARENTS: Das Update zu *pandas 2.0* enthält keine Veränderungen der API. Deshalb ist die API von *pandas 2.0* identisch mit der API von *pandas*. Im Gegensatz dazu weisen die anderen alternativen Bibliotheken zumindest syntaktische Unterschiede auf. Aus diesem Grund hat *pandas 2.0* die größte Kompatibilität zu *pandas*, wodurch die Umstellung sehr einfach wäre. Es muss lediglich das Update der Bibliothek durchgeführt werden.

Auslagerung der Berechnungen in ein DBMS

In diesem Abschnitt werden zwei Bibliotheken vorgestellt, die ihre Berechnungen in ein DBMS auslagern. Diese Strategie verbessert die Performance, da DBMS interne Mechanismen enthalten, um Berechnungen effizient auf großen Datenmengen ausführen zu können. Diese Fähigkeit kann genutzt werden, indem die in Python definierten Anfragen an das DBMS gesendet werden und dort ausgeführt werden.

Grizzly

Grizzly [HKS21; KH21] ist eine Python Bibliothek, die Anfragen der Nutzer*innen als SQL-Anfragen an ein DBMS sendet. Nutzer*innen schreiben ein Python Programm als Sequenz von DataFrame-Operationen. Die von *Grizzly* verwendete Syntax gleicht der *pandas* DataFrame-Syntax. Aus dieser Operationsfolge wird eine SQL Anfrage generiert, die an ein DBMS gesendet wird, in dem die Daten liegen. Dort werden die Berechnungen auf den Daten ausgeführt und die Nutzer*innen erhalten das Resultat [HKS21]. Die Anfrage wird als Lineage Graph dargestellt, der mittels lazy Bottom-Up-Evaluation ausgewertet wird. Das bedeutet die Berechnungen werden erst ausgeführt, wenn das Ergebnis aufgerufen wird [HKS21]. Es ist möglich Daten aus externen Dateien für die

Berechnungen zu verwenden. Dafür müssen die Daten zunächst in die Datenbank geladen werden. Anschließend können Anfragen darauf ausgeführt werden [HKS21]. *Grizzly* kann mit PostgreSQL, Actian Vector oder MonetDB verwendet werden [HKS21].

AFrame und PolyFrame

Das Ziel von *AFrame* [SC19] und *PolyFrame* [SC21] ist, Nutzer*innen *pandas* Syntax zu präsentieren, die im Hintergrund mit Big-Data-Fähigkeiten ausgestattet ist. *AFrame* besitzt eine API mit *pandas* ähnlicher DataFrame-Syntax über die mit den Daten in einer AsterixDB interagiert wird. AsterixDB enthält einen Query Optimizer für große und verteilte Datensätze, der auch verschachtelte Anfragen effizient optimieren kann. *AFrame* transformiert Anfragen inkrementell zu SQL++ Queries und verwendet Lazy Evaluation sowie Query Shipping. Denn erst wenn das Ergebnis abgefragt wird, werden die Queries zur Berechnung an AsterixDB gesendet [SC19]. *PolyFrame* ist eine sprachenunabhängige Erweiterung von AFrame. *PolyFrame* soll nicht an eine bestimmte Anfragensprache gebunden sein, sondern für verschiedene Anfragesprachen verwendet werden können. Dafür wird die Übersetzung der Anfragen in die jeweilige Anfragesprache von der inkrementellen Transformation gelöst. Zunächst wird die komplexe Anfrage inkrementell in ihre Basisoperationen zerlegt. Anschließend werden die Basisoperationen mithilfe einer sprachspezifischen Konfigurationsdatei in die jeweilige Anfragesprache übersetzt [SC21]. Es stehen keine öffentlichen Implementierungen für *AFrame* oder *PolyFrame* zur Verfügung.

Eignung für BARENTS: Neben syntaktischen Unterschieden zu *pandas* gibt es bei der Auslagerung der Berechnungen in ein DBMS einen weiteren Minuspunkt: Da die Berechnungen in einem DBMS durchgeführt werden, müssen die Daten in einer relationalen Datenbank vorliegen. Hierfür müssen sie im Vorfeld in die Datenbank geladen werden. Dies setzt voraus, dass sie in geeigneter Form vorliegen und erfordert Vorbereitung, beispielsweise muss ein Datenbankschema erstellt werden. Außerdem muss eine Verbindung zu der Datenbank hergestellt werden, bevor eine Anfrage abgesendet werden kann. Diese zusätzlichen Schritte führen zu einer deutlich aufwändigeren Umstellung als bei *pandas 2.0*.

Partitionierung und Parallelisierung

In diesem Abschnitt werden drei Bibliotheken vorgestellt, die eine Partitionierung der Daten sowie Parallelisierung der Verarbeitung nutzen, um die Performanz zu verbessern. Indem Datensätze partitioniert werden, kann eine Anfrage parallel für die unterschiedlichen Partitionen ausgeführt werden. Dadurch kann sich die Laufzeit der Anfrage verkürzen sowie der Speicherbedarf der Knoten.

MODIN

MODIN [PMX+20] übersetzt Anfragen zunächst in eine eigene Algebra und verwendet anschließend eine Execution Engine, um die Anfragen parallel auszuführen. *Pandas* Operatoren können mit weniger Operatoren umgesetzt werden, wodurch sich der Optimierungsaufwand reduzieren lässt. Aus diesem Grund überführt *MODIN* die *pandas*-basierten Anfragen zunächst in die eigenen Algebra [PMX+20]. Anschließend wird die Anfrage optimiert und partitioniert. Die einzelnen

Partitionen werden dann mithilfe der gewählten Execution Engine von je einem Worker bearbeitet. Anschließend fasst die koordinierende Komponente alle Zwischenergebnisse zu einem Endergebnis zusammen [PMX+20]. *MODIN* unterstützt die folgenden drei Execution Engines: *Unidist* [Uni23a] ist eine Framework, das eine einheitliche API für verteilte Ausführung von Python Programmen bereitstellt und verschiedene Execution Backends unterstützt. Die zweite Execution Engine ist das Allzweck-Framework für Cluster-Computing namens *Ray* [MNW+18]. Dieses wird sowohl zum Skalieren von AI Rechenlasten als auch allgemeinen Python Applikationen verwendet. Zuletzt kann man die *Dask Execution Engine* [BB20] zur Verarbeitung der Anfragen nutzen. Dabei handelt es sich um eine Bibliothek für die parallele Ausführung von Python Programmen.

Koalas

Die Python Bibliothek *Koalas* [Dat20] verwendet *PySpark* [Apa23b] um Anfragen effizient auf große Datenmengen anzuwenden. Sie übersetzt *pandas* APIs in einen *SparkSQL* [Apa23c] Ausführungsplan. *PySpark* kann große Datenmengen effizient verarbeiten. *Koalas* macht es Nutzer*innen möglich, weiterhin mit der bekannten *pandas* Syntax zu arbeiten und gleichzeitig die Big-Data-Fähigkeiten von *PySpark* zu verwenden. Hierfür können *Koalas* DataFrames verwendet werden. Sie haben die Syntax von *pandas* DataFrames, basieren aber auf *PySpark* DataFrames [Tak20]. *SparkSQL* verwendet einen Optimizer, um einen effizienten Ausführungsplan für die Ausführung auf einem Cluster zu berechnen. Dabei wird eine bessere Laufzeit erzielt als beim klassischen Map-Reduce. Dafür werden die Daten während den Berechnungen im Arbeitsspeicher gehalten, um schnelleren Zugriff darauf zu erlauben. Zudem wird dadurch das Laden und Ablegen von Daten aus dem Hauptspeicher vermieden [Ama23]. Um die Wiederverwendung von Daten möglich zu machen, werden die DataFrames im Arbeitsspeicher behalten. So können sie mit geringer Latenz von mehreren Operationen verwendet werden [Ama23]. *Koalas* wurde nach *PySpark* portiert und ist seit *Apache Spark 3.2* offiziell Teil von *PySpark*.

Polars

Die Bibliothek *polars* [Pol23] verwendet *pandas* Syntax, um eine DataFrame Interface auf einer OLAP Query Engine zu implementieren. Die Query Engine nutzt das *Apache Arrow Columnar Format* [Apa23a] als Speichermodell, in dem Single Instruction Multiple Data (SIMD) Register verwendet werden, um eine optimale Performance zu erreichen [Vin21]. Zusätzlich parallelisiert *polars* die Operationen. Für Operationen, für die Zwischenergebnisse nicht synchronisiert werden müssen, werden herkömmliche Methoden der Parallelisierung verwendet. In den Fällen, in denen sonst eine Synchronisation notwendig ist, beispielsweise die Group-By-Operation, wird Lock-Free-Hashing verwendet. Anstatt die Daten arbiträr zu partitionieren und zu verteilen, werden die Daten so verteilt, dass jeder Thread eine disjunkte Menge von Grouping-Keys bearbeitet. Welche Zeilen er zu bearbeiten hat, prüft jeder Thread für sich. Dadurch entsteht insgesamt mehr Arbeit, die jedoch parellel ausgeführt werden kann. Zudem kann der Synchronisationsschritt übersprungen werden, wodurch Zeit gespart wird [Vin21]. *Polars* bietet zwei APIs an: Eine verwendet Eager- und die andere Lazy-Evaluation. Die Lazy-Evaluation sollte vor allem für Performance-kritische Programmabschnitte verwendet werden, um Zeit zu sparen [Vin21].

Eignung für BARENTS: Durch Partitionierung von Daten und Parallelisierung von Operationen können signifikante Laufzeitverbesserungen erzielt werden. Allerdings lassen sich nicht alle Operationen parallelisieren (zum Beispiel die Group-By-Operation), wodurch sich die Effektivität der Verbesserungen reduziert. Der Lock-Free-Hashing-Ansatz von *polars* ist sehr vielversprechend, da dadurch mehr Operationen parallelisiert werden können. Ein weiteres Problem ist der Laufzeit-Overhead, der durch die Koordination von Clustern entstehen kann, die von *MODIN* verwendet werden. Außerdem müssen alle Arbeitsschritte, die mit dem Cluster in Verbindung stehen, wie beispielsweise die Initialisierung des Clusters, Teil des Programmcodes sein. Zusätzlich zu kleinen Syntaxunterschieden wird die Umstellung von *pandas* dadurch komplexer.

In Kapitel 8 wird geprüft, wie geeignet die vorgestellten Bibliotheken in der Praxis sind, um von einem Data Owner für die effiziente Anwendung von Privacy-Filtern verwendet zu werden. Hierfür wird ein selbst entwickelter Prototyp vorgestellt, welcher zum Verlgeich der Laufzeit von *pandas* mit der Laufzeit der vorgestellten Bibliotheken verwendet wird. Darin wird eine typische Anfrage der Datenverarbeitung an *pandas* und eine Auswahl der vorgestellten Bibliotheken gestellt. Dabei werden jeweils die Laufzeiten gemessen und anschließend werden die Ergebnisse der verschiedenen Bibliotheken verglichen.

8 Prototypische Implementierung

Das Ziel dieses Kapitels ist der Vergleich von *pandas* mit den in Abschnitt 7.2 vorgestellten alternativen Bibliotheken hinsichtlich ihrer Fähigkeit, große Datenmenge zu verarbeiten. Hierfür wurde ein Prototyp einer beispielhaften Aufbereitung eines Datenprodukts mit Privacy-Filtern entwickelt. Damit können die Laufzeiten der verschiedenen Bibliotheken gemessen und verglichen werden. Zunächst wurde der Ablauf der Aufbereitung definiert, der aus dem vorgestellten Anwendungsfall stammt (siehe Kapitel 2). Er ist so gestaltet, wie ein Data Owner bei der Anwendung von Privacy-Filtern vorgehen würde. Die Laufzeit der Aufbereitung wurde für alle Bibliotheken gemessen und anschließend wurden die Ergebnisse verglichen. Da die Implementierungen von *AFrame* und *PolyFrame* nicht zur Verfügung stehen, konnten sie nicht getestet werden. *Koalas* wurde nach *PySpark* portiert und ist daher keine eigenständige Bibliothek mehr. Aus diesem Grund wurde *Koalas* ebenfalls aus dem Prototyp ausgeschlossen. Es wurden die Laufzeiten von *pandas* mit *pandas 2.0*, *Grizzly*, *MODIN* und *Polars* verglichen. Damit ist aus jeder Kategorie ein repräsentativer Vertreter vorhanden. Ferner sind mit *MODIN* und *polars* auch zwei unterschiedliche Ansätze zur Parallelisierung und Partitionierung vertreten. Während *MODIN* eher für die Anwendung in Clustern ausgelegt ist, ist *polars* eher für Desktop-Rechner oder kleinere Server geeignet.

8.1 Vorgehensweise

Um die Laufzeitmessung vorzubereiten, wurden zunächst synthetische Daten generiert. Diese stammen aus dem Szenario der Optimierung der Dienstpläne einer Station (siehe Abschnitt 2.2) und entsprechen den Leistungsdaten einer Fachabteilung. Sie stehen für das Datenprodukt, welches von der Stationsleitung angefragt wird. Bevor der Data Owner den Zugriff darauf gewähren kann, muss er die Erfüllung aller geltenden Privacy-Forderungen prüfen. Daher wurden als nächstes zwei Privacy-Forderungen definiert, welche für die Daten gelten können. Die erste Privacy-Forderung ist, dass Patient*innen die Verwendung von Daten für Analysezwecke ganz verbieten. Hierfür muss das jeweilige Tupel der Patient*in ganz herausgefiltert werden. Die zweite Privacy-Forderung besagt, dass das Attribut „Details“, das die Behandlungsdetails enthält, entfernt werden muss, bevor die Daten für eine Analyse verwendet werden dürfen. Hierfür muss das besagte Attribut aus dem Tupel der Patient*in entfernt werden. Um die Anwendung der Privacy-Filter zu vereinfachen, wurden die geltenden Privacy-Forderungen zu den generierten Datensätzen hinzugefügt. Jedes Tupel in den Datensätzen entspricht einer Patient*in. Welche Patient*innen eine oder beide Privacy-Forderungen haben, wurde zufällig gewählt. Ungefähr 6% der Patient*innen haben die erste Privacy-Forderung und ungefähr 14% der Patient*innen haben die zweite Privacy-Forderung. Damit haben etwa 0.8% der Patient*innen beide Privacy-Forderungen. Da die zweite Privacy-Forderung in der ersten Privacy-Forderung enthalten ist, hat eine Kombination aus beiden keinen weiteren Effekt. Um die Generierung der Daten möglichst simpel zu halten, wurde diese Möglichkeit nicht ausgeschlossen. Ein Ausschnitt aus den Daten ist in Tabelle 8.1 dargestellt.

VersichertenNr	Abteilung	Behandelt von	Leistung	...
46505987	Notaufnahme	11787	Röntgen	...
81207299	Notaufnahme	10992	Wundversorgung	...
25600020	Notaufnahme	38075	Labortest	...
...	...	...	...	...

...	Details	P-F Daten entf.	P-F Details entf.
...	wichtige Anmerkung	n	n
...	normale Anmerkung	j	j
...	peinliche Anmerkung	n	j
...	...	...	...

Tabelle 8.1: Auszug aus synthetischen Leistungsdaten für den Prototyp.

Der Data Owner prüft die Erfüllung der beiden Privacy-Forderungen und stellt fest, dass sie nicht erfüllt sind. Daher muss er die Daten mithilfe von Privacy-Filtern so aufbereiten, dass die Privacy-Forderungen erfüllt sind. Deshalb definiert er Privacy-Filter in der BARENTS-Ontologie, die dann mithilfe von *pandas* auf die Daten angewendet werden. Um zu prüfen, wie geeignet die alternativen Bibliotheken als Ersatz für *pandas* sind, wird die Umsetzung und Anwendung der Privacy-Filter mit den verschiedenen Bibliotheken verglichen. Es wird sowohl die Laufzeit als auch die Ähnlichkeit der Skripte verglichen. Um den Einfluss der Datenmenge auf die Laufzeit zu prüfen, wurden Datensätze in fünf verschiedenen Größen generiert. Die enthaltenen Daten unterscheiden sich für jeden Datensatz, da sie jeweils neu generiert wurden. Die Datensätze wurden in CSV-Datein der Größen 0,24MB (4k Zeilen), 2,44MB (40k Zeilen), 24,4MB (400k Zeilen), 244MB (4000k Zeilen) und 2,44GB (40000k Zeilen) gespeichert. Um das Lesen von Datenbanken testen zu können, wurden die Datensätze zudem in eine *postgreSQL* Datenbank geladen. Deshalb wurden zwei Arten von Abläufen verwendet und jeweils die Laufzeit gemessen: Im ersten Ablauf wurden die Daten aus der CSV-Datei gelesen, die Privacy-Filter wurden angewendet und das Ergebnis wurde wieder in einer CSV-Datei gespeichert. Im zweiten Ablauf wurden die Daten aus der *postgreSQL* Datenbank gelesen, die Privacy-Filter wurden angewendet und das Ergebnis wurde ebenfalls wieder in einer CSV-Datei gespeichert. Die CSV-Datei steht für das fertig aufbereitete Datenprodukt, auf das der anfragenden Domäne Zugriff gewährt wird.

Die Messungen wurden folgendermaßen durchgeführt: Für jede Bibliothek wurden pro Dateigröße die Laufzeiten von fünf Durchläufen dokumentiert. Das arithmetische Mittel dieser fünf Durchläufe wurde als Laufzeit für die jeweilige Kombination aus Ablauf und Bibliothek festgehalten. Hierfür wurde die `timeit`-Methode der gleichnamigen Python-Bibliothek verwendet. Diese führt automatisch die gewünschte Anzahl Durchläufe durch, misst für jeden Durchlauf die Laufzeit und ermittelt am Ende die durchschnittliche Laufzeit. Es wurde die zu messende Methode mit den Aufbereitungsschritten übergeben sowie der `connection-String` für die Verbindung mit der Datenbank. Für *MODIN* wurde das Cluster vor Beginn der Messungen einmal initialisiert. Es wurden keine weiteren Schritte unternommen, um die Durchläufe vor- oder nachzubereiten. Durchgeführt wurden die Messungen auf einem einzelnen Desktop-PC mit einem Intel Core i5 Prozessor mit 6 Kernen und 12 logischen Prozessoren und 16GB Arbeitsspeicher. Die Bibliotheken *pandas 1.5.3*, *pandas 2.0.3*, *Grizzly 0.1.6*, *polars 0.19.2*, *MODIN 0.23.0* mit *Ray 2.6.1* wurden in *Python 3.9* getestet.

In Algorithmus 8.1 ist das Skript der Aufbereitung für *pandas* dargestellt. Zunächst werden die Daten entweder aus der CSV-Datei oder der Datenbank gelesen. Anschließend werden zur Durchsetzung der ersten Privacy-Forderung die Tupel herausgefiltert, die diese Privacy-Forderung haben. Dann werden die „Details“ der Tupel durch einen leeren Eintrag ersetzt, die die zweite Privacy-Forderung haben. Zuletzt wird das Resultat in einer CSV-Datei gespeichert. Für *pandas* und *pandas 2.0* konnten beide Abläufe durchgeführt werden. Um das Skript für *pandas 2.0* anzupassen, musste lediglich das Update auf die neue Version durchgeführt werden. Zudem konnte das *PyArrow* Backend für das Lesen der CSV-Datei verwendet werden, indem `engine='pyarrow'` als zweiter Parameter an die `read_csv` Methode übergeben wird.
Um das Skript für die anderen Bibliotheken anzupassen, waren Änderungen an der Syntax nötig. Allerdings konnten nicht für alle Bibliotheken beide Abläufe genau umgesetzt werden.
Für *Grizzly* war die Syntax nahezu identisch, allerdings kann *Grizzly* nur Daten aus Datenbanken lesen, weswegen nur der zweite Ablauf durchgeführt werden konnte (siehe Algorithmus 8.2). Zudem fehlt im Vergleich zu *pandas* Funktionalität, weswegen der Datensatz zum Entfernen der „Details“ aufgeteilt werden musste und anschließend nicht mehr zusammengesetzt werden konnte. Da *Grizzly* zudem Daten nicht in CSV-Dateien schreiben kann, wurde das Schreiben der Resultate anders umgesetzt, um trotzdem eine möglichst äquivalenten Ablauf der Aufbereitung zu erhalten. Es wurden zwei Varianten verwendet: In der ersten Variante wurde die `show()` Methode verwendet, die *Grizzly* anbietet. Durch sie werden die definierten Transformationen ausgeführt und das Ergebnis in die Konsole geschrieben. In der zweiten Variante wurde die `to_csv()` Methode von *pandas* verwendet. So konnte die Laufzeit der Methode von *Grizzly* mit der Methode von *pandas* verglichen werden.
Als Execution Engine für *MODIN* wurde *Ray* gewählt. *Ray* bietet Nutzer*innen die Möglichkeit die Anzahl der CPUs festzulegen, die für die Berechnungen verwendet werden. Um den Einfluss der Anzahl der CPUs auf die Laufzeit zu prüfen, wurden beide Abläufe einmal mit zwei Kernen und

Algorithmus 8.1 Die Umsetzung der Privacy-Filter für *pandas*.

```
import pandas as pd

def statement(connection, size, source):

  # Daten aus CSV oder DB lesen
  all_data = None
  if source == "csv":
  all_data = pd.read_csv("DataGeneration/behandlungsdaten"+size+".csv")
  elif source == "db":
  all_data = pd.read_sql("SELECT * FROM behandlungsdaten"+size, connection)

  # Privacy-Filter fuer die erste Privacy-Forderung
  result = all_data[(all_data["anf_daten_entf"] == "n")]

  # Privacy-Filter fuer die zweite Privacy-Forderung
  result.loc[result.anf_details_entf == "j", "details"] = ""

  # Resultat wird in einer CSV gespeichert
  result.to_csv("pandasResult.csv")
```

einmal mit vier Kernen durchgeführt. Die Syntax von *MODIN* ist identisch zu *pandas* aber aufgrund von fehlenden Alternativen musste der Datensatz aufgeteilt werden, um die „Details“ zu entfernen (siehe Algorithmus 8.3). Die beiden Teile konnten anschließend nicht wieder zusammengefügt werden.
Für *Polars* wurden beide Abläufe durchgeführt. Die Funktionalität konnte dabei direkt in die *polars*-Syntax übersetzt werden (siehe Algorithmus 8.4). Um die Laufzeit zu verbessern, konnten die Transformationen in einem Schritt durchgeführt werden und der Datentyp `lazyframe` beim Lesen der CSV-Datei verwendet werden. Für diesen Datentyp wird ausschließlich Lazy-Evaluation verwendet. Für das Lesen aus einer Datenbank steht er nicht zur Verfügung.

Algorithmus 8.2 Die Umsetzung der Privacy-Filter für *Grizzly*.

```
import grizzly
import pandas
from grizzly.relationaldbexecutor import RelationalExecutor
from grizzly.sqlgenerator import SQLGenerator

# Verwendet pandas to_csv um Resultat zu speichern
def to_csv(df, csv_file):
  tmp=df.collect(includeHeader=True)
  pdf = pandas.DataFrame(data=tmp[1:], columns=tmp[0])
  pdf.to_csv(csv_file)

def statement(connection, size, output):
  # Daten werden aus DB gelesen (CSV nicht moeglich)
  grizzly.use(RelationalExecutor(connection, SQLGenerator("postgresql")))
  all_data = grizzly.read_table("behandlungsdaten"+size)

  # Privacy-Filter fuer die erste Privacy-Forderung
  data_with_details = all_data[(all_data.anf_details_entf == "n")
                      & (all_data.anf_daten_entf == "n")]

  # Privacy-Filter fuer die zweite Privacy-Forderung
  data_without_details = all_data[(all_data.anf_details_entf == "n")
                      & (all_data.anf_daten_entf == "j")]
  data_without_details = data_without_details[["versichertennr_b", "abteilung",
    "behandelt_von", "leistung", "anf_daten_entf", "anf_details_entf"]]

  # Resultat wird ausgegeben oder als CSV gespeichert
  if output == "show":
    data_with_details.show()
    data_without_details.show()
  elif output == "csv":
    to_csv(data_with_details, "grizzlyResult1.csv")
    to_csv(data_without_details, "grizzlyResult2.csv")
```

Algorithmus 8.3 Die Umsetzung der Privacy-Filter für *MODIN*.

```
import modin.pandas as md

def statement(connection,size, source):
  # Daten aus CSV oder DB lesen
  all_data = None
  if source == "csv":
    all_data = md.read_csv("DataGeneration/behandlungsdaten"+size+".csv")
  elif source == "db":
    all_data = md.read_sql("SELECT * FROM behandlungsdaten" + size, connection)

  # Privacy-Filter fuer die erste Privacy-Forderung
  data_with_details = all_data[(all_data["anf_daten_entf"] == "n")
                       & (all_data["anf_details_entf"] == "n")]

  # Privacy-Filter fuer die zweite Privacy-Forderung
  data_without_details = all_data[(all_data["anf_daten_entf"] == "n")
                       & (all_data["anf_details_entf"] == "j")]
  data_without_details = data_without_details.drop("details", axis=1)

  # Resultat wird in einer CSV gespeichert
  data_with_details.to_csv("modinResult1.csv", index=False)
  data_without_details.to_csv("modinResults2.csv", index=False)
```

Algorithmus 8.4 Die Umsetzung der Privacy-Filter für *polars*.

```
import polars

def statement(connection, size, source):
  result = None
  # Daten aus CSV oder DB lesen
  if source == "csv":
    # Privacy-Filter fuer beide Privacy-Forderungen in einem Schritt
    result = polars.scan_csv("DataGeneration/behandlungsdaten" + size + ".csv")
      .filter((polars.col("anf_daten_entf") =="n"))
      .with_columns(details=polars.when(polars.col("anf_details_entf") == "j").then("")
      .otherwise(polars.col("anf_details_entf")))
  elif source == "db":
    # Privacy-Filter fuer beide Privacy-Forderungen in einem Schritt
    result = polars.read_database(query="SELECT * FROM behandlungsdaten" + size,
      connection=connection).lazy()
      .filter((polars.col("anf_daten_entf") == "n"))
      .with_columns(details=polars.when(polars.col("anf_details_entf") == "j").then("")
      .otherwise(polars.col("anf_details_entf")))

  # Resultat wird in einer CSV gespeichert
  result.sink_csv("polarsResult.csv")
```

8.2 Ergebnisse der Messungen

Tabelle 8.2 und Tabelle 8.3 enthalten die Ergebnisse der Laufzeitmessungen. In Abbildung 8.1 und Abbildung 8.2 sind sie grafisch dargestellt. Abbildung 8.1 zeigt die Ergebnisse für den Ablauf, in dem die Daten aus den CSV-Dateien gelesen wurden. Für *pandas* erhöht sich die Laufzeit bei jeder größeren Datenmenge ungefähr um den Faktor zehn. Bei der letzten Messung (2,44GB) erzielt *pandas* eine Laufzeit von rund 550 Sekunden. *Pandas 2.0* erzielt gegenüber *pandas* konstant bessere Laufzeiten. Bei der letzten Messung benötigt *pandas 2.0* rund 450 Sekunden. Auffällig ist, dass beide *MODIN* Konfigurationen langsamer als alle anderen Bibliotheken sind. Besonders ab der mittleren Datenmenge wird der Abstand immer deutlicher. Erst für die größte Datenmenge nähert sich *MODIN*s Laufzeit wieder *pandas* an. Dabei ist die Variante mit zwei Kernen etwas schneller und die Variante mit vier Kernen etwas langsamer als *pandas*. Beide Varianten sind langsamer als *pandas 2.0*. *Polars* hat die kürzeste Laufzeit von allen. Dies zeigt sich besonders in den letzten beiden Messungen. Das Verarbeiten der größten Datenmenge dauert mit *polars* etwa 30 Sekunden, es geht also mehr als zehn mal so schnell wie mit *pandas*.

Bibliothek	Datenmenge (CSV-Datei)				
	0,24MB	2,44MB	24,4MB	244MB	2,44GB
pandas	0.139	0.581	5.601	56.088	556.802
pandas 2.0	0.069	0.442	4.117	41.263	446.088
MODIN 2 Kerne	11.361	12.402	60.602	121.384	535.388
MODIN 4 Kerne	10.874	10.996	141.164	257.715	583.778
polars	0.157	0.051	0.327	2.869	29.6

Tabelle 8.2: Die Ergebnisse der Laufzeitmessungen mit CSV-Dateien als Quelle in Sekunden.

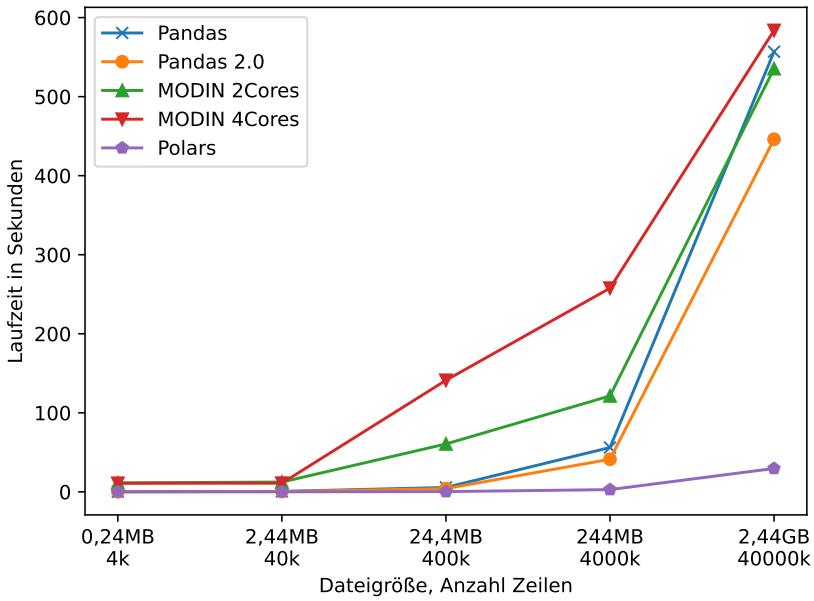

Abbildung 8.1: Die Ergebnisse der Laufzeitmessungen für den Ablauf, in dem die Daten aus den CSV-Dateien gelesen wurden.

Die Ergebnisse des Ablaufs, in dem die Daten aus der *postgreSQL* Datenbank gelesen werden, sind in Abbildung 8.2 dargestellt. Auch hier erhöht sich die Laufzeit von *pandas* zunächst jeweils um den Faktor zehn. Bei der letzten Messung ist eine deutlich stärkere Steigerung zu sehen. Hier beträgt die Laufzeit mit 3000 Sekunden beinahe das vierzigfache der vorangegangenen Messung. *Pandas 2.0* schneidet wieder konsistent besser ab als *pandas*, die letzte Messung ergibt 2500 Sekunden. *MODIN* ist mit beiden Konfigurationen von Beginn an sichtbar langsamer als *pandas*. Die Variante mit zwei Kernen ist zunächst schneller als die Variante mit vier Kernen. In der letzten Messung sind beide Varianten deutlich langsamer als *pandas*. Allerdings ist die Variante mit vier Kernen (3500 Sekunden) deutlich schneller als die Variante mit zwei Kernen (4000 Sekunden). *Polars* ist mit 260 Sekunden in der letzten Messung zwar nicht am schnellsten, aber trotzdem immer noch mehr als zehn Mal so schnell wie *pandas*. *Grizzly* erreicht in den ersten vier Messungen ähnliche Laufzeiten wie *pandas* und *pandas 2.0*.

Bibliothek	Datenmenge (DB)				
	0,24MB	2,44MB	24,4MB	244MB	2,44GB
pandas	0.274	0.754	7.544	77.197	3043.061
pandas 2.0	0.17	0.695	6.933	70.097	2439.674
MODIN 2 Kerne	12.532	45.257	70.138	211.154	3961.16
MODIN 4 Kerne	12.364	18.315	185.275	337.679	3475.021
polars	0.322	0.446	2.846	26.194	265.357
Grizzly `show()`	0.074	0.222	2.057	16.517	163.718
Grizzly `to_csv()`	0.148	0.858	8.608	81.588	1619.538

Tabelle 8.3: Die Ergebnisse der Laufzeitmessungen mit der Datenbank als Quelle in Sekunden.

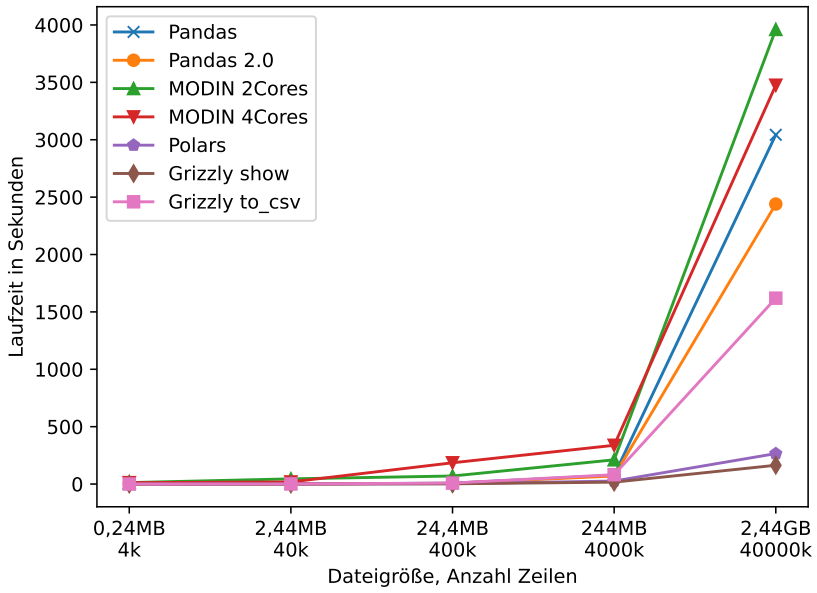

Abbildung 8.2: Die Ergebnisse der Laufzeitmessungen für den Ablauf, in dem die Daten aus der Datenbank gelesen wurden.

Die Variante ohne das Speichern des Resultats, in der stattdessen `show()` verwendet wird, erzielt für die letzte Messung mit 160 Sekunden die kürzeste Laufzeit von allen. Die Variante mit `to_csv()` ist deutlich langsamer, zeigt aber in der letzten Messung mit 1600 Sekunden eine deutliche Verbesserung gegenüber *pandas 2.0*.

Interpretation: Die konsistent schnelleren Laufzeiten von *pandas 2.0* im Vergleich zu *pandas* zeigen, dass die Verbesserungen, die an *pandas 2.0* vorgenommen wurden, effektiv sind.
Im Gegensatz dazu war *MODIN* unerwartet langsamer. Die Initialisierung des *Ray* Clusters wurde aus den Messungen ausgeschlossen. Eine mögliche Erklärung für die langen Laufzeiten ist, dass die Koordination des Clusters einen Overhead darstellt, der sich in allen Messungen bemerkbar macht. Zusätzlich wurde bei der letzten Messung der Arbeitsspeicher voll ausgeschöpft. Das wurde vermutlich durch das Kopieren der Datenpartitionen für die Nodes des Clusters verursacht. Als Folge davon wurden Daten in den Hauptspeicher ausgelagert, wodurch die Laufzeit verlängert wurde. Auch die Unterschiede zwischen den beiden Konfigurationen lassen sich mit dem Overhead durch die Koordination des Clusters erklären. Für die meisten Messungen erzielte die Variante mit weniger Kernen bessere Laufzeiten, da der Koordinationsaufwand für mehr Kerne stärker ins Gewicht fällt, als die Beschleunigung die dadurch erzielt werden kann. Einzig bei der letzten Messung, in der die Daten aus der Datenbank gelesen werden, erzielt die Variante mit vier Kernen eine bessere Laufzeit. Aus den Beobachtungen lässt sich schließen, dass *MODIN* mit *Ray* prinzipiell eher nicht für die Verwendung auf der gegebenen Hardware geeignet ist. Für optimale Ergebnisse wären vermutlich entweder mehr Arbeitsspeicher oder sogar mehrere Rechner notwendig.

Die Messungen zeigen, dass es einen großen Unterschied zwischen den beiden Varianten der Aufbereitung mit *Grizzly* gibt. Das Verwenden der `to_csv()` Methode von *pandas* verschlechtert *Grizzlys* Laufzeit deutlich. Allerdings kann *Grizzly* alleine keine Ergebnisdateien erzeugen. Daher wäre die Bibliothek nur in Verbindung mit der *pandas*-Methode für den Anwendungsfall nutzbar. Auch dann ist *Grizzly* noch deutlich schneller als *pandas* und *pandas 2.0*. *Grizzly* ist außerdem gut zum schnellen Testen von Transformationen geeignet. Dabei kann die `show()` Methode verwendet werden, um zur Überprüfung einen Teil der Ergebnisse anzuzeigen. *Polars* ist für beide Abläufe deutlich schneller als *pandas 2.0*. Zusammen mit *Grizzly* erzielt es die besten Ergebnisse. *Polars* bietet allerdings einige Vorteile gegenüber *Grizzly*: Zum einen kann *Polars* Daten von CSV-Dateien lesen und zudem Ergebnisdateien erzeugen. Außerdem stehen bei *Polars* eine größere Menge an Transformationen zur Verfügung.

Gewonnene Erkenntnisse: BARENTS kann im Data Mesh von den Data-Ownern verwendet werden, um Privacy-Filter zu definieren, die anschließend von BARENTS auf die Daten angewendet werden. Das Ziel dieses Kapitels war herauszufinden, ob die alternativen Bibliotheken dazu geeignet sind, statt *pandas* in BARENTS verwendet zu werden, um die Anwendung von Privacy-Filtern umzusetzen. Dafür sollte die Syntax der Bibliothek der Syntax von *pandas* ähneln, damit eine Umstellung einfach erfolgen kann. Zudem sollte die Bibliothek deutlich kürzere Laufzeiten als *pandas* erzielen, um die Anwendung von Privacy-Filtern mit BARENTS effizienter zu machen. Die Syntax von *pandas 2.0* ist identisch zu *pandas* und *pandas 2.0* erzielt signifikante Laufzeitverbesserungen bei gleicher Funktionalität. Die Syntax von *MODIN* ist identisch zu *pandas*. *MODIN* besitzt jedoch weniger Funktionalität als *pandas* und scheint, aufgrund des Overheads, der durch die Koordination des Clusters entsteht, nicht für die Verwendung auf einzelnen Rechnern geeignet. Auch *Grizzly* besitzt eine Syntax, die nahezu identisch zu *pandas* ist. Zudem erzielt erzielt *Grizzly* gute Laufzeiten für die Transformationen, ist aber in der Menge der Transformationen eingeschränkt und kann selbst keine Ergebnisdateien erzeugen. Die Syntax von *polars* unterscheidet sich von

pandas, weswegen Skripte übersetzt werden müssen. Allerdings können die Verarbeitungsschritte eins zu eins übersetzt werden und *polars* bietet die gleiche Funktionalität wie *pandas*. Dabei erzielt *polars* Laufzeiten, die um ein vielfaches schneller sind als *pandas 2.0*.

Sowohl *pandas 2.0* als auch *polars* sind gute Alternativen zu *pandas* in BARENTS. *Pandas 2.0* bietet die einfachste Umstellung und ermöglicht eine signifikante Laufzeitverbesserung. Um *polars* zu verwenden, muss mehr Aufwand für die Umstellung betrieben werden, aber dafür wird eine sehr große Laufzeitverbesserung erzielt.

9 Evaluation

In diesem Kapitel wird die Eignung des entwickelten Konzepts für die Erweiterung des Data-Mesh-Ansatz um die Privacy-Aspekte evaluiert. Hierfür wird geprüft inwieweit die an das Konzept gestellten Anforderungen (siehe Kapitel 3) erfüllt wurden. Es werden der Reihe nach die sieben Anforderungen betrachtet. Zunächst werden jeweils kurz die zentralen Aspekte der Anforderung wiederholt. Anschließend wird diskutiert, ob und wie sie durch das entwickelte Konzept im Data-Mesh-Kontext erfüllt werden. Zuletzt wird das Konzept mit den verwandten Arbeiten zu Data Mesh aus Kapitel 5 verglichen.

A1: Alle Privacy-Forderungen werden abgebildet.

Im Data-Mesh-Ansatz gibt es drei Arten von Privacy-Forderungen. Es gibt gesetzliche Privacy-Forderungen, Privacy-Forderungen von Domänen und Privacy-Forderungen von Data Subjects. Jede Art von Privacy-Forderung wird von einer anderen involvierten Partei kontrolliert und ist unterschiedlich eng mit den erhobenen Daten gekoppelt. Während die gesetzlichen Privacy-Forderungen unabhängig von den Daten sind, sind die anderen beiden Arten von Privacy-Forderungen abhängig von den Daten. Daher ist es sinnvoll wenn die Erhebung und Abbildung der Privacy-Forderungen Teil der Datenerhebung ist.

Der Prozess der Erhebung von Privacy-Forderungen wird in Abschnitt 6.1 vorgestellt. Die gesetzlichen Privacy-Forderungen werden von einem zentralen Governance-Team aus Experten modelliert, da sie sehr selten und unabhängig von erhobenen Daten angepasst werden müssen. Außerdem sind Experten notwendig, um die Gesetzestexte angemessen zu modellieren. Die Privacy-Forderungen der Domänen werden durch ein domäneninternes Team von Experten modelliert, da sie ebenfalls selten angepasst werden und dafür Domänenexperten notwendig sind. Die Modellierung von den Privacy-Forderungen der Data-Subjects ist an die Erhebung der Daten gekoppelt, da Data-Subjects häufig neue Privacy-Forderungen haben oder Änderungen durchführen möchten. Dies geschieht entweder wenn neue Daten erhoben werden oder spontan, und kann ebenfalls als erneute Datenerhebung umgesetzt werden. Das Data-Subject kann die Privacy-Forderung in natürlicher Sprache definieren. Für die anschließende Formalisierung wurde auf verschiedene existierende Lösungen hingewiesen. Ein Modell für die Abbildung der formalisierten Privacy-Forderungen wurde in Abschnitt 7.1 vorgestellt. Es enthält unterschiedliche Elemente für die unterschiedlichen Arten von Privacy-Forderungen.

Der Prozess der Erhebung von Privacy-Forderungen schließt alle drei Arten von Privacy-Forderungen ein, so dass der jeweilige Kontext berücksichtigt wird. Das Beschreibungsmodell erlaubt eine Unterscheidung der Art der Privacy-Forderungen und bildet für jede Art die relevanten Metadaten ab. So wird sichergestellt, dass alle Privacy-Forderungen abgebildet werden können, wodurch die Anforderung erfüllt ist.

A2: Die Privacy-Forderungen werden gespeichert.

Die erhobenen Privacy-Forderungen werden nicht sofort nach der Datenerhebung verwendet sondern erst, wenn die Daten weiterverarbeitet werden. Auch müssen sie später erneut geprüft werden, wenn Daten von anderen Domänen angefragt werden. Gesetzliche Privacy-Forderungen und Privacy-Forderungen von Domänen gelten zudem auch für in Zukunft erhobene Daten. Aus diesen Gründen müssen die Privacy-Forderungen verfügbar bleiben. Dafür müssen sie gespeichert werden. Dabei müssen alle Verantwortlichen Zugriff auf die Privacy-Forderungen haben und sie dürfen, außer auf Wunsch des Verfassers, nicht verändert werden.

Die Speicherung von Privacy-Forderungen ist Teil des Prozesses der Erhebung in Abschnitt 6.1. Nachdem die Privacy-Forderungen formalisiert werden, werden sowohl die erhobenen Daten als auch die zugehörigen Privacy-Forderungen gespeichert. Die Infrastruktur für die Speicherung der Privacy-Forderungen kann beliebig gewählt werden. Es muss ein zentraler Speicherort sein, auf den alle Domänen Zugriff haben und der keine unrechtmäßigen Veränderungen zulässt. Eine mögliche Technologie, die hierfür verwendet werden kann ist die Blockchain. Dabei wird eine Privacy-Forderung in einem Block gespeichert und kann dann über den Hashwert des Blocks referenziert werden.

Der Prozess der Erhebung der Privacy-Forderungen berücksichtigt also auch die Speicherung der Privacy-Forderungen. Dabei wird keine spezifische Technologie vorgegeben, aber es werden die Bedingungen definiert, die eine passende Technologie erfüllen muss. Die Betreiber des Data Mesh sind frei eine für sie passende Technologie zur Umsetzung der Speicherung zu wählen. Die Anforderung ist erfüllt.

A3: Die Privacy-Forderungen können den Daten, für die sie gelten, zugeordnet werden.

Die verschiedenen Arten von Privacy-Forderungen haben verschiedene Wirkungsbereiche. Die gesetzlichen Privacy-Forderungen gelten für alle Datensätze des Data Mesh, die Privacy-Forderungen einer Domäne gelten für Datensätze, die sich in der Domäne befinden und die Privacy-Forderungen der Data-Subjects gelten nur für die Daten des jeweiligen Data-Subjects. Bevor geprüft werden kann, ob die Privacy-Forderungen eines Datensatzes erfüllt sind oder nicht, muss herausgefunden werden, welche Privacy-Forderungen für diesen Datensatz gelten. Deshalb muss man zuordnen können, welche Privacy-Forderungen für einen Datensatz gelten.

Diese Anforderung wird im Beschreibungsmodell in Abschnitt 7.1 berücksichtigt. Das Modell bildet alle drei Arten von Privacy-Forderungen ab. Im Element „Metadaten“ wird unter anderem der Ursprung der Privacy-Forderung dokumentiert. Darin wird eingetragen, ob die Privacy-Forderung aus der DSGVO, aus einer Domäne oder von einem Data Subject stammt. Damit kann nachvollzogen werden, um welche Art von Privacy-Forderung es sich handelt. Außerdem gibt es das optinale Element „Subject / Domänen ID“, das nur für Privacy-Forderungen von Domänen und Data Subjects ausgefüllt wird. Darin wird die ID des Data Subjects oder der Domäne eingetragen, von der die Privacy-Forderung stammt. So kann zugeordnet werden, wer die Privacy-Forderung verfasst hat.

Mithilfe dieser Elemente kann der Data Owner feststellen, welche Privacy-Forderungen für ein gegebenes Datenprodukt gelten. Entsprechend ihres Wirkungsbereichs werden zunächst alle gesetzlichen Privacy-Forderungen und alle Privacy-Forderungen seiner Domäne gesammelt. Dazu kommen die Privacy-Forderungen mit den IDs der Data-Subjects, deren Daten im Datenprodukt liegen. Diese gelten dann nur für den jeweiligen Teil der Daten, der von diesem Data Subject stammt. So erhält der Data Owner alle Privacy-Forderungen, die für das Datenprodukt bzw. Teile des Datenprodukts gelten. Zusätzlich kann über die vorgeschlagenen Privacy-Filter eine Handlungsempfehlung gemacht werden. Damit ist die Anforderung erfüllt.

A4: Die Einhaltung der Privacy-Forderungen, die für einen Datensatz gelten, muss durch den / die Verantwortlichen für den Datensatz sichergestellt werden.

Laut Art. 24 DSGVO liegt die Durchsetzung des Datenschutzes für einen Datensatz in der Verantwortung der Person, die für den Datensatz verantwortlich ist. Daher ist der / die Verantwortliche, im Data-Mesh-Ansatz Data Owner genannt, auch für die Durchsetzung der Privacy-Forderungen zuständig. Dafür muss der Data Owner auf die Privacy-Forderungen zugreifen, ihre Erfüllung prüfen und gegebenenfalls Schritte einleiten können, um sie durchzusetzen.

In Kapitel 6 gibt es drei Stellen, an denen eine Prüfung der Erfüllung der Privacy-Forderungen relevant ist. Bei der Zugriffsanfrage auf ein Datenprodukt (siehe Abschnitt 6.2) muss die Erfüllung des Datenschutz sichergestellt werden, bevor der Zugriff erteilt wird. Der Data Owner des Datenprodukts prüft, ob Privacy-Forderungen vorhanden sind. Falls ja prüft er, ob sie erfüllt sind. Wenn eine oder mehrere Privacy-Forderungen nicht erfüllt sind, führt der Data Owner eine entsprechende Aufbereitung mit Privacy-Filtern durch. Im Prozess der Erstellung eines neuen Datenprodukts (siehe Abschnitt 6.3) nutzt eine Domäne Teile eines angefragten Datenprodukts, um ein neues Datenprodukt zu erstellen. Dabei werden die Daten des Datenprodukts mit anderen Daten kombiniert oder verändert. Deshalb muss der Data Owner erneut die Erfüllung der Privacy-Forderungen prüfen. Dabei geht er so vor wie im Prozess der Zugriffsanfrage. Seine Verantwortung erlischt erst, wenn das neue Datenprodukt freigegeben wird, da dann die Verantwortung an die andere Domäne übergeht. Bei der Fusion von mehreren Datenprodukten (siehe Abschnitt 6.4) werden mehrere bestehende Datenprodukte zur Erstellung eines neuen Datenprodukts verwendet. In diesem Fall müssen alle Data Owner der bestehenden Datenprodukte die Privacy-Forderungen ihrer Datenprodukte prüfen.

Es ist ersichtlich, dass die Prüfung der Erfüllung der Privacy-Forderungen immer durch den Data Owner erfolgt. Außerdem ist der Zeitpunkt der Übergabe der Verantwortung für das Datenprodukt klar festgelegt. Daher ist zu jedem Zeitpunkt klar, wer für ein Datenprodukt verantwortlich ist und dieser Pflicht nachkommen muss. Damit ist die Anforderung erfüllt.

A5: Das Konzept muss möglichst unabhängig von der vorhandenen Infrastruktur umsetzbar sein.

Soll in einer Organsation ein Data Mesh eingeführt werden, so ist es von Vorteil, wenn Teile der bestehenden Infrastruktur weiterverwendet werden können, da dadurch Kosten und Zeit gespart werden. Deshalb sollen nur die Anforderungen an die Implementierungen der einzelnen Komponenten vorgegeben werden. Welche Technologie dafür verwendet wird, soll den Betreibern überlassen werden.

Der Prozess der Erhebung von Privacy-Forderungen (siehe Abschnitt 6.1) beschreibt für vier der Prozessschritte die Anforderungen an die Umsetzung. Die Beschreibung der Privacy-Forderungen in natürlicher Sprache wird formalisiert, so dass sie besser verarbeitet werden können. Hierfür gibt es existierende Lösungen, die an dieser Stelle eingesetzt werden können. Die Speicherung der Privacy-Forderungen soll zentral stattfinden, so dass alle Domänen darauf zugreifen können. Beispielsweise können sie in einer Blockchain abgelegt werden. Die Datensätze und Datenprodukte werden in der Infrastruktur der jeweiligen Domäne gespeichert. Hierfür gibt es keine speziellen Anforderungen, die Infrastruktur muss lediglich den Anforderungen der Domäne genügen. In drei Prozessen (siehe Abschnitte 6.2 bis 6.4) werden Daten mithilfe von Privacy-Filtern aufbereitet. Der zuständige Data Owner sollte dabei von Werkzeugen, wie zum Beispiel BARENTS, unterstützt werden. Dadurch wird eine effektive und effiziente Definition und Anwendung von Privacy-Filtern ermöglicht. Mit welchen Werkzeugen und Technologien die Anwendung der Privacy-Filter erfolgt, wird nicht festgelegt.

Die Prozesse des Konzepts enthalten für einige Komponenten Hinweise, wie sie implementiert werden können. Dabei wird beschrieben, welche Eigenschaften die Implementierung haben soll. Für einige wird zusätzlich eine Beispieltechnologie angegeben, die aber nicht verwendet werden muss. Daher können bei der Umsetzung beliebige Technologien verwendet werden, solange sie die benötigten Eigenschaften erfüllen. Die vorhandene Infrastruktur kann also weitestgehend weiterverwendet werden, vorausgesetzt die Technologien besitzen die geforderten Eigenschaften. Damit ist die Anforderung erfüllt.

A6: Das Konzept muss die Verwendung von Standardtechnologien ermöglichen.

Wenn bereits eine Dateninfrastruktur besteht, sind die Nutzer*innen im Umgang mit den Schnittstellen vertraut und haben präferierte Technologien. Würden diese beim Wechsel zu einer neuen Dateninfrastruktur ausgetauscht werden, müssten die Nutzer*innen den Umgang mit den neuen Technologien zunächst lernen. Das kostet einerseits Zeit und kann andererseits frustrierend für die Nutze*innen sein. Deshalb ist es von Vorteil Standardtechnologien zu verwenden, die von vielen Fachleuten verwendet werden. Dafür muss das Konzept analog zu **A5** unabhängig von bestimmten Technologien definiert werden.

Im Prozess der Erhebung von Privacy-Forderungen (siehe Abschnitt 6.1) werden die Anforderungen für die Umsetzung von drei Komponenten beschrieben. In drei Prozessen (siehe Abschnitte 6.2 bis 6.4) werden Datensätze vom zuständigen Data Owner mithilfe von Privacy-Filtern aufbereitet. Dabei soll der Data Owner durch Werkzeuge unterstützt werden. Dadurch wird die effektive und effiziente Definition und Anwendung von Privacy-Filtern ermöglicht. BARENTS wurde in Abschnitt 7.2 als mögliche Umsetzung der Aufbereitung mit Privacy-Filtern vorgestellt. BARENTS ist mit der *pandas*-Bibliothek implementiert, die ein weit verbreiteter Standard zum Arbeiten mit Daten ist.

Die Prozesse enthalten für einige Komponenten Hinweise darauf, wie sie umgesetzt werden können. Dabei wird beschrieben welche Eigenschaften geeignete Technologien haben sollen. Zusätzlich werden an einigen Stellen Beispieltechnologien angegeben. Diese müssen aber nicht verwendet werden. Es können beliebige Technologien verwendet werden, die die geforderten Eigenschaften besitzen. Wie mit BARENTS und *pandas* gezeigt wurde, können dabei insbesondere Standardtechnologien verwendet werden. Damit ist die Anforderung erfüllt.

A7: Das Privacy-Konzept muss Skalierbarkeit für große Datenmengen ermöglichen.

In Data Meshes werden große Datenmengen für den Austausch zwischen Domänen vorbereitet. Der dadurch entstehende Zeitaufwand, zum Beispiel die Anwendung von Privacy-Filtern, sollte Verantwortliche und Nutzer*innen nicht in ihrer Arbeit behindern. Deshalb muss die Dateninfrastruktur in der Lage sein, mit großen Datenmengen umzugehen. Das Konzept muss deshalb so umgesetzt werden, dass Skalierbarkeit für große Datenmengen gegeben ist. Dafür können Technologien und Werkzeuge eingesetzt werden, die für den Umgang mit großen Datenmengen geeignet sind.

Das Konzept ist unabhängig von bestimmten Technologien, d.h. es können beliebige Technologien zur Umsetzung verwendet werden. Ein Werkzeug, das für die Umsetzung von Privacy-Filtern verwendet werden kann, ist BARENTS. Es eignet sich sehr gut zur Unterstützung des Data-Owners bei der effektiven Anwendung von Privacy-Filtern. Allerdings kann BARENTS nicht effizient mit großen Datenmengen umgehen, da die zugrundeliegende Python Bibiliothek *pandas* nicht gut skaliert. In Kapitel 8 wurde im Rahmen der prototypischen Implementierung empirisch gezeigt, dass es Alternativen zu *pandas* gibt, die effizient mit großen Datenmengen umgehen können. Diese können statt *pandas* für BARENTS verwendet werden. Damit wird die Anwendung der Privacy-Filter mit BARENTS effizienter.

Indem das Konzept unabhängig von bestimmten Technologien gestaltet ist, wird die freie Wahl von passenden Technologien ermöglicht. Dadurch kann BARENTS zur effektiven Anwendung von Privacy-Filtern verwendet werden. Auch kann die BARENTS zugrundeliegende Technologie *pandas* durch eine besser skalierende Alternative ausgetauscht werden. Dadurch wird für die Anwendung von Privacy-Filtern Skalierbarkeit für große Datenmengen erreicht. Analog können für andere Komponenten des Konzepts Technologien und Werkzeuge gewählt werden, die Skalierbarkeit für große Datenmengen bieten. Damit ist die Anforderung erfüllt.

Vergleich mit verwandten Arbeiten: In Tabelle 9.1 ist dargestellt, inwieweit die verwandten Arbeiten zu Data Mesh (siehe Kapitel 5) die Anforderungen **A1-A7** erfüllen. Das vorgestellte Konzept wurde anhand der Anforderungen mit Bezug auf seine Eignung für den Data-Mesh-Ansatz evaluiert. Daher werden auch die verwandten Arbeiten mit Bezug auf Data Mesh evaluiert. Ein Großteil der verwandten Arbeiten beschäftigen sich im Rahmen eines bestimmten Kontexts, beispielweise im IoT-Kontext, mit einem einzelnen Aspekt eines Privacy-Konzepts. Diese können nicht direkt auf den Data-Mesh-Ansatz übertragen werden. Die Anforderungen beziehen sich auf ein vollständiges Privacy-Konzept im Rahmen des Data-Mesh-Ansatz. Daher wird ihre Erfüllung nur für die verwandten Arbeiten geprüft, die sich mit dem Data-Mesh-Ansatz beschäftigen.

Die erste Gruppe enthält Arbeiten die sich mit dem Konzept des Data-Mesh-Ansatz beschäftigen. Dehghani [Deh22] geht oberflächlich auf die Privacy ein. Die Richtlinien, die den ganzen Data Mesh betreffen, sollen von einem zentralen Governance-Team festgelegt werden (**A1**). Der Data Owner ist für die Umsetzung dieser Richtlinien zuständig (**A4**). Die Arbeit enthält kein Konzept oder eine genauere Spezifikation dazu wie diese Vorgaben erfüllt werden können.
Machado et al. [MCS21; MCS22] stellen ein Domänenmodell und eine konzeptionelle Architektur für Data Mesh vor. Darin ist ein Security-Mechanismus enthalten aber Konzepte für die Einbindung von Privacy fehlen komplett.
Die PricewaterhouseCoopers GmbH [Pri22] hat eine Studie zu Data Mesh durchgeführt. Dabei hat sie herausgefunden, dass die globalen Richtlinien von einem Governance Officer festgelegt werden

Arbeit	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**
Dehghani [Deh22]	(✓)	✗	✗	(✓)	✗	✗	✗
Machado et al. [MCS21; MCS22]	✗	✗	✗	✗	✗	✗	✗
PricewaterhouseCoopers GmbH [Pri22]	(✓)	✗	✗	(✓)	✗	✗	✗
Goedebuure et al. [GKD+23]	✗	✗	✗	(✓)	✗	✗	✗
Bode et al. [BKK+23b]	(✓)	✗	✗	✗	✗	✗	✗
Joshi et al. [JPR21]	✗	✗	✗	✗	✗	✗	✗
Lei et al. [LPS+22]	✗	✗	✗	✗	✗	✗	✗
Chee [Che21]	✗	✗	✗	✗	✗	✗	✗
Eigenes Konzept	✓	✓	✓	✓	✓	✓	✓

Tabelle 9.1: Evaluation der verwandten Arbeiten hinsichtlich der Anforderungen aus Kapitel 3.

(**A1**) und Data Owner die Zugriffsbeschränkungen für ihre Datenprodukte festlegen und umsetzen (**A4**). Sie gehen nicht darauf ein wie diese Aspekte umgesetzt werden. Sie beschreiben außerdem, wie IKEA Data Mesh umsetzt, gehen dabei aber nicht auf die Umsetzung der Privacy-Aspekte ein.

Die zweite Gruppe enthält Arbeiten die sich mit Data Mesh in der Praxis beschäftigen.
Goedebuure et al. [GKD+23] präsentieren einen Grey Literature Review zu Data Mesh in dem sie auch auf Privacy-Aspekte eingehen. In Data Mesh legt eine Global Governance den Security- und Privacy-Standard fest. Die Local Governance der Domänen soll die Regeln für die Zugriffskontrolle der Datenprodukte festlegen (**A4**). Der Review beschreibt nicht mit welchen Prozessen die Regelungen erfüllt werden oder wie sie modelliert werden.
Bode et al. [BKK+23b] stellen das Ergebnis von Interviews mit Data-Mesh-Experten vor. Diese betonen die Wichtigkeit einer zentralen Steuereinheit, die unter anderem auch die Governance Regeln rund um die Privacy festlegen soll (**A1**). Es wird nicht darauf eingegangen wie genau das geschieht.
Einige Arbeiten [Che21; JPR21; LPS+22] befassen sich jeweils mit der Data-Mesh-Lösung eines spezifischen Unternehmens. Dabei gehen sie nicht darauf ein ob und wie Privacy-Aspekte eingebracht werden.

Aus der Evaluation des vorgestellten Konzepts ist ersichtlich, dass es die Anforderungen **A1-A7** im Kontext des Data Mesh erfüllt. Das in Kapitel 6 vorgestellte Konzept trägt zur Erfüllung von **A1** bei und erfüllt **A2** sowie **A4-A6**. Durch das Beschreibungsmodell in Abschnitt 7.1 werden **A1** und **A3** erfüllt. Die in Kapitel 8 vorgestellte prototypische Implementierung zeigt mithilfe von Messungen empirisch, dass **A7** erfüllt ist. Damit erfüllt das vorgestellte Privacy-Konzept in Kombination mit den Implementierungsstrategien und der prototypischen Implementierung alle gestellten Anforderungen. Das entwickelte Konzept für den bedarfsgerechten Schutz von Datenprodukten mit Fokus auf personenbezogene Daten kann in einem Data Mesh verwendet werden.

10 Zusammenfassung und Ausblick

Daten sind einer der begehrtesten Rohstoffe unserer Zeit. Sie erhalten ihren Wert häufig durch die in ihnen enthaltenen Informationen, auf deren Grundlage beispielsweise Geschäftsentscheidungen getroffen werden können. Um die Informationen zu extrahieren müssen die Daten in einer Dateninfrastruktur bereitgestellt werden, die solche Analysen ermöglicht. In den verbreiteten zentralisierten Dateninfrastrukturen wie dem Data Warehouse und dem Data Lake werden die Daten von einem designierten Team aus Datenexperten betreut. Ein Unternehmen besteht aus vielen verschiedenen Domänen mit unterschiedlichen Anforderungen an die Datenhaltung. Die Verwaltung aller Daten ist eine schwierige Aufgabe für die Datenexperten, da die Datenmenge sehr groß ist und außerdem Domänenwissen fehlt, um die Daten fachmännisch zu betreuen. Der Data-Mesh-Ansatz ist ein neuer organisationaler Ansatz für den Umgang mit analytischen Daten. Dabei handelt es sich um einen dezentralen Ansatz in dem die Betreuung und Aufbereitung der Daten in der Verantwortung der Domäne liegt, die die Daten besitzt. Die Daten werden bei Bedarf in Form von Datenprodukten zwischen den Domänen ausgetauscht. Neben anderen Eigenschaften muss ein Datenprodukt auch den Datenschutz erfüllen. Das bedeutet der Zugriff auf die Datenprodukte muss so geregelt werden, dass keine personenbezogenen Daten an unberechtigte Dritte weitergegeben werden.

Als Laufbeispiel für diese Arbeit wurde ein Anwendungsfall eingeführt, der stellvertretend für alle Anwendungsfälle steht, in denen besonders Wert auf den Datenschutz gelegt werden muss. Der Anwendungsfall umfasst drei Szenarien, die sich in einem Krankenhaus abspielen. Das Krankenhaus hat eine komplexe Domänenstruktur in der häufig heterogene und Privacy-problematische Daten zwischen den Domänen ausgetauscht werden. Die Szenarien unterscheiden sich in den Arten der Privacy-Forderungen, die erfüllt werden müssen und der Art der Verarbeitung der angefragten Datenprodukte. Auf Grundlage des Anwendungsfalls wurden sieben Anforderungen an eine mögliche Lösung definiert. Darin sind sowohl Anforderungen an die Privacy-Forderungen, als auch an die Nutzbarkeit der Lösung in der Praxis enthalten.

Zunächst wurden die grundlegenden Prinzipien des Data-Mesh-Ansatz betrachtet. Dadurch wurde gezeigt, dass der Data-Mesh-Ansatz sich gut für den Anwendungsfall eignet. Da das Krankenhaus eine komplexe organisationale Struktur hat, ist die Übergabe der Verantwortung an die Domänen ein sinnvoller Schritt. Der Datenaustausch, der einen wichtigen Aspekt der Szenarien darstellt, wird durch das Konzept des Datenprodukts unterstützt. Außerdem ist die Privacy ein zentraler Aspekt des Anwendungsfalls, da mit personenbezogenen Gesundheitsdaten gearbeitet wird. Die Privacy ist in den Eigenschaften eines Datenprodukts enthalten, aber die bisherige Literatur hat nicht spezifiziert, wie sie umgesetzt wird. Daher wurde eine Literaturrecherche durchgeführt um zu untersuchen, inwieweit bestimmte Privacy-Aspekte im Kontext des Data-Mesh-Ansatzes berücksichtigt werden. Es wurde die Erhebung von Privacy-Forderungen, die Beschreibung von Privacy-Forderungen und die Prüfung und Durchsetzung von Privacy-Forderungen untersucht. Es zeigte sich, dass sie im Kontext des Data-Mesh-Ansatz im Vergleich zu anderen Bereichen noch nicht ausreichend berücksichtigt wurden. Während die Lösungen für die Erhebung von

Privacy-Forderungen unabhängig vom Anwendungsfall einsetzbar sind, müssen sowohl Lösungen zur Beschreibung als auch zur Prüfung und Durchsetzung von Privacy-Forderungen angepasst werden, bevor sie im Data-Mesh-Ansatz verwendet werden können.

Daher wurde ein eigenes Privacy-Konzept für die Erfüllung von Privacy-Aspekten im Data-Mesh-Ansatz vorgestellt. Es wurde jeweils ein Konzept für den Prozess der Erhebung der Privacy-Forderungen, für den Prozess der Prüfung und Durchsetzung der Privacy-Forderungen bei Zugriffsanfragen, für den Prozess der Erstellung eines neuen Datenprodukts aus einem bestehenden Datenprodukt und für den Prozess der Erstellung eines neuen Datenprodukts aus mehreren bestehenden Datenprodukten vorgestellt. Die Privacy-Vorgänge sind nach dem Privacy-by-Design-Konzept von vornherein in die Prozesse integriert. Beispielsweise findet während der Datenerhebung auch die Erhebung der Privacy-Forderungen statt und bei Zugriffsanfragen auf Datenprodukte ist die Prüfung der Privacy-Forderungen mitbedacht. Damit Data Owner der Datenprodukte die Privacy-Forderungen effektiv prüfen und durchsetzen können, müssen die Privacy-Forderungen geeignet abgebildet werden. Daher wurde zusätzlich ein Beschreibungsmodell für die Privacy-Forderungen entwickelt. Das Beschreibungsmodell erlaubt eine Unterscheidung der Art der Privacy-Forderungen und bildet für jede Art die relevanten Metadaten ab. So wird sichergestellt, das alle Privacy-Forderungen abgebildet werden können.

Data Owner müssen zahlreiche und teilweise komplexe Privacy-Forderungen mithilfe von Privacy-Filtern umsetzen. Da sie unter Umständen keine IT-Experten sind, benötigen sie hierfür Unterstützung durch Technologien, um abgebildete Privacy-Forderungen effizient umsetzen zu können. Hierfür kann die maßgeschneiderte Datenaufbereitungszone BARENTS verwendet werden. Diese erlaubt Nutzer*innen, Privacy-Filter zu beschreiben die dann automatisch auf den Datensatz angewendet werden. Da BARENTS mit der Python Bibliothek *pandas* umgesetzt wurden, skaliert BARENTS schlecht für große Datenmengen. Aus diesem Grund wurden sechs alternative Bibliotheken zur Datenverarbeitung betrachtet, die für große Datenmengen skalieren und statt *pandas* mit BARENTS für die Anwendung von Privacy-Filtern verwendet werden können. Neben dem Update für *pandas* wurden zwei Bibliotheken betrachtet, die die Berechnungen in ein DBMS auslagern und drei Bibliotheken, die Partitionierung und Parallelisierung zur Beschleunigung der Berechnungen verwenden. Anschließend wurde eine prototypische Implementierung einer beispielhaften Anwendung von Privacy-Filtern zur Durchsetzung von Privacy-Forderungen für einen Datensatz umgesetzt. Die Laufzeit der verschiedenen Bibliotheken wurde gemessen um empirisch herauszufinden, ob sich diese Bibliotheken für diese Art von Anwendung eignen. Es wurde gezeigt, dass mit diesen alternativen Bibliotheken eine effizientere Durchsetzung der Privacy-Forderungen erreicht werden kann als mit *pandas*. Demnach kann die Skalierung von BARENTS durch Ersetzung von *pandas* durch die Alternativen verbessert werden.

Abschließend wurde das Konzept sowie die Implementierungsstrategien anhand der zuvor definierten Anforderungen hinsichtlich der Einbringung der Privacy-Aspekte in den Kontext des Data-Mesh-Ansatzes evaluiert. Aus der Evaluation ist ersichtlich, dass das vorgestellte Privacy-Konzept in Kombination mit den Implementierungsstrategien und der prototypischen Implementierung alle gestellten Anforderungen erfüllt. Daher kann es in einem Data Mesh für den bedarfsgerechten Schutz von Datenprodukten mit Fokus auf personenbezogene Daten verwendet werden.

Literaturverzeichnis

[ABK18] K. Abouelmehdi, A. Beni-Hessane, H. Khaloufi. „Big healthcare data: preserving security and privacy“. In: *Journal of Big Data* 5.1 (Jan. 2018), S. 1. ISSN: 2196-1115. DOI: 10.1186/s40537-017-0110-7. URL: https://doi.org/10.1186/s40537-017-0110-7 (besucht am 26. 09. 2023) (zitiert auf S. 21).

[AD14] T. Alshugran, J. Dichter. „Extracting and modeling the privacy requirements from HIPAA for healthcare applications“. In: *IEEE Long Island Systems, Applications and Technology (LISAT) Conference 2014*. Mai 2014, S. 1–5. DOI: 10.1109/LISAT.2014.6845198. URL: https://ieeexplore.ieee.org/abstract/document/6845198 (besucht am 01. 10. 2023) (zitiert auf S. 41, 47, 48).

[Ama23] Amazon. *What is Apache Spark? | Introduction to Apache Spark and Analytics | AWS*. en-US. 2023. URL: https://aws.amazon.com/big-data/what-is-spark/ (besucht am 26. 06. 2023) (zitiert auf S. 67).

[ANTK23] M. Ali, F. Naeem, M. Tariq, G. Kaddoum. „Federated Learning for Privacy Preservation in Smart Healthcare Systems: A Comprehensive Survey“. In: *IEEE Journal of Biomedical and Health Informatics* 27.2 (Feb. 2023), S. 778–789. ISSN: 2168-2208. DOI: 10.1109/JBHI.2022.3181823. URL: https://ieeexplore.ieee.org/document/9794622 (besucht am 06. 10. 2023) (zitiert auf S. 36).

[Apa23a] Apache Arrow. *Arrow Columnar Format*. 2023. URL: https://arrow.apache.org/docs/format/Columnar.html (zitiert auf S. 67).

[Apa23b] Apache Spark. *PySpark Overview*. 2023. URL: https://spark.apache.org/docs/latest/api/python/index.html (zitiert auf S. 67).

[Apa23c] Apache Spark. *SparkSQL*. 2023. URL: https://spark.apache.org/sql/ (zitiert auf S. 67).

[Bal23a] S. Balnojan. *Data Mesh Applied*. en. Jan. 2023. URL: https://towardsdatascience.com/data-mesh-applied-21bed87876f2 (besucht am 05. 04. 2023) (zitiert auf S. 32).

[Bal23b] S. Balnojan. *There's More Than One Kind of Data Mesh — Three Types of Data Meshes*. en. Feb. 2023. URL: https://towardsdatascience.com/theres-more-than-one-kind-of-data-mesh-three-types-of-data-meshes-7cb346dc2819 (besucht am 05. 04. 2023) (zitiert auf S. 31, 32).

[Bas15] L. Bassett. *Introduction to JavaScript object notation: a to-the-point guide to JSON*. O'Reilly Media, Inc., 2015 (zitiert auf S. 61).

[BAW+12] J. Benn, G. Arnold, I. Wei, C. Riley, F. Aleva. „Using quality indicators in anaesthesia: feeding back data to improve care“. In: *BJA: British Journal of Anaesthesia* 109.1 (Juli 2012), S. 80–91. ISSN: 0007-0912. DOI: 10.1093/bja/aes173. URL: https://doi.org/10.1093/bja/aes173 (besucht am 26. 09. 2023) (zitiert auf S. 22).

[BB20] S. Böhm, J. Beránek. „Runtime vs Scheduler: Analyzing Dask's Overheads“. In: *2020 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*. arXiv:2010.11105 [cs]. Nov. 2020, S. 1–8. DOI: 10.1109/WORKS51914.2020.00006. URL: http://arxiv.org/abs/2010.11105 (besucht am 05. 04. 2023) (zitiert auf S. 67).

[BE99] M. Böhnlein, A. U. vom Ende. „Xml—extensible markup language“. In: *Wirtschaftsinformatik* 41 (1999), S. 274–276 (zitiert auf S. 61).

[BG13] A. Bauer, H. Günzel. *Data-Warehouse-Systeme: Architektur, Entwicklung, Anwendung*. dpunkt. verlag, 2013 (zitiert auf S. 15, 31).

[BGH+13] M. Backes, S. Gerling, C. Hammer, M. Maffei, P. von Styp-Rekowsky. „AppGuard – Enforcing User Requirements on Android Apps“. en. In: *Tools and Algorithms for the Construction and Analysis of Systems*. Hrsg. von N. Piterman, S. A. Smolka. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2013, S. 543–548. ISBN: 9783642367427. DOI: 10.1007/978-3-642-36742-7_39 (zitiert auf S. 43).

[BKK+23a] J. Bode, N. Kühl, D. Kreuzberger, S. Hirschl, C. Holtmann. *Data Mesh: Motivational Factors, Challenges, and Best Practices*. Techn. Ber. arXiv:2302.01713 [cs] type: article. arXiv, Apr. 2023. DOI: 10.48550/arXiv.2302.01713. URL: http://arxiv.org/abs/2302.01713 (besucht am 26. 09. 2023) (zitiert auf S. 33, 44).

[BKK+23b] J. Bode, N. Kühl, D. Kreuzberger, S. Hirschl, C. Holtmann. *Data Mesh: Motivational Factors, Challenges, and Best Practices*. arXiv:2302.01713 [cs]. Apr. 2023. URL: http://arxiv.org/abs/2302.01713 (besucht am 28. 04. 2023) (zitiert auf S. 84).

[Bor20] J. F. Boris Dinjus. *Krankenhausabteilungen: Überblick der Abteilungen med, non-med.* de-DE. 2020. URL: https://www.klinik-supervision.de/krankenhausabteilungen/ (besucht am 22. 09. 2023) (zitiert auf S. 19).

[Bund23] Bundesbeauftragte für den Datenschutz und die Informationsfreiheit. *BfDI - Gesundheit und Soziales - Datenübermittlung zu Abrechnungszwecken*. 2023. URL: https://www.bfdi.bund.de/DE/Buerger/Inhalte/GesundheitSoziales/Allgemein/DatenuebermittlungZuAbrechnungszwecken.html (besucht am 12. 09. 2023) (zitiert auf S. 25, 26).

[Che21] C. W. Chee. *HelloFresh Journey to the Data Mesh*. en. Okt. 2021. URL: https://engineering.hellofresh.com/hellofresh-journey-to-the-data-mesh-7fe590f26bda (besucht am 02. 05. 2023) (zitiert auf S. 45, 84).

[CM19] C. Chhetri, V. G. Motti. „Eliciting Privacy Concerns for Smart Home Devices from a User Centered Perspective“. en. In: *Information in Contemporary Society*. Hrsg. von N. G. Taylor, C. Christian-Lamb, M. H. Martin, B. Nardi. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, S. 91–101. ISBN: 9783030157425. DOI: 10.1007/978-3-030-15742-5_8 (zitiert auf S. 39, 47).

[CNC11] M. Conti, V. T. N. Nguyen, B. Crispo. „CRePE: Context-Related Policy Enforcement for Android“. en. In: *Information Security*. Hrsg. von M. Burmester, G. Tsudik, S. Magliveras, I. Ilić. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2011, S. 331–345. ISBN: 9783642181788. DOI: 10.1007/978-3-642-18178-8_29 (zitiert auf S. 43).

[Cod72] E. F. Codd. „Relational Completeness of Data Base Sublanguages“. In: *Data Base Systems. Courant Computer Science Symposium 6: May 24–25, 1971*. Hrsg. von R. Rustin. Englewood Cliffs, NJ, USA: Prentice-Hall, 1972, S. 65–98. ISBN: 978-0131967410 (zitiert auf S. 35).

[Dat20] Databricks. *Koalas: pandas API on Apache Spark — Koalas 1.8.2 documentation*. 2020. URL: https://koalas.readthedocs.io/en/latest/ (besucht am 19. 04. 2023) (zitiert auf S. 64, 67).

[Dat23] Data:Algo. *Pandas 2.0 Ditches NumPy for PyArrow: What You Need to Know*. en. März 2023. URL: https://dataalgo.medium.com/pandas-2-0-ditches-numpy-for-pyarrow-what-you-need-to-know-cbba4cb60249 (besucht am 23. 10. 2023) (zitiert auf S. 65).

[Dat89] C. J. Date. *A Guide to the SQL Standard*. Addison-Wesley Longman Publishing Co., Inc., 1989 (zitiert auf S. 35).

[DCL23a] A. Dolhopolov, A. Castelltort, A. Laurent. „Exploring the Benefits of Blockchain-Powered Metadata Catalogs in Data Mesh Architecture“. working paper or preprint. Mai 2023. URL: https://hal.umontpellier.fr/hal-04156089 (zitiert auf S. 48).

[DCL23b] A. Dolhopolov, A. Castelltort, A. Laurent. „Implementing a Blockchain-Powered Metadata Catalog in Data Mesh Architecture“. working paper or preprint. Juli 2023. URL: https://hal.umontpellier.fr/hal-04156134 (zitiert auf S. 48).

[DCL23c] A. Dolhopolov, A. Castelltort, A. Laurent. *Trick or Treat: Centralized Data Lake vs Decentralized Data Mesh*. en. Mai 2023. URL: https://hal.umontpellier.fr/hal-04156120 (besucht am 07. 11. 2023) (zitiert auf S. 61).

[Deh22] Z. Dehghani. *Data Mesh*. O'Reilly, 2022. URL: https://www.oreilly.com/library/view/data-mesh/9781492092384/ (zitiert auf S. 15, 16, 31–33, 43, 47, 51, 83, 84).

[Dix10] J. Dixon. *Pentaho, Hadoop, and Data Lakes*. en. Okt. 2010. URL: https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/ (besucht am 03. 10. 2023) (zitiert auf S. 31).

[DSSK19] S. Dash, S. K. Shakyawar, M. Sharma, S. Kaushik. „Big data in healthcare: management, analysis and future prospects“. In: *Journal of Big Data* 6.1 (Juni 2019), S. 54. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0217-0. URL: https://doi.org/10.1186/s40537-019-0217-0 (besucht am 26. 09. 2023) (zitiert auf S. 20, 21).

[DWS+11] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, W. Joosen. „A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements“. en. In: *Requirements Engineering* 16.1 (März 2011), S. 3–32. ISSN: 1432-010X. DOI: 10.1007/s00766-010-0115-7. URL: https://doi.org/10.1007/s00766-010-0115-7 (besucht am 01. 10. 2023) (zitiert auf S. 39).

[EGG+21] R. Eichler, C. Giebler, C. Gröger, E. Hoos, H. Schwarz, B. Mitschang. „Enterprise-Wide Metadata Management: An Industry Case on the Current State and Challenges“. In: *Business Information Systems* (Juli 2021), S. 269–279. DOI: 10.52825/bis.v1i.47 (zitiert auf S. 40).

[ERM15] K. E. Emam, S. Rodgers, B. Malin. „Anonymising and sharing individual patient data“. en. In: *BMJ* 350 (März 2015), h1139. ISSN: 1756-1833. DOI: 10.1136/bmj.h1139. URL: https://www.bmj.com/content/350/bmj.h1139 (besucht am 26. 09. 2023) (zitiert auf S. 35).

[EU16] Europäisches Parlament und Rat der europäischen Union. *Verordnung (EU) 2016/679 des Europäischen Parlaments und des Rates vom 27. April 2016 zum Schutz natürlicher Personen bei der Verarbeitung personenbezogener Daten, zum freien Datenverkehr und zur Aufhebung der Richtlinie 95/46/EG (Datenschutz-Grundverordnung) (Text von Bedeutung für den EWR)*. 2016. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj (zitiert auf S. 16, 21, 27, 34).

[GGH+19] C. Giebler, C. Gröger, E. Hoos, H. Schwarz, B. Mitschang. „Leveraging the Data Lake: Current State and Challenges“. In: *Big Data Analytics and Knowledge Discovery: 21st International Conference, DaWaK 2019, Linz, Austria, August 26–29, 2019, Proceedings*. Berlin, Heidelberg: Springer-Verlag, Aug. 2019, S. 179–188. ISBN: 9783030275198. DOI: 10.1007/978-3-030-27520-4_13. URL: https://doi.org/10.1007/978-3-030-27520-4_13 (besucht am 05. 10. 2023) (zitiert auf S. 15).

[GGH+20] C. Giebler, C. Gröger, E. Hoos, H. Schwarz, B. Mitschang. „A zone reference model for enterprise-grade data lake management“. In: *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*. IEEE. 2020, S. 57–66 (zitiert auf S. 15, 31).

[GKD+23] A. Goedegebuure, I. Kumara, S. Driessen, D. Di Nucci, G. Monsieur, W.-j. v. d. Heuvel, D. A. Tamburri. *Data Mesh: a Systematic Gray Literature Review*. arXiv:2304.01062 [cs]. Apr. 2023. URL: http://arxiv.org/abs/2304.01062 (besucht am 02. 05. 2023) (zitiert auf S. 33, 44, 84).

[HA03] Q. He, A. Antón. „A Framework for Modeling Privacy Requirements in Role Engineering“. In: *Proceedings of the 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03)* (Juni 2003) (zitiert auf S. 41).

[HKS21] S. Hagedorn, S. Kläbe, K.-U. Sattler. „Putting pandas in a box“. In: *Conference on Innovative Data Systems Research (CIDR);(Online)*. 2021, S. 15 (zitiert auf S. 64–66).

[HLK+16] J. He, B. Liu, D. Kong, X. Bao, N. Wang, H. Jin, G. Kesidis. „PUPPIES: Transformation-Supported Personalized Privacy Preserving Partial Image Sharing“. In: *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. ISSN: 2158-3927. Juni 2016, S. 359–370. DOI: 10.1109/DSN.2016.40. URL: https://ieeexplore.ieee.org/abstract/document/7579755 (besucht am 01. 10. 2023) (zitiert auf S. 43).

[HWW23] E. Hechler, M. Weihrauch, Y. C. Wu. „Data Governance in the Context of Data Fabric and Data Mesh“. en. In: *Data Fabric and Data Mesh Approaches with AI: A Guide to AI-based Data Cataloging, Governance, Integration, Orchestration, and Consumption*. Hrsg. von E. Hechler, M. Weihrauch, Y. C. Wu. Berkeley, CA: Apress, 2023, S. 333–352. ISBN: 978-1-4842-9253-2. DOI: 10.1007/978-1-4842-9253-2_15. URL: https://doi.org/10.1007/978-1-4842-9253-2_15 (besucht am 21. 08. 2023) (zitiert auf S. 16, 21, 41).

[Inm16] B. Inmon. *Data Lake Architecture: Designing the Data Lake and Avoiding the Garbage Dump*. 1st. Denville, NJ, USA: Technics Publications, LLC, März 2016. ISBN: 9781634621175 (zitiert auf S. 15).

[JPR21] D. Joshi, S. Pratik, M. Rao. „Data Governance in Data Mesh Infrastructures: The Saxo Bank Case Study“. Dez. 2021. URL: https://aisel.aisnet.org/iceb2021/52 (zitiert auf S. 44, 84).

[Kai30] H. Kaiser. „Die Krankenhausapotheke“. de. In: Hrsg. von H. Braun, K. W. Clauberg, F. Goldmann, H. Kaiser, G. Kricker, H. Landsberg, I. Linde, H. R. Schinz, H. Schridde. Handbücherei für das Gesamte Krankenhauswesen. Berlin, Heidelberg: Springer, 1930, S. 150–166. ISBN: 9783642995873. DOI: 10.1007/978-3-642-99587-3_5. URL: https://doi.org/10.1007/978-3-642-99587-3_5 (besucht am 22. 09. 2023) (zitiert auf S. 24).

[KH21] S. Kläbe, S. Hagedorn. *Applying Machine Learning Models to Scalable DataFrames with Grizzly*. en. Accepted: 2021-03-16T07:57:09Z ISSN: 1617-5468. 2021. DOI: 10.18420/btw2021-10. URL: http://dl.gi.de/handle/20.500.12116/35793 (besucht am 05. 04. 2023) (zitiert auf S. 65).

[KHJ18] C. Koch, G. K. Hansen, K. Jacobsen. „Missed opportunities: two case studies of digitalization of FM in hospitals“. In: *Facilities* 37.7/8 (Jan. 2018), S. 381–394. ISSN: 0263-2772. DOI: 10.1108/F-01-2018-0014. URL: https://doi.org/10.1108/F-01-2018-0014 (besucht am 26. 09. 2023) (zitiert auf S. 21).

[KS21] Y. Kumar, R. Singla. „Federated Learning Systems for Healthcare: Perspective and Recent Progress“. en. In: Hrsg. von M. H. u. Rehman, M. M. Gaber. Studies in Computational Intelligence. Cham: Springer International Publishing, 2021, S. 141–156. ISBN: 9783030706043. DOI: 10.1007/978-3-030-70604-3_6. URL: https://doi.org/10.1007/978-3-030-70604-3_6 (besucht am 06. 10. 2023) (zitiert auf S. 36).

[LPS+22] B. Lei, G. Pires, J. Shao, K. Chatterjee, S. Jain, V. Sydorenko. *Data Mesh — A Data Movement and Processing Platform @ Netflix*. en. 2022. URL: https://netflixtechblog.com/data-mesh-a-data-movement-and-processing-platform-netflix-1288bcab2873 (besucht am 10. 10. 2023) (zitiert auf S. 45, 84).

[MAP17] H. Mehrpouyan, I. M. Azpiazu, M. S. Pera. „Measuring Personality for Automatic Elicitation of Privacy Preferences“. In: *2017 IEEE Symposium on Privacy-Aware Computing (PAC)*. Aug. 2017, S. 84–95. DOI: 10.1109/PAC.2017.15. URL: https://ieeexplore.ieee.org/abstract/document/8166618 (besucht am 09. 10. 2023) (zitiert auf S. 40).

[McS09] F. D. McSherry. „Privacy integrated queries: an extensible platform for privacy-preserving data analysis“. In: *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. SIGMOD '09. New York, NY, USA: Association for Computing Machinery, Juni 2009, S. 19–30. ISBN: 9781605585512. DOI: 10.1145/1559845.1559850. URL: https://dl.acm.org/doi/10.1145/1559845.1559850 (besucht am 01. 10. 2023) (zitiert auf S. 42).

[MCS21] I. Machado, C. Costa, M. Y. Santos. „Data-Driven Information Systems: The Data Mesh Paradigm Shift“. In: Aug. 2021. URL: https://aisel.aisnet.org/isd2014/proceedings2021/currenttopics/9 (zitiert auf S. 44, 83, 84).

[MCS22] I. A. Machado, C. Costa, M. Y. Santos. „Data mesh: concepts and principles of a paradigm shift in data architectures“. In: *Procedia Computer Science* 196 (2022), S. 263–271 (zitiert auf S. 15, 32, 44, 83, 84).

[MIKG13] H. Mouratidis, S. Islam, C. Kalloniatis, S. Gritzalis. „A framework to support selection of cloud providers based on security and privacy requirements“. In: *Journal of Systems and Software* 86.9 (Sep. 2013), S. 2276–2293. ISSN: 0164-1212. DOI: 10.1016/j.jss.2013.03.011. URL: https://www.sciencedirect.com/science/article/pii/S0164121213000575 (besucht am 01. 10. 2023) (zitiert auf S. 41).

[MKGV07] A. Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam. „L-diversity: Privacy beyond k-anonymity“. In: *ACM Transactions on Knowledge Discovery from Data* 1.1 (März 2007), 3–es. ISSN: 1556-4681. DOI: 10.1145/1217299.1217302. URL: https://dl.acm.org/doi/10.1145/1217299.1217302 (besucht am 05. 04. 2023) (zitiert auf S. 37).

[MMZ08] S. Miyazaki, N. Mead, J. Zhan. „Computer-Aided Privacy Requirements Elicitation Technique“. In: *2008 IEEE Asia-Pacific Services Computing Conference*. Dez. 2008, S. 367–372. DOI: 10.1109/APSCC.2008.263. URL: https://ieeexplore.ieee.org/abstract/document/4780702 (besucht am 01. 10. 2023) (zitiert auf S. 39, 47).

[MNW+18] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan et al. „Ray: A distributed framework for emerging {AI} applications“. In: *13th USENIX symposium on operating systems design and implementation (OSDI 18)*. 2018, S. 561–577 (zitiert auf S. 67).

[MRF19] P. Murmann, D. Reinhardt, S. Fischer-Hübner. „To Be, or Not to Be Notified“. en. In: *ICT Systems Security and Privacy Protection*. Hrsg. von G. Dhillon, F. Karlsson, K. Hedström, A. Zúquete. IFIP Advances in Information and Communication Technology. Cham: Springer International Publishing, 2019, S. 209–222. ISBN: 9783030223120. DOI: 10.1007/978-3-030-22312-0_15 (zitiert auf S. 39).

[Nee19] A. Neely. *Data Valuation – What is Your Data Worth and How do You Value it?* en-US. Sep. 2019. URL: https://opendatascience.com/data-valuation-what-is-your-data-worth-and-how-do-you-value-it/ (besucht am 28. 08. 2023) (zitiert auf S. 15).

[Pan23a] Pandas. *Pandas Documentation*. 2023. URL: https://pandas.pydata.org/docs/dev/index.html (zitiert auf S. 64).

[Pan23b] Pandas. *What's new in 2.0.0 (April 3, 2023) — pandas 2.0.0 documentation*. 2023. URL: https://pandas.pydata.org/docs/whatsnew/v2.0.0.html (besucht am 19. 04. 2023) (zitiert auf S. 64, 65).

[Par23] D. Parmar. *Pandas 2.0 is Here (Coming Soon)*. en. März 2023. URL: https://medium.com/@darshilp/pandas-2-0-is-here-427b026ab913 (besucht am 10. 07. 2023) (zitiert auf S. 65).

[PDTR18] S. M. Palanisamy, F. Dürr, M. A. Tariq, K. Rothermel. „Preserving Privacy and Quality of Service in Complex Event Processing through Event Reordering“. In: *Proceedings of the 12th ACM International Conference on Distributed and Event-based Systems*. DEBS '18. New York, NY, USA: Association for Computing Machinery, Juni

2018, S. 40–51. ISBN: 9781450357821. DOI: 10.1145/3210284.3210296. URL: https://dl.acm.org/doi/10.1145/3210284.3210296 (besucht am 01. 10. 2023) (zitiert auf S. 42, 43).

[PLW02] L. L. Pipino, Y. W. Lee, R. Y. Wang. „Data quality assessment“. In: *Communications of the ACM* 45.4 (Apr. 2002), S. 211–218. ISSN: 0001-0782. DOI: 10.1145/505248.506010. URL: https://dl.acm.org/doi/10.1145/505248.506010 (besucht am 26. 09. 2023) (zitiert auf S. 36).

[PMX+20] D. Petersohn, S. Macke, D. Xin, W. Ma, D. Lee, X. Mo, J. E. Gonzalez, J. M. Hellerstein, A. D. Joseph, A. Parameswaran. „Towards scalable dataframe systems“. 2020 (zitiert auf S. 64, 66, 67).

[Pol23] Polars. *pola-rs/polars*. original-date: 2020-05-13T19:45:33Z. Apr. 2023. URL: https://github.com/pola-rs/polars (besucht am 25. 04. 2023) (zitiert auf S. 64, 67).

[Pri22] PricewaterhouseCoopers. *Data mesh - The next-generation enterprise data platform?* en-US. 2022. URL: https://www.pwc.de/en/digitale-transformation/data-mesh-the-next-generation-enterprise-data-platform.html (besucht am 01. 09. 2023) (zitiert auf S. 44, 83, 84).

[RECC10] S. Rose, D. Engel, N. Cramer, W. Cowley. „Automatic Keyword Extraction from Individual Documents“. en. In: *Text Mining*. John Wiley & Sons, Ltd, 2010. Kap. 1, S. 1–20. ISBN: 9780470689646. DOI: 10.1002/9780470689646.ch1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470689646.ch1 (besucht am 06. 11. 2023) (zitiert auf S. 59).

[Rob23] Robert-Bosch-Krankenhaus. *Alle Abteilungen im Überblick*. 2023. URL: https://www.rbk.de/standorte/robert-bosch-krankenhaus/abteilungen.html (besucht am 22. 09. 2023) (zitiert auf S. 19).

[RZ19] F. Ravat, Y. Zhao. „Data lakes: Trends and perspectives“. In: *Database and Expert Systems Applications: 30th International Conference, DEXA 2019, Linz, Austria, August 26–29, 2019, Proceedings, Part I 30*. Springer. 2019, S. 304–313 (zitiert auf S. 15, 31).

[Sam01] P. Samarati. „Protecting respondents identities in microdata release“. In: *IEEE Transactions on Knowledge and Data Engineering* 13.6 (Nov. 2001), S. 1010–1027. ISSN: 1558-2191. DOI: 10.1109/69.971193. URL: https://ieeexplore.ieee.org/document/971193 (besucht am 12. 11. 2023) (zitiert auf S. 37).

[SBE+20] C. Stach, J. Bräcker, R. Eichler, C. Giebler, C. Gritti. „How to Provide High-Utility Time Series Data in a Privacy-Aware Manner: A VAULT to Manage Time Series Data“. In: *International Journal on Advances in Security* 13.3 & 4 (Dez. 2020). Hrsg. von H.-J. Hof, B. Gersbeck-Schierholz, S. 88–108. ISSN: 1942-2636. URL: http://www.iariajournals.org/security/ (zitiert auf S. 35).

[SBE+22a] C. Stach, J. Bracker, R. Eichler, C. Giebler, B. Mitschang. „Simplified Specification of Data Requirements for Demand-Actuated Big Data Refinement“. en. In: *Journal of Data Intelligence* 3.3 (Aug. 2022), S. 366–400. ISSN: 2577610X. DOI: 10.26421/JDI3.3-5. URL: https://www.rintonpress.com/xjdi3/xjdi3-3/366-400.pdf (besucht am 16. 03. 2023) (zitiert auf S. 42, 49, 56, 57, 64).

[SBE+22b] C. Stach, J. Bräcker, R. Eichler, C. Giebler, B. Mitschang. „Demand-Driven Data Provisioning in Data Lakes: BARENTS — A Tailorable Data Preparation Zone“. In: *The 23rd International Conference on Information Integration and Web Intelligence*. iiWAS2021. New York, NY, USA: Association for Computing Machinery, Dez. 2022, S. 187–198. ISBN: 9781450395564. DOI: 10.1145/3487664.3487784. URL: https://doi.org/10.1145/3487664.3487784 (besucht am 06. 10. 2023) (zitiert auf S. 32).

[SC19] P. Sinthong, M. J. Carey. *AFrame: Extending DataFrames for Large-Scale Modern Data Analysis (Extended Version)*. arXiv:1908.06719 [cs]. Aug. 2019. URL: http://arxiv.org/abs/1908.06719 (besucht am 25. 04. 2023) (zitiert auf S. 64, 66).

[SC21] P. Sinthong, M. J. Carey. *PolyFrame: A Retargetable Query-based Approach to Scaling DataFrames (Extended Version)*. arXiv:2010.05529 [cs]. Feb. 2021. URL: http://arxiv.org/abs/2010.05529 (besucht am 10. 07. 2023) (zitiert auf S. 64, 66).

[SDM+18] C. Stach, F. Dürr, K. Mindermann, S. M. Palanisamy, S. Wagner. „How a Pattern-based Privacy System Contributes to Improve Context Recognition“. In: *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. März 2018, S. 355–360. DOI: 10.1109/PERCOMW.2018.8480227 (zitiert auf S. 40, 42, 59).

[SES23] C. Stach, R. Eichler, S. Schmidt. „A Recommender Approach to Enable Effective and Efficient Self-Service Analytics in Data Lakes“. en. In: *Datenbank-Spektrum* 23.2 (Juli 2023), S. 123–132. ISSN: 1610-1995. DOI: 10.1007/s13222-023-00443-4. URL: https://doi.org/10.1007/s13222-023-00443-4 (besucht am 29. 08. 2023) (zitiert auf S. 32).

[SGB+22] C. Stach, C. Gritti, J. Bräcker, M. Behringer, B. Mitschang. „Protecting Sensitive Data in the Information Age: State of the Art and Future Prospects“. en. In: *Future Internet* 14.11 (Nov. 2022), S. 302. ISSN: 1999-5903. DOI: 10.3390/fi14110302. URL: https://www.mdpi.com/1999-5903/14/11/302 (besucht am 06. 10. 2023) (zitiert auf S. 34).

[SGM20] C. Stach, C. Gritti, B. Mitschang. „Bringing privacy control back to citizens: DISPEL — a distributed privacy management platform for the internet of things“. In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing*. SAC '20. New York, NY, USA: Association for Computing Machinery, März 2020, S. 1272–1279. ISBN: 978-1-4503-6866-7. DOI: 10.1145/3341105.3375754. URL: https://doi.org/10.1145/3341105.3375754 (besucht am 16. 05. 2023) (zitiert auf S. 42, 43).

[Sha18] B. Sharma. *Architecting Data Lakes, 2nd Edition [Book]*. en. 2018. URL: https://www.oreilly.com/library/view/architecting-data-lakes/9781492033004/ (besucht am 05. 10. 2023) (zitiert auf S. 15).

[SM19] C. Stach, B. Mitschang. „Elicitation of Privacy Requirements for the Internet of Things Using ACCESSORS“. en. In: Communications in Computer and Information Science (2019). Hrsg. von P. Mori, S. Furnell, O. Camp, S. 40–65. DOI: 10.1007/978-3-030-25109-3_3 (zitiert auf S. 41, 42).

[SS19] C. Stach, F. Steimle. „Recommender-based privacy requirements elicitation - EPICUREAN: an approach to simplify privacy settings in IoT applications with respect to the GDPR“. en. In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. Limassol Cyprus: ACM, Apr. 2019, S. 1500–1507. ISBN: 978-1-4503-5933-7. DOI: 10.1145/3297280.3297432. URL: https://dl.acm.org/doi/10.1145/3297280.3297432 (besucht am 19. 07. 2023) (zitiert auf S. 40, 41, 48, 58).

[SSM23] C. Stach, F. Steimle, B. Mitschang. „The Privacy Management Platform - An Enabler for Device Interoperability and Information Security in mHealth Applications“. In: Okt. 2023, S. 27–38. ISBN: 9789897582813. URL: https://www.scitepress.org/Link.aspx?doi=10.5220/0006537300270038 (besucht am 06. 10. 2023) (zitiert auf S. 35).

[SSS+19] D. Sarne, J. Schler, A. Singer, A. Sela, I. Bar Siman Tov. „Unsupervised Topic Extraction from Privacy Policies“. In: *Companion Proceedings of The 2019 World Wide Web Conference*. WWW '19. San Francisco, USA: Association for Computing Machinery, 2019, S. 563–568. ISBN: 9781450366755. DOI: 10.1145/3308560.3317585. URL: https://doi.org/10.1145/3308560.3317585 (zitiert auf S. 59).

[Sta15] C. Stach. „How to Deal with Third Party Apps in a Privacy System – The PMP Gatekeeper –“. In: *2015 16th IEEE International Conference on Mobile Data Management*. Bd. 1. ISSN: 2375-0324. Juni 2015, S. 167–172. DOI: 10.1109/MDM.2015.17. URL: https://ieeexplore.ieee.org/document/7264317 (besucht am 06. 10. 2023) (zitiert auf S. 35).

[Sta23a] C. Stach. „Data Is the New Oil–Sort of: A View on Why This Comparison Is Misleading and Its Implications for Modern Data Administration“. en. In: *Future Internet* 15.2 (Feb. 2023), S. 71. ISSN: 1999-5903. DOI: 10.3390/fi15020071. URL: https://www.mdpi.com/1999-5903/15/2/71 (besucht am 20. 09. 2023) (zitiert auf S. 15).

[Sta23b] Statistisches Bundesamt. *Ärztliches und nichtärztliches Personal in Krankenhäusern*. de. 2023. URL: https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Gesundheit/Krankenhaeuser/Tabellen/personal-krankenhaeuser-jahre.html (besucht am 22. 09. 2023) (zitiert auf S. 20).

[Sta23c] Statistisches Bundesamt. *Eckdaten der Krankenhauspatientinnen und -patienten*. de. 2023. URL: https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Gesundheit/Krankenhaeuser/Tabellen/entlassene-patienten-eckdaten.html (besucht am 22. 09. 2023) (zitiert auf S. 20).

[Ste11] M. Stephens. *Hospital Pharmacy*. en. Google-Books-ID: kdAMf8f8RPwC. Pharmaceutical Press, 2011. ISBN: 9780853699002 (zitiert auf S. 24).

[Tak20] X. L. Takuya Ueshin Hyukjin Kwon. *Interoperability between Koalas and Apache Spark*. en-US. Nov. 2020. URL: https://www.databricks.com/blog/2020/08/11/interoperability-between-koalas-and-apache-spark.html (besucht am 24. 04. 2023) (zitiert auf S. 67).

[Uni23a] Unidist. *unidist: Unified Distributed Execution*. 2023. URL: https://github.com/modin-project/unidist (besucht am 24. 10. 2023) (zitiert auf S. 67).

[Uni23b] Universitätsklinikum Heidelberg. *All Medical Departsments & Centers*. 2023. URL: https://www.heidelberg-university-hospital.com/diseases-treatments/all-departments (besucht am 22. 09. 2023) (zitiert auf S. 19).

[Vin21] R. Vink. *I wrote one of the fastest DataFrame libraries*. en. 2021. URL: https://www.pola.rs/posts/i-wrote-one-of-the-fastest-dataframe-libraries/ (besucht am 19. 04. 2023) (zitiert auf S. 67).

[Yıl23] S. Yıldırım. *4 Techniques for Scaling Pandas to Large Datasets*. en. Juli 2023. URL: https://towardsdatascience.com/4-techniques-for-scaling-pandas-to-large-datasets-acf8805d30eb (besucht am 06. 11. 2023) (zitiert auf S. 64).

Alle URLs wurden zuletzt am 12. 11. 2023 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift