Carl von Ossietzky
Universität
Oldenburg

Datenmanagement für dezentrale maritime Forschungs- und Entwicklungsdaten im Testfeldkontext

Von der Fakultät für Informatik, Wirtschafts- und Rechtswissenschaften der Carl von Ossietzky Universität Oldenburg zur Erlangung des Grades und Titels

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

angenommene Dissertation von

Herrn Christian Julius Möller,

geboren am 28. Januar 1996 in Oldenburg

Gutachter:
Prof. Dr.-Ing. Axel Hahn
Prof. Dr. Frank Köster
Prof. Dr. Christoph Schlueter Langdon

Tag der Disputation: 15. Mai 2023

Danksagung

An dieser Stelle möchte ich mich meine aufrichtige Dankbarkeit gegenüber all jenen zum Ausdruck bringen, die mich bei der Anfertigung dieser Dissertation unterstützt haben. Besonders möchte ich meinem Doktorvater Prof. Dr.-Ing. Axel Hahn für seine herausragende und vertrauensvolle Betreuung in sämtlichen Phasen der Arbeit danken. Insbesondere die unkomplizierte Art in der Kommunikation, unsere stets hilfreichen wissenschaftlichen Diskussionen und das Ermöglichen des Zugangs zu zahlreichen benötigten Ressourcen für die Implementierung und Evaluation der Arbeit haben meinen tiefsten Respekt gewonnen. Auch darüber hinaus haben sich mir durch Axels Unterstützung ganz neue Möglichkeiten und Netzwerke eröffnet. Ebenfalls gebührt mein Dank Prof. Dr. Frank Köster, der mir während des Forschungsprozesses wertvolle Einblicke in den Bereich der Data Space Architekturen geben konnte, und als weiterer Experte einen umfassenden Blickwinkel auf die Perspektive der Testfelder bot. Mein aufrichtiger Dank gilt auch Prof. Dr. Christoph Schlueter Langdon, dessen Perspektive aufgrund seiner Verbindungen zu Industriekreisen, die bereits die Data Space Konzepte anwenden, äußerst wertvoll war. Schließlich möchte ich Prof. Dr. Wolfram Wingerath und Dr. Marco Grawunder für ihre Beiträge als Teil der Prüfungskommission meiner Arbeit danken.

Weiterhin möchte ich meinen geschätzten Kolleginnen und Kollegen von der Universität Oldenburg und dem DLR meinen Dank aussprechen. Ihre allgemeine Unterstützung, das wunderbare Miteinander und die inspirierenden Gespräche während unserer Kaffeepausen haben meinen Weg bereichert. Ohne dabei jemanden auszuschließen, möchte ich insbesondere Dennis Jankowski, Klaas Dählmann, Arne Lamm und Janusz A. Piotrowski für ihre Hilfsbereitschaft im Rahmen meiner Promotion hervorheben. Den Studierenden, die durch Abschlussarbeiten, als wissenschaftliche Hilfskräfte oder als Teilnehmende in den Projektgruppen MIRAPLA und MANGO zu der Implementierung meiner Arbeit beitrugen möchte an dieser Stelle ebenfalls danken.

Abschließend gilt mein tiefster Dank meiner Familie und meinen Freunden, die stets an meiner Seite standen und mich auch in den anspruchsvollen Phasen ermutigt haben, weiterzumachen. Danke für eure bedingungslose Unterstützung, die wunderbaren Erlebnisse und eure Liebe!

Zusammenfassung

In den letzten Jahren wurden zahlreiche neue Ansätze für maritime Assistenzsysteme entwickelt, die immer datenintensivere Verfahren nutzen, etwa um Trajektorien zu prädizieren, Steuerbefehle zu berechnen oder Ankunftszeiten vorherzusagen. Zur Evaluation dieser Systeme sind aufgrund komplexer werdender Datenverarbeitungsprozesse immer aufwendigere Testverfahren erforderlich. Um den Gesamtaufwand einer Systementwicklung zu beschränken, ist es daher meist sinnvoll die Entwicklung eines Testfeldes von der eigentlichen Systementwicklung zu trennen und dieses als generisch nutzbaren Service bereitzustellen. Die gemeinsame Nutzung solcher Testfelder durch verschiedenste Stakeholder aus Forschung und Industrie stellt diese jedoch vor diverse Herausforderungen bezüglich des Datenmanagements: Im Prozess der Entwicklung, Validierung und Verifikation (V+V) oder der Demonstration und Zertifizierung maritimer Assistenzsysteme spielen Daten eine essenzielle Rolle und werden zwischen verschiedensten (Teil-)Systemen und Nutzern ausgetauscht und ausgewertet. Gleichzeitig wollen involvierte Parteien ihre Daten schützen und kontrollieren und dadurch etwa mögliche Wettbewerbsvorteile bewahren oder rechtliche Rahmenbedingungen einhalten. Die so entstehende Dezentralität in der Datenverwaltung erschwert die Kollaboration in einem serviceorientierten Testfeld, insbesondere bei der gemeinsamen Verarbeitung und Analyse von Testfelddaten. Diese Arbeit beschäftigt sich daher mit der Fragestellung, wie in diesem Kontext dezentral organisierte maritime Daten mit Hilfe eines Datenmanagementsystems für einen datengetriebenen Forschungs- und Entwicklungsprozess bereitgestellt werden können.

Hierzu wird eine Architektur für maritime, serviceorientierte Testfelder entwickelt, die die Besonderheiten eines dezentral organisierten Datenökosystems und die speziellen Anforderungen bzgl. des Datenmanagements im Testfeld abbilden kann. Insbesondere die Bereitstellung einer historischen Datengrundlage, das Organisieren von Daten im Rahmen von komplexen Datenverarbeitungsketten im Zusammenhang der V+V und das Verwalten von Daten für Demonstrationen und Zertifizierungen im Testfeld werden dabei unterstützt. Die entwickelte Architektur wird in einem etablierten maritimen Testfeld (eMIR) umgesetzt und schließlich bezüglich Anwendbarkeit und Performanz evaluiert.

Abstract

Recently, numerous new approaches have been developed for maritime assistance systems that use increasingly data-intensive processes. For example, to predict trajectories, calculate control commands or predict port arrival times. Due to more and more complex data processing methods, elaborate test procedures are required to evaluate these systems. To limit the overall effort of system development, it can be helpful to separate the development of a test environment or a test bed from the actual system development and to provide it as a generic service. However, the joint usage of test beds by various stakeholders from research and industry presents them with multiple challenges in terms of data management: In the process of development, validation and verification (V+V) or the demonstration and certification of maritime assistance systems, data plays an essential role and is exchanged between and processed in various (sub)systems and users. At the same time, involved parties want to protect and control their data and thus maintain possible competitive advantages or comply with legal framework conditions. The resulting decentralisation in data management complicates collaboration in a service-oriented test bed, especially in the joint processing and analysis of test bed data. This thesis deals with the question of how decentrally organised maritime data can be provided for a data-driven research and development process in a test bed with the help of a data management system.

For this purpose, an architecture for maritime service-oriented test beds is developed that can meet the special requirements of decentrally organised data ecosystems and data management tasks in the test bed. In particular, the provision of a historical data basis, the organisation of data within complex processing chains in the context of V+V, and the management of data for demonstration and certification in the test bed are supported. The developed architecture is implemented in a well-established maritime test bed (eMIR) and finally evaluated with regard to applicability and performance.

Inhaltsverzeichnis

Verzeichnis der Abkürzungen und Akronyme

A1-A18	Anforderungen 1 bis 18 (vgl. Abschnitt 2.6)
ACCSEAS	Accessibility for Shipping, Efficiency Advantages and Sustainability
AIS	Automatic Identification System
ANSI/UL	American National Standards Institute / Underwriters Laboratories
API	Application Programming Interface
BDSG	Bundesdatenschutzgesetz
CKAN	Comprehensive Knowledge Archive Network
COG	Course over Ground
CPA	Closest Point of Approach
CPAd	Distance at Closest Point of Approach
CPSoS	Cyber-physical System-of-Systems
CPU	Central Processing Unit
CSV	Comma-separated Values
DB	Datenbank
DIN	Deutsches Institut für Normung
DMS	Datenmanagementsystem
DSGVO	Datenschutz-Grundverordnung
DSMR	Data Space Metadata Repository
DSSP	Data Space Support Platform
DaaS	Data-as-a-Service
eMIR	e-Maritime Integrated Reference Platform
ECDSA	Elliptic Curve Digital Signature Algorithm
ESAAM	Erweitertes SAAM-Verfahren
FAIR	Findable, Accessible, Interoperable, Re-usable
FDMS	Forschungsdatenmanagementsystem
FEDMS	Forschungs- und Entwicklungsdatenmanagementsystem
FS	Federation Services (Komponente von GAIA-X)
GIS	Geoinformationssystem
GPRS	General Packet Radio Service
HAGGIS	Hybrid Architecture for Granularly, Generic and Interoperable Simulations
HTTPS	Hypertext Transfer Protocol Secure
IALA	International Association of Marine Aids to Navigation and Lighthouse Authorities
IDS	International Data Space
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IHO	International Hydrographic Organization
IMO	International Maritime Organization
ISO	International Organization for Standardization
ITU	International Telecommunication Union
IdP	Identity Provider
IoT	Internet-of-Things
JDBC	Java Database Connectivity
JRE	Java Runtime Environment
JSON	JavaScript Object Notation
JWT	JSON Web Token
KDD	Knowledge Discovery in Databases
LMM	Last-Minute Maneuver
MCP	Maritime Connectivity Platform
ML	Machine Learning
MMSI	Maritime Mobile Service Identity
MONALISA	Motorways and Electronic Navigation by Intelligence at Sea

MTCAS	Maritime Traffic Collision Avoidance System
MTSL	MULTIC Timing Specification Language
MULTIC	Design Paradigms for Multi-Layer Time Coherency in ADAS and Automated Driving
NMEA	National Marine Electronics Association
NoSQL	non-SQL
OAuth	Open Authorization
OCI	Open Container Initiative
ODD	Operational Design Domain
OIDC	OpenID Connect
OPM	Open Provenance Model
OWL	Web Ontology Language
PKI	Public-Key Infrastructure
PROV-DM	Provenance Data Model
PortCDM	Port Collaborative Decision Making
RADAR	Radio Detection And Ranging
RAM	Random-Access Memory
RDF	Resource Description Framework
REST	Representational State Transfer
ROT	Rate of Turn
S01-S22	Evaluationsszenarien 1 bis 22 (vgl. Kapitel 6)
SAAM	Software Architecture Analysis Method
SAML	Security Assertion Markup Language
SOA	Service-oriented Architecture
SOG	Speed over Ground
SPARQL	SPARQL Protocol and RDF Query Language (rekursives Akronym)
SSAP	Smart Ship Application Platform
STM	Sea Traffic Management
SeaSWIM	Sea System Wide Information Management
SoC	Separation of Concerns
SSD	Solid-State-Drive
SuT	System under Test
TDS	Testfeld Data Space
TKG	Telekommunikationsgesetz
TLS	Transport Layer Security
UC1-UC2	Use-Cases 1 – 2 (vgl. Abschnitt 1.2)
UI	User Interface
V+V	Verifikation und Validierung
VDMS	Vehicle Data Management System
VHF	Very High Frequency
VM	Virtuelle Maschine
WEKA	Waikato Environment for Knowledge Analysis
WGS84	World Geodetic System 1984
WSL	Windows-Subsystem für Linux
XML	Extensible Markup Language

Abbildungsverzeichnis

Tabellen- und Listingverzeichnis

Tabellen

Listings

Anmerkung:

Aus Gründen der besseren Lesbarkeit wird in dieser Arbeit das generische Maskulinum bei personenbezogenen Substantiven und Pronomen verwendet. Soweit nicht anders angezeigt soll dies im Sinne der sprachlichen Vereinfachung als geschlechtsneutral zu verstehen sein.

1 Einleitung

> *"People now do not actually look through telescopes. Instead, they are 'looking' through large-scale, complex instruments which relay data to datacenters, and only then do they look at the information on their computers."*
>
> – Jim Grey in (A. J. Hey, Tansley, und Tolle 2009, 19)

Das automatisierte Aufzeichnen und Speichern von riesigen Datenmengen ist sowohl in der Forschung als auch in der Industrie keine neue Entwicklung mehr. Selbst einzelne Personen produzieren mittlerweile beträchtliche Mengen an Daten. Die Verwaltung solcher „Big Data" Datensätze ist längst nicht mehr trivial und wird als eine große Herausforderung für Forschung und Industrie aufgefasst (McAfee und Brynjolfsson 2012). Zusätzlich ergibt sich in der zunehmend vernetzten Welt oft die Anforderung Daten aus verschiedenen Quellen zusammenzuführen, um Analysen in komplexen Daten-Workflows ausführen zu können. Gerade in der Forschung und Entwicklung neuer datengestützter Systeme sind mit der wachsenden Menge, Komplexität und Geschwindigkeit der Daten neue Anforderungen an Verarbeitung und Verwaltung der Daten entstanden. Erhebliche Mengen an Forschungs- und Entwicklungsdaten erfordern Ansätze, um diese zu organisieren. Zudem sind im Bereich der künstlichen Intelligenz zahlreiche Verfahren aufgekommen, die erfolgreich auf großen Datensätzen arbeiten können, dafür aber neue Arten von (Vor-)Verarbeitungsstrukturen voraussetzen (Roh, Heo, und Whang 2021).

Vor allem in Unternehmen werden Daten immer mehr als wertvolle und strategisch relevante Ressource angesehen, auf der in den letzten Jahren neue Geschäftsmodelle aufgebaut wurden (Monino 2021). Ausgehend davon entsteht seit einigen Jahren ein digitaler Markt für den Handel mit Daten unter der Verwendung verschiedenster Abo- und Preismodelle für den Bezug von Daten über diverse Plattformen (F. Liang u. a. 2018). Diese neuartigen Ökosysteme umfassen ebenfalls weitere Dienste zur Aufbereitung, Vorverarbeitung und Analyse der Daten (F. Liang u. a. 2018). Zudem zeigen jüngste Investitionen allein in Deutschland von über hundert Millionen Euro in die Entwicklung von sicheren und vertrauenswürdigen Ökosystemen und Dateninfrastrukturen, wie das europäische Projekt GAIA-X, das große Interesse an neuen Ansätzen für die Datenverwaltung (Rusche 2022, 17).

Im folgenden Abschnitt sollen diese Trends in den Kontext des dezentralen Datenmanagements in maritimen Testfeldern eingeordnet werden. Dabei wird die Motivation dieser Arbeit dargelegt, eine Problemstellung herausgearbeitet und auf die spezifischen Ziele dieser Arbeit eingegangen.

1.1 Daten in maritimen Testfeldern

Auch in der maritimen Wirtschaft ist der Bereich Big Data von bedeutendem Forschungsinteresse. So werden beispielsweise große Mengen an Verkehrsdaten genutzt (Munim u. a. 2020), um maritime Verkehrswege besser zu verstehen (Lamm und Hahn 2019) oder Risiken zu charakterisieren (Silveira, Teixeira, und Soares 2013). In einer Vielzahl von weiteren Publikationen werden zusätzlich Daten aus den Bereichen Logistik, Wetter, Hydrographie oder weiteren Datenquellen für die Forschung genutzt. Dies zeigt auch die Anzahl der wissenschaftlichen Veröffentlichungen pro Jahr aus dem Bereich „Maritime Data“, die im Vergleich zum Jahr 2000 um über 400% angestiegen sind (siehe Abbildung 1).

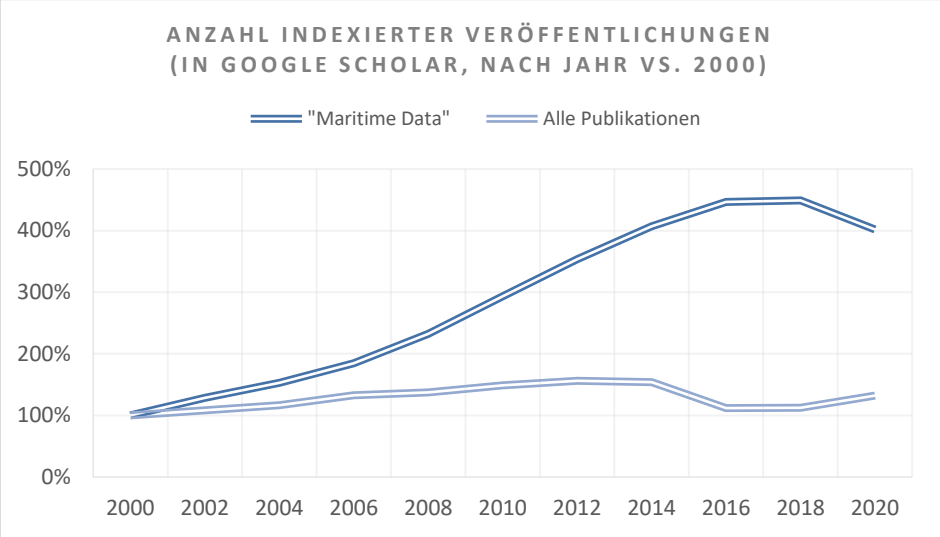

Abbildung 1: Anzahl der jährlichen Anzahl indexierter Veröffentlichungen in Google Scholar (2000-2020 im Vergleich zu 2000): Suchbegriff „Maritime Data“ versus Alle Publikationen.

Insbesondere in maritimen Testfeldern kommt eine große Menge von Datenquellen zusammen, die auf vielfältige Art und Weise für die Erforschung der maritimen Umgebung und der Entwicklung und Demonstration neuer (Assistenz-)Systeme genutzt werden. Testfelder werden dabei nicht nur von Forschenden einer einzelnen Organisation genutzt, sondern oft im Zusammenspiel mehrerer Partner aus Industrie und Wissenschaft.

Eine grundlegende Aufgabe eines Testfeldes aus Datenperspektive ist dabei das Schaffen einer Datengrundlage, die verwendet werden kann, um Umwelt- oder Systemmodelle zu entwickeln. Heutzutage dient eine solche Datensammlung ebenfalls oft als Ausgangspunkt zu Erstellung von Trainingsdatensätzen für Machine Learning-Anwendungen. Gerade die Komplexität neu entwickelter Systeme nimmt insbesondere durch Autonomisierung immer weiter zu und erfordert ein tiefgreifendes Verständnis der Umwelt, bevor neuartige Konzepte entwickelt werden können

(Abbaspour Asadollah, Inam, und Hansson 2015). Daten sind somit auch im Testfeld eine wertvolle Ressource, die zwischen verschiedenen Akteuren kontinuierlich ausgetauscht wird.

Weiterhin werden maritime Testfelder genutzt, um Schiffführungsassistenzsysteme zu testen (vgl. Steidel und Hahn 2019). Im Prozess der Validierung und Verifikation solcher Systeme, spielen Daten ebenfalls eine essenzielle Rolle: Bereits vor der Entwicklung eines Assistenzsystems kann ein Testfeld genutzt werden, um Daten über seine Umgebung zu sammeln und so die Operational Design Domain (ODD) zu definieren. Die ODD beschreibt den Parametermeterbereich für den ein autonomes System entwickelt wurde, um ohne Zwischenfälle seine spezifizierten Funktionen zu erfüllen (Czarnecki 2018).

Beim Testen Cyber-physischer Systeme entstehen zudem große Datenmengen, die für die Auswertung eines Tests analysiert oder überwacht werden müssen (Zhou u. a. 2018). Die Komplexität der Systeme führt häufig dazu, dass ihre Entwicklung durch das Aufteilen in Sub-Systeme auf mehrere Parteien aufgespalten und somit parallelisiert wird. Im Rahmen eines verteilten *Model-based Systems Engineering* Prozesses entstehen dann oft über Schnittstellen zusammengesetzte Systeme mehrerer Vertragspartner, bei denen während eines Integrationstests übergreifende Datenverarbeitungsketten überwacht werden müssen (D'Ambrosio und Soremekun 2017). Damit stellt das Datenmanagement in Testfeldern eine besondere Herausforderung dar: Hier werden in einem Datenökosystem nicht nur eine Vielzahl von verschiedenen Daten erhoben, sondern auch durch eine Vielzahl verschiedener Teilnehmer mit unterschiedlichen Interessen verarbeitet und weitergereicht. Erschwerend kommen hier die wirtschaftlichen Aspekte des Datenteilens hinzu, da das Preisgeben von intern erhobenen Daten zu einem Wettbewerbsnachteil führen kann.

Schließlich werden maritime Testfelder eingesetzt, um fertiggestellte Systeme zu demonstrieren oder zu zertifizieren (vgl. auch Abschnitt 2.2.3). Auch hier spielt das Austauschen von Daten in einem Ökosystem mit mehreren Teilnehmern eine essenzielle Rolle: Bei Demonstrationen ist es relevant, dass aus einer Vielzahl von Datenquellen und Datenströmen verschiedener (Sub-)Systeme genau jene ausgewählt und unabhängig präsentiert werden können, die die Anforderungen eines bestimmten Anwendungsfalls erfüllen können. Bei der Zertifizierung hingegen müssen der Zertifizierer und (mehrere) Entwickler Daten über den Betrieb entsprechender Systeme miteinander austauschen, damit eine vollständige Analyse und Bewertung des Systemverhaltens erfolgen können.

Globale Trends des kollaborativen Datenhandels lassen sich also ebenfalls auch in den lokalen Dimensionen eines maritimen Testfeldes beobachten. Auf der anderen Seite ist der Aufbau und Betrieb eines Testfeldes (insbesondere für Cyber-physische Systeme) sehr kostenintensiv (Parveen und Nandan 2022) sodass Testfeldbetreiber zudem ein Interesse daran haben eine

Testfeldnutzung möglichst zu parallelisieren und vielen Teilnehmern gleichzeitig den Zugang zu einem entsprechenden Ökosystem zu ermöglichen. Soll ein maritimes Testfeld eingesetzt werden, kommen also nun mehrere Entwickler, Betreiber des Testfeldes und weitere Akteure (wie etwa überwachende Behörden oder Sensordatenprovider) zusammen. Zwischen diesen Parteien werden verschiedene Daten ausgetauscht, die jeweils nicht für alle Teilnehmer sichtbar sein dürfen, um wertvolle Daten wie Szenariendatenbanken, aufbereitete historische Beobachtungsdaten oder Live-Informationen zu Schiffsparametern zu schützen. Trotzdem müssen bestimmte Daten zur Laufzeit des Experiments ausgetauscht werden, damit die zu testenden Systeme gemeinsam funktionsfähig sind und eine sichere Durchführung und Auswertung des Tests möglich ist.

Diese Trends in der Entwicklung und dem Test von modernen maritimen Assistenzsystemen führen dazu, dass Testfelder immer generischer aufgebaut werden und als Service für Entwickler oder Wissenschaftler bereitgestellt werden (Hossain u. a. 2017), (Brinkmann, Hahn, und Hjøllo 2017). So können Systementwicklung und Tests im Rahmen des Simultaneous Engineerings (Schäuffele und Zurawka 2016, 29) einfacher parallelisiert, und Testfelder vielfältiger und effizienter eingesetzt werden. Durch die Verwendung von standardisierten Schnittstellen können zu testende Systeme zudem einfacher integriert werden und mit Datenquellen eines Testfeldes verbunden werden.

1.2 Problemstellung

In Entwicklungsprozessen maritimer Assistenzsysteme müssen sich Forscher und Entwickler immer stärker auf datenbasierte Verfahren stützen. Testfelder mit riesigen Datengrundlagen über die Operational Design Domain solcher Systeme und hochgenauen Sensorsystemen können beim Testen besonders hilfreich oder sogar ausschlaggebend sein. Allerdings stellen das Verwalten und Bereitstellen von Daten verschiedenster Datentypen, -modelle und -qualitätsausprägungen in Testfeldern eine große Herausforderung dar. Dazu kommt, dass solche Datenquellen oft dynamisch für bestimmte Tests integriert werden. Daraufhin müssen benötigte Daten häufig einer aus einer Vielzahl von unabhängigen Quellen selektiert, vorverarbeitet, transformiert und analysiert werden. Gleichzeitig muss aber darauf geachtet werden, dass die verschiedenen Teilnehmer die Hoheit über ihre Daten nicht verlieren, und, dass für eine spätere Analyse nachvollziehbar ist, welche Prozesse zum Datenaustausch überhaupt stattgefunden haben. Um diese Herausforderungen anzugehen, wird ein Konzept zum Forschungs- und Entwicklungsdatenmanagement benötigt, sodass die verfügbaren Datensätze einheitlich verwaltet und Zugriffe auf verfügbare Datenquellen koordiniert werden können. Der wirtschaftliche Forschungs- und Entwicklungskontext, in dem maritime Testfelder häufig genutzt werden

resultiert in vielen Fällen in Szenarien mit zahlreichen unabhängigen Stakeholdern, die jeweils einzelne Teilsysteme bereitstellen, um ein komplexes System-of-Systems zu erschaffen. Diese verteilten und dezentralisierten Konstellationen erschweren das Management von wissenschaftlichen Daten jedoch enorm, sodass klassische Systeme nicht mehr eingesetzt werden können. Hinzu kommt, dass häufig nicht nur Datengrundlagen aus dem Testfeld selbst genutzt werden, sondern auch externe (kommerzielle) Datenprovider hinzugezogen werden, die Spezialdaten bereitstellen können (z.B. bestimmte Strömungsmodelle oder spezifische Wetterdaten).

Existierende Architekturen für maritime Testfelder ermöglichen nach aktuellen Forschungsstand zwar das dynamische Integrieren von zu testenden Systemen (vgl. etwa N. Rüssmeier, Lamm, und Hahn 2019), das Management der anfallenden und benötigten Daten oblag dabei bisher allerdings ausschließlich den Nutzern. Aus der Problemstellung des maritimen Forschungs- und Entwicklungsdatenmanagements in Testfeldern ergeben sich daher die folgenden Teilprobleme:

Dezentralität. Maritime Forschungs- und Entwicklungsdaten aus unabhängig verwalteten Quellen können nicht ohne Weiteres zentralisiert werden und für beliebige Interessenten zur Verfügung gestellt werden. Im Rahmen dieser Arbeit ist hiermit organisatorische Dezentralität gemeint. D.h. es existiert keine zentrale Organisation, die die volle Kontrolle über die vorhandenen Datenquellen ausübt. Vielmehr stellen einzelne Stakeholder des Testfeldes den Zugriff auf ihre selbst organisierten Datenquellen als Teil eines Datenökosystems bereit. Daraus resultiert auf technischer Ebene das Problem der dezentralen Datenquellen, welche zunächst über keine gemeinsame Zugriffsschnittstelle verfügen (Franklin, Halevy, und Maier 2005). So muss also eine Kommunikation über Datensätze mit den einzelnen Systemen stattfinden, um Zugriffe auf diese sinnvoll zuzuordnen.

Integrität und Souveränität. Weitere Herausforderungen ergeben sich außerdem in Bezug auf die Integrität der verfügbaren Daten (Wallis u. a. 2007): Die Dokumentation der Verarbeitungsschritte, Fehlererkennung oder Informationen zur Aufzeichnung der Daten sind hier relevant, damit zum einen die Datenverarbeitungsprozesse (welche ebenfalls nicht zentral organisiert sind) von den Stakeholdern inhaltlich nachvollzogen werden und Ergebnisse ausgewertet werden können. Dem gegenüber stehen jedoch oft ebenso relevante Anforderungen bezüglich des Datenschutzes von personen- oder unternehmensbezogenen Informationen (Sun u. a. 2014) und der Wahrung der Souveränität über die eigenen Daten (Zrenner u. a. 2019).

Bereitstellung im Testfeld. Es gilt schließlich die Daten so bereitzustellen, dass Forscher und Entwickler aus dem industriellen Kontext sinnvoll und effizient mit diesen arbeiten können. Für das maritime Testfeld bedeutet das insbesondere, dass Daten aus Tests so bereitgestellt werden müssen, dass eine Nutzung und Auswertung für einen kollaborativen Systementwicklungsprozess

möglich ist. Weiterhin besteht in der fortschreitenden Entwicklung von Cloud Services eine großes Interesse an serviceorientierten Architekturen (Duan u. a. 2015), sodass Nutzer eines Testfeldes zu testende Systeme flexibel integrieren können und dabei die benötigten Teilfunktionen des Datenmanagements für ihren Anwendungsfall kombinieren können.

Besonderheiten maritimer Testfelder. Die o.g. Kernherausforderungen sind zunächst nicht vollständig spezifisch für den maritimen Kontext. Auch in Testfeldern aus anderen Domänen (z.B. Automobildomäne) können sich diese Probleme ergeben. Bei der Integration der eigentlichen Datenquellen in ein gemeinsames System ergeben sich jedoch spezielle Herausforderungen der maritimen Domäne. Die Art der verwendeten Daten unterscheidet sich im maritimen Bereich klar von anderen Domänen. Die offene Übertragung von Positions- und Schiffsinformationen über das Automatic Identification System (AIS) spielt in Kombination mit RADAR-Daten eine wichtige Rolle (siehe Abschnitt 2.5). Aufgrund großer Unterschiede in der Ausstattung von Schiffen und Landstationen ergibt sich aber eine große Varianz in der Qualität solcher Sensordaten (vgl. Harati-Mokhtari u. a. 2007). Entsprechende Metriken und Vorverarbeitungsschritte müssen speziell auf diese Besonderheiten angepasst werden. Zudem verfügen Akteure im maritimen Umfeld (Schiffe, Bojen, Sensorstationen) aufgrund ihrer Distanz zum Festland nicht immer über Breitband-Datenverbindungen (Jo und Shim 2019) und auch bei der Entwicklung fortschrittlicher maritimer Assistenzsysteme sind verschiedene Rahmenbedingungen und Performanz-Kriterien vorgegeben, die in Anforderungen an Datenverarbeitungsketten berücksichtigt werden müssen.

Aus diesen Gründen ist es sinnvoll ein Lösungskonzept zu entwickeln, welches auf maritime Testfelder zugeschnitten ist und welches neben den allgemeinen Problemstellungen auch die speziellen Anforderungen einer maritimen Umgebung berücksichtigen kann. Zur weiteren Veranschaulichung sollen zwei Anwendungsfälle dienen, anhand derer die Ziele und relevanten Grundlagen dieser Arbeit im folgenden Abschnitt und dem nächsten Kapitel verdeutlicht werden.

Anwendungsfall 1 (UC1). Zur Untersuchung von Kollisionsrisiken sollen historische Verkehrsdaten in der deutschen Bucht ausgewertet werden. Dazu sollen AIS-Daten mit RADAR-Daten aus einem Testfeld fusioniert werden, um die Positionsgenauigkeit zu erhöhen und die Datenqualität messbar zu verbessern. Zudem werden Wetterdaten von einem Drittanbieter mit einbezogen, um zu untersuchen ob raue Wetterbedingungen einen Einfluss auf das Kollisionsrisiko haben. Aus den Daten wird ein Trainingsdatensatz erstellt, der ein ML-Modell trainieren soll. Außerdem soll nachvollziehbar dokumentiert werden, wie die Daten für das Modell verarbeitet wurden.

Anwendungsfall 2 (UC2). Ein Testfeld wird gleichzeitig von zwei Firmen (A und B) genutzt, um prototypische Systeme zu testen. Beide Firmen arbeiten an einem ähnlichen Produkt zur

Fernsteuerung eines Schiffes. Firma A hat zudem eine weitere Firma (C) beauftragt ein Dynamikmodell des Testträger-Schiffes bereitzustellen, welches von der Fernsteuerungssoftware benötigt wird, um korrekte Steuerbefehle zu berechnen. Zum Schutz der Daten vor der Konkurrenz müssen die Systeme der Firmen A und B vollständig voneinander abgekapselt werden, während eine parallele Nutzung bestimmter Testfeldkomponenten (z.B. Live-AIS-Datenquellen) erforderlich ist. Zudem muss es möglich sein das Dynamikmodell von Firma C in einer Kollaboration mit Firma A innerhalb des Testfeldes zu verwenden, ohne den vollständigen Quellcode oder Entwicklungsprozess offenzulegen. Schließlich strebt Firma A eine baldige Zertifizierung des entwickelten Systems an.

1.3 Zielsetzung

Aus der Problemstellung ergibt sich die folgende Forschungsfrage, welche im Rahmen der Dissertation beantwortet werden soll:

„Wie können dezentral organisierte maritime Daten für einen datengetriebenen Forschungs- und Entwicklungsprozess im Testfeld bereitgestellt werden?“

Zur Beantwortung dieser Forschungsfrage soll im Rahmen dieser Dissertation ein klassischer Systementwicklungsprozess durchlaufen werden. Im Folgenden werden auf Basis der Forschungsfrage und der Problemstellung fünf Teilziele hergeleitet, welche dieses System erfüllen muss.

Teilziel 1: *Zusammenfassung mehrerer unabhängiger bzw. dezentraler Datenquellen zu einer kohärenten Einheit.*

Es stellt sich zunächst eine essenzielle Frage des Datenmanagements im dezentralen Kontext: Wie können die dezentral organisierten Quellen in einem Testfeld zusammengeführt und für einen gemeinsamen Anwendungsfall genutzt werden (vgl. AIS, RADAR und Wetterdaten in UC1)? Um mit diesen Daten einen Forschungs- bzw. Entwicklungsprozess zu untermauern, müssen sie zunächst zusammengeführt werden. Damit dieser Prozess jedoch nicht für jedes einzelne Vorhaben neu konfiguriert werden muss, sollte das zu entwickelnde System dabei alle verfügbaren maritimen Daten als kohärente Einheit bereitstellen, auf die einheitlich von Forschern und Entwicklern zugegriffen werden kann. Hiermit lässt sich eine Abstraktion von mehrfachen und unkoordinierten Zugriffen auf verschiedene Datenquellen erreichen. Die Fusion von AIS- und RADAR-Daten stellt etwa ein typisches Anwendungsszenario in der Verarbeitung maritimer Daten dar (Guerriero u. a. 2008). Für das Szenario aus UC1 ist dabei aber der Zugriff auf mindestens zwei separate und häufig nicht zentral verwaltete Datenquellen notwendig. Eine kohärente Dateneinheit würde hier die Schnittstellen zweier verschiedener Datenquellen auf eine

einheitliche Zugriffsschnittstelle reduzieren. Dabei muss es ebenso möglich sein, Datenquellen dynamisch in das System einzubinden, die nicht fester Bestandteil des Testfeldes sind (z.B. spezielle Verkehrsmodelle als mögliche Erweiterung des Modells aus UC1). So können Testfeldnutzer ihre eigenen Datenquellen anbinden, um das Testen beliebiger Systeme zu ermöglichen.

Teilziel 2*: Bereitstellung des Zugriffes auf Forschungsdaten in Abhängigkeit von den Testfeldnutzern.*

Ausgehend von Teilziel 1 kann die Frage gestellt werden, ob eine zentralisierte Zusammenführung der Daten an einem Ort die angesprochenen Probleme lösen könnte. Allerdings gestaltet sich ein solches Vorhaben nur selten problemlos, wenn die Daten bereits dezentral verwaltet werden. Es wäre ein enormer organisatorischer Aufwand nötig, um in einem Forschungsnetzwerk aus verschiedenen Teilnehmern einen Konsens über das Zusammenführen aller Daten an einem zentralen Ort zu finden. In solch einem Prozess spielen rechtliche Aspekte und unternehmerische Interessen, wie z.B. der Wettbewerbsdruck der beiden Firmen in UC2, eine nicht zu vernachlässigende Rolle (vgl. Zrenner u. a. 2019), da die Daten vor der Zusammenführung unter der Aufsicht einer einzelnen Organisation verwaltet wurden. Hier müsste etwa auch geprüft werden, ob persönliche Daten nach Datenschutz-Grundverordnung (DSGVO) verarbeitet wurden, oder ob das Teilen bestimmter Daten im Interesse des Datenbesitzers liegt. Speziell im Falle der kommerziellen Bereitstellung dieser Daten, ist dies ein hochrelevantes Thema. Wie in UC2 dargestellt, haben datenbereitstellende Organisationen häufig wenig Interesse am freien Teilen oder Veröffentlichen von vollständigen Datensätzen (wie einem Dynamikmodell eines Schiffes), da diese immer mehr zu einem wertvollen Gut werden. Weiterhin müssten Änderungen in den dezentral verwalteten Daten überwacht und auf den hypothetischen zentralen Speicher übertragen werden. Daher sollte das zu entwickelnde System die Bedürfnisse der verschiedenen Stakeholder im maritimen Testfeldkontext berücksichtigen: Hierbei muss unterschieden werden zwischen Datenbasen, die nur durch ihre Besitzer verwaltet werden, Daten, die für begrenzte Teile des Testfeldbetriebes bzw. der Testfeldnutzung notwendig sind und Daten, die allen Beteiligten zur Verfügung gestellt werden können. So können die Souveränität und die unabhängigen Interessen der Stakeholder berücksichtigt werden.

Teilziel 3: *Unterstützung dezentraler Datenverarbeitungsprozesse im maritimen Testfeld.*

Datengetriebene Forschungsprozesse ermöglichen heutzutage das Erlangen neuer wissenschaftlicher Erkenntnisse durch das reine Analysieren bereits vorhandener Datensätze. Allerdings müssen an solch einen Prozess gewisse Anforderungen gestellt werden. Wie für die Herleitung eines Kollisionsrisikomodells in UC1 müssen beginnend mit der Überprüfung der Datenqualität, entsprechende Daten vorverarbeitet und selektiert werden. Neueste

datengetriebene Forschungsansätze stützen sich häufig auf Methoden des maschinellen Lernens (ML), die später beispielsweise in maritimen Assistenzsystemen eingesetzt werden (Park, Jeong, und Park 2021). Daneben können auch beim Testen von Assistenzsystemen komplexe, verteilte Datenverarbeitungsprozesse eine Rolle spielen, die für die Bewertung des Testerfolges überwacht nachverfolgt werden müssen. Außerdem kommt bei der Entwicklung maritimer Assistenzsysteme insbesondere das Szenario-basierte Testen zum Einsatz (Reiher und Hahn 2021). So sollte etwa die in UC2 vorgestellte Fernsteuerung unter verschiedensten, systemrelevanten Bedingungen (Normalbetrieb aber z.B. auch abbrechende Verbindungen, widersprüchliche Steuereingaben, etc.) evaluiert werden. Das zu entwickelnde System muss solche Anforderungen abbilden können. Um einen maximalen Mehrwert für die Nutzer des Systems zu bieten, sollte der gesamte Prozess einer Datenverarbeitungskette von Ende-zu-Ende unterstützt werden können. Auch die Teile des Prozesses die lokal auf der Infrastruktur eines Testfeldnutzers stattfinden, müssen dabei nachverfolgt werden können. Weiterhin gilt es bereits vorhandene datengetriebene Forschungsprozesse abbilden zu können und durch das System zu unterstützen. Besonders im Bereich der Modellbildung sind klassische Prozesse, wie der KDD-Prozess relevante Vorgehensweisen, die mit dem entworfenen System kompatibel seien sollten (Fayyad, Piatetsky-Shapiro, und Smyth 1996). So könnten typische Anwendungsfälle der Modellbildung (wie UC1) durch Testfelddatendienste vereinfacht oder automatisiert werden.

Teilziel 4: *Gewährleistung von Sicherheit und Nachverfolgbarkeit der Datenverarbeitungsschritte in einem datengetriebenen Forschungsprozess.*

Aufbauend auf Ziel 2 muss die Frage gestellt werden, wie eine Kooperation verschiedenster dezentral verwalteter Daten sicher und nachverfolgbar gestaltet werden kann. Gerade in Forschungsprozessen ist die Nachvollziehbarkeit der Ergebnisse besonders relevant (vgl. Eisner 2018). Nicht reproduzierbare Forschungserkenntnisse werden im Allgemeinen nicht von der wissenschaftlichen Community anerkannt. Ähnlich verhält es sich bei der Zertifizierung sicherheitskritischer Systeme (vgl. Ruiz, Sabetzadeh, und Panaroni 2011). Ebenfalls kann eine Nachverfolgbarkeit zur Stabilisierung eines Vertrauensverhältnisses zwischen unabhängigen Testfeldnutzern beitragen. So muss also entgegengesetzt zu der Dezentralität der Daten eine zentrale Möglichkeit bestehen, die die Interaktionen von den Nutzern des Systems mit diesem sichtbar und nachvollziehbar macht. Dies ist in beiden vorgestellten Anwendungsfällen relevant: In UC1 geht es primär um die Nachvollziehbarkeit und Reproduzierbarkeit, während sich die drei Firmen aus UC2 bei durchgeführten Tests darauf verlassen können müssen, dass nur dazu berechtigte Teilnehmer auf sensitive Daten und Verarbeitungsprozesse zugegriffen haben. Hierbei müssen die üblichen Prinzipien der IT-Sicherheit berücksichtigt werden, da es sich um wertvolle und sensible Informationen handeln kann. Es muss sichergestellt werden können, dass eine Dokumentation von Datenverarbeitungsprozessen mittels der dezentral verwalteten

Datenquellen konsistent, korrekt und verfügbar ist. Im Kontext eines Testfeldes muss das zu entwickelnde System dazu beitragen, dass Konflikte zwischen konkurrierenden Stakeholdern aufgelöst werden können und, dass nachgewiesen werden kann, welche Originaldaten von welchen Teilnehmern eines Experimentes zu welchem Zeitpunkt erhoben wurden. Zudem muss sichergestellt sein, dass Unberechtigte keinen Zugriff auf sensible Daten haben.

Teilziel 5: *Vereinfachung des datengetriebenen Forschungs- und Entwicklungsprozesses für maritime Testfelddaten.*

Ein zentraler Anwendungsfall maritimer Testfelder findet sich in der Validierung und Verifikation von komplexen Systemen. Das zu entwickelnde System soll hierbei helfen komplexe Datenverarbeitungsprozesse zu verstehen, analysierbar zu machen, verschiedene Testergebnisse zu verwalten und somit den gesamten V+V Prozess aus Datensicht zu erfassen. Weiterhin werden in verschiedenen maritimen Testfeldern strukturell ähnliche Daten erhoben, die für reproduzierbare Anwendungsfälle verwendet werden. Datenquellen wie AIS, RADAR, Wetterdaten oder Seekarten sind charakteristisch für maritime Testfelder. Das zu entwerfende System muss diese gängigen Datenquellen ohne maßgeblichen Aufwand integrieren können und grundlegende Datenverarbeitungsoperationen durchführen können, um die datengetriebenen Forschungsprozesse durch angemessene Vorverarbeitungsschritte zu vereinfachen und die Datenqualität zu erhöhen (vgl. UC1). Allerdings sollte es trotzdem weiterhin möglich sein das System durch (vor der Laufzeit) unbekannte Datenquellen zu erweitern, um so flexibel auf die Anforderungen der Testfeldnutzer eingehen zu können. Zudem sollte beachtet werden, dass aufgrund der maritimen Umgebungsbedingungen für Feldexperimente nicht immer eine hohe Bandbreite für den Datenaustausch zur Verfügung steht.

Diese Arbeit soll sich im Bereich der maritimen Testfelder auf solche Testfelder beschränken, die genutzt werden, um schiffszentrierte Systeme zu untersuchen. Hiermit werden Testfelder ausgeschlossen, die sich ausschließlich mit landseitiger Infrastruktur, z.B. in Häfen befassen. Weiterhin beschränkt sich diese Arbeit auf lokale, nicht mobile Testfelder. Der Austausch von Daten aus maritimen Testfeldern zwischen weltweit verteilten Testfeldnutzern oder Kontrollzentren wird somit nicht in dieser Arbeit betrachtet.

1.4 Aufbau

Diese Forschungsarbeit beschäftigt sich mit der Konzeptionierung, Umsetzung eines Systems zum Verwalten von Forschungs- und Entwicklungsdaten. Der Aufbau und das Vorgehen dieser Arbeit ist als ingenieurwissenschaftlicher Forschungsprozess ausgerichtet und an die Methodik nach Peffers u. a. (2007) angelehnt. Abbildung 2 zeigt eine Übersicht über das ingenieurwissenschaftliche Vorgehen und die Struktur dieser Arbeit.

Kapitel 1 führt in das Thema dieser Arbeit ein und liefert eine Motivation für die Auseinandersetzung mit Datenverarbeitungsprozessen in maritimen Testfeldern. Darauf aufbauend werden die Problemstellung und Forschungsfrage hergeleitet. Schließlich werden fünf Teilziele entwickelt, die eine klare Definition des Forschungsvorhabens dieser Arbeit darstellen. Weiter wird in Kapitel 2 der aktuelle Stand von Konzepten und Methoden aus Technik und Wissenschaft vorgestellt, die für das Erheben von Anforderungen an ein maritimes Forschungs- und Entwicklungsdatenmanagementsystem relevant sind. Hierbei wird auf Datenmanagement im Allgemeinen und Forschungsdatenmanagement im Speziellen eingegangen. Weiter werden Prozesse für den dezentralen Datenaustausch diskutiert und der Bezug zur maritimen Domäne und entsprechenden Testfeldern hergestellt. Eine Betrachtung von möglichen Use-Cases und einer Herleitung konkreter Anforderungen schließt das Kapitel ab. Danach folgt in Kapitel 3 eine Analyse von verwandten Arbeiten aus der Literatur und relevanten Forschungsprojekten. Hierbei werden klassische Forschungsdatenmanagementsysteme, Ansätze dezentral organisierter Datenökosysteme und Datenmanagementarchitekturen in Testfeldern betrachtet und mit Hilfe der zuvor definierten Anforderungen und der Teilziele ein Handlungsbedarf hergeleitet. Darauffolgend wird in Kapitel 4 ein Lösungskonzept vorgestellt, das die identifizierte Forschungslücke schließen und den definierten Anforderungen gerecht werden soll. Konkret passiert dies durch den Entwurf eines Forschungs- und Entwicklungsdatenmanagementsystems für maritime Testfelder. Die prototypische Umsetzung dieses Systems wird in Kapitel 5 beschrieben. Diese wird in Kapitel 6 dazu verwendet das Lösungskonzept in realistischen Anwendungsfällen in drei Fallstudien zu validieren und zu verifizieren, ob die definierten Anforderungen und Ziele erreicht wurden. Außerdem wird eine quantitative Evaluation der Performanz des Systems durchgeführt. Die Arbeit endet mit einer Zusammenfassung, der Darstellung der Ergebnisse und einer Diskussion der Limitierungen und einem Ausblick auf mögliche anschließende Forschungsaktivitäten.

Abbildung 2: Ingenieurwissenschaftliches Vorgehen und Aufbau dieser Arbeit.

Teile der Ergebnisse dieser Dissertation wurden bereits in mehreren wissenschaftlichen Publikationen veröffentlicht: (Möller, Fröschle, und Hahn 2021), (Möller, Jankowski, und Hahn 2021), (Möller u. a. 2022).

2 Datengetriebene Forschungs- und Entwicklungsprozesse

Das Ziel dieser Arbeit ist es den maritimen datengetriebenen Forschungs- und Entwicklungsprozess mit Daten aus dezentral verwalteten Datenquellen in maritimen Testfeldern abzubilden und ein Lösungskonzept für die in Kapitel 1 definierten Problemstellungen zu entwickeln. In diesem Kapitel werden Konzepte, Technologien und Prozesse vorgestellt, die für das Erheben der Anforderungen an ein maritimes Forschungs- und Entwicklungsdatenmanagementsystem (im Folgenden: FEDMS) notwendig sind. Dabei werden insbesondere die Grundlagen der datengetriebenen Forschung, des Forschungsdatenmanagements und des Austausches dezentral organisierter Daten dargestellt. Die Themen werden speziell im Kontext maritimer Testfelder betrachtet. Des Weiteren wird auf den normativen Rahmen der Arbeit und konkrete Typen maritimer Daten eingegangen und herausgearbeitet, welche Eigenschaften der Daten beim Entwurf eines FEDMS für Testfelder berücksichtigt werden müssen. Schließlich werden die Erkenntnisse aus diesen Abschnitten gezielt dazu verwendet Anforderungen zu erheben und diese auf den Bezugsrahmen dieser Arbeit zu spezialisieren.

2.1 Perspektiven der datengetriebenen Forschung und Entwicklung

Diese Arbeit baut auf den grundlegenden Konzepten der datengetriebenen Forschung und Produktentwicklung auf, welche in den letzten Jahrzenten durch die Entwicklungen der Data Science, e-Science und dem Aufkommen des „Big Data“-Phänomens maßgeblich beeinflusst wurden. Insbesondere relevant ist zudem die Anwendung von Methoden dieser Bereiche in maritimen Testfeldern.

Big Data. Mit einer zunehmend vernetzten Welt ändert sich nicht nur die Menge an aufgezeichneten und übermittelten Daten, sondern auch die Art und Beschaffenheit der erhobenen Daten. In der heutigen Gesellschaft ist es immer wichtiger die korrekten Daten zu bestimmten Zeitpunkten zu verarbeiten und dem richtigen Nutzer zur Verfügung zu stellen. Im Kontext der Verarbeitung solcher erheblichen Datenmengen aus den verschiedensten Bereichen wird immer wieder der Begriff „Big Data“ erwähnt. Die eigentliche Definition dieses Begriffes beschränkt sich jedoch nicht allein auf die Menge der Daten, sondern bildet eine abstraktere Struktur ab. In der Literatur gibt es allerdings keine einheitlich anerkannte Definition des Begriffes. Häufig wird „Big Data“ als Datenmenge bezeichnet, die aufgrund verschiedener Eigenschaften nicht mit klassischen Datenverarbeitungsmechanismen behandelt werden kann. Die vier bekanntesten dieser Eigenschaften sind: Velocity, Value, Variety und Volume (Chen, Mao, und Liu 2014) und werden häufig auch als „V’s of Big Data“ bezeichnet: Die Geschwindigkeit (Velocity), mit welcher neue Daten erhoben werden, kann stark variieren und führt zu Echtzeitanforderungen,

Schwankungen in der Datenerhebungsgeschwindigkeit und neuen Herausforderungen beim Reagieren auf hochfrequente Datenströme (McAfee und Brynjolfsson 2012). Der Wert (Value) großer Datensätze für Industrie und Forschung spielt ebenfalls eine wichtige Rolle, da Erkenntnisse aus Daten genutzt werden, um Optimierungen in jeglichen Bereichen zu erwirken (Chen, Mao, und Liu 2014). Auch für den Konsumenten kann die Auswertung von Big Data Datensätzen entscheidende Wertschöpfungsprozesse hervorbringen. So können durch Big-Data-Methoden etwa Staus vermieden werden, oder das Wetter durch die Analyse historischer Datensätze besser vorhergesagt werden. Die Varietät (Variety) von Daten beschreibt das Vorhandensein verschiedenster Datentypen und ist ein großes Problem in universellen Datenverarbeitungsprozessen, da nicht alle Datentypen sinnvoll in eine abstrakte Form (wie etwa multidimensionale Feature-Vektoren) überführt werden können (Mao u. a. 2015). Ebenfalls ist die vorhandene Menge (Volume) an Daten eine Herausforderung in der Verwaltung und Speicherung von Big Data (Kaisler u. a. 2013). Als weitere Eigenschaft wird in der Literatur häufig auch noch die „Veracity of Big Data“ betrachtet: Bei dem Erheben und Zusammentragen von Daten aus verschiedensten Quellen muss beachtet werden, dass in diesen Daten möglicherweise Unsicherheiten, Fehler oder Abweichungen vorliegen (Rubin und Lukoianova 2013). Es gibt außerdem zahlreiche Erweiterungen der V-Eigenschaften, die jedoch nicht einheitlich verwendet werden.

Data Science. Mit dem Aufkommen von Big Data stieg ebenfalls das Interesse am Forschungsbereich Data Science. Nach (Dhar 2013, 1) bezeichnet der Begriff Data Science „die generalisierbare Extraktion von Wissen aus Daten“. Dazu werden verschiedene Methoden aus der Mathematik, Statistik, und der künstlichen Intelligenz, sowie Datenbanktechnologien und Optimierungsverfahren eingesetzt (Dhar 2013). Die Data Science wird häufig konkret durch das sogenannte Data Mining, dem tatsächlichen Prozess der Wissensextraktion aus Daten, angewandt (Provost und Fawcett 2013). Das durch das Data Mining gewonnene Wissen kann dann zum Beispiel in Unternehmen dazu genutzt werden, um Entscheidungen zu treffen (data-driven decision making) wird aber auch eingesetzt, um Risiken frühzeitig zu erkennen und darauf zu reagieren (Provost und Fawcett 2013). Meist wird in der Data Science ein fest definierter Prozess verfolgt, um Wissen aus Daten zu extrahieren. Hierbei sei der Knowledge Discovery in Databases (KDD) Prozess erwähnt, bei welchem sich die Analyse von Daten in fünf Schritte aufteilt (Fayyad, Piatetsky-Shapiro, und Smyth 1996): Zunächst wird die Teilmenge der zu betrachtenden Daten selektiert, dann vorverarbeitet, transformiert, eine Mustererkennung ausgeführt und folglich die Ergebnisse evaluiert und interpretiert. Dieses Modell stellt die einzelnen Schritte dar, die notwendig sind, um Wissen aus einem Datensatz zu extrahieren. Hierdurch wird auch der Zusammenhang zwischen Big Data und Data Science klar: Nur durch das Vorhandensein von

entsprechend umfangreichen und hochwertigen Datensätzen können durch einen Data Mining Prozess interpretierbare Ergebnisse geschaffen werden.

e-Science und Datenökosysteme. Wenn Methoden aus der Data Science kollaborativ in der Forschung eingesetzt werden, wird in diesem Zusammenhang häufig von e-Science gesprochen. Hey und Trefethen (2003, 1809) definieren e-Science nach John Taylor folgendermaßen: „Bei e-Science geht es um die globale Zusammenarbeit in Schlüsselbereichen der Wissenschaft und um die nächste Generation der Infrastruktur, die dies ermöglichen wird." Weiterhin werden digitale Daten- und Rechenservices als Kernkomponente verstanden und dabei nicht nur das verteilte Rechnen und Abrufen von Daten berücksichtigt, sondern auch das kollaborative Arbeiten in „virtuellen Organisationen", die aus kollaborierenden, aber dezentral organisierten Teilnehmern bestehen (Hey und Trefethen 2003). Die e-Science beschäftigt sich also vor allem mit Aspekten, die die kollaborative Nutzung von Daten ermöglichen und vereinfachen. Um kollaborative Dateninfrastrukturen nutzen zu können, sind Kollaborationsmanagement, Werkzeuge für gemeinsames Aufzeichnen, Analysieren und Visualisieren von Daten, Methoden zur Archivierung und Bereitstellung von Daten, Infrastruktur zum Veröffentlichen von Daten wichtige Instrumente, die thematisch ebenfalls die Kernbereiche der e-Science abbilden (Jankowski 2007). Ein ähnlicher Trend ist in der Wirtschaft zu beobachten, in der immer mehr Teilnehmer aus der Industrie kollaborieren, um Daten für Entwicklungsprozesse mit anderen zu Teilen und somit ökonomische Vorteile zu erhalten (Oliveira und Lóscio 2018).

Maritime Testfelder. Nach IEEE ist ein Testfeld „eine Umgebung, die Hardware, Messgeräte, Simulatoren, Software-Werkzeuge und andere unterstützende Elemente enthält, die für die Durchführung eines Tests benötigt werden." (ISO/IEC/IEEE 2010, 466). Seit der Entwicklung hochautomatisierter Schiffssteuerungssysteme spielen besonders physikalische Testfelder in der maritimen Domäne eine immer wichtigere Rolle, besonders vor dem Hintergrund der Verifikation und Validierung (V+V) sicherheitskritischer Anwendungen (Brinkmann, Hahn, und Hjøllo 2017). Diese Testfelder werden zum einen dazu genutzt Daten zu sammeln, um später statische Analysen auf ihnen ausführen zu können, zum anderen können die Daten auch dynamisch genutzt werden, um Realweltexperimente von zu testenden Systemen durchzuführen (Brinkmann, Abdelaal, und Hahn 2018). Hierbei stehen meist diverse heterogene Datenquellen zur Verfügung (Brinkmann, Stasch, und Hahn 2016), die auch von verschiedenen Stakeholdern in das Testfeld integriert werden können und für das Testen der zu testenden Systeme benötigt werden. Die Daten bilden also die Grundlage eines maritimen Testfeldes und der zu Testenden Systeme, um die Aktivitäten der V+V durchführen zu können.

Für das Forschungs- und Entwicklungsdatenmanagement in maritimen Testfeldern ergeben sich damit drei relevante Perspektiven aus den aktuellen Forschungsdisziplinen:

1. Die Perspektive der **Datengrundlage** des Testfeldes, die sich zum heutigen Zeitpunkt nur noch sinnvoll durch das Konzept Big Data mit den V-Eigenschaften modellieren lässt.

2. Die Perspektive der **Wissensgenerierung** aus Testfelddaten und Daten aus Testläufen, die durch die Methoden der Data Science charakterisiert wird und Prozesse vorgibt, wie von Forschern und Entwicklern mit der Datengrundlage interagiert wird.

3. Die Perspektive der **kollaborativen Methodik** der e-Science und der Datenökosysteme in Testfeldern, die die Prozesse der Data Science und des Datenteilens in den Kontext des kollaborativen Arbeitens und der häufig dezentral vorliegenden Infrastrukturen und verschiedenen Stakeholder setzt.

Die genauen Zusammenhänge der drei übergeordneten Forschungsbereiche sollen im Folgenden nun für die Anwendungsperspektive eines FEDMS im Kontext maritimer Testfelder konkretisiert werden.

2.2 Forschungsdatenmanagement

Um den Begriff des Forschungsdatenmanagements zu definieren, wird zunächst das allgemeine Datenmanagement in Abschnitt 2.2.1 betrachtet und dann in Abschnitt 2.2.2 auf die Eigenschaften des Forschungsdatenmanagements eingegangen. Der Abschnitt 2.2.3 beschäftigt sich dann speziell mit der Situation in maritimen Testfeldern.

2.2.1 Allgemeine Aspekte des Datenmanagements

Das Verwalten von Daten erfordert neben dem reinen Speichern der Daten und dem Zugreifen auf die Daten noch mehrere erweiterte Aktivitäten. Diese müssen ausgeführt werden, um eine konsistente Datengrundlage für weitere Prozesse wie etwa die Nutzung der Daten in maritimen Testfeldern sicherzustellen. Solche Aktivitäten sind eng verbunden mit den Prozessen der Datenerhebung, -verarbeitung und -bereitstellung. Dies entspricht der Definition von Mosley u. a. (2010), die Datenmanagement wie folgt beschreiben:

> *„Data management (...) is the business function of planning for, controlling and delivering data and information assets." – (Mosley u. a. 2009, 4)*

Weiterhin liefern sie eine genauere Definition des Begriffes durch das Beschreiben der einzelnen Funktionen des Datenmanagements (siehe Abbildung 3). Das Datenmanagement setzt sich insgesamt aus verschiedenen einzelnen Bereichen bzw. Funktionen zusammen, die durch das

Data Governance Konzept geplant, beobachtet und kontrolliert werden (vgl. Otto und Weber 2011).

Zum Datenmanagement gehören zunächst die klassischen Bereiche der Datenverarbeitung wie Datenbankzugriff, Datenarchitektur, Datenanalyse und -weiterverarbeitung (Data Development). Weiterhin folgen übergeordnete Aktivitäten wie Data Security Management, welches sich mit dem Sicherheitskonzept und Zugriffsrechten beschäftigt und Data Quality Management, das durchgeführt wird, um die Qualität der Daten zu analysieren. In Kombination mit dem Metadatenmanagement geht es beim Reference & Master Data Management darum, den Prozess der Datenerhebungs- und -verarbeitungsstadien nachzuvollziehen und mit zusätzlichen Meta-Informationen anzureichern. Das Reference & Master Data Management orientiert sich dabei meist an einem geschäftlichen Kontext und soll geschäftliche Zusammenhänge in den vorhandenen Daten beschreiben. Weitere Spezialisierungen des Datenmanagements finden sich im Data Warehousing & Business Intelligence Management, welches sich mit der Modellierung der Prozesse zur Informationsgewinnung aus den Daten beschäftigt und im Document & Content Management beschäftigt, welches sich mit dem Management von nicht strukturierten Daten beschäftigt. (Mosley u. a. 2009)

Einzelne oder mehrere dieser Unterkategorien sind in diversen Arbeiten zu finden, oft mit gegenseitigen Abhängigkeiten (Wende 2007), (Khatri und Brown 2010), (Deelman und Chervenak 2008).

Oft verschwimmen jedoch die Grenzen zwischen den von Mosley u. a. vorgestellten Unterkategorien des Datenmanagements. So existieren etwa Datenmanagementsystem-Architekturen, wie der Data Lake (Terrizzano u. a. 2015), die sowohl strukturierte als auch unstrukturierte Daten, sowie die zugehörigen Metadaten zentral verwalten. Dabei werden außerdem längst nicht immer alle Kategorien des Datenmanagements nach Mosley u. a. berücksichtigt: Im Kontext des maritimen Testfeldes sind z.B. Data Warehousing & Business Intelligence Management und Reference & Master Data Management weniger relevant, da hier meist eindeutig definierte und fixierte Prozesse zugrunde liegen, die der benötigten Flexibilität eines Testfelddatenmanagementsystems nicht gerecht werden würden.

Abbildung 3: Funktionen des Data Managements (vgl. Mosley u. a. 2009).

Die allgemeine Perspektive von Mosley u.a. bildet eine ganze Reihe an Anwendungsfällen ab. In dieser Arbeit soll aber speziell das Datenmanagement von Forschungs- und Entwicklungsdaten in Testfeldern betrachtet werden. Hier lässt sich der Teilbereich des Datenmanagements eingrenzen, da zur Entwicklungszeit eines FEDMS nicht notwendigerweise klar ist, welche Datenquellen in Zukunft in welchen Anwendungsszenarien verwendet werden sollen. Da maritime Testfelder selbst als Plattform für eine große Menge von Anwendungsszenarien dienen, lassen sich hier keine vollständigen, standardisierten Prozesse vordefinieren, die konsistent benötigt werden. So kann das Testfeld etwa genutzt werden, um eine Fernsteuerungssoftware wie in UC2 zu testen, es ist aber auch möglich das Testfeld lediglich zu nutzen, um den maritimen Verkehr zu beobachten und das Verhalten von Schiffen zu analysieren wie in UC1.

2.2.2 Grundlegende Aspekte des Forschungsdatenmanagements

Forschungsdatenmanagement (in der englischsprachigen Literatur meist als „*scientific data management*“ bezeichnet, wörtlich übersetzt: Wissenschaftliches Datenmanagement) beschäftigt sich mit den speziellen Aspekten von Datenmanagement im wissenschaftlichen bzw. Forschungskontext. Der grundlegende Unterschied zum allgemeinen Datenmanagement liegt hier in der Tatsache, dass die Daten meist vielfältiger eingesetzt werden als in einem nicht-Forschungskontext und ein klarer Verwendungszweck der Daten nicht eindeutig festgelegt werden kann: Datensätze werden (öffentlich) verwaltet und für andere Zwecke wiederverwendet (vgl. auch Abschnitt 3.1). Dies führt u.a. zu einer sehr hohen Verfügbarkeit von Datensätzen aus

verschiedensten Forschungsdomänen, die eine große Heterogenität aufweisen. Somit entstehen neue Herausforderungen in der Nachvollziehbarkeit von großen Datenverarbeitungsworkflows und einer erschwerten Auffindbarkeit von Datensätzen (Shoshani und Rotem 2009). Im Jahr 2016 veröffentlichten Wilkinson u. a. (2016) die sogenannten FAIR-Prinzipien für wissenschaftliches Datenmanagement. In ihrer Arbeit, die mittlerweile zu einer der meistzitierten Arbeiten im Bereich des Forschungsdatenmanagements gehört, wurden die Herausforderungen des wissenschaftlichen Datenmanagements dargelegt und vier neue Leitprinzipien für das Verwalten wissenschaftlicher Datensätze vorgestellt, die den angesprochenen Problemen entgegenwirken sollen (Wilkinson u. a. 2016):

- Auffindbarkeit (**F**indability): Um die Auffindbarkeit von Datensätzen für die Forschung zu verbessern, sollen (Meta-)Daten eindeutig identifizierbar sein, Metadaten sollen umfangreich erhoben werden und einem Datensatz eindeutig zugeordnet werden können. Außerdem sollen (Meta-)Daten in durchsuchbaren Registern registriert oder indexiert werden.

- Zugänglichkeit (**A**ccessability): Um eine bessere Zugänglichkeit zu Datensätzen zu ermöglichen sollen (Meta-)Daten durch einen Identifier und mittels standardisierter Kommunikationsprotokolle abrufbar sein. Diese Protokolle sollen offen, frei verfügbar und universal implementierbar sein und ebenfalls Prozeduren zur Authentifizierung und Autorisierung beim Datenabruf bieten. Weiterhin sollen Metadaten auch nach dem eventuellen Entfernen der eigentlichen Daten verfügbar bleiben.

- Interoperabilität (**I**nteroperability): Für das Herstellen von Interoperabilität sollen (Meta-)Daten eine formale, zugängliche, geteilte und allgemein anwendbare Sprache verwenden, um Wissen zu repräsentieren. Dabei sollen Vokabeln verwendet werden, die ebenfalls den FAIR-Prinzipien folgen und qualifizierte Verweise auf andere (Meta-)Daten verwenden.

- Wiederverwertbarkeit (**R**eusability): Damit Datensätze einfacher wiederverwertet werden können, sollen (Meta-)Daten reichhaltig und akkurat beschrieben werden, mit einer klaren Nutzungslizenz versehen werden, Informationen zu ihrer Herkunft beinhalten und domänenrelevante Standards unterstützen.

Diese Prinzipien spielen in der klassischen akademischen Forschung eine wichtige Rolle, müssen hier aber in den Kontext der Forschungs- und Entwicklungsdaten in maritimen Testfeldern gesetzt werden: Hier kommen Stakeholder aus Industrie und Wissenschaft zusammen. Neben den Zielen der Nachvollziehbarkeit in der Wissenschaft, sind es ökonomische Interessen, die darauf abzielen bestimmte Daten nicht der breiten wissenschaftlichen Community (und damit auch der

Konkurrenz) öffentlich bereitzustellen, um wie in UC2 einen Wettbewerbsvorteil zu bewahren. Die FAIR-Kriterien tragen hier zwar weiterhin zur Effizienzsteigerung in Forschungs- und Entwicklungsprozessen bei, müssen aber durch eine Zugriffskontrolle eingeschränkt werden. Die verwalteten Daten in einem Testfeld sind in diesem Szenario nur punktweise für festgelegte Gruppen an Stakeholdern FAIR. Ein typisches Beispiel hierfür sind Trainingsdatensätze für ML-Anwendungen in Anwendungsfällen wie UC1. Solche Datensätze werden oft in mühsamer Handarbeit gelabelt und gewinnen somit stark an Wert, da es mit ihnen möglich ist besonders performante ML-Modelle zu trainieren. In einem maritimen Testfeld könnte es sich hierbei konkret um annotierte Kameraaufzeichnungen handeln, in denen Schiffe markiert wurden. Betrachtet man also nicht öffentlich verfügbare Datensätze ist es zwar erwünscht, dass Entwickler von Assistenzsystemen die Daten einfach und zugänglich auffinden können und interoperabel in anderen Systemen wiederverwenden können, unautorisierte Parteien sollen sie aber nicht nutzen können (vgl. auch UC2).

In Ergänzung zu diesen Grundprinzipien, wird das wissenschaftliche Datenmanagement häufig in Zyklen bzw. Prozessen dargestellt. Hierbei werden die Schritte vom Einlesen der Quelldaten bis zur Wissensgewinnung aus den verarbeiteten Daten beschrieben. Dieses Vorgehen wird in der Literatur häufig als Teil der e-Science dargestellt (Humphrey 2006). Ein fundiertes und detailliertes Prozessmodell liefern Crowston und Qin (2011). In einem Vergleich von neun Data Management Zyklen/Prozessmodellen durch Ball (2012) wird dieses Modell als eines der umfangreichsten identifiziert. Abbildung 4 zeigt eine Übersicht über das Modell: Zunächst werden die Datenerhebung, Vorverarbeitung für die Speicherung und Qualitätsanalyse beschrieben. Im 2. Schritt werden die zu betrachtenden Daten u.a. durch Metadaten beschrieben und dokumentiert. In dem Modell mit wissenschaftlichem Hintergrund werden zudem mit den Daten zusammenhänge Publikationen mit einbezogen. Ebenfalls wichtig ist hier die Sicherstellung von Interoperabilität und die Standardisierung. Im dritten Schritt geht es um die Veröffentlichung der Daten. Dazu werden Zusammenhänge zwischen verschiedenen Daten analysiert und typische Maßnahmen wie etwa die Verwaltung von Zugriffen ergriffen, um die Daten verfügbar zu machen. Im 4. Schritt geht es um eine langfristige Konsistenz der Daten. Diese wird durch Datensicherung, Validierung und Erweiterung des Datenbestandes angestrebt.

Um die FAIR-Prinzipien einhalten zu können, muss also der gesamte Prozess von der Datenerhebung bis zur Dissemination der Daten abgebildet werden. Gerade in dezentral organisierten Strukturen stellt dies eine Herausforderung dar, da die verschiedenen Prozessschritte nicht von einer zentralen Entität durchgeführt werden, sondern auf viele verschiedene Teilnehmer verteilt sind.

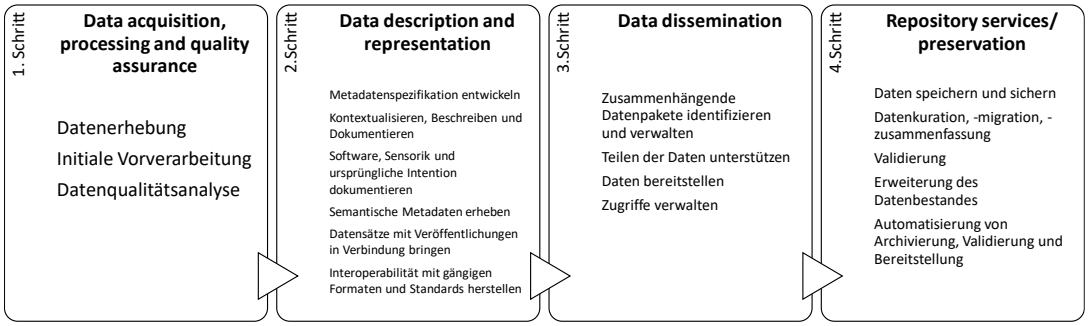

Abbildung 4: Scientific Data Management Prozess (vgl. Crowston und Qin 2011).

Auch in serviceorientierten Testfeldern werden Datenerhebung, Datenverarbeitung und -archivierung nicht notwendigerweise von einer Entität durchgeführt. Dies ergibt sich allein schon durch die Anwesenheit verschiedener Stakeholder und dem Plattformcharakter eines Testfeldes: Durch das Nutzen generischer Testfeldarchitekturen wird das Testfeld klar von zu testenden Systemen abgegrenzt (Nickovic u. a. 2018). Hierbei ist das mit Umweltsensorik ausgestattete Testfeld maßgeblich für die Erhebung der Daten zuständig, die in einem weiteren Schritt vom zu testenden System weiterverarbeitet werden. Denkbar sind hier auch weitere Parteien, wie unabhängige dritte (Testabnehmer oder Behörden, etwa bei der in UC2 angesprochenen Zertifizierung des Fernsteuerungssystems), die die Testergebnisse konservieren, oder externe Datenprovider, die als weitere Urheber von Daten in diesen Prozess integriert werden.

2.2.3 Forschungs- und Entwicklungsdaten in maritimen Testfeldern

Forschungs- und Entwicklungsdaten aus maritimen Testfeldern werden vielfältig eingesetzt und erfüllen in der wissenschaftlichen Literatur drei wesentliche Funktionen:

Grundlage für die Modellbildung und Systementwicklung in der maritimen Domäne. Das Schaffen einer Datengrundlage mittels eines maritimen Testfeldes stellt eine klassische Datenmanagementaufgabe dar, wie sie im vorigen Abschnitt definiert wurde: Verkehrsdaten, Schiffsbewegungsdaten, Wetterdaten, und weitere Daten müssen gespeichert, aufbereitet und durch eine Datenmanagementarchitektur bereitgestellt werden können, damit sie zur Modellbildung und Systementwicklung genutzt werden können (Brinkmann, Hahn, und Hjøllo 2017), (Martins u. a. 2014). Dies möglichst generisch und standardisiert zu tun entspricht den FAIR-Prinzipien und ermöglicht vielfältige Anwendungsfälle und das Generieren von neuem Wissen. So können Testfelddaten in Anwendungsfällen wie UC1 genutzt werden, um beispielsweise Kollisionsrisikomodelle zu entwickeln.

Testen von Systemen während des Entwicklungsprozesses im Rahmen der V+V. Spezifischere Anwendungen der Daten eines maritimen Testfeldes leiten sich aus den Anforderungen zu testender Systeme im Rahmen der Verifikation und Validierung ab (Brinkmann, Stasch, und Hahn 2016). Hierbei geht es darum, die Funktionalität von (Teil-)Systemen in einer kontrollierbaren Umgebung experimentell zu untersuchen, und festzustellen, ob es die Bedürfnisse der Stakeholder erfüllt (bzw. das erwartete Verhalten in der Einsatzumgebung des Systems zeigt) und, ob die spezifischen Anforderungen an das System erfüllt wurden (Engel 2010, 19). Dabei wird bei hochkomplexen maritimen Systemen oft auf Szenario-basierte Testverfahren zurückgegriffen, bei denen repräsentative Szenarien genutzt werden, um die Systemfunktionalitäten zu verifizieren und zu validieren (Lamm und Hahn 2018). Neurohr u. a. (2020) präsentieren in ihrer Arbeit zu diesem Thema eine schematische Darstellung des Prozesses zum Szenario-basierten Testen von automatisierten Steuerungssystemen im Automotive Bereich (Abbildung 5): Zu Beginn des Verfahrens werden neue Szenarien generiert, indem je nach Methode Expertenwissen, Use-Case und Datengrundlage (z.B. große Datensätze von Fahrverhalten (Pütz u. a. (2017)) genutzt werden, um Szenarioklassen zu entwickeln. Diese werden dann mit entsprechenden Sicherheitsanforderungen dazu genutzt konkrete Testfälle abzuleiten, auszuführen und zu evaluieren.

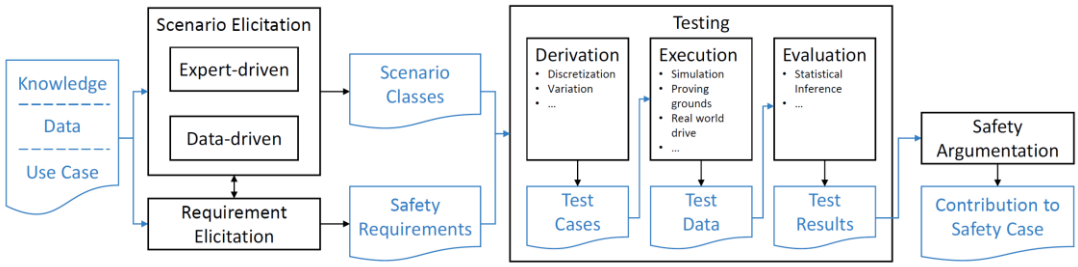

Abbildung 5: Ablauf und Artefakte des Szenario-basierten Testens (Neurohr u. a. 2020).

Wie zu erkennen ist, sind relevante Aspekte des Datenmanagements im Testfeld hier vor allem in der datengetriebenen Szenariogenerierung und bei der Testdurchführung und -auswertung zu beachten. Schließlich fällt beim Durchführen von Tests eine große Menge relevanter Daten aus Sensorik und Informationssystemen an, die auch schon während eines Tests dazu dienen kann, bestimmte Parameter auszuwerten. So wird im Bereich der Automatisierung von Fahrzeugen das Konzept der Operational Design Domain (ODD) genutzt, die einen Zustandsraum der Umgebung und des Fahrzeuges (inkl. interner Zustände) definiert, für den ein automatisiertes System entwickelt wurde (Czarnecki 2018). Zu erkennen ob sich das SuT in einem Zustand in der ODD befindet, kann entscheidend für die Durchführung eines Tests sein.

Demonstration und Zertifizierung bestehender Systeme. Bei der Zertifizierung von maritimen Systemen werden Anforderungen an Daten von Tests gestellt. So wird beispielsweise im Kontext

autonomer Schiffführungssysteme empfohlen, operationale Daten bzgl. Erhebung, Verarbeitung und Speicherung von zu testenden Systemen zu sammeln und zu dokumentieren (DNV GL AS 2018, 33ff.). Auch bei der Generierung von Testberichten von Systemen kann das Sammeln dieser Daten Sinn ergeben (DNV GL AS 2018, 32f.). Dies spielt insbesondere eine Rolle für Contract-basierte Zertifizierungsverfahren, bei deinen kontinuierlich geprüft werden muss, ob bestimmte Daten in vereinbarten Wertebereichen liegen (vgl. Hake, Feuerstack, und Hahn 2020). Weiterhin werden Testfelder genutzt, um Systeme zu demonstrieren (vgl. z.B. Dixon und Morrison 2008). Dabei kann der Fokus etwa auf dem Aufzeigen einzelner Systemeigenschaften liegen, sodass eine spezifische Menge an Daten aus dem Testfeld und dem zu demonstrierenden System ausgewählt und für Beobachter visualisiert wird. Hier tritt zudem besonders der Aspekt der sicheren Nachvollziehbarkeit bestimmter Datenverarbeitungsprozesse im Testfeld in den Vordergrund.

Aus diesen Aktivitäten ergeben sich drei verschiedene Phasen, in denen Datenmanagementstrategien in maritimen Testfeldern benötigt werden (siehe Abbildung 6). *Phase I* findet vor der Entwicklungsphase von zu testenden Systemen statt und zielt auf ein den Aufbau eines allgemeinen Datenpools aus maritimen Testfelddaten ab. Diese Aufgabe kommt dem klassischen Forschungsdatenmanagement sehr nah und baut auf den eingeschränkten (vgl. Abschnitt 2.2.2) FAIR-Prinzipien auf. *Phase II* behandelt das Datenmanagement während der Entwicklung von zu testenden Systemen speziell im Bereich der V+V. Dabei ist es besonders wichtig komplexe Datenflüsse in (Teil-)Systemen nachvollziehen zu können und Daten aus einzelnen Testszenarien zu verwalten. *Phase III* ergibt sich nach der Entwicklungszeit von Systemen und besteht daraus gezielt das Beobachten bestimmter Daten zu ermöglichen und diese für Demonstration oder Dokumentation bereitzustellen. Typischerweise sind diese Daten eine Teilmenge der Daten aus *Phase II*, da das in *Phase II* entwickelte System weiterhin im Fokus der Beobachtung steht.

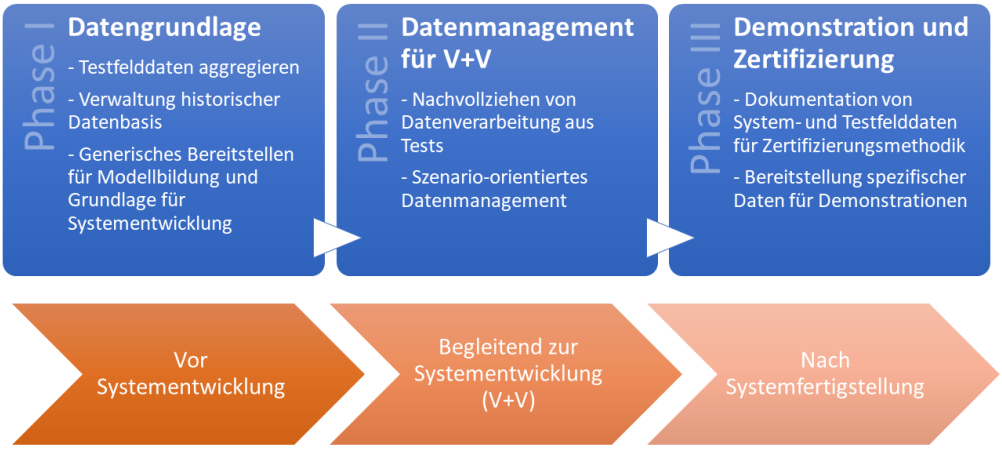

Abbildung 6: Drei Phasen des Datenmanagements in maritimen Testfeldern: Datengrundlage, Datenmanagement für V+V und Demonstration und Zertifizierung.

Weiterhin gibt es die übergeordneten Herausforderungen des Datenqualitätsmanagements und des Datensicherheitsmanagements (vgl. Abschnitt 2.2.1) im maritimen Testfelddatenmanagement, die sich keiner einzelnen Phase zuordnen lassen.

Datenqualitätsmanagement. Die Bewertung von Datenqualität ist überwiegend kontextabhängig und multidimensional (R. Y. Wang und Strong 1996). Es lassen sich allerdings einige Oberkategorien und Merkmale erkennen, die in den meisten Kontexten Anwendung finden. So teilen Strong, Lee, und Wang (1997) diese Merkmale in folgende Kategorien ein:

- **Intrinsische Datenqualität**: Hierunter fallen Qualitätskriterien, die von der Repräsentation und der Organisation der Daten unabhängig sind: Genauigkeit, Objektivität, Glaubhaftigkeit und Reputation.

- **Zugänglichkeit**: Unter die Zugänglichkeit fallen Zugriffsarten auf mehreren Ebenen: Zum einen zählen die technische Zugänglichkeit der Daten, zum anderen aber auch die Zugänglichkeit der Daten im Zusammenhang mit Verständlichkeit, Interpretierbarkeit und Konsistenz. Zusätzlich kann auch das Volumen der Daten zu Zugänglichkeitsschwierigkeiten führen.

- **Kontextuelle Datenqualität**: Die kontextuelle Datenqualität beschreibt die Nutzbarkeit der Daten im Kontext der Anwendung. Hierzu zählen Relevanz, Mehrwert, Aktualität, Vollständigkeit und verfügbare Menge.

- **Repräsentationsqualität**: Die Repräsentationsqualität sagt aus, wie nützlich das Format bzw. ihre Repräsentation für die Verwendung der Daten ist. Hierunter fallen Interpretierbarkeit, Verständlichkeit, Übersichtlichkeit und Konsistenz der Repräsentation.

Für den Entwurf von Datenqualitätsmechanismen ist aufgrund der Tatsache, dass sie kontextspezifisch ist, domänenspezifisches Wissen notwendig. Im maritimen Kontext ist insbesondere das Sicherstellen der intrinsischen Datenqualität eine Herausforderung. Ein großer Teil der in der datengetriebenen, maritimen Forschung verwendeter Datensätze besteht aus Daten von maritimer Sensorik (vgl. auch Abschnitt 2.4). Hierbei fällt eine große Menge an Rohdaten an, die noch Ungenauigkeiten und Rauschen beinhalten. Zudem kommt die Tatsache, dass eine der am häufigsten verwendeten Datenbasen aufgezeichnete Daten aus dem Automatic Identification System sind (siehe Abschnitt 2.4). Diese setzen sich aus einer Kombination von dezentral organisierter Sensorik (verteilt auf einzelne Schiffe) aufgezeichneten Werte zusammen und weisen im Allgemeinen keine hohe Datenqualität auf (Harati-Mokhtari u. a. 2007).

Datensicherheitsmanagement. Unter Datensicherheitsmanagement versteht man „die Planung, Entwicklung und Ausführung von Sicherheitsrichtlinien und -verfahren, um eine ordnungsgemäße Authentifizierung, Autorisierung, Zugriff und Prüfung von Daten und Informationsbeständen zu gewährleisten“ (Mosley u. a. 2009, 151). Primär geht es hierbei darum Informationen (repräsentiert durch Daten) unter Berücksichtigung von Privatsphäre und Vertraulichkeit zu schützen, und dies mit weiteren Anforderungen zu vereinbaren. Diese Anforderungen werden insbesondere durch Stakeholder, gesetzliche Rahmenbedingungen oder anwendungsbezogene Gegebenheiten vorgegeben. Konkrete Ziele in der Umsetzung sind das Bereitstellen von Authentifizierungs-, Autorisierungs-, Zugriffs- und Prüfungsmechanismen. (Mosley u. a. 2009)

2.3 Dezentraler Datenaustausch

Mit dem zunehmenden Vernetzungsgrad von Akteuren in Wirtschaft und Wissenschaft, sowie im alltäglichen Umgang mit digitalen Diensten findet die Verwaltung aller zu einem Dienst zugehörigen Daten standardmäßig längst nicht mehr zentral bei ebendiesem statt. Vielmehr werden Daten aus verschiedensten Quellen zusammengeführt, vorverarbeitet, transformiert und erst dann für eine Anwendung genutzt. In einer Arbeit, die sich mit dem Vergleich von zentralen und dezentralen Informationssystemen beschäftigt, definiert Hugoson (2009) den Begriff der Dezentralisierung wie folgt:

> *“In a business perspective, decentralization means that business can make decisions locally.” - (Hugoson 2009, 107)*

Im Zusammenhang dieser Arbeit geht es in der Frage der Verteiltheit oder Zentralisierung vor allem um die Kontrolle über verschiedene Datenquellen. So haben im Szenario der „dezentralen Forschungs- und Entwicklungsdaten“ Stakeholder im maritimen Testfeld keine zentrale Kontrolle über alle Datenquellen: Ein Testfeldnutzer verfügt in der Regel über eigene Datenquellen, nutzt aber auch die Infrastruktur, die vom Testfeldbetreiber bereitgestellt wird und fragt Daten aus anderen Quellen ab. Alle Stakeholder haben jedoch weiterhin die Freiheit lokal über ihre eignen Datenquellen zu entscheiden. Die Definition der organisatorischen Dezentralisierung wird von Hugoson weiter erläutert und dabei auf die Dezentralisierung von Informationssystemen eingegangen:

> *„The system in the [decentralized] structure must fulfil specified demands on interaction with other systems, but it should be possible to develop (and change) the inner structure in each system, including data storage, without dependences to other systems, as long as the specified interaction stands. It*

must for instance be possible to insert systems of different origin into the structure. The main condition is that each system must interact with other systems as specified." - (Hugoson 2009, 107)

Das Argument der gemeinsamen Interaktion wird dadurch begründet, dass die Grenzen der dezentralisierten Struktur erkennbar sein müssen, denn sonst könnte auch die Gesamtheit aller existierender Informationssysteme als Teil einer einzigen dezentralisierten Struktur interpretiert werden (Hugoson 2009). Die Dezentralität eines FEDMS für Testfelddaten wird also durch die Gesamtheit aller Datenquellen im Kontext des Testfeldes eingeschränkt.

2.3.1 Das maritime Testfeld als Datenökosystem

Durch Kooperationen zwischen verschiedenen Partnern entstehen in Wirtschaft und Wissenschaft kontinuierlich Verbindungen zwischen autonom agierenden Akteuren. Im Hinblick auf den Datenaustausch haben sich in den letzten Jahren industrieübergreifende, sozio-technische Netzwerke ausgebildet, die im Allgemeinen als „Datenökosysteme" bezeichnet werden (Gelhaar und Otto 2020). Solche Datenökosysteme finden sich in den verschiedensten Bereichen. So findet man sie in der Industrie und Wirtschaft, beispielsweise für das Austauschen von Daten über Energiesysteme (Diran u. a. 2020) oder im Bereich des öffentlichen Verkehrs (Le Dantec u. a. 2015). Vermehrt treten sie bisher allerdings im Bereich der öffentlichen Verwaltung bzw. im Bereich „Open Government Data (Ecosystems)" oder in Teilbereichen der wissenschaftlichen Forschung auf. Daten der öffentlichen Verwaltung werden dabei häufig frei verfügbar zur Verfügung gestellt (Sanaei u. a. 2019) und für Anwendungen wie die Verwaltung von öffentlichen Ressourcen oder die Beschaffung von Informationen über diese genutzt. Im Bereich der Wissenschaft werden Datenökosysteme typischerweise für einzelne Forschungsdisziplinen, wie etwa für die Krebsforschung (Grossman 2018) oder den Bereich der Neurowissenschaften (Wiener u. a. 2016) angelegt. Auch hier handelt es sich häufig um öffentlich verfügbare Daten.

In der Literatur finden sich verschiedene Definitionen eines Datenökosystems, die häufig ähnliche Merkmale beschreiben. Oliveira und Lóscio führten 2018 eine Literaturanalyse durch, in der sie aus den verschiedenen Definitionen gemeinsame Merkmale eines Datenökosystems ableiteten. Diesen sollen im Folgenden auf den Anwendungsfall (insbesondere auch auf die Anwendungsfälle aus Abschnitt 1.2) eines maritimen Testfeldes übertragen werden (vgl. Tabelle 3 in Oliveira und Lóscio 2018):

- *Ein Datenökosystem besteht aus mehreren Akteuren.* Akteure eines maritimen Testfeldes können durch Betreiber, Nutzer, Datendienstanbieter und Beobachter charakterisiert werden. Diese Rollen können jeweils auch von mehreren Akteuren übernommen werden (vgl. Abschnitt 2.3.4). In den Anwendungsfällen UC1 und UC2 sind dies z.B. die

Forscher, die das Kollisionsrisikomodell entwickeln, Datendienstleister, der Testfeldbetreiber und die Firmen, welche die Fernsteuerungssysteme konstruieren.

- *Die Akteure haben verschiedene Interessen, Fähigkeiten und Anforderungen an das Datenökosystem.* Auch dies ist hier der Fall. Selbst innerhalb der verschiedenen Arten von Akteuren können konfliktäre Interessen in Bezug auf Daten oder (Teil-)Systeme festgestellt werden (vgl. Abschnitt 2.3.4). Dies zeigt sich z.B. durch den Wettbewerb der beiden Firmen in UC2.

- *Datenökosysteme verwenden häufig Rollenmodelle, um die verschiedenen Merkmale der Akteure einzuordnen und den Bezug zur Produktion und Konsum von Daten beschreiben.* Diese Eigenschaft ist nicht zwangsläufig erfüllt. Zwar gibt es klare Unterschiede zwischen Testfeldnutzer und Testfeldbetreiber und Beobachtern, die ggf. auch vertraglich festgehalten werden. Ein explizites Rollenmodell muss dafür aber nicht vorhanden sein. Es kann jedoch sinnvoll sein, ein solches Modell im Testfeld zu etablieren, um Akteure klarer voneinander zu differenzieren. In UC1 unterscheidet sich z.B. der Forscher in seiner Rolle als Datenkonsument vom externen Bereitsteller der Wetterdaten.

- *Zwischen den Akteuren bestehen Beziehungen, die es ermöglichen miteinander zu agieren und Ressourcen auszutauschen. Ggf. wird dazu ein vorher definiertes Business Model verwendet.* Die klare Beziehung ist hier eindeutig durch das gemeinsame Nutzen des Testfeldes gegeben. Ein vorher definiertes Business Model (falls vorhanden) wird typischerweise vom Testfeldbetreiber zur Bereitstellung des Testfeldes ausgearbeitet, kann aber auch zwischen Nutzern zum Austausch von Daten entwickelt werden (Wetterdatenbezug in UC1 oder Nutzung eines Dynamikmodells in UC2). Ressourcen stellen hier Datensätze aus dem Testfeld, aber auch Services und Infrastruktur zur Bereitstellung und Verarbeitung von Daten dar (vgl. Oliveira und Lóscio 2018).

- *Der Austausch von Ressourcen findet möglicherweise standardisiert statt, unterliegt Lizenzen oder Qualitätsuntersuchungen.* In existierenden Arbeiten wird hierbei speziell der Fokus auf die Schnittstelle zwischen Testfeldarchitektur und zu testendem System gelegt (vgl. z.B. Brinkmann, Hahn, und Hjøllo 2017): Systementwickler und Bereitsteller des Dynamikmodells in UC2 müssen sich bspw. auf eine gemeine Schnittstelle zwischen den Systemen (und dem Testfeld) einigen.

Das allgemeine Verständnis eines Datenökosystems ist also größtenteils zutreffend auf die gemeinsame Nutzung eines maritimen Testfeldes durch die verschiedenen Stakeholder.

2.3.2 Auffindbarkeit und Interoperabilität

Zunächst einmal muss geklärt werden, wie in einer dezentral organisierten Struktur aus Teilnehmern mit verschiedenen Zielen und Interessen relevante Datensätze überhaupt aufgefunden werden können. Natürlich ist hier eine konventionelle Kommunikation (z.B. persönlicher Austausch oder Kommunikation via Mail oder Telefon) zwischen den Teilnehmern möglich, lässt sich allerdings nur schwer auf komplexere oder umfangreichere Szenarien skalieren. Für nicht öffentliche Daten wird es daher mit zunehmender Verfügbarkeit schwieriger entsprechende Datensätze überhaupt aufzufinden.

Zur Lösung dieses Problems werden Datensätze mittels speziell dafür entwickelter Dienste bzw. Plattformen bereitgestellt (Data-as-a-Service). Dieses Paradigma ermöglicht auch die gezielte Implementierung von Maßnahmen zur besseren Auffindbarkeit der Daten. Häufig ist die Komplexität solcher Plattformen im privaten bzw. wirtschaftlichen Sektor deutlich größer als in öffentlicheren Sektoren wie der Verwaltung oder Wissenschaft. Dies lässt sich durch den potenziellen Wert von nicht öffentlichen Datensätzen erklären, der sich im Interesse des Datenbesitzers auf die Verfügbarkeit und damit die Zugriffsrechte auswirkt. (Zhuang und Lee 2016)

Des Weiteren ergeben sich in Datenökosystemen Herausforderungen in Bezug auf die Interoperabilität zum Datenaustausch verwendeter Schnittstellen und der Daten selbst (Otto 2022, 10). Dabei definiert das Standard Computer Dictionary der IEEE Interoperabilität als „Die Fähigkeit von zwei oder mehr Systemen oder Komponenten, Informationen auszutauschen und die ausgetauschten Informationen zu nutzen.“ (IEEE 1990, 114). Auch hier spiegelt sich das Problem der dezentralen Verwaltung der Datenbestände wider: Da keine einheitliche Harmonisierung stattfindet, liegen Daten in Datenökosystem häufig in verschiedenen Datenmodellen vor und werden mithilfe unterschiedlicher Technologien gespeichert. So entstehen also sowohl Probleme bei dem reinen Austausch der Daten als auch beim Nutzen der Daten selbst, die je nach Herkunft in unterschiedlichen Datenmodellen vorliegen. Das Problem der Interoperabilität in großen Datenökosystemen wird mit der zeitlichen Dimension zunehmend komplex und beschränkt sich nicht ausschließlich auf den technischen Bereich, sondern schließt auch organisatorische Aspekte mit ein (Agostinho u. a. 2016). Im maritimen Bereich ergeben sich die angesprochenen Probleme ebenfalls: So wird eine breite Menge an verschiedenen Datenarten und -formaten sowie Technologien verwendet (Herodotou u. a. 2021). Wie in anderen Bereichen können sich auf der organisatorischen Ebene in der Testfelddatennutzung Probleme ergeben, insbesondere bei der dynamischen Verarbeitung von Live-Daten zur Laufzeit von Tests ist eine systematische organisatorische Einigung über die Nutzungsbedingungen notwendig (vgl. Munoz-Arcentales u. a. 2019). Hier kann der Testfeldbetreiber eingreifen und die Nutzung eines gemeinsamen Datenmodells für seine Datenquellen festlegen. So könnte etwa das universell

erweiterbare, standardisierte hydrografische Datenmodell S-100, welches von der International Hydrographic Organisation entwickelt wird (Contarinis, Pallikaris, und Nakos 2020), als testfeldweites Datenmodell verwendet werden (Brinkmann und Hahn 2017). Eine vollständige Durchsetzung eines solchen Datenmodells, auch für dynamische hinzugefügte Datenquellen kann zwar angestrebt werden und hätte den Vorteil einer einfacheren Integration, ist aber aufgrund der dezentralen Organisation der Stakeholder nicht immer vollständig umsetzbar.

2.3.3 Datensouveränität

Neben den eher technisch-organisatorischen Problemen der Auffindbarkeit und Interoperabilität, ergeben sich in Datenökosysteme auch Herausforderungen, die eher von organisatorischer oder rechtlicher Relevanz sind. Ausschlaggebend sind hier die verschiedenen Interessen der Teilnehmer eines Datenökosystems. Im Zusammenhang mit dem Austauschen von Daten zwischen Personen oder Institutionen mit verschiedenen Interessen wird häufig der Begriff der Datensouveränität verwendet, für welchen jedoch keine einheitliche Definition in der Literatur zu finden ist (Zrenner u. a. 2019). Im Rahmen dieser Arbeit soll die Definition nach (Otto u.a. 2019) verwendet werden:

> *„Datensouveränität kann als die Fähigkeit einer natürlichen oder juristischen Person definiert werden, die volle Kontrolle über ihre Daten zu haben“ - (Otto u.a. 2019, 9)*

Es geht hier also darum, den Austausch eigener Daten in einem Ökosystem vollständig kontrollieren zu können. Dies wäre z.B. nicht ohne Weiteres gewährleistet, wenn eine (juristische) Person ihren vollständigen Datenbestand an eine dritte Partei weitergeben würde, um die Verwaltung dieser Daten unabhängig durchzuführen. Ein typisches Beispiel hierfür wären etwa klassische kommerzielle Cloud-Anbieter, die für Datenverwaltung und -Austausch genutzt werden (vgl. z.B. Peterson, Gondree, und Beverly 2011).

Dies muss jedoch nicht heißen, dass jeder Teilnehmer eines Datenökosystems seine Daten ausschließlich bei sich behält und nicht teilt, um vollständige Souveränität zu gewährleisten. Vielmehr beinhaltet das allgemeine Verständnis der Datensouveränität das Vorhandensein von Konzepten zum organisierten und abgesicherten Austausch, der die Rechte der Datenbesitzer wahrt. Probleme in der Umsetzung ergeben sich hierbei vor allem durch technische Hindernisse. (Hummel u. a. 2021)

2.3.4 Akteure im dezentralen Datenaustausch in Testfeldern

Innerhalb eines Datenökosystems müssen verschiedene Interessen berücksichtigt werden. Die Herausforderung komplexe Austausch- und Verarbeitungsverfahren zwischen verschiedenen Parteien nachvollziehen und überwachen zu können ist ein bekanntes Problem (Cuno u. a. 2019). Trotzdem lassen sich in diesen Prozessen typische Rollen erkennen, die in verschiedenen Datenökosystemen wiedergefunden werden können. Einige Datenökosysteme berücksichtigen diese Rollen bereits in der Entwurfsphase (siehe unten). Zu Rollenmodellen in dezentral organisierten Datenaustauschsystemen gibt bereits einige Untersuchungen in wissenschaftlichen Publikationen oder Referenzarchitekturmodelle, die Stakeholder und ihre typischen Interessen und Verantwortlichkeiten bei der Datenverarbeitung identifizieren. Drei Modelle aus der Literatur sollen im Folgenden betrachtet werden.

Xu u. a. (2014) stellen ein detailliertes Rollenmodell für datenschutzkonformes Data Mining mit verschiedenen Teilnehmern vor (siehe Abbildung 7). Ihr Fokus liegt auf der Untersuchung des „Knowledge Discovery from Data“-Prozesses im Hinblick auf die Wahrung der Privatsphäre aus verschiedenen Stakeholder-Perspektiven. Daher basiert das Modell hauptsächlich auf den Akteuren in einem Data-Mining-Prozess.

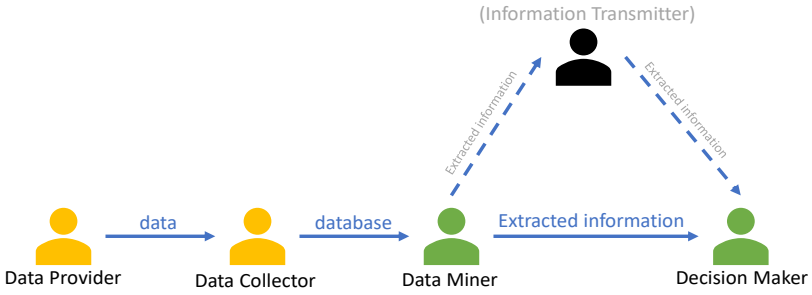

Abbildung 7: Modell der Stakeholder für datenschutzkonformes Data Mining (vgl. Xu u. a. 2014).

Weiterhin ist die International Data Space (IDS) - Referenzarchitektur (Otto, Steinbuß, und et al. 2019) ein standardisiertes Modell zur Etablierung der Datenintegration in einem dezentralen Data Space (siehe auch Abschnitt 2.3.5). Es wurde bereits in mehreren Publikationen angewandt und auch in industriellen Anwendungsfällen realisiert. Ein wichtiger Bestandteil der Referenzarchitektur ist das Rollenmodell (siehe Abbildung 8), welches die verschiedenen Entitäten eines Datenraums auf einer eher generischen Ebene und teilweise aus industrieller Sicht betrachtet.

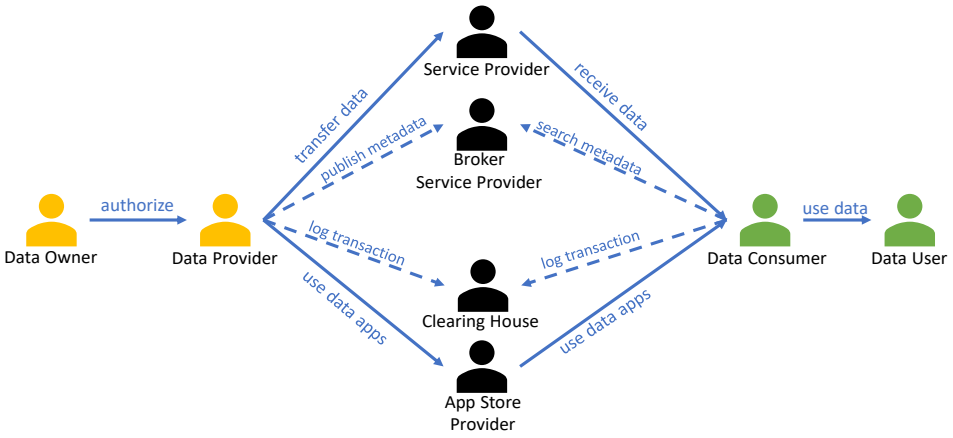

Abbildung 8: Auszug aus dem Rollenmodell der International Data Space Referenzarchitektur (vgl. Otto, Steinbuß, und et al. 2019).

Oliveira u. a. (2017) stellen schließlich ein Rollenmodell für den Austausch von Daten in einem öffentlich zugänglichen Datenökosystem vor (siehe Abbildung 9). Ihr Fokus liegt auf der Darstellung der Aspekte des Austauschs von Open Data (öffentlich verfügbare Daten) und typischer Akteure in diesem Szenario.

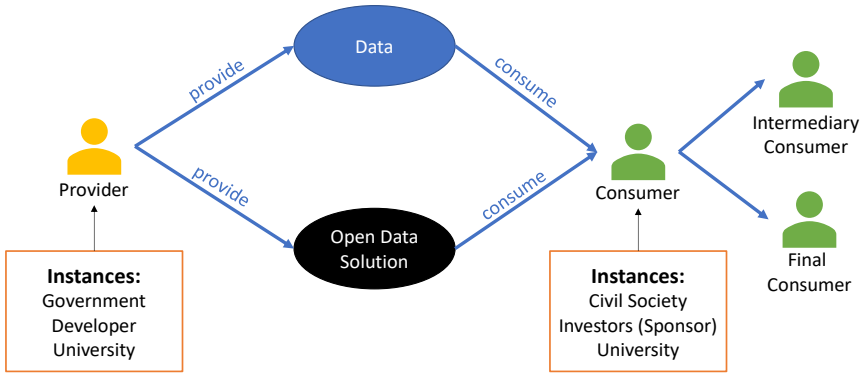

Abbildung 9: Rollenmodell für den Datenaustausch für ein öffentlich zugänglichen Datenökosystem (vgl. Oliveira u. a. 2017).

Alle drei Modelle unterscheiden zwischen einer Art Datenprovider, der die Daten generiert und/oder bereitstellt, und dem Datenkonsumenten, der die Daten für seine Zwecke nutzt. In der detaillierten Ausgestaltung dieser Gruppen unterscheiden sich die Modelle: Xu u.a. stellen die Datenkonsumenten als Data Miner und Decision Maker dar, während die IDS-Referenzarchitektur zwischen einem Datenkonsumenten und einem Datennutzer unterscheidet, der die Daten tatsächlich nutzt (und nicht nur erhält). Dieser Unterschied findet sich auch im Modell von Oliveira et al. wieder, in welchem die Konsumenten in Intermediary Consumer und Final Consumer aufgeteilt werden. Die meisten anderen Akteure in den Modellen von Xu et al. und dem IDS sind Intermediäre im Prozess der kollaborativen Datenverarbeitung. Diese

Intermediäre sind architekturspezifisch und erfüllen Aufgaben wie Datenaustausch, Bereitstellen von Datendiensten, Datenauffindbarkeit und Vertrauensbildung (in der erweiterten Version des IDS-Modells in (Otto, Steinbuß, und et al. 2019)). Oliveira u.a. geben auch einige spezifische Instanzen des abstrakten Stakeholders im Use-Case eines öffentlich verfügbaren Data-Ökosystems an.

Das Modell des Datenproviders, Intermediärs und des Datenkonsumenten (vgl. auch Otto 2022, 8f.) kann nun in Kombination mit dem 3-Phasen Modell aus Abschnitt 2.2.3 verwendet werden, um die Stakeholder eines maritimen Testfelddatenökosystems abzuleiten (siehe Abbildung 10). In *Phase I* ist zunächst der Testfeldbetreiber der offensichtlichste Datenprovider: Er stellt die Datenquellen des Testfeldes bereit und ermöglicht so einen Zugriff auf die Datengrundlage des Testfeldes. Testfeldnutzer agieren in dieser Phase ausschließlich als Datenkonsumenten, um die Datengrundlage für Forschungs- oder Entwicklungsprojekte zu nutzen. Generell lassen sich die Testfeldnutzer in Forscher bzw. Systementwickler unterteilen, die ggf. leicht abweichende Anforderungen haben. (Akademische) Forscher stellen die lediglich die Anforderung an FAIR-konformes Datenmanagement, während (industrielle) Systementwickler zusätzliches Interesse an der Wahrung von Wettbewerbsvorteilen haben (vgl. Abschnitt 2.2.2 und UC2). Die Rolle des Intermediärs lässt sich dem Testfeldbetreiber zuordnen, da dieser ein Interesse hat, seinen Nutzern alle verfügbaren Daten sicher und vertrauenswürdig als Dienstleistung mit Hilfe eines FEDMS zur Verfügung zu stellen. Er besitzt diese Rolle eines Treuhänders allerdings auch als einziger, direkt involvierter Stakeholder, weshalb eine klare Trennung von seiner Datenproviderrolle anzustreben ist, um Interessenkonflikte zu vermeiden (z.B. bevorzugtes Bereitstellen eigener Datenquellen). Ist eine solche Trennung nicht möglich, oder erhält diese kein ausreichendes Vertrauen durch andere Stakeholder, so lässt sich die Bereitstellung der Daten auch durch einen externen Datentreuhänder auslagern, welcher die Aufgabe der Datenübermittlung als vertrauenswürdige neutrale Partei übernimmt und keine weiteren eigenen Interessen verfolgt (Thomas und Leiponen 2016). Schließlich ist es ebenfalls denkbar, dass Dritte weitere Daten zur Erweiterung der Testfelddatengrundlage (Data-as-a-Service Modell) bereitstellen (z.B. Wetterdaten wie in UC1). Diese Rolle ist eindeutig als reiner Datenprovider zu klassifizieren. In *Phase II* kann ein Testfeldnutzer zusätzlich die Rolle eines Datenproviders übernehmen. Dies ist dadurch zu erklären, dass im V+V-Prozess ggf. spezifische Datenquellen benötigt werden, um bestimmte Funktionalitäten von zu testenden Systemen zu überprüfen. Diese können durch die Testfeldnutzer selbst bereitgestellt werden, oder durch einen externen Datenprovider. Auch die Services des Testfeldbetreibers können in *Phase II* und *III* unverändert weitergenutzt werden. In *Phase III* kommen schließlich die Beobachter hinzu, die ausschließlich als Datenkonsumenten zu klassifizieren sind. Sie nutzen speziell ausgewählte Datenquellen des Testfeldes, um etwa

Zertifizierungsverfahren mit einer Datengrundlage zu legitimieren, oder das Verhalten von zu testenden Systemen zu visualisieren.

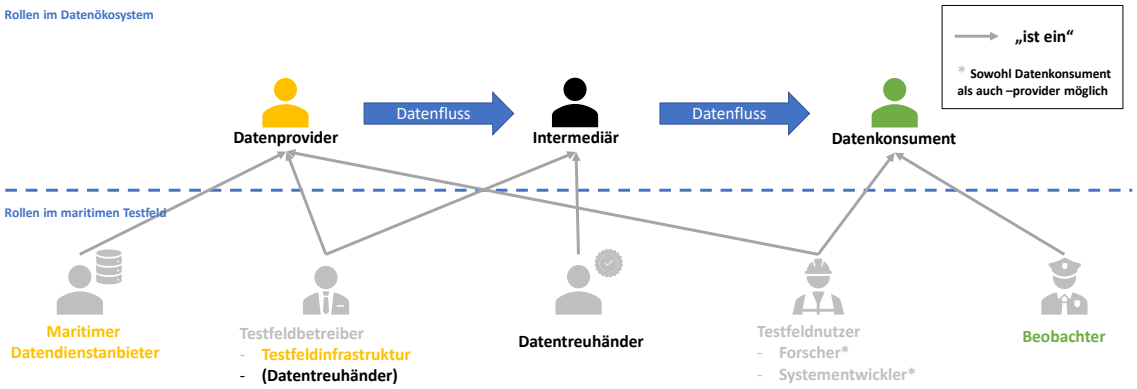

Abbildung 10: Stakeholder im maritimen Testfeld aus Datenperspektive, abgeleitet von Rollen im Datenökosystem (vgl. Möller u. a. 2022).

2.3.5 Data Spaces

Bisher wurden Eigenschaften des dezentralen Datenaustausches, dabei auftretende Probleme und die involvierten Stakeholder diskutiert. In der Literatur findet sich in dem vorgestellten Zusammenhang immer wieder das Konzept des „Data Spaces“. Ursprünglich von Franklin, Halevy, und Maier (2005) beschrieben, wurde das Konzept kontinuierlich weiter erforscht und fand seit den 2010er Jahren sowohl großen Anklang in der Industrie, als auch in wissenschaftlichen Anwendungsfällen. Laut Franklin, u.a., handelt es sich beim Konzept des „Data Spaces“ um ein neuartiges Architekturkonzept zum Management von Daten. Der Begriff Data Space wird seitdem in unterschiedlichen Kontexten verwendet. Im Rahmen dieser Arbeit soll aber die ursprüngliche Definition des Data Spaces von Franklin u.a. verwendet werden. Diese definieren einen Data Space als eine koexistente (und somit dezentral organisierte) Menge von Daten, die durch eine „*Data Space Support Platform*“ (DSSP) verbunden ist. Dieses zunächst abstrakte System muss eine Reihe von Anforderungen erfüllen, um als solches anerkannt zu werden (Franklin, Halevy, und Maier 2005):

- Unterstützung einer breiten Palette von Datentypen und -formaten, die alle Daten im Data Space abdecken.
- Anbieten von Methoden zum Suchen, Abfragen und Aktualisieren von Daten und Verwalten des Data Spaces. Abfragen müssen allerdings nicht zu einem vollständigen Ergebnis der verfügbaren Daten führen, eine Annäherung ist ausreichend.
- Unterstützung von Werkzeugen, um eine engere Integration der Daten im Data Space zu schaffen.

Data Spaces enthalten typischerweise eine Vielzahl an unterschiedlichen Daten in verschiedenen Formaten und Datenmodellen, die über nicht zwingend standardisierte Schnittstellen abgerufen werden können. Die Integration der Daten durch ein unterstützendes System passiert hier schrittweise bzw. inkrementell und die Zugriffe auf Daten erfolgen nicht immer exakt, sondern durch eine „best-effort“ Approximation auf den verfügbaren Datenquellen. (Y. Wang, Song, und Chen 2016)

Es ist also zu erkennen, dass die in den vorherigen Kapiteln diskutierten Herausforderungen moderner Datenökosysteme durch Data Spaces bereits auf der architekturellen Ebene modelliert werden. In der ursprünglichen Definition von Franklin u.a. wird der Data Space durch Teilnehmer und Beziehungen zwischen diesen modelliert (vgl. Eigenschaften von Datenökosystemen in 2.3.1). Abbildung 11 zeigt den konzeptionellen Aufbau eines Data Spaces mit DSSP nach Curry (2020): Hier werden die vielfältigen Beziehungen zwischen den durch die verschiedenen Teilnehmer bereitgestellten Datenquellen und die beschriebene Heterogenität der Daten und verwendeten Technologien klar. Durch ein unterstützendes System werden verschiedene Dienste zum Suchen, Verwalten und Verbessern der Zugriffsmöglichkeiten bereitgestellt. Otto (2022) beschreibt einen sogenannten *Federator*, in dessen Verantwortung es liegt solche Dienste in einem Data Space bereitzustellen (vgl. auch *Intermediär* in Abschnitt 2.3.4). Diese Dienste selbst müssen weiterhin auch nicht zentral umgesetzt werden, sondern können wie die Datenquellen dezentral und verteilt organisiert werden (Otto 2022).

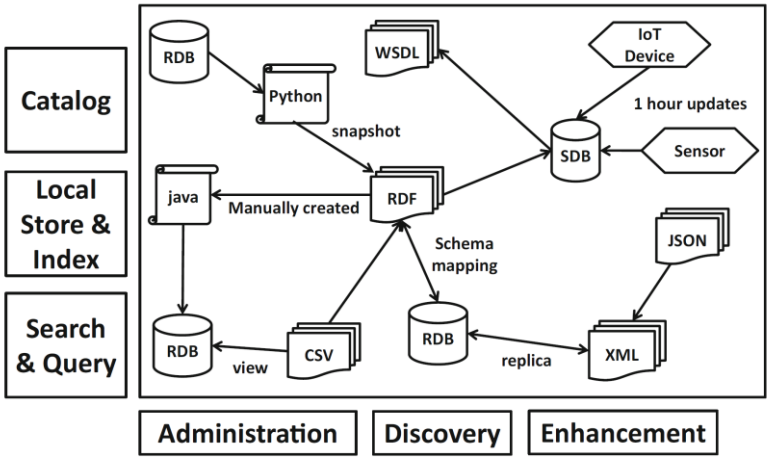

Abbildung 11: Konzeptioneller Aufbau eines Data Spaces (Curry 2020).

Eine weitere Sicht auf die Architektur eines Data Spaces liefern Shakhovska und Bolubash (2019), welche die verschiedenen Abstraktionsebenen in einem Data Space darstellen: Die Grundlage bildet die (Quell-)datenebene, die alle koexistenten Datenquellen beinhaltet, die mit der nächsten Ebene in Interaktion stehen. Dabei handelt es sich um die Verwaltungsebene, die allgemeine Anfragen an die Datenquellen in deren native Anfragesprachen transformiert und etwa Datenkataloge oder Data Warehouse Funktionalitäten bereitstellt. Zuletzt wird die Metaebene

genutzt, um dem Nutzer eine Gesamtsicht auf den Data Space zu präsentieren und Metadaten über die Datenquellen zu verwalten.

Im Vergleich zu anderen Architekturen, die zur Verwaltung heterogener Datenquellen verwendet werden (wie zum Beispiel der Data Lake Architektur), werden die Daten nicht aus den vorhandenen Quellen extrahiert und zentral verarbeitet, sondern die Datenquellen als solche belassen und in die vorhandene Architektur integriert. Hier kommt wieder die Datensouveränität (vgl. Abschnitt 2.3.3) ins Spiel: Die Dezentralität der Daten bleibt in einem Data Space erhalten, und somit auch die Kontrolle der Datenprovider über ihre Daten. (Jarke und Quix 2017)

Eine klassische DSSP stellt alle Funktionalitäten zentral für alle Nutzer des Data Spaces zur Verfügung. Je nach Implementierung werden durch die DSSP Metadaten verwaltet, Datenanfragen analysiert und weitergeleitet und ggf. auch Anfrageergebnisse verarbeitet. Daher gibt es weitere Ansätze, die das Konzept einer zentralen DSSP weiter aufbrechen, und die Funktionalitäten in einzelne Services kapseln, die durch Nutzer frei wählbar genutzt, oder selbst instanziiert werden können. In diese Richtung geht auch der Ansatz des International Data Spaces, bei welchem unterstützende Funktionen zur Bereitstellung von Daten durch unabhängige Services umgesetzt werden. Dabei wird zwischen Brokern und Apps unterschieden: Broker sammeln Metadaten über verfügbare Datenquellen im Data Space und tragen damit zur besseren Auffindbarkeit und Durchsuchbarkeit bei und Apps stellen Funktionen und Algorithmen zur weiteren Verarbeitung dieser Daten bereit. Durch weitere Dienste kann Vertrauen hergestellt werden und eine sichere Abrechnung von Datenprodukten ermöglicht werden. (Otto, Steinbuß, und et al. 2019)

Im Kontext des maritimen Testfelds könnten die vorgestellten Datenprovider ihre Datenquellen mittels eines Data Space weiterhin dezentral zur Verfügung stellen. Die Dezentralität der Datenquellen, die jeweils von Testfeldbetreiber, -nutzer und Datendienstanbietern zur Verfügung gestellt werden, könnten weiterhin unter ihrer Kontrolle bleiben. Zudem könnten Integrations- und Auffindbarkeitsprobleme von maritimen Datenquellen (vgl. Abschnitt 2.3.2) mit Hilfe einer DSSP oder ihrer weiter dezentralisierten, serviceorientierten Variante gelöst werden. Dieses System könnte weiterhin die Anforderungen an das Testfeld-FEDMS abbilden. Insgesamt bietet sich der Data Space also dazu an, die dezentrale Organisationsstruktur des Testfelddatenökosystems architekturell abzubilden.

2.4 Normativer Rahmen

Das Erheben, Verarbeiten und Teilen von Daten bringen immer eine gewisse Verantwortung mit sich. Spätestens wenn in den aufgezeichneten Daten personenbezogene Informationen enthalten sind, muss darauf geachtet werden, dass die Aufzeichnung und Verarbeitung dieser Daten

rechtlich einwandfrei geschehen. Bei dem in der Literatur beschriebenen Problem des „Privacy-Preserving Data Publishing“ haben neben den inhaltlich betroffenen Personen (auch: „Data Subject“) häufig auch die Datenprovider Bedenken beim Teilen von internen Daten (Jurczyk und Xiong 2009). Weiterhin kann das Offenlegen von Daten auch das Aufgeben eines Wettbewerbsvorteils bedeuten (Cavanillas, Curry, und Wahlster 2016, 3f.).

Zwei Beispiele sollen im Folgenden die Herausforderungen im Kontext maritimer Testfelder darstellen. Zunächst wird auf die Seite des Data Subjects eingegangen, im zweiten Beispiel geht es um die Datenprovider.

Ein klassischer Anwendungsfall in *Phase I* des maritimen Testfelddatenmanagements (siehe Abschnitt 2.2.3) ist das Aufzeichnen und Bereitstellen historischer AIS-Daten (z.B. notwendig für UC1). Zunächst einmal muss festgestellt werden, dass es sich bei AIS-Daten um eine unverschlüsselte Kommunikation handelt. Da AIS-Nachrichten nicht an spezielle Empfänger zugestellt werden, sondern von allen Teilnehmern in einem bestimmten Umkreis empfangen werden können, ist davon auszugehen, dass es sich hier um Daten handelt, die für die Allgemeinheit zur Kenntnis bestimmt sind. Somit ist es nach deutscher Rechtslage bzw. § 89 Telekommunikationsgesetz (TKG) grundsätzlich für jeden erlaubt, diese Daten zur Kenntnis zu nehmen. Weiterhin ist zu prüfen, ob mit dem Verarbeiten von AIS-Daten auch personenbezogene Informationen verarbeitet werden. Laut Art. 4, Abs. 1 der europäischen Datenschutzgrundverordnung (DSGVO) handelt es sich bei personenbezogenen Daten um „alle Informationen, die sich auf eine identifizierte oder identifizierbare natürliche Person beziehen“. Im weiteren Sinne fallen hierunter auch die MMSI Nummer, die IMO Nummer und der Name eines Schiffes, da hieraus durch Kombination mit weiteren Datenquellen (z.B. Schiffsregister) personenbezogene Daten ermitteln lassen (vgl. Port of Rotterdam o. J.). Es ist zwar also der Fall, dass AIS-Daten personenbezogene Daten enthalten, allerdings dürfen diese nach §27 Abs. 1 Bundesdatenschutzgesetz (BDSG) auch ohne Einwilligung der jeweiligen Personen für wissenschaftliche Zwecke verarbeitet werden, wenn „die Verarbeitung zu diesen Zwecken erforderlich ist und die Interessen des Verantwortlichen an der Verarbeitung die Interessen der betroffenen Person an einem Ausschluss der Verarbeitung erheblich überwiegen“. Dies ist hier offenbar der Fall. Weiterhin müssen aber die personenbezogenen Daten nach §27 Abs. 3 Bundesdatenschutzgesetz (BDSG) anonymisiert werden „sobald dies nach dem Forschungs(…)zweck möglich ist (…)“. Wenn diese Voraussetzungen erfüllt wurden, dürfen die Daten für Forschungszwecke bereitgestellt werden. Betrachtet man den allgemeineren Fall, in dem Testfeldnutzer oder -betreiber mit industriellem Hintergrund und wirtschaftlichen Interessen AIS-Daten verarbeiten, wird ist die Lage weniger eindeutig. In Art. 6 Abs.1 f) der DSGVO heißt es, dass personenbezogene Daten verarbeitet werden dürfen, wenn ein berechtigtes Interesse bewahrt wird, und dieses Interesse die „Grundrechte und Grundfreiheiten der betroffenen Person,

die den Schutz personenbezogener Daten erfordern, überwiegen“. Da es hier also verschiedene fallspezifische Auslegungen geben kann, ist nicht davon auszugehen, dass jegliche AIS-Daten durch ein FEDMS anonymisiert werden müssen, da selbst im Fall der Forschungsaktivitäten nicht immer klar ist, ob der Forschungszweck eine Anonymisierung zulässt. Es bietet sich also an die Entscheidung auf die Nutzer des FEDMS auszulagern, sodass je nach Anwendungsfall im Testfeld entschieden werden kann. Allerdings besteht selbst bei der Anonymisierung ein Restrisiko, dass durch Kombination verschiedener Datensätze Reidentifikationen möglich sind und personenbezogene Informationen wiederhergestellt werden können (Ji u. a. 2014). Auch dadurch gibt es in der e-Science zunehmende Bestrebungen weniger Daten mit anderen zu Teilen oder Datensätze komplett zurückzuhalten (Tene und Polonetsky 2012).

Für ein zweites Beispiel kann *Phase II* des maritimen Testfelddatenmanagements (siehe Abschnitt 2.2.3) betrachtet werden. Hier werden interne Daten aus zu testenden Systemen stärker einbezogen. So könnte sich also eine Situation ergeben, in der ein zu testendes System aus mehreren Teilsystemen verschiedener Hersteller besteht oder mehrere Hersteller das Testfeld gleichzeitig verwenden, wie in UC2. Möglicherweise entwickelt einer der Hersteller durch ihm verfügbare, interne Datenquellen ein Testszenario, welches eine bestimmte Funktionalität seines Teilsystems prüfen soll. Dafür werden aber interne Parameter benötigt, die nur durch eine Teilmenge der anderen Teilsysteme verarbeitet werden sollen, weil sie Geschäftsgeheimnisse enthalten. Je nach Anwendungsfall ist es nun notwendig diese Parameter zu anonymisieren oder durch weitere Maßnahmen nur den vorgesehenen Empfängern zur Verfügung zu stellen. In diesem Hinblick können sich weitere Szenarien ergebenen, in denen Hersteller nur Daten beziehen wollen, die Verarbeitung aber komplett offline bzw. lokal beim Hersteller erfolgen muss. Hier könnten also auch Methoden aus dem Bereich des Privacy-Preserving Data Minings (vgl. Verykios u. a. 2004) relevant sein.

Neben den gesetzlichen Rahmenbedingungen zum Speichern und Aufzeichnen von Daten gibt es noch weitere Standards, die sich auf die verwendeten Datenmodelle in maritimen Systemen beziehen wie etwa die IHO S-100 Standardfamilie (Ward u. a. 2008). Zudem existieren Standards und Richtlinien, die die Übertragung dieser Daten betreffen (z.B. im e-Navigation Bereich) wie IALA G1128 (IALA 2021), G1157 (IALA 2020) oder IEC 63173-2 (IEC TC80 2022) und Standards und Regularien, die das gesamte Systemverhalten betreffen wie IMO SOLAS (IMO 1974) und für die Sicherheit generell befolgt werden müssen. Bezüglich des Testfeldes beziehen sich diese Standards zwar hauptsächlich auf ein zu testendes System, sollten aber trotzdem nicht im Widerspruch zur Architektur eines FEDMS stehen. So sollte es etwa kein Problem darstellen S-100 Daten über das FEDMS zu übertragen.

2.5 Verwendung von maritimen Daten in der datengetriebenen Forschung und Entwicklung

Wie in Abschnitt 2.3 herausgestellt, ergeben sich in dezentral organisierten Datenökosystemen die Probleme der Auffindbarkeit und Interoperabilität der Daten, welche mit der Integration der einzelnen Datenquellen in das Datenökosystem einhergeht. In Data Spaces wird dieses Problem durch inkrementelles Verbessern der Datenanbindungen gelöst. Nach Lenzerini (2002, 233) ist Datenintegration „*(…) das Problem Daten aus verschiedenen Quellen zu kombinieren und dem Nutzer eine einheitliche Sicht auf diese Daten zur Verfügung zu stellen.*“ Dafür müssen die Datenquellen bekannt sein, sodass Anfragen an ein übergeordnetes System in das Quellschema der einzelnen Datenquellen übersetzt werden können (Lenzerini 2002). Es ist also notwendig Wissen über die Eigenschaften von Datenquellen eines Datenökosystems zu haben, damit eine Datenintegration neuer Quellen in das Datenökosystem sinnvoll möglich ist. Im Folgenden wird zunächst analysiert, welche Daten in der maritimen, datengetriebenen Forschung überhaupt relevant sind und wofür sie eingesetzt werden. Gemeinsamkeiten können später ausgenutzt werden, um auch nicht vollständig bekannte Datenquellen mittels eines Forschungs- und Entwicklungsdatenmanagementkonzeptes für maritime Testfelddaten besser integrieren zu können. Aufgrund der Tatsache, dass ein Testfeldbetreiber möglicherweise eine größere Menge an Datenquellen bereitstellt, ist es hier ebenfalls denkbar, dass ein einheitliches Datenmodell dafür verwendet wird (vgl. Abschnitt 2.3.2).

Um das Feld der maritimen, datengetriebenen Forschung und Entwicklung näher zu untersuchen, soll eine Analyse von wissenschaftlicher Literatur dieses Feldes durchgeführt werden. Hierzu gibt es bereits mehrere Studien, die Literatur im Bereich der maritimen Datenverarbeitung analysieren:

Mirović, Miličević, und Obradović (2018) konzentrieren ihre Literaturanalyse auf Arbeiten, die sich mit Anwendungen von Big Data in der maritimen Industrie beschäftigen und legen dabei einen speziellen Fokus auf die Optimierung von Prozessen der Logistik, die Verbesserung von Sicherheit und Effizienz und die Herausforderungen von Systemen, die Big Data nutzen. Sanchez-Gonzalez u. a. (2019) untersuchen die Aspekte der Digitalisierung in der maritimen Industrie und unterscheiden hier zwischen Schiffsdesign und -bau, Schifffahrt und Häfen. Eine dritte, umfangreichere Studie in diesem Bereich wurde zum Thema „Big data and artificial intelligence in the maritime industry” im Jahr 2020 von Munim u. a. durchgeführt. Sie untersuchten dabei Arbeiten im zeitlichen Bereich von 1995 bis 2019. Zusätzlich zu den Themenbereichen „Big Data“ und „Artificial Intelligence“ wurden auch Arbeiten aus den Bereichen „Business Intelligence“ und „Data Analytics“ untersucht. Allerdings wurden dabei lediglich die Anwendungsbereiche der Arbeiten und nicht ihre eigentliche Datengrundlage betrachtet. Für eine Recherche zur grundlegenden Nutzung von maritimen Daten im Bereich der

datengetriebenen Forschung wurde daher eine Rückwärtssuche durchgeführt: Dazu wurden 70 von Munim u. a. zitierte Quellen untersucht, von denen 54 einen direkten thematischen Bezug hatten. 8 dieser 54 Arbeiten waren ebenfalls Literaturreviews, weitere 8 hatten zwar einen thematischen Bezug, konnten aber nicht der datengetriebenen Forschung zugeordnet werden, weil eine Datengrundlage fehlte. Insgesamt konnten 38 Arbeiten identifiziert werden, die als Grundlage für die weitere Analyse in Frage kamen. Eine detaillierte Auflistung der verwendeten Quellen und der Untersuchungsergebnisse findet sich im Anhang 8B. Bei der Analyse der Arbeiten wurden folgende Fragen berücksichtigt:

1. Welche Arten von maritimen Daten wurden in der jeweiligen Arbeit verwendet?
2. In welchem Prozessschritt der Forschung wurden die Daten verwendet?

Zu Frage 1:

Zunächst einmal fiel auf, dass Daten des **Automatic Identification System** mit Abstand (in 31,6% der untersuchten Arbeiten) am häufigsten und oft in Kombination mit weiteren Daten (z.B. weiterführende Schiffsinformationen) verwendet werden. Danach folgen **Wetterdaten** (21,1%), Daten im Zusammenhang mit **Häfen** (18,4%) und **Motor- und Treibstoffdaten** (15,8%). Etwas weniger häufig werden Daten in Bezug auf **Handel und Wirtschaftlichkeit** (13,2%) und **bildgebende Daten** (10,5%) verwendet. Eher selten (5,3%) kommen **Kartendaten** zum Einsatz. Sonstige Daten, die im Zusammenhang mit der datengetriebenen maritimen Forschung identifiziert werden konnten, sind Daten zur Wasserqualität, Daten zu Schiffsantrieben, Navigationswarnungen, Twitter-Posts, textuelle Berichte, Daten aus verschiedenen Arten von Interviews, Begegnungssituationen zwischen Schiffen, Anlegeplanungsdaten, Analyseresultate von Meerwasser und Daten zur Rekrutierung von Besatzungsmitgliedern. Zudem fiel auf, dass größtenteils reale Daten für die Forschungsarbeiten verwendet wurden. Nur drei der 38 Arbeiten nutzten Daten aus Simulationen.

Weiter untersuchten Sidibé und Shu (2017) 19 Arbeiten, die sich ausschließlich mit der datengetriebenen Forschung mit AIS-Daten beschäftigen, was die Relevanz des AIS für den maritimen Anwendungsbereich erneut unterstreicht. Das Automatic Identification System (AIS) ermöglicht es, Nachrichten über den Status und die Beschaffenheit von Objekten aus dem maritimen Umfeld über das VHF-Frequenzband drahtlos zu übermitteln. Bei den Sendern und Empfängern handelt es sich meist um Schiffe, die Informationen über aktuelle Reisedaten, Positionierung und Geschwindigkeit austauschen (Last u. a. 2014). Zudem können mithilfe von weiteren AIS-Nachrichtentypen statische Informationen wie etwa die Länge, Breite und der Tiefgang des Schiffes übermittelt werden (Last u. a. 2014). Aufgrund ihrer hohen Relevanz werden AIS-Daten im restlichen Teil der Arbeit als beispielhafte Datengrundlage angesehen, für die Datentyp-spezifische Architekturelemente stellvertretend entwickelt werden.

Zu Frage 2:

Der Einsatz der Daten in den einzelnen Arbeiten ist vielfältig: So werden die Daten beispielsweise genutzt, um einfache Kennzahlen zu berechnen, Modelle zu trainieren oder neu entwickelte Verfahren auf einer Realdatenbasis zu testen. Insgesamt lassen sich die Verwendungszwecke grob in die Kategorien **Datenanalyse** (36,8% der Arbeiten), **Trainingsdaten** (ebenfalls 36,8%), **Evaluation** (26,3%) und **Datenarchitektur** (10,5%) aufteilen (Mehrfachzuordnung möglich). In der Datenanalyse werden die Daten in einem klassischen Prozess (vgl. KDD-Prozess) aufbereitet, analysiert und so neues Wissen mit Hilfe der vorhandenen Daten generiert. Trainingsdaten werden speziell für das Training von Machine-Learning Modellen eingesetzt. Weiterhin nutzen einige Autoren die Daten lediglich für die Evaluation eines neuen Konzeptes oder Modells, um die Anwendbarkeit oder Nutzbarkeit zu demonstrieren. Schließlich gibt es auch noch Arbeiten, die die Daten als Grundlage für den Entwurf neuer Datenarchitekturen nutzen, die speziell auf die genutzten Datenarten ausgerichtet sind. Insgesamt lassen sich erneut Zusammenhänge zu den drei Phasen des Testfelddatenmanagements aus Abschnitt 2.2.3 feststellen: Datenanalyse und Trainingsdaten kommen in *Phase I* und *II* zum Einsatz, während Daten zur Evaluation neuer Konzepte schwerpunktmäßig in *Phase II* verwendet werden.

2.6 Anforderungen an ein FEDMS im maritimen Testfeldkontext

Im folgenden Abschnitt werden Anforderungen an ein FEDMS (im Folgenden auch als „**das System**“ bezeichnet) für dezentrale, maritime Daten hergeleitet. Diese basieren auf den Inhalten der vorangegangenen Grundlagenkapitel und den definierten Zielen der Arbeit. In den folgenden Kapiteln werden sie genutzt, um die verwandten Arbeiten näher zu untersuchen und dienen als Grundlage für den Entwurf des Systems.

2.6.1 Annahmen und Vorgehen

Wie in Abschnitt 1.3 beschrieben, wird davon ausgegangen, dass die betrachteten Forschungs- und Entwicklungsdaten in einem maritimen, schiffszentrierten Testfeld vorliegen. Zudem lässt die Anordnung der Stakeholder in einem serviceorientierten maritimen Testfeld eine Zentralisierung der Daten nicht zu (vgl. Abschnitt 2.2.3 bzw. 2.3.4). Aufgrund dieser Voraussetzungen und der Charakterisierung von dezentral organisierten Datenökosystem und den diskutierten Gegebenheiten im Testfeldkontext wird von einem Data Space als grundlegender Struktur für das zu entwerfende System ausgegangen. Außerdem ist davon auszugehen, dass in einem existierenden Testfeld bereits Prozesse und Tools etabliert sind, um die Forschungs- und Entwicklungsdaten in den drei Phasen des Testfelddatenmanagements zu verarbeiten. Das System

soll also nicht dazu dienen bestehende Datenverarbeitungsprozesse durch neue, systeminterne Prozesse zu ersetzen, sondern soll bestehende Prozesse integrieren, überwachen und im dezentralen Kontext des Testfeld Data Spaces unterstützen. Der Entwurf eines allgemeinen Systems zur zentralen Datenanalyse im Rahmen der Data Science oder der Verifikation und Validierung in maritimen Testfeldern ist folglich nicht Teil dieser Arbeit. Um die Anforderungen abzuleiten, werden im Folgenden die allgemeinen Use-Cases der identifizierten Stakeholder für ein FEDMS diskutiert (siehe Abbildung 12).

Zunächst ist es im Interesse des Testfeldbetreibers ein FEDMS bereitzustellen, was den Datenaustausch zwischen Testfeldinfrastruktur und Testfeldnutzern, -beobachtern und -dienstleistern ermöglicht. Diese sollen sowohl auf die testfeldeigene Infrastruktur zugreifen können als auch externe Daten von Data-as-a-Service Providern nutzen können, und untereinander Daten austauschen können. Als Betreiber einer Infrastruktur zum Austausch von Daten innerhalb des Testfeldes tritt der Testfeldbetreiber als Datentreuhänder auf. Je nach auftretenden Interessenkonflikten ist es aber auch denkbar, dass weitere, externe Datentreuhänder Komponenten bereitstellen, die zum sicheren und neutralen Datenaustausch zwischen den Stakeholdern im Testfeld genutzt werden können. Der Testfeldbetreiber hat weiterhin das Ziel in allen drei Phasen des Testfelddatenmanagements grundlegende Daten des Testfeldes (Sensordaten, Statische Informationen, historische Aufzeichnungen etc.) bereitzustellen. Diese werden durch externe Datendienstleister ergänzt. Die Rolle der Testfeldnutzer lässt sich anhand der ersten beiden Phasen unterteilen: In *Phase I* sind maßgeblich Data Scientists, Data Engineers oder allgemeinere Analysten aktiv, um auf der Datengrundlage des Testfeldes Modelle zu entwickeln und Zusammenhänge im Testfeld analysieren zu können. In *Phase II* des Testfelddatenmanagements spielen der System Engineer und der Data Management Engineer als Testfeldnutzer eine primäre Rolle. So agiert der System Engineer größtenteils als Datenkonsument, der Daten aus dem Testfeld abruft und in das SuT einspeist. Diese Tätigkeit wird durch den Data Management Engineer ergänzt, der zunächst für die Bereitstellung von Interfaces zum Austausch von Daten mit anderen Systemen zuständig ist, aber auch an der Verwaltung von Datenflüssen innerhalb des Testfeldes interessiert ist, um später das Systemverhalten während der Testdurchläufe analysieren zu können. Es ist an dieser Stelle denkbar, dass die Aufgaben des System Engineers und des Data Management Engineers von einer einzelnen Person, oder gemeinsam durch ein Team von Personen ausgeübt werden. Speziell diese Stakeholder können einen wirtschaftlichen Hintergrund haben und daran interessiert sein, die Kontrolle über ausgetauschte Daten zu behalten. Letztlich haben Testfeldbeobachter in *Phase III* des Testfelddatenmanagements ein Interesse daran ausgewählte Teilmengen der Testfelddaten zu beziehen und etwa auszuwerten oder zu visualisieren. Denkbare Akteure sind hier Gutachter in einem Systemzertifizierungsverfahren oder Zuschauer und Veranstalter einer Demonstration.

Insbesondere den Zertifizierern ist es wichtig Datenprovenienzinformationen sicher nachvollziehen zu können, damit Zertifizierungen später nicht auf der Grundlage einer falschen Datengrundlage zurückgenommen werden müssen. Dies gilt auch für andere Testfeldnutzer, die die Nachvollziehbarkeit und Reproduzierbarkeit von Datenverarbeitungsketten im Testfeld sicherstellen müssen.

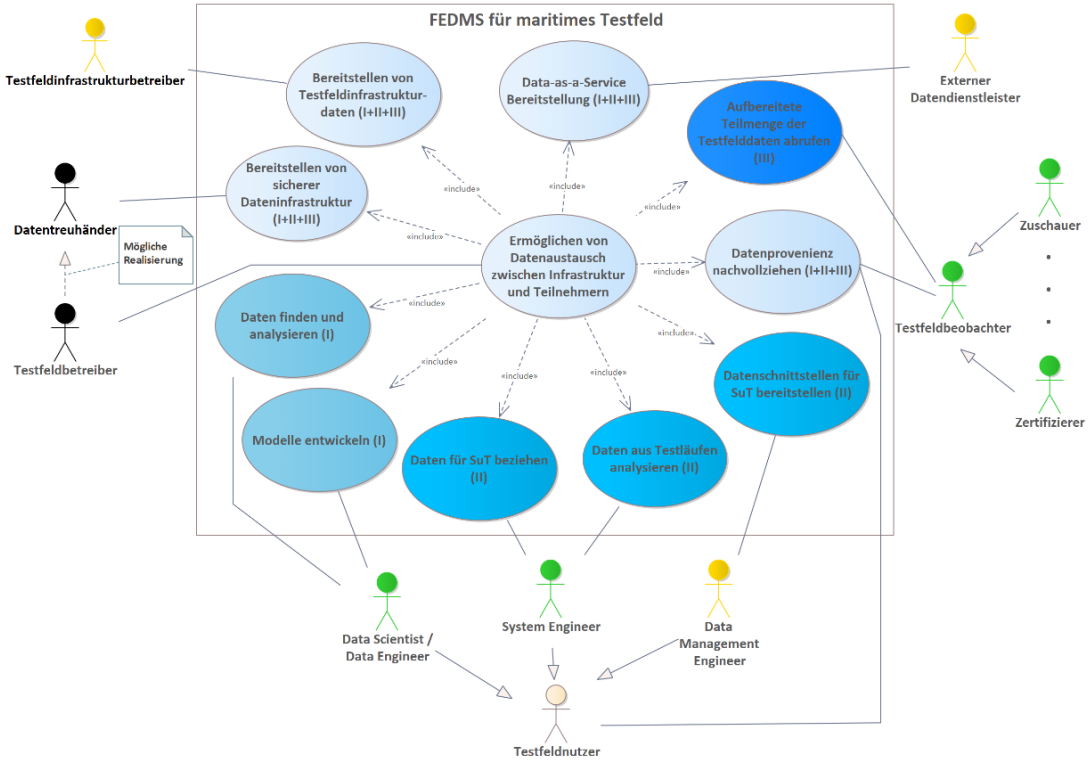

Abbildung 12: Use-Cases im Testfelddatenmanagement.

Aus den Use-Cases ergeben sich genauere Anforderungen, die im Folgenden mithilfe von Erkenntnissen der wissenschaftlichen Literatur hergeleitet werden sollen. Dazu werden die Anforderungen nach Shams-ul-Arif und Gahyyur (2009) in 5 Bereiche eingeteilt (siehe Abbildung 13): grundlegende Voraussetzungen an das System (architekturelle Kernanforderungen), Bedürfnisse der Stakeholder (Anforderungen an bzgl. des Teilens und Verarbeitens von Daten) in Kombination mit organisatorischen Regeln und Standards (Anforderungen bzgl. Sicherheit und Vertrauensbildung) und Domäneninformationen (spezielle Anforderungen an maritime Testfelddaten). Diese entsprechen ebenfalls den Zielen dieser Arbeit.

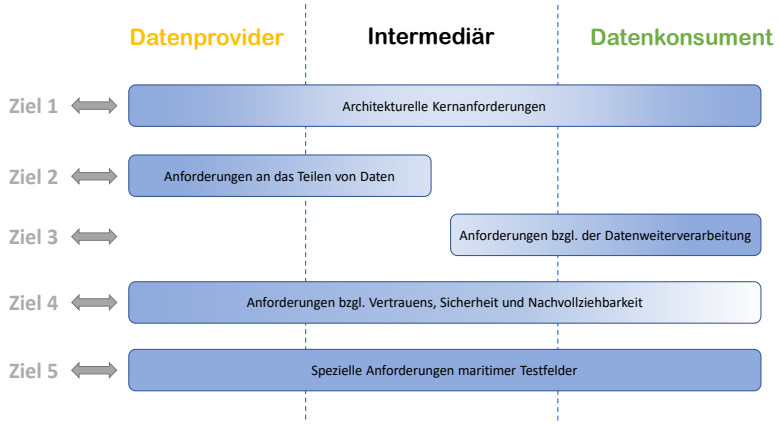

Abbildung 13: Anforderungsbereiche für den Entwurf des FEDMS unter Berücksichtigung der drei Stakeholdergruppen.

2.6.2 Architekturelle Kernanforderungen

Die physische Verteilung von Daten über mehrere Datenquellen im Testfeld setzt einige grundlegende Architekturelemente voraus, die für ein FEDMS vorhanden seien müssen. Gleichzeitig sollten die eingeschränkten FAIR-Prinzipien (siehe Abschnitt 2.2.2) für das Management von Forschungsdaten in einem dezentralen Kontext nicht vernachlässigt werden und werden in den folgenden architekturellen Kernanforderungen berücksichtigt.

A1 – Integration dezentraler Testfelddatenquellen. Die Aufgabe eines Forschungsdatenmanagementsystems ist es, eine Vielzahl von Datenquellen zu verwalten und den Zugriff auf einzelne Daten zu ermöglichen (Wilkinson u. a. 2016). Im Szenario der dezentral organisierten Datenquellen eines Testfeldes muss dieses System in der Lage sein, die Datenquellen zu erkennen und, unabhängig von den verwendeten Technologien, in seine Architektur zu integrieren (Chung und Liao 2008). Diese Funktionalität erfüllt eine zentrale Eigenschaft einer Data Space Support Platform, wie sie von Franklin, Halevy und Maier (Franklin, Halevy, und Maier 2005) definiert wurde (vgl. Abschnitt 2.3.5), und bildet die Grundlage für den zentralen Zugriff und die Nutzung von dezentralen Datenquellen in einem Data Space. In einem Testfeld-FEDMS müssen dezentrale Datenquellen der grundlegenden Testfeldinfrastruktur, wie AIS-Daten, Wetterdaten oder bildgebende Daten integriert werden können. Diese werden ergänzt durch Daten aus SuTs und Daten von externen DaaS-Anbietern.

A2 – Verarbeitung von Datenanfragen. Ein wesentlicher Teil eines dezentralen Datenmanagementsystems liegt in seiner Fähigkeit, eine zentrale Schnittstelle für die Suche oder Auswahl von verteilten Daten bereitzustellen (Zhong, Liu, und Chen 2008). Ein FEDMS muss dabei ein Gleichgewicht zwischen der Expressivität der Suchanfragen und der Benutzerfreundlichkeit der bereitgestellten Suchschnittstellen bieten (Freitas, O'Riáin, und Curry

2020), damit z.B. sowohl System Engineers als auch einfache Beobachter eines Testfeldes Daten abfragen können. Besonders in *Phase I* des Testfelddatenmanagements ist es relevant aus einer großen Menge von Daten passende Daten abrufen zu können, um bspw. Verkehrsprädiktionsmodelle zu entwickeln oder Hypothesen über Wettereinflüsse im Testfeld zu untersuchen. In *Phase II* müssen zur Laufzeit integrierte Datenquellen aus SuTs ebenfalls gezielt abfragbar sein, damit Datenströme ausgetauscht und dokumentiert werden können. Schließlich müssen selektierbare Teilmengen der Daten für die Analyse und Demonstration von Systemen in *Phase III* bereitgestellt werden können.

A3 – Zugriffsverwaltung für Datenquellen. Ein Data Space ist besonders durch die Heterogenität seiner Daten gekennzeichnet (Curry 2020). Neben dieser Heterogenität sind in einem Data Space auch unterschiedlichste Konzepte zum Austausch von Daten vorhanden und verschiedene Organisationsstrukturen zu finden (Cindy u. a. 2009). So ist es möglich, dass einige Daten im Testfeld komplett frei verfügbar sind (z.B. Wetterdaten aus öffentlichen Quellen), während andere eine strikte Regelung des Zugangs zu ihren Daten definieren (Daten aus SuTs). Neben den Zugriffsrechten unterscheiden sich in einem Data Space auch die Schnittstellen, über die auf die Daten zugegriffen werden kann (Curry 2020). In Testfeldern kommen hier verschiedene, häufig unabhängig voneinander entwickelte Systeme zusammen, zwischen denen ein Datenaustausch stattfinden soll. Ein FEDMS muss es ermöglichen, diese unterschiedlichen Anforderungen abzubilden. Außerdem muss es ein modulares Zugriffsmanagement bieten und auf die verschiedenen Zugriffskonzepte anpassbar sein. Dies ist besonders relevant vor dem Hintergrund schützenswerter Daten, die etwa von Testfeldnutzern mit Wettbewerbsdruck genutzt werden.

A4 – Metadatenmanagement für Datenquellen. Metadaten werden für eine Vielzahl von Datenfindungs- und Verarbeitungsaufgaben verwendet (Rajasekar und Moore 2001). Qin, Ball, und Greenberg (2012) diskutieren funktionale und architektonische Anforderungen an wissenschaftliche Metadaten im Kontext des Datenmanagements: Neben dem reinen Zugriff auf die dezentralen Datenquellen sollte ein FEDMS die Möglichkeit bieten, weitere Metadaten zu den Datenquellen aufzunehmen, und zu verwalten. Dies ermöglicht eine bessere Klassifizierung der Datenquellen und kann sich positiv auf den datengetriebenen Forschungsprozess auswirken, indem Daten schneller aufgefunden werden können. Besonders relevant sind die Metadaten im Testfelddatenmanagement für das effizientere Auffinden und Suchen (vgl. W. Zhang u. a. 2019) von Datenquellen in *Phase I*, aber auch als Element der Dokumentation von Testläufen in den *Phasen II* und *III*. Hierfür können äquivalente Anforderungen an das dezentrale Metadatenmanagement, wie in A1 und A2 für das Datenmanagement beschrieben, angenommen werden (vgl. (Gaspar, Braga, und Campos 2011), (F. Wang u. a. 2006)).

2.6.3 Anforderungen an das Teilen von Daten

Um ein FEDMS in einem dezentralen Testfeldkontext umzusetzen, müssen gegenüber konventionellen, zentralen Systemen die Interessen des Testfeldbetreiber und der Testfeldnutzer stärker berücksichtigt werden, die ihre Daten im Testfeld bereitstellen. Bei ihnen besteht oft ein klares Interesse daran ihren Datenbestand nicht vollständig in die Obhut eines dritten Systems zu übergeben, welches diese dann unabhängig verwaltet. Als Vermittler zwischen Datenkonsument und Datenprovider stellt das FEDMS einen Intermediär dar und muss hierbei ggf. konfliktäre Interessen dieser beiden Stakeholder berücksichtigen. Die folgenden Anforderungen basieren auf datenschutzrechtlichen Grundlagen und Design-Prinzipien (vgl. European Union Agency for Cybersecurity u. a. 2015) und sollen die Interessen der Datenprovider im Testfeld abbilden.

A5 – Flexible Zwischendatenspeicherung. Daten oder Metadaten, die innerhalb der Grenzen eines FEDMS als Teil des datengetriebenen Forschungs- und Entwicklungsprozesses gespeichert werden, können sensible Informationen aus einzelnen Datenquellen des Testfeldes und den SuTs enthalten. Daher sollte es für die Beteiligten möglich sein, zu konfigurieren, wo, wie und wann Zwischenergebnisse im datengetriebenen Forschungs- und Entwicklungsprozess innerhalb eines solchen Systems gespeichert und verarbeitet werden sollen (Zichichi u. a. 2020). So kann sichergestellt werden, dass konkurrierende Nutzer eines Testfeldes sensible Daten sicher austauschen können, ohne die Kontrolle über diese zu verlieren. Konkret ist das durch die flexible Wahl eines Datentreuhänders oder die Möglichkeit selbst als ein solcher zu agieren zu realisieren (vgl. Abschnitt 2.3.4). Gegebenenfalls müssen auch zusätzliche Informationen zur Verwendung der Daten übermittelt werden können: So könnten etwa Eigentums- und Nutzungsbedingungen der Daten relevant für die Weiterverwendung sein (Bader u. a. 2020), sodass einwandfrei festgestellt werden kann, wie die Daten weiterverwendet werden dürfen.

A6 – Datensouveränität bei der Integration. Um Daten für ein FEDMS verfügbar zu machen, müssen die Datenprovider im Testfeld ihre Datenquellen mit dem System verbinden. Ein solcher Prozess macht es (je nach verwendeter Technologie) nicht immer möglich, Zugriffsanfragen auf die Datenquelle feingranular zu unterscheiden. Der Kern des Datensouveränitätsprinzips (vgl. Abschnitt 2.3.3) besteht jedoch darin, die Kontrolle über die gemeinsam genutzten Daten hinsichtlich Identität, Zugriff und Nutzung beim Datenprovider zu belassen (Munoz-Arcentales u. a. 2019). Um Transparenz zwischen den Stakeholdern im Testfeld und dem System herzustellen und von den verwendeten Technologien zu abstrahieren, muss das System ein entkoppeltes Artefakt bereitstellen, das eine Datenquelle mit dem System verbindet und vollständig vom Datenanbieter kontrolliert und beobachtet werden kann (vgl. Zrenner u. a. 2019). Auf diese Weise kann die Souveränität der Testfeldnutzer gewahrt werden. In der Anwendung könnte das Artefakt ein Open-Source Connector sein, der die Verbindung zwischen der eigentlichen Datenquelle, z.B. einer historischen AIS-Datenquelle, und dem System auf Seiten

des Datenanbieters herstellen kann. Zusätzlich ist mit einer dynamischen und heterogenen Menge von Datenquellen zu rechnen, da beliebige Systeme durch Stakeholder an das FEDMS angebunden werden können. Daher sollte das System stets eine dynamische Verwaltung der Connectoren zur Laufzeit unterstützen.

A7 – Anonymisierung und Datenschutz. Aufgrund der in Abschnitt 2.4 diskutierten Gegebenheiten kann es Sinn ergeben bestimmte Daten vor dem Austausch zu anonymisieren. Die Anonymisierung (z.B. von AIS-Daten) sollte auf Seiten des Datenproviders möglich sein (Salheddine und Benslimane 2014), da die Verarbeitung oder Zwischenspeicherung der noch nicht anonymisierten Daten innerhalb des Systems gegen rechtliche Rahmenbedingungen verstoßen könnte. Auf diese Weise können anonymisierte Forschungs- und Entwicklungsdaten unter Gewährleistung des Datenschutzes für die Modellbildung oder in einem SuT genutzt werden. Umgekehrt muss durch den Datenprovider darauf geachtet werden, dass Dritte nicht zusätzliche Datenquellen nutzen können, um eine De-Anonymisierung durchzuführen (vgl. Falorsi, Liseo, und Scannapieco 2019).

A8 – Rechtemanagement und Rollenmodell. In modernen Datenökosystemen kommen zahlreiche Teilnehmer mit unterschiedlichen Interessen, Rollen und Zugriffsrechten zusammen (Oliveira, Barros Lima, und Farias Lóscio 2019), (Demchenko, De Laat, und Membrey 2014). Wie in den Abschnitten 2.2.3 und 2.3.4 diskutiert, ist dies in maritimen Testfeldern ebenso der Fall. Um in einer solchen Anordnung nicht den Überblick zu verlieren, muss klar definiert werden, welche Teilnehmer welche Ziele verfolgen (Xu u. a. 2014), (Oliveira u. a. 2017) und auf welche Daten sie zugreifen (Munoz-Arcentales u. a. 2019). Auf diese Weise können Interessenkonflikte, z. B. zwischen zwei unabhängigen Entwicklern eines SuTs, identifiziert werden. Dazu sollte das Testfeld FEDMS ein klares Rollenmodell definieren und dieses vor allem für die Verwaltung des Zugriffs auf einzelne Datenquellen verwenden (Cuno u. a. 2019).

2.6.4 Anforderungen bezüglich der Datenweiterverarbeitung

Die folgenden Anforderungen sollen dazu beitragen, moderne Datenverarbeitungsprozesse in das Testfeld FEDMS zu integrieren und die Daten so bereitzustellen, dass bereits etablierte Prozesse und Methoden effizient darauf arbeiten können. Dies wird insbesondere von den Konsumenten der Daten gefordert.

A9 – Integration von Datenverarbeitungsprozessen. Diverse Prozesse im Datenökosystem Testfeld finden dezentral statt und werden durch die einzelnen Stakeholder verwaltet. Um einen nahtlosen Übergang beim Datenaustausch zu ermöglichen, muss ein FEDMS den Prozess der Datenverarbeitung im Testfeld in seiner Gesamtheit unterstützen. Neben der Bereitstellung von Datensätzen aus den Datenquellen des Testfeldes, gehört auch die Begleitung des gesamten

zwischengelagerten Prozesses (vgl. Begriff des Scientific Data Lifecycle Management von Demchenko u. a. (2013)) dazu. Nicht nur allgemeine Werkzeuge müssen in den Datenmanipulationsprozess integriert werden, sondern auch die speziellen Verarbeitungsschritte, die *in Phase II* in SuTs zum Einsatz kommen oder jene, die in *Phase III* dazu verwendet werden die Daten für eine Analyse oder visuelle Darstellung zu filtern und aufzubereiten. Dazu muss das System Schnittstellen bereitstellen, die es ermöglichen, die Datenverarbeitungsschritte an externe Werkzeuge und (Teil-)Systeme auszulagern, sodass die Dezentralität erhalten werden kann. Mehrere Prozessmodelle, auch speziell für Big-Data-Verarbeitung, werden derzeit in Data-Science-Engineering-Prozessen eingesetzt und können dazu als Referenz verwendet werden (Volk u. a. 2020), (Aho u. a. 2020).

A10 – Unterstützung für Live-Datenverarbeitung. Der zentrale Anwendungsfall eines Testfelds ist das Durchführen von Tests. Um während der Testdurchführung den Stakeholdern ein direktes Feedback zu liefern, ist es besonders bei komplexeren Aufbauten notwendig Daten aus Tests als Datenstrom zwischen verschiedenen Komponenten auszutauschen und zu analysieren. Hierzu eignen sich insbesondere nachrichtenorientiere Middlewares bzw. Nachrichtenbroker (Chowdhury u. a. 2018). Weiterhin sollte es möglich sein bestimmte Live-Daten in ein SuT einspeisen zu können. Hierbei kann es sich insbesondere um Messungen von Testfeldsensorik oder die Ausgabedaten von (Simulations-)Modellen handeln (Swanson u. a. 2013). Auch für das Überwachen interner SuT-Zustände, der ODD, oder der Einhaltung bestimmter Contracts ist eine Live-Datenverarbeitung in *Phase II* und *Phase III* notwendig (vgl. Abschnitt 2.2.3).

A11 – Szenario-orientiertes Datenmanagement. Assistenzsysteme, autonomisierte Schiffssteuerungen oder überwachende Systeme sind typische Beispiele für Systeme, die in einem maritimen Testfeld untersucht werden sollen. Alle diese Systeme agieren auf einer dynamischen und größtenteils sensorbasierten Datengrundlage, welche zur Laufzeit durch komplexe Prozesse verarbeitet wird. Beim Testen dieser datenbasierten Systeme in Feldexperimenten im Rahmen der V+V entstehen diverse Verarbeitungszustände aus den ursprünglichen Daten, die kontinuierlich zwischen verschiedenen Systemen unter der Kontrolle verschiedener Datenprovider und -konsumenten ausgetauscht werden (vgl. Abschnitt 2.2.3). Gerade in komplexeren Workflows ist es später oft schwer nachzuvollziehen, wo eventuelle Fehler aufgetreten sind. Diese Aufgabe manuell zu erledigen ist sehr anspruchsvoll und zeitintensiv. Da beim Testen von komplexen maritimen Assistenzsystem oft das Szenario-basierte Testen zum Einsatz kommt (siehe Abschnitt 2.2.3), bietet es sich an in Feldexperimenten anfallende Daten neben einer Modellierung mittels Workflows auch Szenario-orientiert verwalten zu können (vgl. Klitzke u. a. 2019). So können große Blöcke von Testdaten in einzelne Testszenarien aufgeteilt, und effizienter ausgewertet werden.

2.6.5 Anforderungen bezüglich Vertrauens, Sicherheit und Nachvollziehbarkeit

In einem von mehreren Teilnehmern genutzten Testfeld, in dem Daten aus dezentral organisierten Datenquellen ausgetauscht werden kann es zu Vertrauensproblemen zwischen den Testfeldnutzern kommen, was eine generelle Herausforderung beim Datenhandel und -austausch darstellt (F. Liang u. a. 2018). Darüber hinaus macht die rein beschreibende Dokumentation von Verarbeitungsschritten den Datenverarbeitungsprozess zwar nachvollziehbar, aber nicht überprüfbar. Dies ist etwa bei der Zertifizierung von Systemen oder dem Nachweisen von wissenschaftlichen Datenverarbeitungsprozessen in Veröffentlichungen notwendig. Die folgenden Anforderungen zielen darauf ab, eine sichere und überprüfbare Dokumentation der Datenmanipulation zu ermöglichen, während gleichzeitig die konträren Interessen der Beteiligten berücksichtigt werden.

A12 – Dokumentation von Zwischenergebnissen. Eine angemessene Dokumentation der Datenverarbeitungsschritte (Datenprovenienz) ermöglicht es, einen Datenverarbeitungsprozess nachzuvollziehen und ggf. zu reproduzieren (Simmhan, Plale, und Gannon 2005) (vgl. Abschnitt 2.2.2). Allerdings dürfen Rohdaten oder Zwischenergebnisse im Datenmanipulationsprozess nicht vernachlässigt werden, um den genauen Nachweis zu erbringen, wie die Daten verarbeitet wurden. Die Speicherung und Bereitstellung genau dieser Zwischenergebnisse und ihrer Provenienzinformationen kann in einem kollaborativen Forschungs- und Entwicklungsprozess im Testfeld, an dem mehrere Forscher, Entwickler und unterschiedliche Methoden beteiligt sind, von großem Nutzen sein (Rousidis u. a. 2014). Spätestens beim Zusammenspiel mehrerer Systeme in *Phase II* bzw. dem System-of-Systems Engineering ist eine akkurate Dokumentation der Datenflüsse ein hilfreiches Werkzeug, um den Entwicklungsprozess zu unterstützen und nachzuvollziehen welche Daten zwischen welchen Komponenten ausgetauscht wurden.

A13 – Sicheres Tracking des Verarbeitungsprozesses. Die Nachvollziehbarkeit von Datenverarbeitungsprozessen kann deutlich erhöht werden, indem deren Zwischenergebnisse gespeichert und verfügbar gemacht werden (siehe A12). Berücksichtigt man jedoch die konfliktären und wirtschaftlichen Interessen der verschiedenen Teilnehmer im Data Space des Testfeldes, so könnte einer reinen Speicherung von Zwischenergebnissen ad hoc nicht mehr vertraut werden (vgl. Saad, Jalil, und Manaf 2014, 388f.). Gerade bei kritischen Datenaustauschprozessen wie Datendienstleistungen im Testfeld oder datengestützten Zertifizierungsprozessen, die vertraglich geregelt oder mit finanziellen Transaktionen verknüpft sind muss sichergestellt werden können, dass die Daten auch wirklich in vereinbarter Form ausgetauscht wurden (vgl. Tocco und Lafaye 2022). So könnte z.B. sicher dokumentiert werden, in welchem Rahmen ein Service zur Bereitstellung von hochauflösenden Kartendaten für das Testfeld eingesetzt wurde. Es kann weiterhin nicht automatisch davon ausgegangen werden, dass die tatsächlichen Datenverarbeitungsschritte vollständig mit den zur Verfügung gestellten und

damit nachweisbaren Zwischenergebnissen übereinstimmen. Daher sollte ein Datenmanagementsystem für dezentrale Daten den Prozess der Datenmanipulation überwachen und eine kryptographisch sichere Lösung zur Nachverfolgung des Prozesses bereitstellen (Bertino u. a. 2014).

A14 – Sicherheit dezentraler Testfelddatenquellen und der Datentransportwege. Der Entwurf eines Systems zum Datenaustausch erfordert immer auch allgemeine Überlegungen zu Sicherheitsaspekten. Beim Entwurf von Systemen zur Datenverwaltung sind diese Merkmale oft notwendig, um die verwalteten Daten und die Benutzer des Systems vor Angreifern zu schützen (Tocco und Lafaye 2022, 390). Allgemeine Sicherheitsmaßnahmen und -technologien wie z. B. Transportverschlüsselung sind jedoch architekturübergreifend und werden in dieser Arbeit nicht im Detail behandelt. Die Arbeit von Brost u. a. (2018) definiert formale Anforderungen an einen Data Space hinsichtlich der Sicherheit: Datendienste in einem Data Space sollten nur dann veröffentlicht werden, wenn sie von einem Betreiber freigegeben wurden, sollten zusätzlich streng voneinander isoliert sein und von einem vertrauenswürdigen Datenconnector zur Verfügung gestellt werden, außerdem sollte die Datennutzung reguliert, durchgesetzt und protokolliert werden. Die Zugriffskontrolle, die Isolation verschiedener Datenquellen, das Protokollieren und Überwachen von Datennutzung sind bereits durch die Anforderungen A3, A6, A12 und A13 abgedeckt. Zusätzlich muss ein FEDMS für Testfelder die Authentifizierung von Nutzern unterstützen und eine Möglichkeit bieten, neue Datendienste selektiv verfügbar zu machen. Hierdurch kann die Vertrauenswürdigkeit der verfügbaren Dienste sichergestellt werden. Sensible Daten dürfen ausschließlich über abgesicherte Kommunikationskanäle ausgetauscht werden.

2.6.6 Anforderungen der maritimen Domäne

Die domänenspezifischen Anforderungen ergeben sich im zu entwerfenden System grundsätzlich durch die Bereitstellung konkreter maritimer Datenquellen. Da sich diese Arbeit auf die Entwicklung eines FEDMS fokussiert und nicht auf die Integration einzelner Datenquellen, soll die Integration zweier maritimer Datenquellen stellvertretend für die Nutzung maritimer Datenquellen in den drei vorgestellten Phasen des Testfelddatenmanagements konzeptionell gezeigt werden. Hierbei fällt die Wahl auf AIS-Daten, da diese eine der relevantesten Datenquellen in der maritimen Domäne darstellen (vgl. Abschnitt 2.4). Um die Charakteristiken der drei Phasen abzudecken, werden neben historischen AIS-Daten auch Live-AIS Daten verwendet. Als weitere domänenspezifische Eigenschaft wird die schlechte Verfügbarkeit von Kommunikationskanälen mit hoher Bandbreite in vielen Gewässern berücksichtigt (Wei u. a.

2021), die es erfordert Daten möglichst effizient zu übermitteln. Zudem werden Anforderungen an Latenz und Durchsatz für die Verarbeitung maritimer Daten definiert.

A15 – Integration maritimer Datenquellen. Für die Nutzung im FEDMS sollen exemplarisch zwei Connectoren bereitgestellt werden, die jeweils eine historische, sowie eine Live-Datenquellen mit AIS-Daten anbinden. Die Connectoren sollen die Struktur von AIS-Daten vollständig abbilden und das feingranulare Abfragen von historischen und Live AIS-Daten ermöglichen. Die zu entwerfenden Connectoren sollten eine allgemeine Struktur aufweisen, die später für weitere Connectoren wiederverwendet werden kann. Es sollen dabei Datentechnologien unterstützt werden, die standardmäßig in Forschung und Industrie Anwendung finden.

A16 – Möglichkeit zur Bereitstellung von Vorverarbeitungsschritten bzgl. Datenqualität. Beim Arbeiten mit maritimen Sensordaten werden als einer der ersten Schritte die Rohdaten vorverarbeitet. Hierbei wird in der Regel das Ziel verfolgt, fehlerhafte oder fehlende Daten zu erkennen und diese zu korrigieren bzw. zu entfernen. Hintergrund ist hier die Aufwertung der Datenqualität, um später im Prozess aussagekräftigere Ergebnisse erzielen zu können. Dies könnte besonders im Fall von AIS- Daten oder Daten von ähnlich geringer Qualität (vgl. Harati-Mokhtari u. a. 2007) eine große Rolle spielen. Damit diese Prozesse nachvollziehbar durchgeführt werden können, ist es zunächst notwendig die Datenqualität durch Metriken zu messen, sodass evaluiert werden kann, ob ein Datensatz verwendbar ist, oder ob Vorverarbeitungsschritte die Datenqualität tatsächlich verbessern.

Durch vordefinierte Verarbeitungsschritte könnten mithilfe des FEDMS die Datenqualität gemessen und Datensätze bekannter Daten aufbereitet werden. Fehler in bekannten Attributen könnten erkannt und mit datenspezifischen Modellen korrigiert werden (vgl. z.B. Steidel u. a. 2019). Diese Aufgabe konzentriert sich aufgrund ihrer Komplexität auf bekannte Datenarten und ist damit in *Phase I* anzusiedeln. In Ausnahmefällen, in denen verwendete Datenquellen explizit bekannt sind, können die Verfahren auch in *Phase II* angewendet werden. Zur Überwachung dieser Verfahren sind Metriken notwendig, die objektive Verbesserungen der Datenqualität nachweisen können. Aufgrund der Kontextspezifizität solcher Verfahren (siehe Abschnitt 2.2.3) ist durch ein FEDMS eine Architektur bereitzustellen, die beliebige vordefinierte Verarbeitungsschritte unterstützt.

A17 – Speichereffiziente Übermittlung von Testfelddaten. Die maritime Domäne zeichnet sich durch die Besonderheit aus, dass Datenverbindungen auf hoher See meist nur über wenig Bandbreite verfügen oder mit hohen Kosten verbundenes Equipment benötigt wird, um Daten über Sattelitenverbindungen auszutauschen (Jo und Shim 2019). Diese Tatsache sollte bei der Serialisierung von Testfelddaten, die mithilfe des FEDMS (etwa in Feldexperimenten auf See) übertragen werden, berücksichtigt werden. Nutzer sollten die Möglichkeit haben beliebige

Datenstrukturen in ein speichereffizientes Serialisierungsformat zu überführen, welches durch das FEDMS genutzt werden kann.

A18 – Latenz- und Durchsatz bei Verarbeitung maritimer Daten. Big Data-Architekturen müssen große Mengen von Daten verarbeiten und sollten bestimmte Performanz-Kriterien erfüllen, damit dies in angemessener Weise möglich ist (Ivanov und Singhal 2018). Daher soll das FEDMS angemessene Latenzzeiten sowie einen entsprechenden Datendurchsatz für die Verarbeitung maritimer Daten aufweisen. Performanceergebnisse verschiedener Architekturen sind of schwer vergleichbar und unterscheiden sich je nach Use-Case (Cooper u. a. 2010). Daher dienen zwei Extrembeispiele als Orientierung: Für die Latenz wird eine sicherheitskritische Anwendung angenommen, bei der ein Schiff im Ernstfall von Land aus ferngesteuert werden muss. Yim und Park (2021) ermittelten, dass bei einer solchen Anwendung eine Latenz von minimal 0.2 Minuten kritisch sein kann. Im Automobilbereich argumentieren Lin u. a. (2018), dass ein autonomes System innerhalb von 100ms reagieren können sollte. Für den Durchsatz wird die Analyse einer großen Menge von historischen AIS-Daten betrachtet: Typische Datensätze, die für Data Mining genutzt werden, bewegen sich hier im Bereich von mehreren hundert Gigabyte (vgl. AbuAlhaol u. a. 2018). Das FEDMS sollte Daten mit einem angemessenen Durchsatz übermitteln können, sodass entsprechende Verarbeitungsprozesse effizient durch Data Scientists durchgeführt werden können.

3 Verwandte Arbeiten

In diesem Kapitel sollen verwandte Arbeiten vorgestellt und bezüglich der Ziele dieser Arbeit eingeordnet werden. Beginnend mit Abschnitt 3.1 wird die Klasse konventioneller Forschungsdatenmanagementsysteme untersucht. Weiterhin sollen in Abschnitt 3.2 aktuelle Arbeiten aus der Forschung betrachtet werden, die für die Verwaltung von Daten in Datenökosystem zum Einsatz kommen. Abschnitt 3.3 beschäftigt sich mit Testfeldarchitekturen und darin vorgesehenen Datenmanagementkomponenten, die in den meisten Fällen allerdings nur als Teil einer größeren Architektur diskutiert werden. Aus diesem Grund werden weitere Lösungen aus dem Automotive Bereich untersucht. Die einzelnen Lösungen werden abschließend nach ihren Erfüllungsgraden der fünf Teilziele dieser Arbeit beurteilt.

3.1 Konventionelle Forschungsdatenmanagementsysteme

Das Verwalten von Entwicklungs- und insbesondere von Forschungsdaten ist keine gänzlich neue Disziplin und wurde auch schon vor dem digitalen Zeitalter in analoger Form praktiziert (Gray 1996). Mit fortschreitender Zeit wurden aber immer mehr Daten digitalisiert oder schon in digitaler Form aufgezeichnet und verwaltet. Dabei haben sich mehrere Systeme zur klassischen, zentralen Verwaltung von Forschungsdaten etabliert und werden von zahlreichen Forschungseinrichtungen verwendet, häufig um Forschungsdatensätze zu veröffentlichen.

Einen grundlegenden Vergleich populärer Systeme dieser Art präsentieren Amorim u. a. (2017). Die meisten der untersuchten, konventionellen Forschungsdatenmanagementplattformen Figshare[1], Zenodo[2], CKAN[3], DSpace (Tansley u. a. 2003), ePrints[4] und EUDAT[5] verfolgen das Ziel der langfristigen Konservation von Forschungsdatensätzen in großen Repositorien. Diese Systeme verwalten resultierende Daten am Ende eines datengetriebenen Forschungsprozesses. Dabei sind sie zum großen Teil darauf ausgerichtet die entsprechenden Daten auch zu speichern und sie als eine Art Veröffentlichung (ähnlich zu wissenschaftlichen Publikationen) bereitzustellen. Das Dokumentieren oder Aufzeichnen früherer Zustände der Daten im Datenverarbeitungsprozess (Rohdaten, Zwischenverarbeitungsschritte) werden dabei meist vernachlässigt. Die untersuchten Plattformen können teilweise lokal aufgesetzt werden, bei

[1] https://figshare.com/, abgerufen am 01.10.2021
[2] https://zenodo.org/, abgerufen am 01.10.2021
[3] https://ckan.org/, abgerufen am 01.10.2021
[4] https://www.eprints.org/uk/, abgerufen am 01.10.2021
[5] https://www.eudat.eu/, abgerufen am 01.10.2021

anderen Systemen wie etwa Zenodo existiert global nur eine einzige, zentrale Instanz. Eine weitere bedeutende Aufgabe solcher Systeme ist die Verwaltung Metadaten über die gespeicherten Datensätze sowie die Dissemination der Datensätze durch Herstellung von Interoperabilität und dem Bereitstellen von konfigurierbaren Suchfunktionalitäten. (Amorim u. a. 2017)

Ein populäres Beispiel für ein konventionelles Forschungsdatenmanagementsystem, welches u.a. maritime Daten bereitstellt ist das PANGAEA System, welches vom Alfred-Wegener-Institut betrieben wird (Grobe u. a. 2006). Weitere öffentliche maritime Daten (zum Klima) werden z.B. durch das Helmholtz-Zentrum Hereon auf deren Website bereitgestellt[6]. PANGAEA nutzt ein standardisiertes Modell, um den Prozess der Datenerhebung und weitere Metadaten wie etwa die geografische Position der Aufzeichnung der Daten zu dokumentieren und ermöglicht auch eine Suche auf Basis dieser Parameter (Grobe u. a. 2006). Des Weiteren gibt es Ansätze wie das „Data Portal German Marine Research“, welches Metadaten von mehreren öffentlichen Forschungsdatenmanagementsystem indexiert und somit eine Metaschnittstelle zum Zugriff auf die Daten zur Verfügung stellt (Koppe und Schäfer 2015). Grundlage dieser Portale sind allerdings immer meist vollständig offene Datenökosysteme, die freien Zugriff auf die Daten ermöglichen und verstärkt nach den FAIR-Prinzipien (siehe Abschnitt 2.2.2) aufgebaut sind (vgl. Zuiderwijk, Janssen, und Davis 2014).

Allein die Tatsache, dass konventionelle Forschungsdatenmanagementsysteme die zu verwaltenden Daten bevorzugt zentralisiert speichern, widerspricht bereits dem Prinzip der Dezentralität im dargestellten Testfeldkontext. Bei einer Zusammenführung aller Testfelddaten in einer zentralen Instanz würden die Souveränitätsinteressen der Datenprovider verletzt und ggf. Datenschutzvorschriften nicht eingehalten werden können. Weiterhin ist die starke Fokussierung auf die Verwaltung von Metadaten und konfigurierbarer Suchfunktionalitäten zwar nicht konträr zu den Zielen dieser Arbeit, stellt aber eine Komponente dar, die im Testfeld deutlich weniger stark gewichtet werden kann. Konventionelle Forschungsdatenmanagementsysteme müssen teilweise hunderttausende Datensätze aus ähnlich vielen verschiedenen Quellen durchsuchbar machen, während die Anzahl gleichzeitig relevanter Datenquellen im Testfeldkontext deutlich geringer ist.

[6] https://www.hereon.de/central_units/hcdc/data/index.php.de, abgerufen am 15.12.2022

3.2 Datenmanagementsysteme für Datenökosysteme

Neben den klassischen Forschungsdatenmanagementsystemen, die wissenschaftliche Daten in zentralen Repositorien verwalten, finden sich in der aktuellen wissenschaftlichen Literatur ebenfalls Ansätze, die das Managen von wissenschaftlichen oder industriell genutzten Daten in Datenökosystemen (vgl. Abschnitt 2.3.1) ermöglichen. In einer der früheren Arbeiten zu diesem Thema beschreiben Elsayed und Brezany (2012) eine Plattformarchitektur zur Verwaltung wissenschaftlicher Daten in einem Data Space (siehe Abbildung 14). Dabei charakterisieren sie den „*Scientific Data Space*“ durch Input-Daten, die für ein Experiment verwendet werden, Analysemethoden, die in Experimenten verwendet werden und Datensätze, die (Zwischen-)Ergebnisse von Experimenten darstellen. Hierbei werden die verschiedenen Datenquellen im Data Space zunächst durch den Data Space Indexer durchsucht und indiziert. Ein Data Space Browser ermöglicht das Durchsuchen der Daten und kann über einen Anfragenprozessor und SPARQL-Queries Daten aus dem Data Space abrufen. Es wird hierbei jedoch vorausgesetzt, dass Daten im Data Space über standardisierte Metadaten im OWL-Format verfügen. Mit einer weiteren Abstraktion werden zusätzlich Metadaten über die Verknüpfung von Ressourcen im Data Space in einer RDF-Datenbank verwaltet, um wissenschaftliche Datenverarbeitungsprozesse abzubilden. (Elsayed und Brezany 2012)

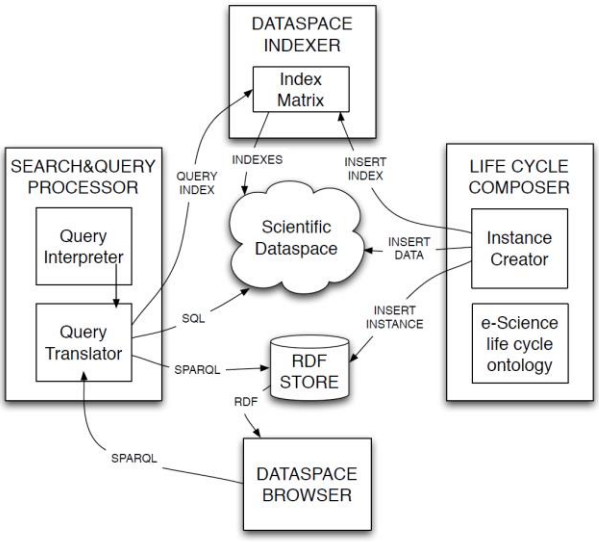

Abbildung 14: Architektur einer Data Space Support Platform für e-Science Anwendungen (Elsayed und Brezany 2012).

Ein technisch ähnliches Konzept schlägt Curry (2012) für das Verwalten von Daten in Systems-of-Systems (SoS) vor. Auch hier werden Architekturkomponenten genutzt, die Metadaten über Datenverarbeitungsprozesse verwalten (Datenprovenienz), einen Datenkatalog bereitstellen, und (Such-)Anfragen verarbeiten können. Dafür werden ebenfalls SPARQL und RDF verwendet.

Zusätzlich wird hier eine Wrapper-Schicht verwendet, um verschiedene Datenformate in RDF zu konvertieren, welches als global standardisiertes Datenformat im Data Space dient und mit zuvor definierten Vokabeln und Ontologien interoperabel verwendet werden kann. (Curry 2012)

Weiterentwickelt wurde das Konzept zum Datenmanagement in Data Spaces von Curry (2019), welcher als Resultat den Real-time Linked Data Space vorstellt. Dabei handelt es sich um eine Anpassung von Data Spaces auf Szenarien in denen Daten u.a. live und eventbasiert verarbeitet werden müssen. Als spezielle Anwendungsdomäne wird das Internet of Things (IoT) angeführt. Eine Plattformlösung (Data Space Support Platform) liefert Funktionalitäten und Dienste zur Verwaltung des Data Spaces. Dazu gehören neben Suchfunktionalitäten und Zugangskontrollen, u.a. Dienste, die das Streaming von Daten und komplexe Eventverarbeitung unterstützen. Diese sind von den klassischen Datendiensten in einer weiteren Architekturschicht abgegrenzt. (Curry 2019)

Eine Standardisierung von Data Space Architekturen strebt die International Data Space (IDS) Association an. Hierzu wurde ein Architekturmodell (vgl. Abbildung 15) ausgearbeitet, was verschiedenste Stakeholder und Interessen abbilden kann. Kern des Konzeptes ist ein größtenteils dezentral organisiertes System, das mittels standardisierter Connectoren und einem oder mehreren unabhängigen Brokern den Datenaustausch in Datenökosystemen ermöglicht. Dabei werden diverse standardisierte Modelle verwendet, um Metadaten über den Broker auszutauschen und durchsuchbar zu machen und Rahmenbedingungen für den ressourcenbasierten Datenaustausch klären zu können. Weitere Elemente sind ein Identitätsprovider, ein App Store, für das anwendungsorientierte Bereitstellen und Beziehen von Data-Assets, eine Vermittlungsstelle zur Dokumentation und Überwachung von Transaktionen und ein Vocabulary Store, der Datenschemata und Vokabeldefinitionen bereitstellt. Außerdem wird ein System zur Einschränkung und Durchsetzung von Datennutzungsrechten implementiert. (Pettenpohl, Spiekermann, und Both 2022)

Das Konzept des IDS fokussiert sich stark auf die Connectoren, die sowohl von Datenprovider als auch Datenkonsument genutzt werden, um mit dem Data Space zu interagieren. Die aktuelle Referenzimplementierung des IDS Connectors nutzt eine standardisierte REST-Schnittstelle für die Verwaltung und Abfrage der Daten und ist modular aufgebaut. Externe Datenquellen können u.a. via REST angebunden werden. Es können zwar Live-Daten übermittelt werden, die effiziente und performante Bereitstellung von Datenströmen ist jedoch nur schwer umsetzbar und damit in der Standardimplementierung ohne Erweiterungen für die Nutzung für Sensordatenströme nicht geeignet. (vgl. Fraunhofer ISST 2021)

Neben der Referenzimplementierung des IDS Connectors existiert auch noch der „Eclipse Dataspace Connector“, welcher die Architektur erweitern soll, und als vollständige Open Source

Lösung beliebige Data Space Protokolle (auch vom IDS unabhängige) unterstützen soll. Zum Zeitpunkt des Verfassens dieser Arbeit befindet sich der Eclipse Data Space Connector jedoch noch im frühen Entwicklungsstadium, und es nicht abzusehen, in welchem Umfang das Konzept die Anforderungen eines Testfeld-Data Spaces erfüllen kann. (Pampus 2022)

Eine Umsetzung des IDS-Modells liefern z.B. Sarabia-Jácome u. a. (2019) im Kontext eines Hafens und Nesheim u. a. (2021) für einen operativen Datenaustausch zwischen verschiedenen maritimen Stakeholdern.

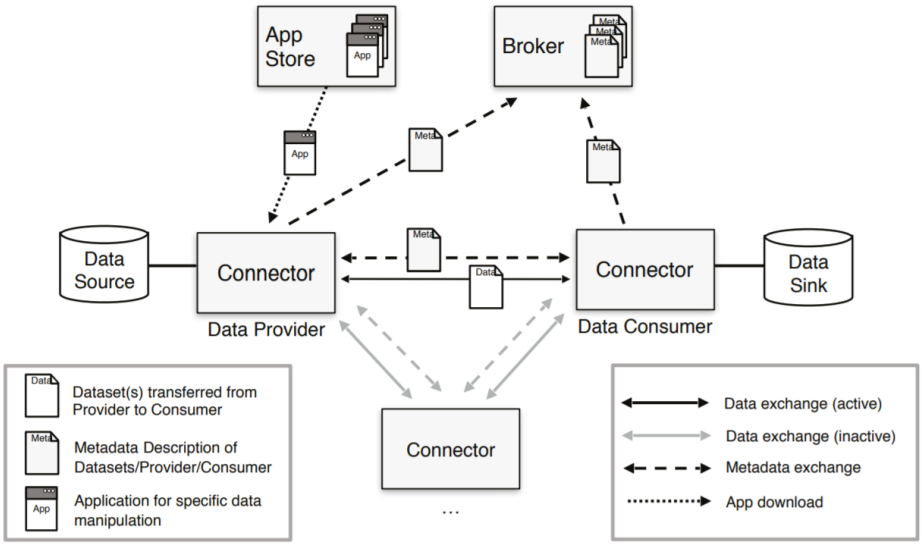

Abbildung 15: Referenzarchitektur der International Data Spaces Association (Otto, Steinbuß, u. a. 2019).

Die IDS-Referenzarchitektur wird vom europäischen Dateninfrastrukturprojekt GAIA-X aufgegriffen und in einen größeren Kontext integriert (siehe Abbildung 16). Data Spaces aus verschiedenen Bereichen wie Energie, Mobilität und Landwirtschaft und zugehörige Datendienste sollen über die GAIA-X Architektur föderiert werden. Zusätzlich zum Datenökosystem wird hier auch noch ein Architekturmodell für das Bereitstellen eines Infrastrukturökosystems implementiert: So können z.B. Cloud-Architekturanbieter oder HPC-Betreiber ebenfalls in GAIA-X eingebunden werden und Services zum Speichern, Verarbeiten und Übertragen von Daten anbieten. Außerdem werden Zertifizierungsprozesse expliziter definiert, um eine tatsächliche Umsetzung einer globalen GAIA-X Instanz zu ermöglichen und Interoperabilität zu gewährleisten. (Eggers u. a. 2020)

Mit den GAIA-X Federation Services wird außerdem ein Framework bereitgestellt, welches technische Dienste zur Föderation von Daten- und Infrastrukturdiensten im GAIA-X Ökosystem spezifiziert (GAIA-X 2022). Hierunter fallen Dienste zur Katalogisierung, zum Identitätsmanagement, zum Aushandeln von Verträgen für die Nutzung bestimmter Ressourcen,

zur Dokumentation von Datenhandel und der Durchsetzung von gemeinsamen Richtlinien (GAIA-X 2022). Die Federation Services ermöglichen einen sicheren und nachverfolgbaren Austausch von Daten und legen den Fokus auf die Erhaltung der Souveränität ihrer Nutzer unter Einsatz eines Rollenmodells (GAIA-X 2022). Konkrete Spezifikationen versuchen dabei allerdings auf einer abstrakten Ebene zu arbeiten, die alle Use-Cases von GAIA-X abdeckt und bieten somit nur ein Basis-Framework was durch Erweiterungen an konkretere Use-Cases wie den eines serviceorientierten Testfelds angepasst werden müsste (vgl. z.B. den Data Exchange Logging Service zur sicheren Dokumentation von Datenaustauschprozessen (GAIA-X 2021)).

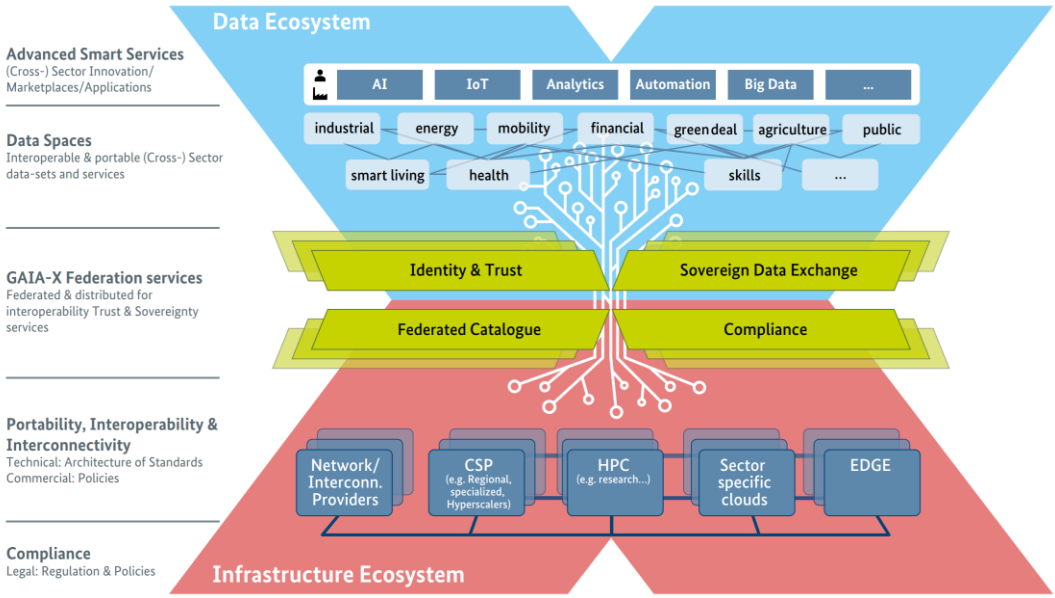

Abbildung 16: Architekturansatz von GAIA-X: Datenökosystem, Infrastrukturökosystem und Federation Services (BMWi 2020).

Der Eclipse Data Space Connector hat neben seiner geplanten Verwendung im IDS auch in GAIA-X eine große Relevanz (Pampus, Jahnke, und Quensel 2022). Er unterstützt im Vergleich zum IDS Connector auch alternative Protokolle, verfügt über eine bessere Erweiterbarkeit und wird stärker durch die Community vorangetrieben (Pampus 2022).

Neben der Modellierung von Datenökosystemen als Data Spaces schlagen Li u. a. (2019) eine dreischichtige Architektur für ein globales System zum Verwalten von und Handeln mit Gesundheitsdaten vor, die auf einer Edge-Cloud Lösung basiert. Zunächst erheben Nutzer in der Datenschicht dezentral Daten und übermitteln diese an die sogenannte Edge-Schicht, in der Daten unter der Aufsicht einer lokalen Autorität zusammengetragen, vorverarbeitet und analysiert werden. Zudem gibt es ein lokales Nutzermanagement und eine kryptografische Absicherung der Datenübertragung. Im Modell von Li et al. existieren mehrere Instanzen der Daten und der Edge-Schicht, die durch eine zentrale Autorität global vernetzt und überwacht werden und der die Identitäten aller Teilnehmer bekannt sind. Dies soll es ermöglichen Daten lokal schneller zu

verarbeiten und bereitzustellen aber trotzdem den globalen Austausch von Daten nicht einzuschränken. Für den Austausch von Daten zwischen Datenkonsumenten und Datenprovidern wird ein standardisiertes und abgesichertes Protokoll implementiert, welches Verhandlungen ermöglicht, um die Konditionen eines Datenhandels festzulegen. (Li u. a. 2019)

Weitere Ansätze, wie etwa von X. Liang u. a. (2017) oder B. Liu u. a. (2017), verwenden Blockchain-Technologien, um eine Vertrauensbasis zwischen verschiedenen Teilnehmern im Datenökosystem herzustellen und den Datenaustausch abzusichern. Meist werden die eigentlichen Daten dabei aber nicht direkt über eine Blockchain ausgetauscht. Diese wird lediglich als vertrauenswürdige Instanz genutzt, um Datenaustauschprozesse zu verhandeln, zu dokumentieren und zu überwachen, wenn kein Intermediär vorhanden ist, der die Anforderungen der Teilnehmer des Datenökosystems erfüllt und diese Aufgaben übernehmen kann.

3.3 Anwendungen in Testfeldern

In den folgenden Unterabschnitten werden Lösungsansätze präsentiert, die sich auf die Anwendung in Testfeldern spezialisieren. Abschnitt 3.3.1 geht dabei auf maritime Testfelder ein, Abschnitt 3.3.2 auf automobile Testfelder.

3.3.1 Maritime Testfelder

Für die Untersuchung der verwandten Arbeiten, die sich konkret mit dem Anwendungsfall von maritimen Testfeldern beschäftigen, wurde zunächst eine Recherche auf Basis der von der International Association of Marine Aids to Navigation and Lighthouse Authorities (IALA) gepflegten Liste (IALA o. J.) an Testfeldern durchgeführt. Hierbei wurden insgesamt 50 Testfelder in Betracht gezogen. Im Folgenden werden aus diesen Testfeldern nur jene berücksichtigt, die eine generische Architektur aufweisen und nicht Use-Case spezifisch sind, und somit das Testen von unterschiedlichen Szenarien ermöglichen (vgl. Hahn und Noack 2016). Zudem wurden nur Testfelder berücksichtigt, die Architekturelemente zum aktiven Managen von Daten aufweisen. Beide Kriterien können sich hier gegenseitig bedingen. So weisen Use-Case spezifische Testfelder, etwa zum Testen bestimmter Services zur Eis-Navigation, meist keine Komponenten zur nachhaltigen Verwaltung von Daten auf, sondern nur eine angepasste Lösung, um Daten für den speziellen Use-Case zu übertragen.

Smart Ship Application Platform 2 (SSAP 2). Im Rahmen der SSAP 1 und 2 Projekte wurde eine Architektur zum Austausch von Daten von Schiffen und Küstenstationen entwickelt. Dabei wurde ein Ansatz verfolgt, in dem die gesamten Daten des Testfeldes in küstenseitigen Master-Databases mit standardisierten Schnittstellen und einem „Onboard data server“ an Bord des

Schiffes für schiffsspezifische Applikationen verwaltet werden (siehe Abbildung 17). Dabei werden die Master-Databases, welche die Daten über mehrere Schiffe aggregieren in einem Rechenzentrum von einer neutralen Partei betrieben. Der Fokus wurde speziell auf IoT-Daten und typische maritime Daten wie Wetter-, Verkehrs- und Motordaten gelegt. Weiterhin werden die Daten standardisiert in einer SOA ausgetauscht und können durch Vorgaben der Schiffsbesitzer kontrolliert werden. Neben der Speicherung der Daten auf Schiffs- und Küstenseite können Live-Daten auch zwischen diesen gestreamt werden. Die Datenaufzeichnung und Übertragung basiert auf den Standards ISO 19848 und IEC61162. (Ando 2017)

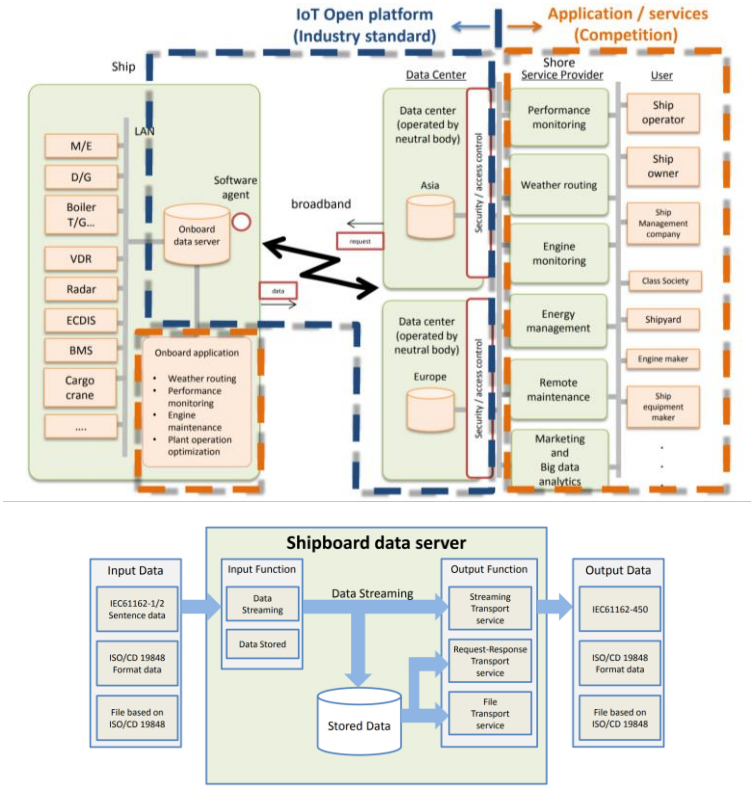

Abbildung 17: SSAP2 Architektur für maritime Testfelder: Gesamtübersicht (oben) und schiffsseitiger Datenserver (unten) (Ando 2017).

datAcron. Das datAcron-Projekt fokussiert sich auf das Management und die Analyse heterogener Daten im Zusammenhang mit Überwachungssystemen aus der maritimen Domäne und der Luftfahrt. Ausgewählte maritime Datenquellen beinhalten dabei AIS-Daten, Kartendaten, Schiffsrouten, Schiffsdaten, (Unfall-)Berichte, Wetterdaten und Daten über spezielle Geografische Bereiche (z.B. Fischereigebiete). Dabei wird im Speziellen zwischen „data-at-rest“ (archivierten Daten) und „data-in-motion“ (Live/Streaming-Daten) unterschieden. Die Daten werden nach einem klassischen wissenschaftlichen Workflow verarbeitet (siehe Abschnitt 2.2.2). Dabei werden alle Daten mittels einer zuvor definierten Ontologie in das RDF-Format transformiert und zentral gespeichert. Zusätzlich werden Verkehrs- und Wetterdaten assoziiert,

für Überwachungsaufgaben aufbereitet und über eine Apache-Spark basierte RDF-Query-Schnittstelle zur Verfügung gestellt. (Vouros u. a. 2018)

Auch auf der architekturellen Ebene (vgl. Abbildung 18) wird zwischen Datenquellen mit archivierten Datenquellen und Streaming-Datenquellen unterschieden. Die Datenquellen werden an eine zentrale Managementkomponente angeschlossen, wobei die Streaming-Daten zuvor komprimiert und aufbereitet werden. Weiterhin werden spezielle Komponenten für den Anwendungsfall der Analyse und Vorhersage von Überwachungsdaten, sowie zur Visualisierung bereitgestellt. (Claramunt u. a. 2017)

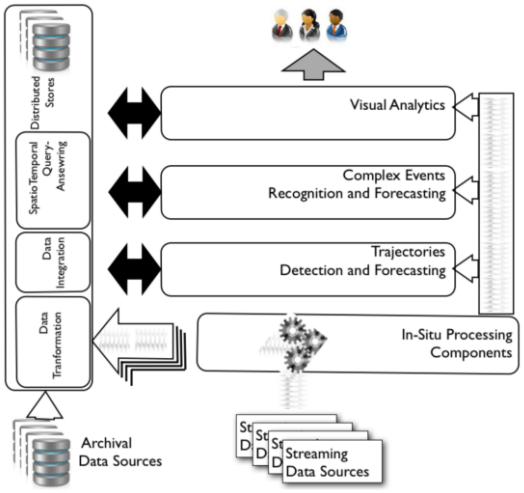

Abbildung 18: datAcron Datenarchitektur: Aufteilung in Streaming-Daten und archivierte Datenquellen (Claramunt u. a. 2017).

STM Validation. Ziele der STM (Sea Traffic Management) Initiative sind das Erhöhen der Sicherheit im maritimen Verkehr durch das Verringern des Workloads auf Schiffsbrücken mittels besserer Organisation von relevanten Informationen, die Steigerung der Effizienz der Schifffahrt mittels digitaler Infrastrukturen. Diese Maßnahmen sollen zudem eine stärkere Berücksichtigung von Nachhaltigkeitsaspekten erzielen. Im Rahmen des STM Validation Projektes wurden mehrere Konzepte und Services des STMs untersucht und innerhalb mehrerer Testfelder evaluiert. Hierbei wurden im Speziellen die drei Bereiche der kollaborativen Entscheidungsfindung in Häfen, der Routenmanagementsysteme und der Verkehrssimulation berücksichtigt. (Andreasson u. a. 2019)

Im Bereich der kollaborativen Entscheidungsfindung in Häfen wurde eine Plattformlösung entwickelt, die das Austauschen von Daten unter Berücksichtigung verschiedener Interessen und Sicherheitsaspekte ermöglicht. So werden die Interessen der Anbieter und Konsumenten von Daten-Services berücksichtigt. Eine plattformbasierte Lösung hilft beim souveränen Austausch der Daten zwischen den verschiedenen Teilnehmern (vgl. Abbildung 19). Dazu wird das Datenökosystem maritimer Stakeholder analysiert und eine Rollenverteilung hergeleitet: Der „*Core Provider*“ stellt die essenzielle, standardisierte Infrastruktur zur Verfügung, um Daten

auszutauschen, zu verwalten und Services zu integrieren. Weiterhin können Datenprovider Connectoren nutzen, um ihre Daten bereitzustellen und Service Provider können Services bereitstellen, die die vorhandenen Daten verarbeiten und den Servicekonsumenten zur Verfügung stellen. Es wird also eine klassische Service-orientierte Architektur umgesetzt, bei der der Core Provider als Intermediär zwischen Datenkonsumenten und Providern steht und erinnert mit den Connectoren stark an einen Data Space. Der sogenannte „*PortCDM council*“ dient als überwachendes Organ und erfüllt allgemeine Governance-Aufgaben (Standardisierung, Wartung, Genehmigung neuer Services). Das Konzept ist stark auf den Betrieb operativer Systeme und Services ausgerichtet. (Lind u. a. 2015)

Konkretisiert wird das Prinzip der Connectoren im „Voyage Management Testbed“ des STM Validation Projektes. Mit den sogenannten *SeaSWIM* (System Wide Information Management) Connectoren und der Maritime Connectivity Platform (früher: Maritime Cloud), die für das Identitätsmanagement und das Verwalten von Services in einer SOA genutzt wird, können mittels direkter Verbindungen (ohne Intermediär) Daten zwischen verschiedenen Providern und Konsumenten nachrichtenorientiert ausgetauscht werden. Typische Daten, die unterstützt werden, sind: Routendaten, Navigationswarnungen, Textnachrichten und Hafendaten in jeweils standardisierten Datenformaten. (Andreasson u. a. 2019)

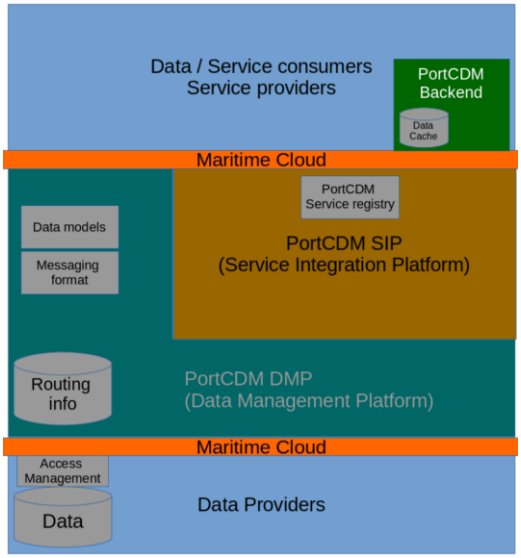

Abbildung 19: Datenmanagementarchitektur im STM Validation Projekt: Kommunikation von Daten über die Maritime Cloud (Lind u. a. 2015).

e-Maritime Integrated Reference Platform (eMIR). Das eMIR-Testfeld ist ein generisch aufgebautes Testfeld, welches die Entwicklung und Erprobung hochautomatisierter und autonomer Systeme ermöglichen soll. Dabei wird das Ziel verfolgt eine möglichst nachhaltige und interoperabel verwendbare (und damit generische) Testfeldarchitektur bereitzustellen, und

damit die Anbindung zu testender Systeme an das Testfeld zu vereinfachen. Der Fokus hierbei wurde auf die Aspekte der Entwicklung Cyber-physischer Systems of Systems (CPSoS) gelegt, die spezielle Anforderungen bezüglich Sicherheit, Zuverlässigkeit, Validierung und Verifikation, und des komplexen Zusammenspiels mehrerer Subsysteme mit sich bringen. Weiterhin werden Schnittstellen angeboten, die das kombinierte Testen in physischen und simulierten Testumgebungen ermöglichen. Zwischen allen Komponenten der Testfeldarchitektur werden die Daten im standardisierten S-100 Format ausgetauscht. Genauer besteht eMIR aus diversen Simulationskomponenten, die Verkehr, Umgebung und Sensorik simulieren können, einer Referenzwasserstraße im Bereich der deutschen Bucht und der Elbe, die Realdaten über den maritimen Verkehr und Umweltbedingungen aufzeichnet, einem Forschungsboot, einer mobilen Schiffsbrücke und einer Near-Collision Datenbank. (Rüssmeier, Lamm, und Hahn 2019)

Der zentrale Datenaustausch wird im eMIR-Testfeld durch eine nachrichtenorientierte Middleware umgesetzt, die die in S-100 kodierten Nachrichten zwischen den verschiedenen Komponenten austauscht. Zur Konvertierung zwischen verschiedenen anderen Standards und dem standardisierten Datenformat wird eine polymorphe Schnittstelle verwendet. Über diese Schnittstelle werden auch zu testende Systeme angeschlossen, die sich zum Austauschen von Daten mit der restlichen Testfeldinfrastruktur an die Spezifikation der Schnittstelle anpassen muss. Dieser allgemeine Aufbau ermöglicht das Einbinden unbekannter (physischer und simulierter) Systeme in die Testfeldumgebung von eMIR. (Brinkmann, Hahn, und Hjøllo 2017)

Ausgewählte Daten des Testfeldes werden in einem Data-Warehouse gespeichert und vor dem Hintergrund der Verkehrsanalyse aufbereitet. Dazu wird ein klassischer wissenschaftlicher Datenverarbeitungsprozess angewendet (vgl. Abschnitt 2.2.2), bei dem AIS und RADAR-Daten aufgezeichnet, bereinigt und miteinander fusioniert werden. Für den Anwendungsfall der Erkennung kritischer Verkehrssituationen werden die Daten zu einzelnen Schiffsbahnen zusammengefügt, analysiert und basierend auf verschiedenen Arten von Schiffbegegnungssituationen kategorisiert. Die Rohdaten, Tracks und Near-Collision-Begegnungssituationen werden zentral in verschiedenen Datenbanken gespeichert. Dieser Workflow deckt jedoch nur einen Teil der Daten aus dem eMIR-Testfeld ab, spezielle Datenmanagementprozesse für andere Datenquellen, wie etwa weitere Sensorik werden nicht betrachtet. (Lamm und Hahn 2018)

Weiterhin sind noch das Hermitage Testbed[7] und die Testfelder der Projekte SMART Navigation[7], ACCSEAS[7] und MONALISA[7] zu nennen, bei denen zu vermuten ist, dass aufgrund ihrer generischen Ausrichtung auch Methoden zur nachhaltigen Verwaltung von Daten zum Einsatz kamen. Hierzu werden jeweils jedoch keine weiteren öffentlichen Angaben gemacht.

3.3.2 Automobile Testfelder

Zwar nicht mit dem maritimen Anwendungsfall identische, aber durchaus ähnliche Anforderungen an Testfelddatenmanagementsysteme lassen sich in der Automobilindustrie und -forschung wiederfinden. Auch hier werden etwa hochautomatisierte und autonome Systeme für Fortbewegungsmittel eingesetzt, die auf ähnliche Art und Weise auf ihre Umwelt reagieren müssen und ähnlich getestet werden, wie im maritimen Anwendungsfall. Aus diesem Grund werden im folgenden Abschnitt ausgewählte Arbeiten aus dem Automotive Bereich vorgestellt, die sich mit dem Management von Testfelddaten beschäftigen.

UCLA C-Vet. Das C-Vet Testfeld der University of California, Los Angeles ist ein generisches Testfeld zur Erprobung von Modellen und Protokollen, die in vernetzten Fahrzeugen eingesetzt werden. Dabei werden sowohl die Erforschung der Kommunikationskanäle für vernetzte Fahrzeuge (physische Kommunikationsebene, Protokollebene und im Speziellen der Datenaustausch via 802.11p) als auch die experimentelle Validierung von Mobilitätsmodellen ermöglicht. Letzteres zielt auf das Testen innerhalb einer realistischen Umgebung ab, sodass Einflüsse wie die Verkehrsdichte, realistische Bewegungsmuster und weitere Verkehrsbedingungen in Experimenten berücksichtigt werden und naturalistische Verhaltensmuster im Verkehr untersucht werden können. Weiter wurde das Ziel verfolgt, das Testfeld auch für externe Nutzer verfügbar zu machen und nachfrageorientiert aufzubauen, was sich ebenfalls in der gewählten Testfeldarchitektur widerspiegelt. Das Testfeld besteht aus Hardware und Software zur Erprobung neuer Netzwerkschichten und -protokolle, einem testfeldweiten Mesh-Netzwerk, mehreren mit Sensorik ausgerüsteten Fahrzeugen, Sensorik zur Überwachung der Umgebung (Verkehr, Luftqualität und Bildgebende Daten durch stereoskopische Kameras), einer Webschnittstelle zur Verwaltung der Testfeldkomponenten und einer zentralen Live-Datenbank zur Aggregation der aufgezeichneten Testfelddaten. Dabei wird das Testfeld mittels mehrerer, virtualisierter, sogenannter *Mobile Nodes* organisiert, die durch Testfeldnutzer frei konfiguriert werden können. Durch eine Middleware kann aus diesen Knoten

[7] Vgl. https://www.iala-aism.org/technical/e-nav-testbeds/iala-testbeds-guideline/, abgerufen am 08.10.2021

auf die Testfeldinfrastruktur zugegriffen werden: So werden die Authentifizierung und Autorisierung, die Steuerung von Experimenten, die Verwaltung von Mobile Nodes und die Datensammlung organisiert. Die Sensordaten werden dann mithilfe der C-VeT-Census platform bereitgestellt und aufgezeichnet. Auf diese Art entstehen Testfeld-Datengrundlagen, wie eine Mobilitätsdatenbank, eine Schadstoffdatenbank, sowie eine Bilddatenbank, die für 3D Lokalisierung und Rekonstruktionen verwendet werden kann. Typischerweise beinhalten diese Daten zudem zeitliche und spatiale Attribute. (Lutterotti u. a. 2008)

VFbed. Das Testfeld VFbed ist eine Testumgebung zur Erprobung von Konzepten im Bereich des „*Vehicular Fog Computing*“. Hierbei werden Berechnungen in eine Netzwerkschicht zwischen Endgerät und Cloud (der sogenannten „*Edge*“) ausgelagert. Neben der Kommunikation zwischen den Fahrzeugen im Testfeld (Vehicle-to-Vehicle) spielt also auch die Kommunikation zwischen Fahrzeug und Infrastruktur (Vehicle-to-Infrastructure) eine wichtige Rolle. Sowohl die Testträger (durch On-Board-Units), als auch die Infrastruktur (sogenannte „*Fog Nodes*“) sind mit dem Backend des Testfeldes verbunden. Dabei werden statt realen Fahrzeugen, Miniaturroboter verwendet, die einen Raspberry Pi zur Steuerung nutzen. Das VFbed Testfeld ist stark auf eine Nutzung als Service ausgerichtet: Nutzer des Testfeldes können vollautomatisch Testumgebungen buchen, konfigurieren und erweitern. Dazu werden verschiedene Management-Komponenten in einer Kombination aus Cloud und einem lokalen Testfeld-Manager (siehe Abbildung 20) verwendet, die eine benutzerdefinierte Experimentspezifikation validieren, basierend darauf das Testfeld konfigurieren, benötigte Ressourcen buchen und die Durchführung des Experiments überwachen. Nach der Durchführung eines Experimentes werden Log-Daten, Aufzeichnungen der Kommunikation im Testfeldnetzwerk, Sensordaten und Benchmarkdaten an eine Datenmanagementkomponente weitergeleitet, die die Ergebnisse analysiert und in einer zentralen Experimentdatenbank ablegt. Nutzer des Testfeldes können diese dann über ein Webinterface abrufen. Konfigurationen und die Belegung der Testfeldressourcen werden in separaten Datenbanken intern verwaltet. Eine Live-Nutzung der Testfelddaten ist hier nicht ohne weiteres möglich. (Hoque und Hasan 2020)

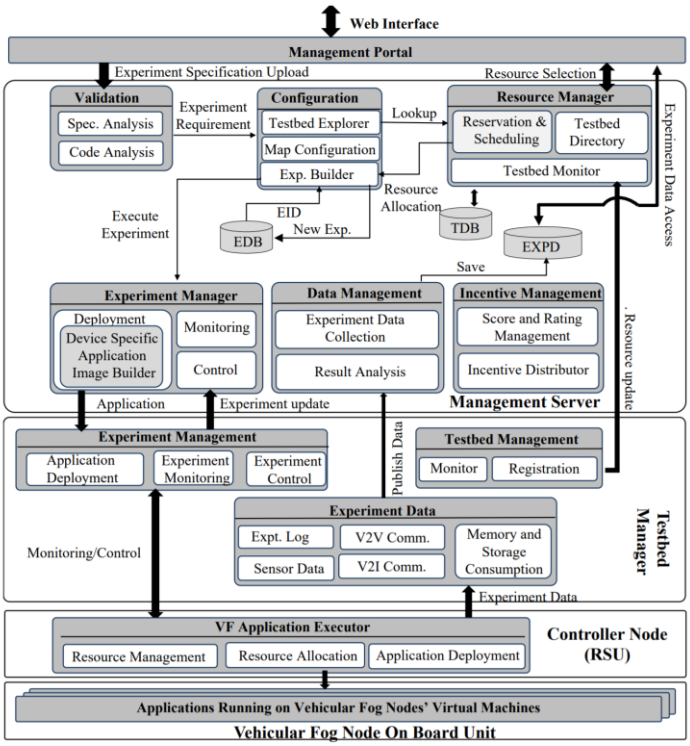

Abbildung 20: VFbed Systemarchitektur: Verwaltung und automatische Konfiguration von Experimenten und anschließende Datensicherung (Hoque und Hasan 2020).

Klitzke et al. Eine der wenigen Arbeiten, die sich ausschließlich mit dem Thema des Datenmanagements in Testfeldern bzw. Testumgebungen im Automobilbereich beschäftigt, wurde 2019 von Klitzke, Koch, Haja und Köster veröffentlicht. Sie beschäftigen sich mit der Frage, wie Daten für Realwelt-Testfahrten organisiert werden können, insbesondere geht es hierbei um die Identifikation und das Verwalten von Daten in Form von Szenarios. Dazu wird ein Vehicle Data Management System (VDMS) entwickelt, welches die Rohdaten aus den Testträger-Fahrzeugen in Szenen einteilt, aggregiert und diese durch weitere Verarbeitung und externe Datenbanken (z.B. Kartendaten) anreichert. Zur Anforderungserhebung wird ein Rollenmodell von Stakeholdern im Testfeld genutzt, welches operative Aspekte in der Testdurchführung modelliert (Akteure sind hier z.B. Kampagnenleiter oder Testfahrer). Die gesamte Architektur und die Verarbeitungsmodule sind darauf ausgerichtet die Daten aus den Tests in eine Szenenrepräsentation zu überführen und verfügbar zu machen, um einen Szenariokatalog aufzubauen, relevante Szenarios für die Zertifizierung von SuTs zu identifizieren, relevante Systemparameter in bestimmten Szenarios zu identifizieren und die Performance von Realwelttests zu evaluieren. Für das Datenmanagement wird eine dreischichtige Architektur vorgestellt (siehe Abbildung 21): Die Datenschicht besteht aus den Rohdaten aus den Testträgern, die in einem standardisierten Format aufgezeichnet werden, einer relationalen Datenbank und Kartendaten von OpenStreetMap, die für das Anreichern der Testdaten verwendet werden. Die Modulschicht enthält die zentrale Datenmanagementlogik des Systems und ist in die Bereiche

Import, Diskretisierung, Aggregation, Anreicherung, Kompression, Parser, Statistik und Bildexport aufgeteilt. Die Kommunikation dieser Module untereinander findet über ein gemeinsames Dateisystem und einer statischen Konfiguration von Ordnern statt, die von den Modulen zum Lesen und Schreiben von Daten genutzt werden. Eine Eventschnittstelle ermöglicht das Benachrichtigen von anderen Modulen bei Updates im Verarbeitungsprozess. Letztlich bietet die Schnittstellenschicht die Möglichkeit die Daten über mehrere APIs und ein Benutzerinterface aus dem System abzurufen und zu analysieren. (Klitzke u. a. 2019)

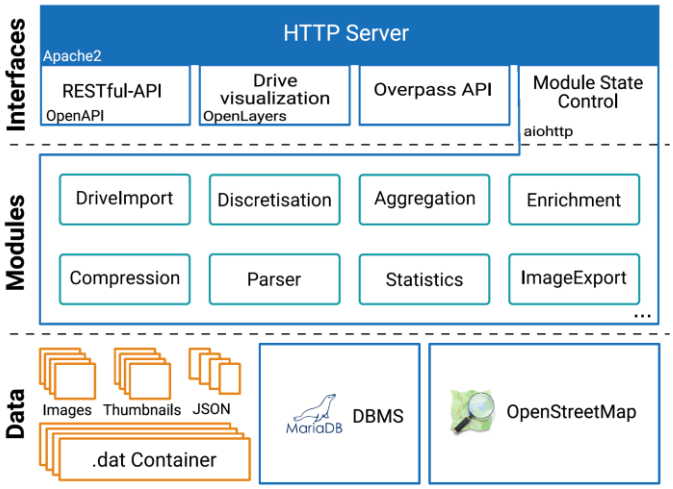

Abbildung 21: Dreischichtige Datenmanagementarchitektur zum Verwalten von Fahrzeugdaten (Klitzke u. a. 2019).

Ähnliche Konzepte zum Szenario-orientierten Datenmanagement finden sich PEGASUS-Projekt wieder, welches sich aber nicht mit Testfelddaten, sondern mit Daten aus natürlichem Fahrverhalten (außerhalb von Testfeldern) beschäftigt. (PEGASUS 2019)

HarborNet. HarborNet (siehe Abbildung 22) hat mit seiner Positionierung im Hafen von Leixões, Portugal einen klaren maritimen Bezug, basiert neben ausgewählten Schiffen aber u.a. auf LKWs als Testträgern. Diese sind jeweils mit mobilen Einheiten ausgerüstet, die das Austauschen von Daten über IEEE 802.11p, Wi-Fi oder 3G ermöglichen. Dabei ist sowohl eine Verbindung zu Basisstationen des Testfeldes möglich als auch die direkte Kommunikation zwischen den Testteilnehmern. Das Testfeld wird hauptsächlich zur Erforschung von Netzwerkprotokollen für „*connected vehicles*“ verwendet. Als Datenmanagementlösung zum Verwalten der aufgezeichneten Testfelddaten wurde ein Cloud-basiertes Backendsystems entwickelt, welches in Form eines Data-Warehouses die Daten aus dem Testfeld in einer NoSQL-Datenbank speichert und über eine Web-API verfügbar macht. Weiterhin werden mehrere Monitoring-Komponenten eingesetzt und eine Echtzeit Auswertung von Experimenten, sowie eine verzögerte Analyse sind möglich. (Ameixieira u. a. 2014)

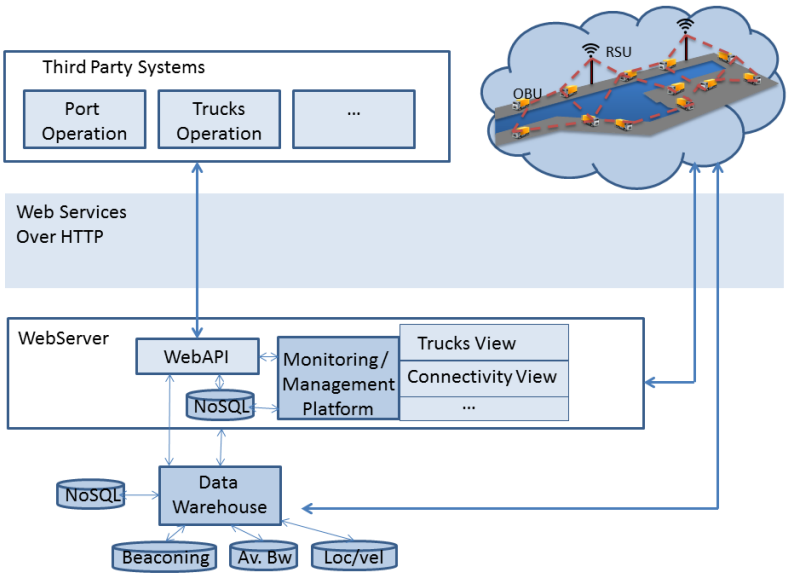

Abbildung 22: Systemarchitektur von HarborNet: Datawarehouse und Webinterface (Ameixieira u. a. 2014).

In der Literatur finden sich viele weitere Arbeiten zum Aufbau von Automobiltestfeldern, die in den vorgestellten Architekturen zwar Komponenten zur Datenspeicherung vorsehen, diese aber nicht detailliert darstellen. Wie in den Testfeldern von Marquez-Barja u. a. (2019), Chowdhury u. a. (2018) oder dem U.S. Department of Transportation (2012) handelt es sich dabei meist um Lösungen, die neben Methoden zur Übertragung und einfachen Speicherung keine expliziten Datenmanagementkonzepte spezifizieren.

3.4 Zusammenfassung und Bewertung der verwandten Arbeiten

Im Folgenden soll nun die Abdeckung der Ziele dieser Arbeit durch die vorgestellten verwandten Arbeiten (Abschnitte 3.1 bis 3.3) untersucht werden.

Zusammenfassung mehrerer unabhängiger bzw. dezentraler Datenquellen zu einer kohärenten Einheit.

Das erste Ziel dieser Arbeit wird durch die verwandten Arbeiten bereits größtenteils erfüllt. Gerade das Zusammenfassen von Daten aus verschiedenen Quellen an einem zentralen Zugriffspunkt ist essenziell für nahezu alle Arbeiten. Klassische FDMS, sowie DMS für Datenökosysteme und spezielle Testfeldkomponenten erfüllen somit die Anforderungen A1 und A2. Auch A3 wird überwiegend durch standardisierte Schnittstellen oder erweiterbare Komponenten erfüllt und ist in DMS für Datenökosysteme aufgrund ihrer Dezentralität unabdingbar. Lediglich HarborNet aufgrund seiner statischen Data-Warehouse Lösung mit vordefinierten Datenbankstrukturen und der Ansatz von Klitzke et al., die sich auf ein spezielles

Datenformat fokussieren, können diese Anforderung nicht erfüllen. Bezüglich des Metadatenmanagements (A4) wird bei den meisten Architekturen nicht explizit dargelegt, wie verschiedene Datenquellen verwaltet wurden. Es ist jedoch davon auszugehen, dass bei einer dynamischen Einbindung neuer Datenquellen notwendige Metadaten gespeichert werden. Erweiterte Metadatenkonzepte findet man typischerweise bei den klassischen FDMS oder bei DMS für Datenökosysteme. Insbesondere das Referenzarchitekturmodell der IDS Association verfügt über ausführliche Metadatenschemata, die das Auffinden und Durchsuchen von Daten(-dienstleistungen) erleichtern sollen.

Bereitstellung des Zugriffes auf Forschungsdaten in Abhängigkeit von den Testfeldnutzern.

Bei der Abdeckung der Anforderungen, die sich dem zweiten Teilziel dieser Arbeit zuordnen lassen, zeigt sich ein gemischtes Bild. Die auf Data Spaces basierenden DMS für Datenökosysteme erfüllen die Anforderungen A5 und A6 durch ihre dezentral ausgerichtete Architektur grundsätzlich, es ist aber (abgesehen vom IDS und GAIA-X) nicht immer ganz klar, inwiefern die Durchsetzung von Nutzungsbedingungen stattfindet, oder ob der Einsatz von Datentreuhändern möglich ist. Der Edge-Cloud Ansatz von Li u. a. (2019) ist hier außerdem etwas kritischer zu sehen, da nicht klar definiert ist, unter welcher Kontrolle die Edge-Schicht steht und somit nicht klar ist, ob eine Souveränität bzw. flexibel konfigurierbare Zwischendatenspeicherung gewährleistet werden können. A8 wird durch die DMS für Datenökosysteme erfüllt, da hier jeweils klar definiert ist welche Rolle ein Teilnehmer im Datenaustausch hat, damit dieser stattfinden kann. Bei den Testfelddatenarchitekturen ergibt sich ein anderes Bild. Da HarborNet und der Ansatz von Klitzke et al. die modulare Integration weiterer Datenquellen nicht vorsehen ist hier eine Betrachtung der Anforderungen A5, A6 und A8 nicht sinnvoll. datAcron, eMIR und VFBed verteilen und aggregieren die Testfelddaten zentral und entziehen damit externen Nutzern die Souveränität, wenn diese Daten in das Testfeld einspeisen wollen. C-Vet unterstützt zwar eine Isolierung externer Architekturelemente durch Virtualisierung und Authentifizierung bzw. Autorisierung von Nutzern, aggregiert die Testfelddaten dann aber auch in einer zentralen Datenbank. SSAP2 unterstützt zwar das lokale Erheben und Speichern von Daten auf Schiffen als Testträgern, überträgt diese dann ebenfalls in eine zentrale Masterdatenbank unter der Kontrolle einer neutralen Partei. Die volle Flexibilität ist hier also nicht gegeben und A5 und A6 nur teilweise erfüllt. Lediglich das STM Testbed verfolgt mit dem alleinstehenden SeaSWIM-Connector einen Ansatz der A5, A6 und A8 erfüllen kann. Bezüglich des inhaltlichen Datenschutzes und der Anonymisierung (A7) von Daten finden sich keine konkreten Konzepte.

Unterstützung dezentraler Datenverarbeitungsprozesse im maritimen Testfeld.

Die Integration von Datenverarbeitungsprozessen (A9) wird von nahezu allen der vorgestellten Arbeiten in grundlegender Form zunächst dadurch ermöglicht, dass standardisierte (Datenbank-)Schnittstellen zum Abrufen und Einspeisen von Daten vorhanden sind. Gerade die DMS für Datenökosysteme unterstützen dies aus der Notwendigkeit überhaupt Daten austauschen zu können. Überwiegend wird die Datenkonsumentenseite jedoch nicht weiter betrachtet und die Anbindung an häufig genutzte Tools aus der Data Science muss selbst implementiert werden. Lediglich der IDS-Connector und der SeaSWIM-Connector erlauben hier auch das direkte Weiterverarbeiten auf der Datenkonsumentenseite. Spezielle Voraussetzungen für die Integration klassischer Data Science Werkzeuge werden so aber auch nicht geschaffen. Explizite Unterstützung für die Verarbeitung von Live-Daten in Form von Datenströmen (A10) liefert unter den FDMS und den DMS für Datenökosysteme nur der Real-time Linked Data Space. Li u. a. (2019) und die IDS-Referenzarchitektur, machen nur abstrakte Angaben über eine mögliche Live-Daten Unterstützung. Der IDS Connector unterstützt dabei zwar Live-Daten, aber standardmäßig keine Datenströme, eine weitere Implementierung befindet sich momentan noch in Entwicklung. In den untersuchten Testfeldarchitekturen wird das Verarbeiten von Live-Daten grundsätzlich ermöglicht. Ausnahmen sind hier VFBed und der Ansatz von Klitzke et al., die ohne Weiteres keine Live-Datenverarbeitung unterstützen und die datAcron-Architektur, die nur die Live-Analyse von Daten unterstützt. Das Szenario-orientierte Datenmanagement (A11) in *Phase II* des Testfelddatenmanagements unterstützt nur die Arbeit von Klitzke et al., die auch speziell auf diesen Use-Case ausgerichtet ist. Hier ist jedoch auch anzumerken, dass kein Live-Mitschnitt ausgewählter Testszenarien beabsichtigt ist, sondern eine nachträgliche Analyse von Testdaten, die erst im Verarbeitungsprozess in Szenarien aufgeteilt wird. Der Fokus liegt hier also weniger auf einer aktiven Unterstützung der V+V (*Phase II*), sondern eher dem Aufbau einer Datenbasis.

Gewährleistung von Sicherheit und Nachverfolgbarkeit der Datenverarbeitungsschritte in einem datengetriebenen Forschungsprozess.

Die Dokumentation beliebiger Zwischenschritte in der Datenverarbeitung (A12) wird nur in einer kleineren Teilmenge der vorgestellten verwandten Arbeiten betrachtet. Unter den klassischen Forschungsdatenmanagementsystem ist zumeist das Verwalten verschiedener Versionen eines Datensatzes möglich. Integrationen von dedizierten Datenprovenienzmodellen sind aber eher selten vorzufinden. Unter den DMS für Datenökosysteme wird diese Anforderung etwa durch den Life-Cycle-Composer von Elsayed und Brezany (2012), durch das Dokumentieren von Datenaustausch mittels eines Intermediärs in der IDS-Referenzarchitektur (z.B. durch die „Clearing House“-Komponente, die für das Überwachen aller Daten- und Finanztransaktionen zuständig ist) oder dem Data Exchange Logging Service von GAIA-X erfüllt. Bei anderen Ansätzen wie etwa dem Realtime Linked Dataspace ist dies nicht ohne weiteres möglich. In

weiteren Arbeiten werden außerdem Blockchain-Technologien eingesetzt, um Meta-Informationen zu den ausgetauschten Daten und dem Datenaustausch selbst nicht nur zu dokumentieren, sondern auch nachweisbar und fälschungssicher zu machen (A13), dies bringt allerdings weitere Anforderungen für den Betrieb einer Blockchain in einem Datenökosystem mit sich. In den Datenmanagementanwendungen in den untersuchten Testfeldern ist eine einheitliche Dokumentation der Datenverarbeitungsschritte grundsätzlich nicht vorgesehen. Lediglich VFBed speichert bei der Ausführung von Experimenten und der Erhebung von Daten einfache Logs. Die Ansätze von datAcron und Klitzke basieren auf statischen Datenverarbeitungsworkflows, die offen einsehbar sind, können aber dadurch frei konfigurierte Verarbeitungsschritte nicht abbilden. Da die meisten dieser Systeme eine einfache Dokumentation von Verarbeitungsschritten nicht unterstützen, kann Anforderung A13 ebenfalls nicht erfüllt werden. Der Erfüllungsgrad der Anforderungen zur Sicherheit dezentraler Testfelddatenquellen und der Datentransportwege (A14) lässt sich für konventionelle FDMS nicht beurteilen, da hier schon nicht von einer dezentralen Struktur ausgegangen wird, und daher einige Sicherheitsprobleme aus dem vorgestellten Testfeldkontext gar nicht betrachtet werden. Im Bereich der Datenökosysteme existieren dagegen bereits sehr ausgeprägte Methoden: So werden in der IDS-Referenzarchitektur (vgl. Otto, Steinbuß, und et al. 2019) Daten grundsätzlich nur verschlüsselt und nur an authentifizierte und autorisierte Parteien übertragen. Weiterhin existiert ein allgemeines Konzept zur Zertifizierung verschiedenster Komponenten in den IDS und GAIA-X Ökosystemen mit dem etwa Datenservices kontrolliert werden können. In den untersuchten Testfeldern sind Sicherheitsvorkehrungen in Bezug auf das Datenmanagement selten ein Thema, in SSAP2 werden allgemeine Standards für maritime Kommunikationskanäle genutzt, im STM Validation Testfeld wird eine Governance-Komponente durch den PortCDM Council eingeführt, der aus Personen mit bestimmten Kompetenzen besteht und die Aktivitäten im Testfeld überwachen soll und übertragene Daten werden verschlüsselt. Einige Testfelder bieten außerdem ein User-Management. Generelle Lösungsansätze zur Absicherung, Authentifizierung und Autorisierung für dezentral organisierte Daten (ähnlich zu dem IDS-Ansatz) werden in den Testfeldern aber bisher nicht angewandt.

Vereinfachung des datengetriebenen Forschungs- und Entwicklungsprozesses für maritime Testfelddaten.

In diesem letzten Abschnitt werden nun nur noch die maritimen Testfelder betrachtet, da sich die verbleibenden Anforderungen (A15-A18) explizit auf maritime Daten beziehen. Sowohl die klassischen FDMS als auch die DMS für Datenökosysteme bieten eher abstrahierte oder allgemeine Ansätze und sind i.A. nicht auf maritime Daten spezialisiert. Aufgrund der abweichenden Spezialisierung werden die Arbeiten aus dem Automobilbereich ebenfalls nicht betrachtet. Alle untersuchten Testfelder integrieren bestimmte maritime Datenquellen. Für die

Testfelder SSAP2 und STM Validation wird allerdings kein expliziter Mechanismus beschrieben, um beliebige weitere maritime Datenquellen ohne großen Integrationsaufwand für die aktive Nutzung im Testfeld einzubinden (A15). datAcron (vgl. Vouros u. a. 2018) setzt eine maritime Ontologie für bewegte Objekte zur allgemeinen Repräsentation unterstützter Datentypen ein: Zu Beginn des Verarbeitungsprozesses werden alle Rohdaten in das RDF-Format der Ontologie transformiert, was allerdings weiteren Implementierungsaufwand pro Datenquelle mit sich bringt. Auf diesem abstrahierten Modell können dann weitere Datenanalysen aufgebaut werden (vgl. Vouros u. a. 2018). In eMIR kann eine manuelle Transformation neuer Datenquellen in das S-100 Datenmodell durchgeführt werden, um Daten nativ in das Testfeld zu integrieren. Eine umfassende Beschränkung auf ein standardisiertes Datenmodell ist jedoch im dezentralen Kontext schwierig, da nicht vorhergesehen werden kann, welche Datenquellen zur Laufzeit eines FEDMS integriert werden müssen. Bei unbekannten Datenquellen wäre eine dynamische Integration nicht ohne weiteres möglich, da ein standardisiertes Datenmodell die Daten möglicherweise nicht ohne Anpassungen abbilden könnte. Dies würde manuelle Anpassungen erfordern und die Integration eines SuT in das Testfeld verzögern. Trotzdem ist es denkbar das ein bevorzugtes Datenmodell durch den Testfeldbetreiber unterstützt wird. eMIR und datAcron bieten zusätzlich einen klassischen Datenaufbereitungsprozess zur Verbesserung der Datenqualität an, sind aber spezifisch was die Architektur für Verarbeitungsschritte und Datentypen (z.B. Aufbereitung von AIS und RADAR Daten) betrifft. Die allgemeine Anwendbarkeit oder Erweiterbarkeit dieser Prozesse ist somit nicht gegeben (A16). Ein anpassbares speichereffizientes Datenübertragungsformat (A17) unterstützt keines der betrachteten Testfelder. SSAP2, STM Validation und eMIR nutzen zwar standardisierte Datenformate für maritime Daten, diese werden aber nicht auf Speichereffizienz evaluiert und sind ebenfalls nicht dynamisch anpassbar. Zu Latenzen und maximalen Datenübertragungsraten bei der Datenverarbeitung (A18) machen die Arbeiten grundsätzlich keine konkreten Angaben. Lediglich eMIR kann die Anforderungen zur Latenz erfüllen (M. Brinkmann 2018, 114).

3.5 Handlungsbedarf

Für die Einordnung in den Zusammenhang des (dezentralen) Forschungs- und Entwicklungsdatenmanagements wurden zunächst konventionelle FDMS und DMS in Datenökosystem betrachtet. Konventionelle FDMS können zwar grundlegende Anforderungen zur Zusammenführung von Daten aus verschiedenen Datenquellen erfüllen und bieten die Möglichkeiten weitere Metadaten zur Datenverarbeitungshistorie zu speichern, scheiden aber aufgrund ihrer Zentralisiertheit für einen Einsatz im dezentral organisierten Testfeld aus. Im Bereich der Datenökosysteme können die IDS-Referenzarchitektur und die GAIA-X Federation Services Anforderungen in Bezug auf Vertrauen und Sicherheit, also besonderen Interessen aus

der Datenproviderperspektive, erfüllen. Für den Einsatz im Forschungs- und Entwicklungszusammenhang gibt es bisher zwar vereinzelte Ansätze, die aber die Anforderungen der Anwenderseite (insbesondere der Datennutzer im Testfeld) durch mangelnde Methoden zur Live-Datenstromverarbeitung, insbesondere bei der Verarbeitung von Sensordatenströmen aus dem Testfeld und dem Szenario-orientierten Datenmanagement bzw. der Unterstützung von V+V in Testfeldern durch ein angepasstes Datenmanagement bisher nicht abdecken können. Diese betrachteten Arbeiten beziehen sich zum größten Teil auch auf den formalen Austausch von Daten im Kontext eines Datenhandels und weniger auf das Ermöglichen eines kollaborativen Systems Engineering Prozess, wie er im Testfeld stattfindet. Keine dieser Arbeiten ist somit auf die speziellen Bedürfnisse in maritimen Testfeldern ausgerichtet, die sich aus den drei Phasen der Testfelddatennutzung ergeben (vgl. Abschnitt 2.2.3). Insbesondere die allgemeinen bzw. abstrakten Frameworks (IDS und GAIA-X) können somit das letzte Ziel dieser Arbeit und Anforderungen aus dem Testfelddatenmanagement ohne signifikante Erweiterungen nicht erfüllen und befinden sich außerdem noch zu großen Teilen in der aktiven Entwicklung.

Schließlich wurden die direkt verwandten Arbeiten analysiert, die sich mit Datenmanagement für Testfelder beschäftigen. Selbst diese können das letzte definierte Ziel bisher nur teilweise erfüllen: So werden zwar einfache Prozesse zur Datenqualitätsverbesserung genutzt, eine allgemeine Anwendbarkeit bzw. Erweiterbarkeit auf architektureller Ebene ist jedoch nicht gegeben. Auch das modulare Anbinden neuer Datenquellen wird nicht ohne Weiteres unterstützt: Hier sind Transformationen notwendig, damit die restliche Testfeldinfrastruktur die Daten verarbeiten kann. Besonders im Bereich der dezentralen Verwaltung von Datenquellen kann die Wahrung von Datensouveränität für Testfelddatenprovider oder die Einhaltung von Datenschutzmaßnahmen im Testfeld, sowie die (sichere) Dokumentation von Verarbeitungsschritten nicht gewährleistet werden. Eine dynamische Organisation von Daten aus komplexen Workflows für Testszenarien ist zur Laufzeit ebenfalls nicht möglich. So können die untersuchten Testfeldarchitekturen unter Berücksichtigung eines dezentralen Kontextes nicht dazu eingesetzt werden Daten in einem Testfeld für externe Nutzer serviceorientiert bereitzustellen. Dies führt zu einer beschränkten Durchführbarkeit und Effizienz für kollaborative Forschungs- und Entwicklungsprozesse im beschriebenen Testfeldkontext. Schließlich ergibt sich also der Bedarf für ein neues FEDMS für Testfelder zu konzeptionieren, welches

- dezentral verwaltete, maritime Datenquellen modular einbinden kann und dabei die Souveränitäts- und Datenschutzansprüche der Testfelddatenprovider berücksichtigen kann.
- einen kollaborativen Forschungs- und Entwicklungsansatz in Testfeldern ermöglicht, indem sowohl historische als auch Live-Datenströme durch das System bereitgestellt

werden und Datenverarbeitungsschritte sicher und nachvollziehbar dokumentiert werden können.

- den Datenaustausch innerhalb des Testfeldes bezüglich der Bereitstellung einer Datengrundlage zur Modellbildung, der Unterstützung von Szenario-orientierter Verifikation und Validierung von Systemen und der sicheren Bereitstellung ausgewählter Daten für Demonstrationen und Zertifizierungsprozesse verwalten kann.

Der nachfolgende Teil dieser Arbeit beschäftigt sich mit der Konzeption, Umsetzung und Evaluation eines solchen FEDMS für Testfelder.

4 FEDMS-Architektur für maritime Testfelder

Im folgenden Kapitel soll nun eine Architektur entworfen werden, die dem in Abschnitt 3.5 empfohlenen Handlungsbedarf gerecht wird. Abbildung 23 zeigt den Zusammenhang zwischen den in den vorangegangenen Kapiteln diskutierten Themen: Für den Entwurf der Architektur wird davon ausgegangen, dass es sich beim betrachteten Systemkontext um ein Datenökosystem handelt. Wie in Abschnitt 2.6.1 begründet, wird von einem Data Space als konzeptionelles Modell ausgegangen. Schließlich wird eine Methode für den Entwurf der Architektur benötigt, die basierend auf den Anforderungen Lösungsmechanismen hervorbringt. Dazu wird nach einer kurzen Einordnung des Testfeldes als Data Space in Abschnitt 4.1, der Ansatz des schichtenbasierten Architekturentwurfes genutzt. Das nachfolgende Kapitel 5 beschäftigt sich dann mit der Implementierung und Umsetzung der entworfenen Architektur.

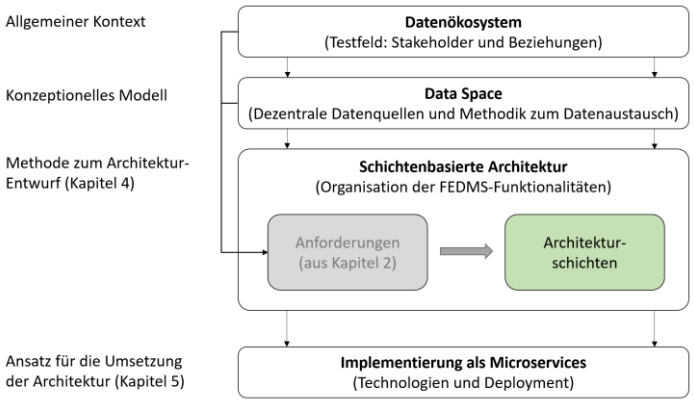

Abbildung 23: Weiteres Vorgehen zum Entwurf einer Architektur für ein Testfeld-FEDMS (vgl. Möller u. a. 2022).

4.1 Der Testfeld Data Space

Dass ein serviceorientiertes Testfeld aus der Datenmanagementsicht als Data Space modelliert werden kann, wurde bereits in Abschnitt 2.3 diskutiert. Im Folgenden sollen die Komponenten eines solchen Testfeld Data Spaces (TDS) nun genauer definiert werden und als Grundlage für den Entwurf eines FEDMS dienen. Für das unterstützende System im TDS wird das abstrakte Konzept der Data Space Support Platform nach Franklin, Halevy, und Maier (2005) und Curry (2020) als Modell genutzt, um ein System zu entwerfen, welches die in Abschnitt 2.6 definierten Anforderungen erfüllt. Im Hinblick auf Anforderung A5 soll jedoch sichergestellt werden, dass die typischen Funktionalitäten einer DSSP modular als Dienste durch verschiedene Stakeholder bereitgestellt werden können (vgl. z.B. Datentreuhänder). Da Anforderungen wie A9-A13 oder A15-A16 auf Hilfsfunktionen für Testfelddaten abzielen, wird das FEDMS Daten aber auch direkt verarbeiten müssen. Hierbei ist zu unterscheiden zwischen Funktionen, die lediglich auf

Metadaten basieren (z.B. sichere Dokumentation der Verarbeitungsschritte) und Funktionen, die die tatsächlichen Daten aus dem Testfeld verarbeiten. Letztere können durch das FEDMS nur als modulares Element zur Verfügung gestellt werden und im Fall eines Misstrauens oder der Nichtübereinstimmung von Anforderungen zwischen involvierten Stakeholdern durch dezentral bereitgestellte Verarbeitungsschritte ersetzt werden. In einem klassischen Aufbau könnte der Testfeldbetreiber als Datentreuhänder eingesetzt werden, der den Testfeldnutzern diverse Dienste mittels des FEDMS zur Verfügung stellt. Ein hohes Maß an Misstrauen zwischen Betreiber und Nutzer eines Testfeldes ist eher unwahrscheinlich, da die Nutzung eines privat betriebenen Testfeldes typischerweise durch einen Nutzungsvertrag geregelt wird (vgl. Hoque und Hasan 2020). Ein Misstrauen ist ggf. zwischen unterschiedlichen Testfeldnutzern zu erwarten, die in wirtschaftlicher Konkurrenz zueinanderstehen oder in Fällen, in denen besonders schützenswerte Daten übertragen werden.

Im Folgenden wird im TDS zwischen den Bereichen der maritimen Datenquellen, des FEDMS und Datenkonsumaktivitäten unterschieden, um das Verhältnis Datenprovider – Intermediär – Datenkonsument (vgl. Abschnitt 2.3.4) abzubilden. Abbildung 24 zeigt eine Übersicht über die drei Bereiche, die im Folgenden genauer erläutert werden.

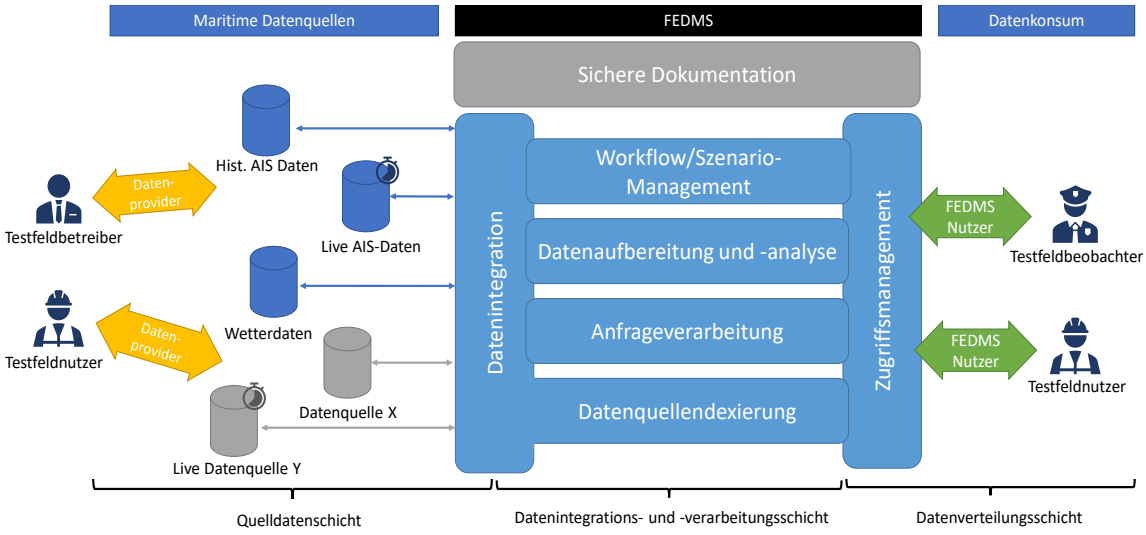

Abbildung 24: Übersicht über den Testfeld Data Space (TDS): Datenquellen, FEDMS und Konsum der Daten.

Maritime Datenquellen. Die maritimen Datenquellen stellen nach der Definition eines Data Spaces (vgl. Abschnitt 2.3.5) den Gesamtbestand der koexistenten Daten im Testfeld dar. Standardmäßig werden einige Datenquellen durch den Testfeldbetreiber verwaltet, welcher beispielsweise AIS-Daten über fest installierte Sensorik im Testfeld aufzeichnet, persistent speichert und für die Nutzer des Testfeldes zur Verfügung stellt. In diesem Zusammenhang sind beliebige Datenquellen, wie etwa Wetterdaten oder Dynamikdaten denkbar, die dem Testfeldnutzer als Service durch den Betreiber des Testfeldes bereitgestellt werden. Bei

komplexeren oder länger andauernden Testkampagnen ist es ebenfalls denkbar, dass ein Testfeldnutzer für seine Experimente eigene Datenquellen in das Testfeld integrieren möchte, um sie für verschiedene Subsysteme eines SuTs einfach verfügbar zu machen oder in einer Kollaboration mit anderen Testfeldnutzern kontrolliert zu teilen. Zudem ist zu beachten, dass die im Testfeld vorhandenen maritimen Datenquellen verschiedene Zwecke erfüllen können, die auf die verschiedenen Phasen des Testfelddatenmanagements abgestimmt sind: So sind neben statischen Datenquellen (z.B. Kartendaten), auch kontinuierlich aktualisierte Datenquellen oder Live-Datenquellen möglich, die für Tests im Testfeld benötigt werden. Zu erwartende Daten sind die in Abschnitt 2.4 diskutierten Klassen maritimer Daten. Es ist nach Curry (2020) aber davon auszugehen, dass die Datenquellen eine große Vielfalt an Datenformaten und Schnittstellen aufweisen, besonders, wenn sie durch verschiedene Akteure im Testfeld bereitgestellt werden.

FEDMS. Das Forschungs- und Entwicklungsdaten Management System stellt das zu entwerfende System dieser Arbeit dar. Es soll durch Akteure im Testfeld genutzt werden, um effizient und sicher auf die dezentral organisierten Datenquellen zugreifen zu können und hat ähnlich zu einer DSSP nach Curry (2020) das Ziel mehrere unterstützende Services im Data Space anzubieten, die die Integration von Datenquellen unterstützen, sodass sich Entwickler auf anwendungsspezifische Probleme fokussieren können. Im TDS entsprechen diese anwendungsspezifischen Probleme genau den in Abschnitt 2.2.3 definierten Phasen.

Für die Integration der Datenquellen in das FEDMS ist zunächst eine Datenintegrationskomponente notwendig, die von den unterschiedlichen nativen Technologien der Datenquellen abstrahiert und eine Verbindung zum FEDMS ermöglicht (vgl. Anforderung A1). Da von keiner statischen Menge von Datenquellen ausgegangen werden kann, ist eine Indexierung auf Datenquellenebene notwendig, um Datenanfragen den entsprechenden Quellen zuordnen zu können und Metadaten durchsuchbar zu machen (Anforderung A4). Weiterhin müssen Datenanfragen über das FEDMS an die einzelnen Datenquellen weitergeleitet werden können und die von den Datenquellen gelieferten Ergebnisse einheitlich an die Nutzer zurückgeliefert werden können (Anforderung A2). Mit A16 wurden Anforderungen bzgl. der Aufbereitung und Analyse von maritimen Daten gestellt. Diese Verarbeitungsschritte stellen ein weiteres Modul in der FEDMS-Architektur dar. Es ist jedoch ebenfalls zu erwarten, dass benutzerdefinierte Verarbeitungsschritte auch außerhalb des FEDMS ausgeführt werden und die Daten danach wieder durch das System verarbeitet werden (vgl. Anforderung A9). Sowohl das Zustellen von Daten aus den Datenquellen als auch das Verarbeiten muss live (zur Laufzeit einer Testdurchführung) und als Datenstrom möglich sein (A10). Ausgehend von dem Bedarf, Datenverarbeitungsworkflows verwalten zu können (A9) und insbesondere für das Szenario-orientierte Datenmanagement (A11) wird ein weiteres Modul benötigt. Um Souveränitätsansprüche zu erfüllen und Datenaustauschprozesse im TDS transparent und sicher

nachvollziehbar zu machen wird eine übergeordnete Dokumentations- und Monitoringkomponente eingeführt, die den Transfer von Daten von Ende-zu-Ende überwachen und dokumentieren soll (vgl. Anforderungen A5, A6 und A12, A13). Ein Rechte- und Rollenmanagement (A8) wird durch eine Datenverteilungsschicht umgesetzt, die das FEDMS nach außen absichert und alle Zugriffe auf Daten oder Funktionen authentifiziert und autorisiert (vgl. Anforderung A14). Nur über diese Schicht können Nutzer auf die FEDMS-Funktionalitäten zugreifen.

Datenkonsum. Akteure im Testfeld können das FEDMS nutzen, um Daten aus den dezentral organisierten Datenquellen abzurufen und in anwendungsspezifische Prozesse zu integrieren. So kann das FEDMS beispielsweise genutzt werden, um mit historischen Verkehrsdaten aus dem Testfeld ein Prädiktionsmodell zu trainieren und dieses später mit Live-Daten aus dem Testfeld zu evaluieren. Denkbar ist also sowohl eine Nutzung zum Abrufen der Testfelddatengrundlage als auch die Nutzung von Live-Daten aus dem Testfeld zur aktiven Verwendung in zu testenden Systemen. Zudem können die Datenkonsumenten das FEDMS für die Aufbereitung der Daten und als Kollaborationsmöglichkeit in der Verifikation und Validierung in Szenarien mit mehreren Akteuren im Testfeld nutzen. Dies wird insbesondere durch das Workflow- und Szenario-orientierte Management ermöglicht, welches Daten aus verschiedenen Quellen einem definierten Testszenario oder einem wissenschaftlichen Workflow zuordnen kann und in Abschnitt 4.4.2 und 4.4.3 näher beschrieben wird.

Der Feinentwurf der einzelnen Komponenten wird in den folgenden Unterabschnitten diskutiert.

4.2 Systemarchitektur des FEDMS

Im vorigen Abschnitt wurde der Aufbau des TDS diskutiert, in dem das FEDMS eine unterstützende Funktion erfüllen, und Datenmanagementfunktionalitäten für die Nutzer des Testfeldes zur Verfügung stellen soll. Der folgende Abschnitt gibt eine Übersicht über die Systemarchitektur des FEDMS, Details werden in den nachfolgenden Unterkapiteln diskutiert.

Die in Abschnitt 2.6 definierten Anforderungen wurden auf Grundlage verschiedener Stakeholder-Typen (Testfeldbetreiber, Testfeldnutzer, Datenprovider und Beobachter) erhoben. Um alle Anforderungen der Stakeholder zu erfüllen, aber zugleich zu vermeiden, dass eine zu hohe Komplexität die Umsetzung des FEDMS erschwert, wird eine schichtenbasierte Architektur verwendet. Hierbei erfüllt jede einzelne Architekturschicht eine bestimmte Aufgabe und ergibt sich als Gesamtheit ihrer Bestandteile, ihrer Komponenten. Dies hat den Vorteil der „Separation of Concerns“ (SoC) und trennt die verschiedenen Softwareschichten logisch voneinander. D.h., dass Verantwortlichkeiten klar unter den Schichten aufgeteilt werden und Entwicklungs-, Test-

und Verwaltungsaufgaben auf Aufgaben mit geringerer Komplexität heruntergebrochen werden können. (Richards 2015)

Die Aufgaben des FEDMS werden dabei in vier Schichten eingeteilt:

- **Quelldatenschicht**: Bildet die dezentral verwalteten Datenquellen im TDS und die Datenintegrationskomponenten des FEDMS ab.

- **Datenverarbeitungsschicht**: Beinhaltet alle Komponenten, die zum Verarbeiten und Übermitteln von Daten durch das FEDMS benötigt werden.

- **Datenverteilungsschicht**: Besteht aus den öffentlichen Schnittstellen des FEDMS und beinhaltet das Rollen- und Nutzermanagement.

- **Kontrollschicht**: Stellt Möglichkeiten bereit, um den Austausch von Daten sicher zu dokumentieren und an verifizierbare Identitäten zu knüpfen.

Abbildung 25 zeigt ein Komponentendiagramm der FEDMS-Systemarchitektur. Die Schichten sind voneinander getrennt und kommunizieren jeweils über standardisierte Schnittstellen. So können die Dezentralisierung der Quelldatenschicht erhalten werden, die Erweiterbarkeit oder Austauschbarkeit einzelner Schichten durch äquivalente Implementierungen gewährleistet werden und mehrere Instanzen durch verschiedene Stakeholder betrieben werden können (vgl. Abschnitt 4.7). Die zentrale Einheit ist in jeder Schicht durch eine Anfrageverarbeitungskomponente gegeben, die die Funktionsaufrufe von außerhalb einer Schicht interpretiert und an die zuständigen Komponenten innerhalb der Schicht weiterleitet. Schnittstellenaufrufe und Metadatentransfers sind als schwarze (innerhalb einer Schicht) und graue Pfeile (zwischen Schichten) dargestellt, der Fluss der eigentlichen Daten ist in orange dargestellt.

Durch FEDMS-Nutzer erreichbare Schnittstellen des FEDMS sind grün eingefärbt. Dies sind neben der öffentlich-verfügbaren Schnittstelle der Datenverteilungsschicht, eine Programmierbibliothek sowie eine graphische Benutzeroberfläche, die aber beide lediglich eine Abstraktionsschicht der Zugriffsschnittstelle darstellen, um eine einfachere Zugänglichkeit und Integration der FEDMS-Funktionalitäten zu ermöglichen. Eine Identitäts- und Zugriffsmanagement Komponente wird genutzt, um Anfragen zu verifizieren und unautorisierte Zugriffe zu verhindern. Nutzer können dabei lediglich über die Datenverteilungsschicht auf die Funktionen des FEDMS zugreifen, sodass jeder Zugriff einheitlich analysiert und dokumentiert werden kann. Eine Manipulation interner Prozesse durch externe Nutzer ist somit nicht möglich. Details zu dieser Schicht werden **in Abschnitt 4.5** erläutert.

Auf der anderen Seite setzt sich die Quelldatenschicht aus zwei Teilen zusammen: Zur Integration der Datenquellen werden Connectoren eingesetzt, die als Adapter zwischen dem FEDMS und der eigentlichen nativen Datenquelle fungieren, aber zur Erhaltung der Datensouveränität unter der vollen Kontrolle der jeweiligen Datenprovider stehen. So kann auf der Datenproviderseite genau überwacht und kontrolliert werden, welche Daten durch das FEDMS angefragt und an entsprechende Nutzer verteilt wurden. Das Gegenstück zu den Connectoren bildet die Connector Zugriffsschicht, welche die Verbindung des FEDMS zu den Connectoren verwaltet, Metadaten zu den einzelnen Datenquellen aggregiert und benutzerdefinierte Datenanfragen automatisch an verfügbare Datenquellen zuweisen kann. Hierzu werden eine Metadata Management Komponente sowie das Data Space Metadata Repository eingesetzt, die Informationen darüber liefern, welche Datentypen und -modelle durch die externen Connectoren abgerufen werden können, und wie diese erreichbar sind. Datenflüsse, oder Komponenten, die mit Datenflüssen arbeiten sind im Komponentendiagramm (Abbildung 25) orange markiert. Metadaten-Speicher sind lila gefärbt. Die Quelldatenschicht wird **in Abschnitt 4.3** genauer diskutiert.

Das zentrale Element des FEDMS bildet ein Streaming Cluster, der als Middleware verwendet werden kann, um die dezentral verwalteten Daten aus den Connectoren für Testfeldnutzer abrufbar zu machen. Nutzer können Anfragen in einer allgemeingültigen Anfragesprache stellen und erhalten das Resultat über den Streaming Cluster zurück. So können über die gleiche Schnittstelle sowohl Daten aus statischen Datenquellen gestreamt werden als auch Live-Daten, etwa aus Testfeldsensorik. Datenübertragungen mittels des Streaming Clusters werden dabei in sogenannte Topics gekapselt, die von der Topic Management Komponente verwaltet werden. Die Datenverarbeitungsschicht kann weiterhin für den Einsatz von internen Datenverarbeitungsschritten eingesetzt werden, die mit entsprechender Berechtigung direkt auf den Streaming-Cluster zugreifen und Daten (z.B. für Datenqualitätsaufwertungen) manipulieren können. Vervollständigt wird die Datenverarbeitungsschicht durch das Workflow & Scenario Management, welches die verarbeiteten Daten in einer Workflow-Struktur organisiert und entsprechenden realen Testszenarien zuordnen kann. Im Workflow & Scenario Repository wird außerdem eine sichere Datenverarbeitungshistorie dokumentiert, die das Nachverfolgen von Datenverarbeitungsprozessen ermöglicht. Details zu diesen Prozessen sind **in Abschnitt 4.4** nachzulesen.

Die Kontrollschicht ist keine unabhängige Schicht, sondern ergibt sich aus Zusatzfunktionalitäten der Datenverarbeitungs- und der Datenverteilungsschicht. Hierzu werden die Metadaten aus der Workflow & Scenario Management Komponente verwendet und an die Identitäten und Zugriffsrechte aus der entsprechenden Komponente der Datenverteilungsschicht gebunden. So werden den Workflows, Szenarien und weiteren Provenienzmetadaten die Identitäten der Nutzer zugeordnet, die Zugriff auf entsprechende Daten auf dem Streaming Cluster haben, oder diese

weiterverarbeitet haben. Zudem können alle Zugriffe auf das FEDMS zur Modifikation dieser Daten über die Datenverteilungsschicht dokumentiert werden und Datenverarbeitungsketten durch ihre Nutzer signiert werden. Die Kontrollschicht wird **in Abschnitt 4.6** näher erläutert.

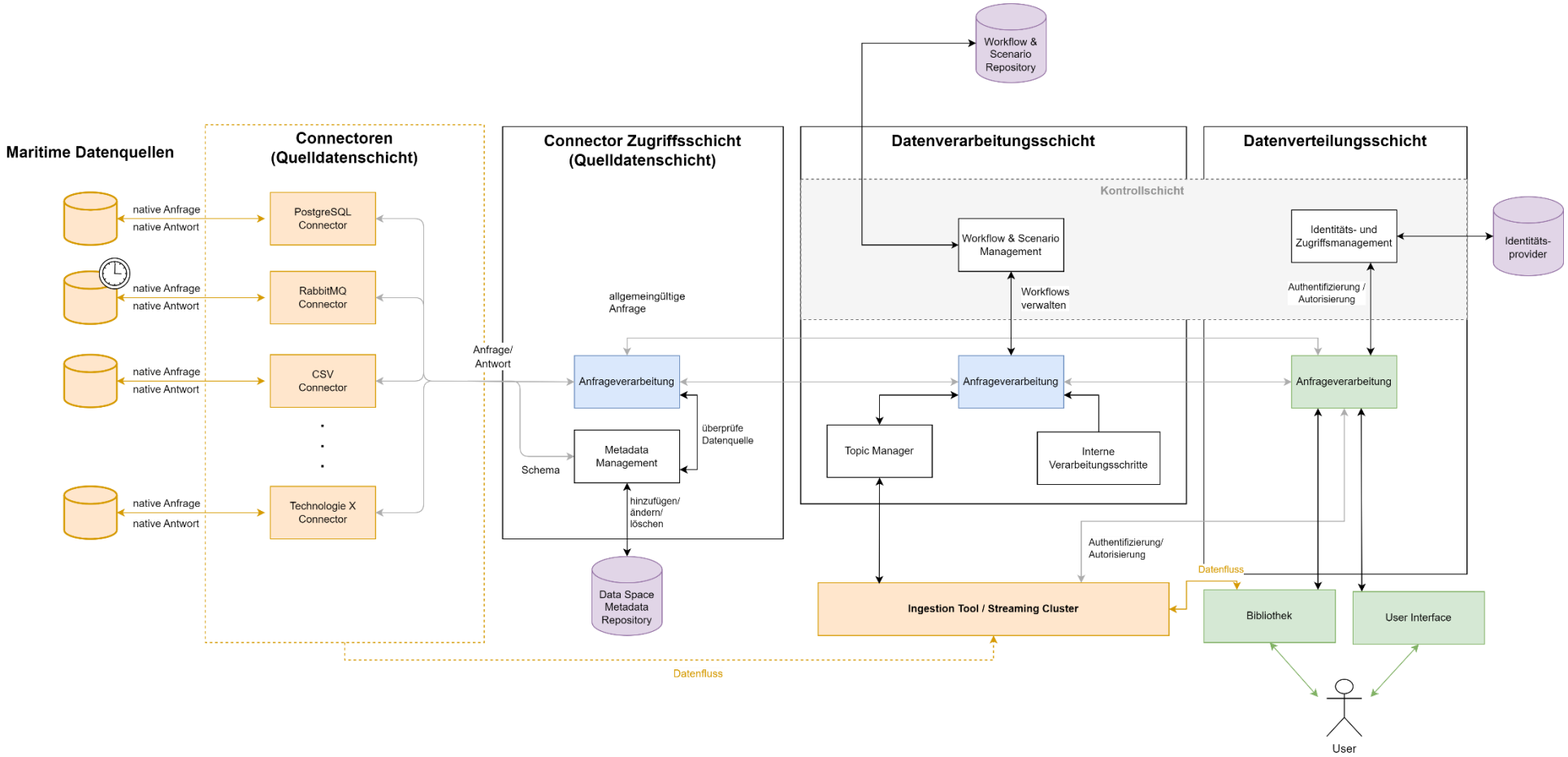

Abbildung 25: Systemarchitektur des Testfeld FEDMS: Datenquellen, Quelldatenschicht, Datenverarbeitungs- und Datenverteilungsschicht (vgl. Möller u. a. 2022).

4.3 Quelldatenschicht

Die Quelldatenschicht beinhaltet die Datenquellen des Testfeldes und notwendige Komponenten, um auf diese über das FEDMS zuzugreifen. Wesentliche Bestandteile dieser Schicht sind die Connectoren, die in Abschnitt 4.3.1 vorgestellt werden und eine Integration der Datenquellen unter der Kontrolle der Datenprovider gewährleisten. Das Gegenstück auf der Seite der FEDMS-Infrastruktur, die Connector Zugriffsschicht, wird in Abschnitt 4.3.2 diskutiert.

4.3.1 Connector

Für die Kommunikation zwischen dem FEDMS und den dezentral verwalteten Datenquellen muss eine Datenintegrationsschicht eingesetzt werden, um die technologische Heterogenität im TDS zu adressieren. Zudem muss die Souveränität der Datenprovider über ihre Daten gewahrt werden (vgl. Anforderung A6). Die Software-Artefakte, die diese Aufgabe spezifisch für jede Datenquelle erfüllen, werden als Connectoren bezeichnet. Sie müssen mehreren Anforderungen gerecht werden, um zwischen den nativen Technologien im TDS und den standardisierten Schnittstellen des FEDMS zu vermitteln. Das allgemeine Prinzip der Connectoren in einem Data Space wurde bisher insbesondere im Kontext des IDS geprägt und in GAIA-X adaptiert (vgl. Abschnitt 3.2). In der Norm DIN SPEC 27070:2020-03, die sich ebenfalls auf die IDS-Referenzarchitektur bezieht wird das Konzept des Connectors unter dem Namen „Security Gateway“ aufgegriffen und im wirtschaftlichen Zusammenhang definiert. Solch ein Gateway „*bindet die unternehmensinterne Infrastruktur an die Außenwelt an und steuert dabei Datenverarbeitung, Datenverdichtung, Zugriffskontrolle und Datenflüsse*.“ (Teuscher u. a. 2020, 9)

In der allgemeinen IDS-Referenzarchitektur werden die Connectoren also als Verbindung eines Akteurs (z.B. ein Unternehmen) zum Data Space beschrieben. Dabei ist ein bidirektionaler Austausch von Daten über die Connectoren vorgesehen. Im Kontext des Testfeldes bietet es sich an intensiv genutzte Datenquellen über einen solchen Connector bereitzustellen. In diesem Fällen handelt es sich jedoch meistens um Datenquellen, die durch den Testfeldbetreiber oder Stakeholder mit einem Interesse andere Testfeldnutzer auf längere Sicht mit Daten zu versorgen bereitgestellt werden. Dies sollte von der kollaborativen Bereitstellung von Daten für einen bestimmten Zweck (etwa durch einen System Engineer im Rahmen eines Systemtests) klar getrennt werden. Für diese Zwecke jedes Mal einen klassischen Connector zu konfigurieren, stellt einen zu hohen Aufwand dar. Lässt man dieses Szenario außen vor, lassen sich die meisten Stakeholder im Testfeld der Konsumentenseite solcher Connector-Daten zuordnen. Ein Trennen von Datenkonsum und Datenbereitstellung ergibt hier im Vergleich zum IDS Connector also

Sinn, um den Konfigurationsaufwand zu verringern und ungenutzte Schnittstellen zu vermeiden. Gleiches gilt für die Datenverarbeitungsfunktionalitäten, die typischerweise ebenfalls nicht vom Testfeldbetreiber genutzt werden. Lediglich das Abrufen der Daten aus der nativen Datenquelle, ihre Transformation und die Kontrolle der Datenflüsse von den Datenquellen zum FEDMS werden also durch den Connector umgesetzt.

Um den FAIR-Prinzipien des Datenmanagements gerecht zu werden und insbesondere die Zugänglichkeit und Interoperabilität (vgl. Abschnitt 2.3.2) der Daten im TDS herzustellen, wird für das Abrufen jeglicher Daten über das FEDMS eine allgemeine Anfragesprache verwendet, die unabhängig von den in den Quelldaten vorhandenen Technologien eingesetzt werden kann (vgl. Anforderung A3). Dies hat folglich den Vorteil, dass Daten aus verschiedenen Quellen einfacher zusammengeführt werden können und die Nutzer des FEDMS einen geringeren Implementierungsaufwand haben, um Daten automatisiert abfragen und weiterverarbeiten zu können. Um die verschiedenen Arten von maritimen Datenquellen im Testfeld abzudecken, wird eine Abfragesprache eingesetzt, die sowohl das Abfragen statischer Daten als auch das Konfigurieren von Datenströmen ermöglicht. Die zentrale Funktionalität eines Connectors liegt also darin eine solche allgemeine Datenanfrage anzunehmen und in eine Datenanfrage zu übersetzen, die durch die native Technologie der zugrundeliegenden Datenquelle verarbeitet werden kann und die zurückgegebenen Daten in einem standardisierten Format an das FEDMS zu übermitteln, damit sie für den anfragenden Nutzer bereitgestellt werden können. Um dabei die Datensouveränitätsanforderungen (Anforderung A6) der Datenprovider erfüllen zu können, ist es notwendig die Connectoren nicht zentral und getrennt von der restlichen Infrastruktur zu betreiben und dabei die üblichen Sicherheitsstandards einzuhalten (Anforderung A14). Daher werden die Connectoren – wie die nativen Datenquellen – dezentral von den zugehörigen Datenprovidern im Testfeld betrieben. Abbildung 26 zeigt den allgemeingültigen Aufbau eines Connectors. Zur Kommunikation mit den anderen Schichten des FEDMS und zur Beantwortung von Datenanfragen werden folgende Aufgaben durch einen Connector umgesetzt:

Bereitstellung einer standardkonformen Schnittstelle: Da die Implementierung der Connectoren außerhalb der Kontrolle des FEDMS liegen kann, muss eine standardisierte Schnittstelle implementiert werden, über die die Kommunikation mit den restlichen Schichten des FEDMS stattfinden kann. Hierzu zählen neben der Entgegennahme von Datenanfragen durch den Connector auch das Übermitteln einer Selbstbeschreibung (vgl. auch Otto, Steinbuß, u.a. 2019) in Form von Metadaten zum Status des Connectors und der angebundenen Datenquelle, damit dieser später für Nutzer einfacher auffindbar ist. Eine dafür benötigte standardisierte Schnittstellenspezifikation wird im TDS öffentlich verfügbar gemacht, damit neue Connectoren (auch durch Testfeldnutzer) entwickelt werden können.

Zugriffskontrollmodul: Über die Schnittstelle des Connectors kann festgestellt werden, welche Anfragen des FEDMS auf dem Connector ausgeführt werden, und welcher Nutzer diese Daten abfragt. Um die Souveränität des Datenproviders zu schützen und die Nutzungsvereinbarungen bzgl. seiner Daten durchzusetzen, kann ein benutzerdefiniertes Zugriffskontrollmodul im Connector eingesetzt werden, welches nicht autorisierte Zugriffe auf die Daten blockiert und somit eine zusätzliche Absicherung bietet.

Übersetzung der Datenanfrage und nativer Antwort: Die in der allgemeinen Anfragesprache verfasste Datenanfrage wird nach dem Entgegennehmen in eine native Anfrage transformiert, die von der eigentlichen Datenquelle beantwortet werden kann. Eine Transformation der gesamten Datengrundlage in ein global kompatibles Datenmodell ist nicht vorgesehen, da dies gerade für größere Datenquellen einen enormen Rechen- und Speicheraufwand mit sich bringen würde. Außerdem entstehen durch das direkte Anfragen der Datenquellen keine Inkonsistenzen, falls eine transformierte Kopie der Daten nicht auf dem aktuellen Stand der eigentlichen Datenquelle wäre. Nachdem die angefragten Daten durch die Datenquelle zurückgegeben wurden, werden sie in einer internen Repräsentation an die Schnittstelle zum Streaming-Cluster übergeben.

Interface zur Datenquelle: Spezifisch für die Entwicklungssprache des Connectors muss ein Interface bereitgestellt werden, welches das Senden von nativen Anfragen an die Datenquelle ermöglicht, und die Antwort auf diese Anfragen entgegennehmen kann.

Transformation und Schnittstelle zum Streaming-Cluster: Damit die Daten einheitlich weiterverarbeitet werden können, müssen Sie in ein Container-Format transformiert werden, das durch das FEDMS verarbeitet werden kann. Die eigentliche der Struktur der Daten bleibt dabei aber unverändert, um den Datenverarbeitungsprozess nicht durch den Verlust von Daten bei einer Modell-Transformation negativ zu beeinflussen. Die transformierten Daten werden dann über eine Schnittstelle zum Streaming-Cluster des FEDMS übermittelt.

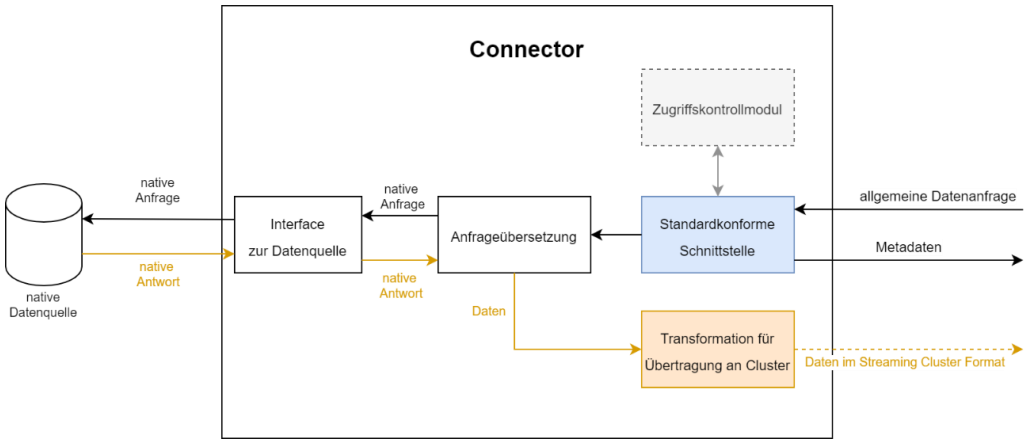

Abbildung 26: Architektur des FEDMS-Connectors inkl. der Datenanfrage und -Rückgabe.

Insgesamt entsteht so also ein dezentrales Netz aus Datenquellen im Testfeld, die durch ihren jeweiligen Connector vom FEDMS abgekapselt sind, es aber trotzdem ermöglichen Datenanfragen von Nutzern entgegenzunehmen, zu autorisieren und entsprechende Daten in einem FEDMS-kompatiblen Format für die weitere Verarbeitung zurückzugeben. Schließlich ist es auch möglich, den grundlegenden Aufbau eines Connectors zu erweitern. So ist es denkbar, vor der Transformation zur Übertragung an den Streaming-Cluster weitere Module zu installieren, die die ausgehenden Daten zuvor noch modifizieren. Insbesondere für die Umsetzung von Anforderung A7 ergibt es Sinn hier einen zusätzlichen Prozess zu etablieren, der abgefragte Daten automatisiert anonymisieren oder pseudonymisieren kann. So sind die zu schützenden Daten unter der vollen Kontrolle des Datenproviders und werden vor jeder Datenanfrage durch einen Nutzer anonymisiert, sodass rechtliche Rahmenbedingungen ebenfalls eingehalten werden können. In der Umsetzung ist hier eine einfache Pseudonymisierung durch das lokale Ersetzen von identifizierenden Attributen oder die Anonymisierung durch Generalisierung denkbar, um Eigenschaften wie k-Anonymity zu erreichen (Sweeney 2002). Wenn es um die gemeinsame Anonymisierung von Daten aus mehreren dezentralen Quellen geht, müssen komplexere Module in die Connectoren integriert werden, um beispielsweise eine gemeinsame Anonymisierung mittels eines Secure Multiparty Computation Protokolls wie in (Kohlmayer u. a. 2014) durchzuführen.

4.3.2 Connector Zugriffsschicht

Das Netz aus dezentralen Datenquellen als Teil des TDS zu verwalten und die Datenanfragen den richtigen Quellen zuzuweisen erfordert weitere Datenmanagementkonzepte, die durch die Connector Zugriffsschicht des FEDMS umgesetzt werden. Diese werden im folgenden Abschnitt erläutert.

In dieser Schicht kommen zwei Kernfunktionen zusammen, die auch durch eine klassische DSSP unterstützt werden sollten. ul Hassan, Ojo, und Curry (2020) argumentieren, dass es für die datenbasierte Entscheidungsfindung notwendig ist, Wissen über die verwendbaren Datenquellen zu haben. In Data Spaces sollten Katalogisierungsdienste eingesetzt werden, um verfügbare Datenquellen und Datensätze unabhängig von ihrer Beschaffenheit mittels Metadaten für die Nutzer einer DSSP auffindbar zu machen. Die Ausprägung solcher Dienste kann durch fünf verschiedene Ausprägungslevel kategorisiert werden. Ausgehend von einer einfachen Liste der Datenquellen (Level 1), können Metadaten über die Datenquellen hinzugefügt werden (Level 2), die maschinenlesbar zur Verfügung gestellt werden können (Level 3) und durch die Beschreibung von Beziehungen zwischen verschiedenen Datenquellen ergänzt werden (Level 4). Schließlich ist

eine umfassende, semantische Aufschlüsselung der Zusammenhänge im Data Space möglich (Level 5). (ul Hassan, Ojo, und Curry 2020)

Für die Anwendung im TDS ist abzuwägen, welches dieser Level erforderlich ist. Besonders Level 4 und 5, die bereits inhaltlich auf die Daten Bezug nehmen, sind sinnvoll, wenn eine sehr große Datengrundlage mit tausenden oder hunderttausenden verschiedenen Datensätzen vorliegt. Auf diese Weise können Daten einfacher exploriert werden und Zusammenhänge hergestellt werden, die für die weitere Nutzung der Daten von entscheidender Bedeutung sind. In einem einzelnen Testfeld ist die Menge der Datenquellen jedoch meist begrenzt – etwa durch Verfügbarkeit von Sensorik. Weiterhin ist zu beachten, dass bei einer Nutzung des Testfeldes oder der vorliegenden Datengrundlage meist schon ein zu testendes System mit fest definierten Anforderungen an die Datenquellen vorhanden ist. Somit ist eine explorative Analyse der verfügbaren Datenquellen häufig nicht mehr notwendig. Somit ist in der Architektur des FEDMS Level 3 der von ul Hassan, Ojo, und Curry (2020) vorgeschlagenen Kategorisierung für den Anwendungsfall ausreichend. Die durch Level 3 geforderte Maschinenlesbarkeit ist hier notwendig, um eine Suchfunktionalität umsetzen (vgl. Anforderung A4), die Datenquellen nach Metadaten filtern kann. Eine solche Datenquellensuche ermöglicht das schnelle Auffinden der Datenquellen und die effiziente Nutzung der verfügbaren Anfragemöglichkeiten für die Bereitstellung von Daten in SuTs oder zur Modellbildung.

Als zweite Funktionalität wird in der Zugriffsschicht das Zuordnen von Datenfragen zu den korrekten Datenquellen umgesetzt. Dies trägt zur Realisierung der Kernfunktionalität der zentralen Abfrage von Daten aus dem TDS bei (Anforderung A2). Das FEDMS stellt hiermit eine Abstraktion der dezentral verwalteten Datenquellen bereit, indem Nutzer die Daten über eine zentral erreichbare Schnittstelle abfragen können, ohne dabei zwangsläufig zwischen Schnittstellen von einzelnen Datenquellen unterscheiden zu müssen und stellt somit Interoperabilität im TDS her. Im Testfeld ermöglicht dieser Ansatz das schnelle Anbinden von Testfelddatenquellen an zu testende Systeme oder Modelle, während gleichzeitig der Integrationsaufwand von Datenquellen minimiert wird. Abbildung 27 zeigt den Prozess der Zuordnung von Datenanfragen zu den dezentral verwalteten Datenquellen im TDS: Jegliche Datenanfragen werden in einer standardisierten und allgemeinen Datenanfragesprache formuliert und über die Datenverteilungsschicht (siehe Abschnitt 4.5) an die Schnittstelle der Connector Zugriffsschicht weitergeleitet. Mittels Informationen zum authentifizierten Nutzer des FEDMS wird dann geprüft, für welche Datenquellen dieser die Zugriffsrechte besitzt. Für die so definierte Menge an Datenquellen werden aus einem Metadatenrepository Informationen zu den Abfrageschemas der Datenquellen abgerufen (übermittelte Selbstbeschreibung der Connectoren) und eine Validierung der Datenanfrage bezüglich dieser Schemas durchgeführt. Falls ein Schema die Anfrage unterstützt, wird diese zur Beantwortung an den zugehörigen Connector

weitergeleitet. Im Beispiel aus Abbildung 27 werden die Anfragen A und B durch die Anfragezuordnung interpretiert. Beide Anfragen fragen die gleiche Art von Daten ab (RadarTargetState), Anfrage B zielt allerdings auf ein Abonnieren der Daten (subscription) in einem Datenstrom ab, während Anfrage A eine Eingrenzung der Daten auf einen absoluten Bereich in einer Datenbank enthält und somit nicht von einem Live-Datendienst beantwortet werden kann. Dies wird durch das Validieren der Datenanfragen anhand der Schemas der Datenquellen erkannt, und die Anfragen können dem richtigen Connector zugeordnet werden.

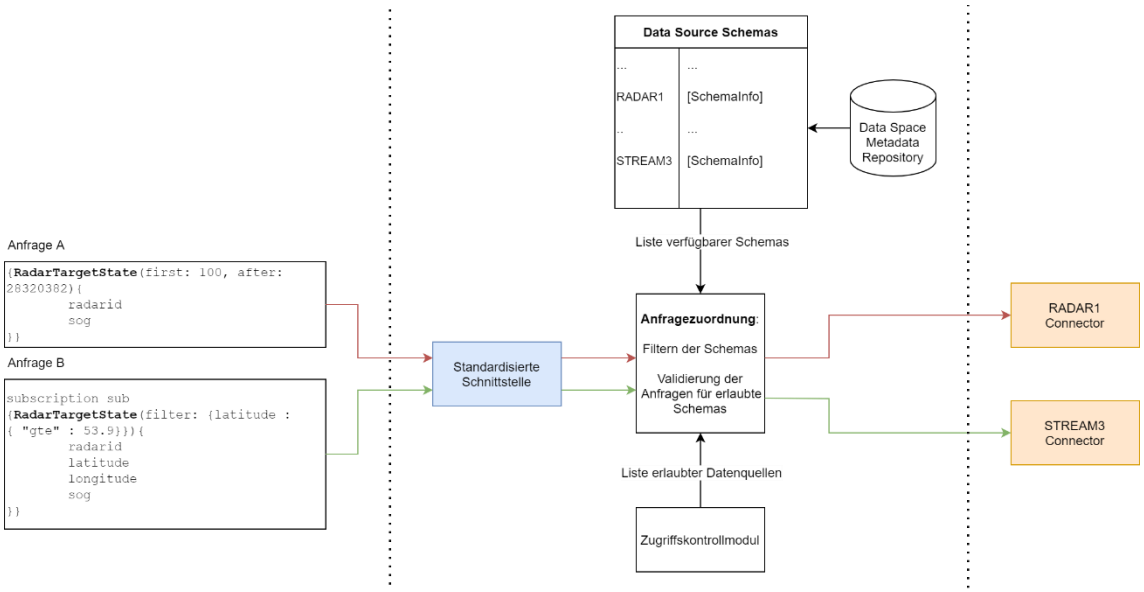

Abbildung 27: Beispiel zur Zuordnung der Datenanfragen zu den dezentral verwalteten Connectoren durch die Connector-Zugriffsschicht.

Der gesamte Aufbau der Connector Zugriffsschicht ist in Abbildung 28 visualisiert und setzt sich aus den folgenden Komponenten zusammen:

Standardisierte Schnittstelle: Wie in Abschnitt 4.2 diskutiert, stellt eine Anfrageverarbeitungskomponente das zentrale Element jeder Architekturschicht dar. Diese nimmt die Anfragen anderer Schichten entgegen und leitet sie zur Bearbeitung innerhalb der Schicht weiter.

Anfragezuordnung: Wie oben erläutert, führt die Anfragezuordnungskomponente auf Basis der gespeicherten Schemas eine Zuordnung einer Datenanfrage durch, prüft ob Zugriffsberechtigungen für den anfragenden Nutzer bestehen, und leitet die Anfrage ggf. weiter.

Suchfunktionalität: Für den Katalogisierungsdienst wird die Suche genutzt, um mittels einer Stichwortsuche die Metadaten über die verfügbaren Datenquellen zu durchsuchen und Ergebnisse zurückzuliefern.

Zugriffskontrollmodul: Diese Komponente wird eingesetzt, um zu prüfen auf welche Datenquellen der mit der eingehenden Datenanfrage assoziierte Nutzer berechtigt Zugriffsrechte besitzt. Bei einer Suche nach Datenquellen werden detaillierte Informationen ebenfalls nur angezeigt, wenn Nutzer Zugriffsrechte besitzen.

Data Space Metadata Repository (DSMR): Das DSMR ist eine Datenbank, die jene Metadaten speichert, die durch die Connectoren als Selbstbeschreibung an die Zugriffsschicht bei ihrer Registrierung beim FEDMS übermittelt wurden.

Metadata Service: Der Metadata Service verwaltet das DSMR. Er erfüllt dabei zum einen die Funktion einer Einbindung der Datenbanktechnologie des DSMR in die restliche Architektur der Zugriffsschicht, und kontrolliert zum anderen alle Lese- und Schreiboperationen.

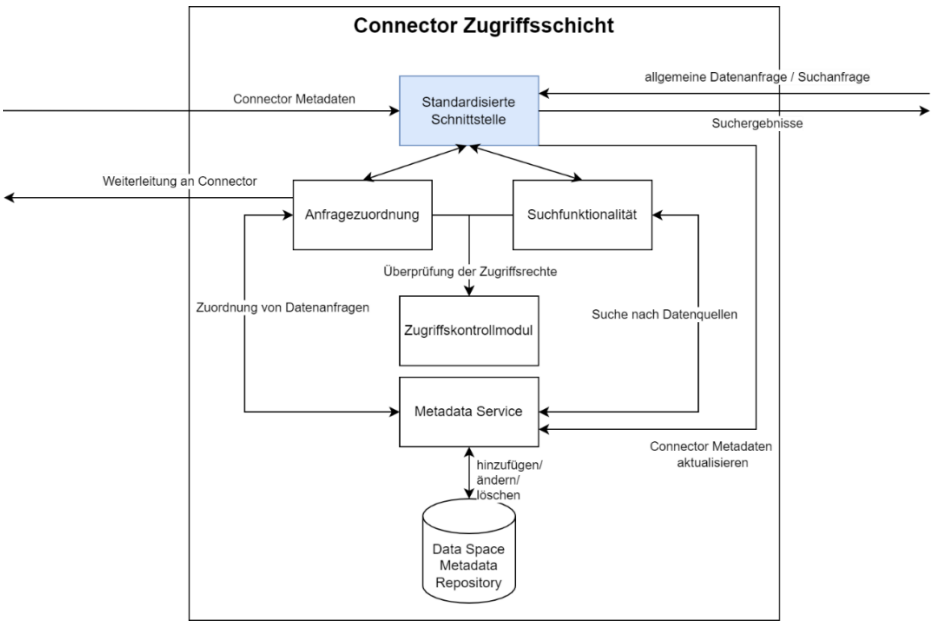

Abbildung 28: Aufbau der Connector Zugriffsschicht.

Die Kommunikation der Zugriffsschicht mit den Connectoren (vgl. Abbildung 29) erfordert aufgrund der dezentralen Verwaltung zunächst eine Registrierung des Connectors bei der Connector Zugriffsschicht, um Metadaten, wie den Host-Namen, den Port, den Namen, eine Beschreibung und das Schema des Connectors auszutauschen. Nachdem diese Selbstbeschreibung im DSMR gespeichert wurde, kann ein Nutzer eine Daten- bzw. Suchanfrage stellen. Da der Nutzer die FEDMS-Funktionalitäten nur über die Datenverteilungsschicht erreicht (siehe Abschnitt 4.5), werden sowohl Daten als auch Suchanfragen von hieraus an die Connector Zugriffsschicht weitergeleitet. Bei einer Datenanfrage findet die Anfragezuordnung statt und die Datenanfrage wird an den entsprechenden Connector weitergeleitet. Anfragen für die Datenquellensuche können direkt durch den Zugriff auf das DSMR beantwortet werden, bei Datenanfragen stellt der Connector die Daten dem Nutzer über den Streaming Cluster zur Verfügung.

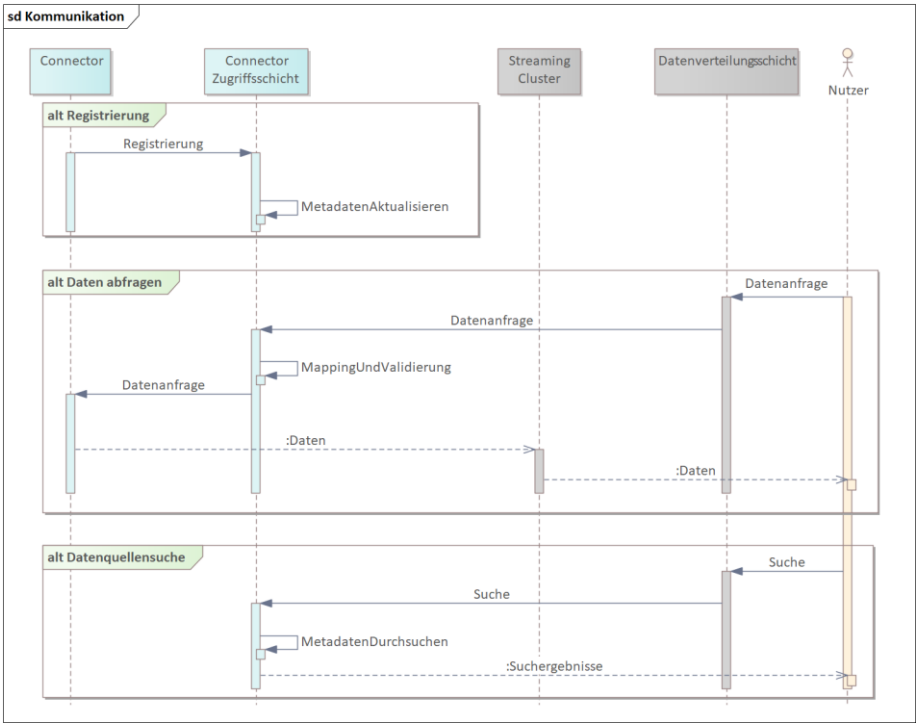

Abbildung 29: Kommunikation der Connector Zugriffsschicht mit anderen Schichten des FEDMS.

4.4 Datenverarbeitungsschicht

Die Datenverarbeitungsschicht stellt die Kernkomponente des FEDMS dar und ist dafür zuständig den eigentlichen Datenaustausch zwischen Datenprovider und Datenkonsument zu realisieren, zu vereinfachen und zu dokumentieren. Als Kernelement dient ein Streaming Cluster, dessen Funktionsweise in Abschnitt 4.4.1 vorgestellt wird und der durch ein Modell zur Verwaltung von Datenverarbeitungsketten (Workflows) ergänzt wird (siehe Abschnitt 4.4.2). Basierend darauf werden das speziell auf das Testfeld angepasste Szenario-orientierte Datenmanagement in Abschnitt 4.4.3 und die FEDMS-unterstützte Datenverarbeitung (Abschnitt 4.4.4) diskutiert. Daraufhin folgt eine Betrachtung spezieller maritimer Datenverarbeitungsprozesse in Abschnitt 4.4.5.

4.4.1 Datenstromverarbeitung

Für das Speichern, Verarbeiten, Analysieren und Visualisieren von Big Data sind in der wissenschaftlichen Community bereits viele etablierte Ansätze vorhanden. Im vorgestellten Testfeldkontext werden diese Aufgaben zwar teilweise dezentral und durch verschiedene Stakeholder durchgeführt. Durch die bereits in Abschnitt 4.3 entworfene Abstraktionsschicht zur Datenintegration, werden die dezentralen Datenquellen jedoch zentral verfügbarer und ein

Vergleich mit etablierten Big Data Architekturen zur Verarbeitung und Analyse der Daten ist möglich.

Wenn man die Fähigkeit zur Verarbeitung von Live-Datenströmen (Anforderung A12) berücksichtigt, finden sich zwei besonders häufig verwendete und etablierte Architekturen in der Literatur. Als klassischere Variante dieser beiden gilt die sogenannte Lambda Architektur (siehe Abbildung 30 (a)). Hierbei wird der eingehende Datenstrom auf einen „*Batch Layer*“ und einen „*Speed Layer*“ verteilt. Der Batch Layer ist für die Speicherung und Analyse eines Master-Datensatzes zuständig. Analysen auf diesem großen und vollständigen Datensatz werden über längere Zeit ausgeführt und nach Fertigstellung verfügbar gemacht. Da dies aber normalerweise verhältnismäßig lange dauert, werden durch den Speed Layer Analysen für eine kleinere Menge von aktuellen Daten durchgeführt und bereitgestellt, um die Verzögerung durch den Batch Layer auszugleichen. (Feick, Kleer, und Kohn 2018)

Der Nachteil der Lambda-Architektur liegt in der Synchronisation von Batch und Speed Layer. Sowohl auf der Datenebene stellt dies Probleme dar als auch in der Weiterentwicklung der Code-Basis, die sich für Batch und Speed Layer unterscheidet. Eine mögliche Lösung für diese Probleme bietet die Kappa-Architektur (siehe Abbildung 30 (b)), in welcher kein Batch Layer verwendet wird, und eine Abwandlung des Speed Layers auch das Verarbeiten größerer Datenmengen ermöglicht. Durch den Einsatz von speziellen Ingestion Tools (bzw. Streaming Cluster Plattformen) wie Apache Kafka im Speed Layer, wird das temporäre Speichern des eingehenden Datenstroms ermöglicht. So können durch eine alleinstehende Code-Basis sowohl historische als auch aktuelle Daten verarbeitet und analysiert werden. Kappa-Architekturen werden folglich eingesetzt, wenn Live-Datenströme in der Datenverarbeitung eine wichtige Rolle spielen. (Kalipe und Behera 2019)

Der stärkere Fokus der Kappa-Architektur auf die Verarbeitung von Live-Datenströmen ist auch im TDS wiederzufinden, in welchem zur Laufzeit von Tests ein aktives Datenmanagement notwendig ist (vgl. *Phase II* in Abschnitt 2.2.3). Die testfelddatenbasierte Modellbildung und Demonstration von zu testenden Systemen (Phasen I und III) sind hingegen nicht zeitkritisch, bzw. haben meist nicht die Anforderung riesige Mengen an Daten kontinuierlich aufzuzeichnen und zur gleichen Zeit auszuwerten. Der Einsatz eines Speed Layers (vgl. auch dezentralisierte Daten-Workflows in Abschnitt 4.4.2) mit der Möglichkeit der temporären Speicherung von Daten für andauernde Analysen bietet sich hier also insgesamt an. Ein Unterschied zur Kappa-Architektur im klassischen Einsatzgebiet ist allerdings dadurch gegeben, dass kein statischer Datenstrom verarbeitet wird. Das Ziel des FEDMS ist es nicht vordefinierte Analysen auf immer gleichen Daten auszuführen, die etwa einen zur Laufzeit des Systems bereits bekannten Entscheidungsprozess unterstützen, sondern benutzerdefinierte Datenfragen von Testfeldnutzern entgegenzunehmen und zu beantworten. Somit kann bereits der eingehende Datenstrom vom

Nutzer definiert und die Verarbeitung von nicht benötigten Daten ausgeschlossen werden, sodass unautorisierte Zugriffe auf andere Daten aus dem Testfeld gar nicht erst möglich sind. Bei der im FEDMS eingesetzten Variante handelt es sich also um eine *anfrageorientierte* Version der Kappa-Architektur (siehe Abbildung 30 (c)). Weiterhin lässt sich der Speed-Layer im FEDMS als dezentralisiertes Element interpretieren: Die eigentlichen Datenverarbeitungsschritte werden dezentral durch die Stakeholder des Testfeldes implementiert.

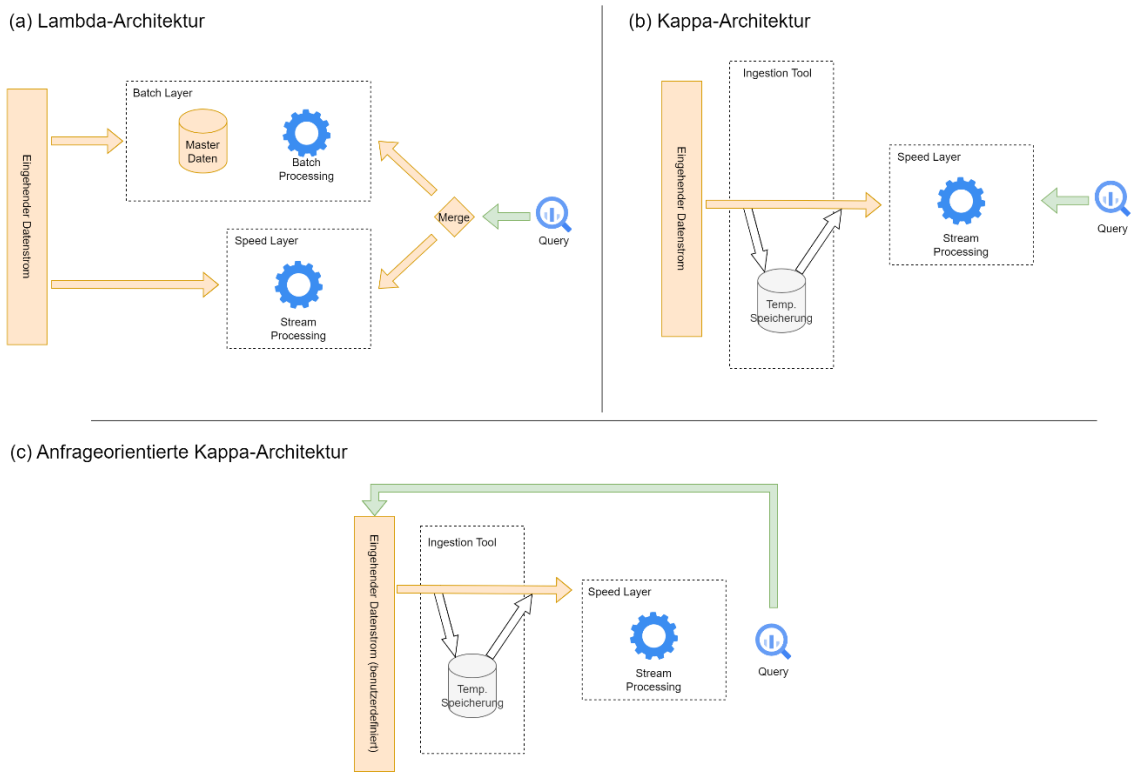

Abbildung 30: Mögliche Datenarchitekturen für das FEDMS: (a) Lambda-Architektur, (b) Kappa-Architektur, (c) Anfrageorientierte Kappa-Architektur (vgl. Kalipe und Behera 2019).

Typische Ingestion Tools wie Apache Kafka (vgl. Vohra 2016) oder Google Cloud Pub/Sub (vgl. Krishnan und Gonzalez 2015, 277ff.) sind als nachrichtenbasierte Publish/Subscribe Systeme aufgebaut. Diese Systeme bilden eine Middleware für den Nachrichtenaustausch zwischen Provider (auch: Publisher, der Produzent einer Nachricht) und Consumer (auch: Subscriber, der Konsument einer Nachricht) (Krishnan und Gonzalez 2015, 278). Zwischen Producern und Consumern entsteht durch das Ingestion Tool eine n:m Verbindung zur Übermittlung von Informationen (vgl. Krishnan und Gonzalez 2015, 278). Um größere Mengen von Nachrichten besser zu organisieren, werden die Nachrichten innerhalb der Tools mittels sogenannter Topics verwaltet (siehe Abbildung 31): Ein Topic ist eine logische Einheit von thematisch verwandten Nachrichten (Wu, Shang, und Wolter 2019). So könnten im TDS etwa Nachrichten über ein Topic übertragen werden, die nur Daten aus einer speziellen Datenquelle enthalten und für einen

bestimmen Nutzer zur Verfügung gestellt werden sollen. Systeme wie Apache Kafka können dann durch die Replikation von Topics in einem Cluster skaliert werden (Wu, Shang, und Wolter 2019).

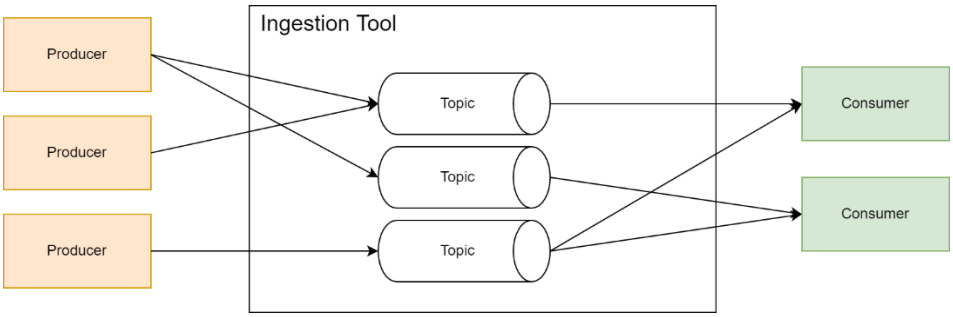

Abbildung 31: Datenaustausch mittels Ingestion Tool: Nachrichtenbasiertes Consumer/Producer Pattern (vgl. Krishnan und Gonzalez 2015, 278).

4.4.2 Datenverarbeitungsketten im Testfeld

Die verwendete Kappa-Architektur unterstützt das Anfragen und Entgegennehmen von Daten aus den dezentralen Datenquellen des Testfeldes. Weiterhin bleibt jedoch die Frage, wie eine Vielzahl von darauffolgenden Verarbeitungsschritten, die unter der Kontrolle diverser Nutzer durchgeführt werden, in das FEDMS integriert und verwaltet werden können (vgl. Anforderung A9). Um die Gesamtheit eines solchen Prozesses zu überwachen, ist es zunächst notwendig, einzelne Verarbeitungsschritte im Kontext einer Modellbildung, Verifikation und Validierung oder Demonstration eines Systems im Testfeld abbilden zu können.

Für den folgenden Teil wird ein *Verarbeitungsschritt* als die Anwendung einer Transformation auf eine oder mehrere Datenartefakte durch einen klar definierten Stakeholder im Testfeld bezeichnet, die als Ergebnis ein neues Datenartefakt generiert. Ein *Datenartefakt* ist dabei eine endliche Menge oder ein Strom von Datentupeln (vgl. Gedik, Turaga, und Andrade 2014), die in einem gemeinsamen Zusammenhang genutzt werden. Folglich ist es möglich, dass ein Verarbeitungsschritt nicht terminiert, sondern auf einem kontinuierlichen Strom von Daten ausgeführt wird. Ein Beispiel hierfür wäre das Aufbereiten eines AIS-Datenstroms für ein Assistenzsystem, welches periodisch AIS-Daten zur Auswertung des aktuellen Kollisionsrisiko eines Schiffes benötigt. Terminierende Verarbeitungsschritte lassen sich etwa bei der Auswertung von historischen Verkehrsdaten finden: Nach der Verarbeitung des gesamten Datensatzes steht ein Ergebnis bereit und alle Verarbeitungsschritte sind beendet. Wie diese Beispiele zeigen, können sich mögliche Verarbeitungsschritte im TDS stark unterscheiden: Sowohl in der Art der Transformation als auch bzgl. des Systems, durch welches ebendiese ausgeführt werden. Eine Zentralisierung aller Verarbeitungsschritte auf der Infrastruktur des FEDMS ist daher nicht möglich und würde den Entwicklungsprozess vom Testfeld unabhängiger Systeme behindern.

Die Integration von Datenverarbeitungsprozessen (Anforderung A9) ist also nur durch das Verwalten der Datenartefakte als Verarbeitungsergebnisse zwischen Verarbeitungsschritten in einem Workflow und der Speicherung von Meta-Informationen zu den durchgeführten Verarbeitungsschritten umsetzbar.

Der in Abschnitt 4.4.1 vorgestellte Streaming Cluster (Ingestion Tool), welcher das Kernelement zum Datenaustausch in der FEDMS-Architektur darstellt, muss also genutzt werden können, um Datenartefakte aus verschiedensten Daten-Workflows im Testfeld zu verwalten. Jedes Datenartefakt kann an die Verwendung eines separaten Topics im Streaming Cluster gekoppelt werden. Producer sind in diesem Fall die Systeme, die einen Verarbeitungsschritt im TDS ausführen und das Ergebnis über das FEDMS teilen. Consumer sind dann Systeme, die diese Ergebnisse vom FEDMS abrufen. In einem kollaborativen Workflow mit mehreren geteilten Datenartefakten schließt die Durchführung eines Verarbeitungsschrittes also sowohl das Konsumieren als auch das Produzieren ein (Abbildung 32). Einen Ausnahmefall bilden die Datenprovider, die keinen Datenverarbeitungsschritt ausführen, sondern nur die angefragten Daten über einen Connector in einem Topic bereitstellen. Im Sinne des Stream Processings kann jedes Topic als eine eigene Datenstromquelle interpretiert werden, auf die (außerhalb des FEDMS) verschiedene Verarbeitungsoperatoren angewandt werden können (vgl. Gedik, Turaga, und Andrade 2014).

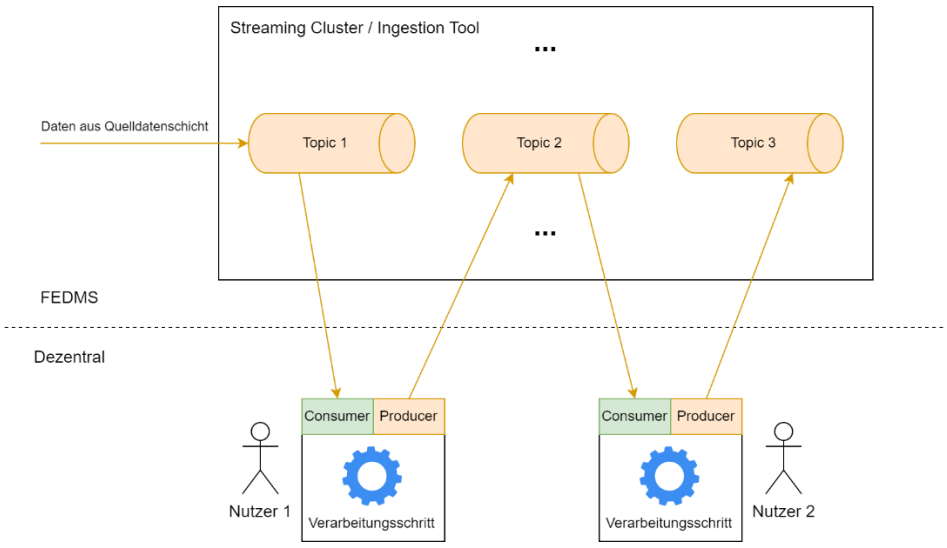

Abbildung 32: Nutzung von Topics für die Verwaltung von Datenartefakten im Datenworkflow.

Eine Bereitstellung von allgemeinen, vordefinierten Stream Processing Operatoren durch das FEDMS selbst ist jedoch nicht vorgesehen: Hier ist davon auszugehen, dass etablierte Toolchains durch die Nutzer des Testfeldes außerhalb des Systems eingesetzt werden (vgl. auch Anforderung A9). Für das Verarbeiten im Batch-Modus kann auch ein endliches Datenartefakt über das Topic abgerufen werden: Nach der Übertragung des letzten Datentupels endet der Datenstrom und der

im Batch zu verarbeitende Datensatz wurde vollständig übertragen. Zusammenfassend stellt das FEDMS mittels des Streaming Clusters also nur eine Infrastruktur zur Übertragung der Zwischenstände in einem Datenworkflow bereit. Die Verknüpfung der Topics durch Verarbeitungsschritte wird von den Nutzern des FEDMS selbst umgesetzt.

Um nach Vollendung eines Datenworkflows nachvollziehen zu können, welche Daten verarbeitet wurden, können so zwar die Zwischenstände zwischen den Verarbeitungsschritten nachvollzogen werden, bei einer Kollaboration mehrerer Teilnehmer in einem Workflow (z.B. bei einem Integrationstest zweier kooperierender Systeme) ist es jedoch nicht immer für jeden einzelnen Teilnehmer möglich daraus abzuleiten, wie die Daten tatsächlich verarbeitet wurden. Dies ist jedoch für die Vergleichbarkeit und Reproduzierbarkeit eines Workflows notwendig (vgl. Anforderung A12). Da eine zentrale Überwachung aller Datenverarbeitungsschritte nicht umsetzbar ist (siehe oben), muss hier auf Metadaten zur Beschreibung dieser Prozesse zurückgegriffen werden. Des Weiteren sollen auch benutzerdefinierte Metadaten gespeichert werden können, um etwa Nutzungsbedingungen für ein Datenartefakt zu dokumentieren (vgl. Anforderung A5).

Das *Open Provenance Model (OPM)* ist ein technologieunabhängiges Provenienzmodell, mit dem es möglich ist Metadaten von Datenverarbeitungsschritte abzubilden. Hierzu wird eine graphenbasierte Repräsentation verwendet, die zwischen drei verschiedenen Knotenarten unterscheidet: Ein *Artefakt* beschreibt einen unveränderlichen Zustand eines Objektes, ein *Prozess* ist eine Aktion, bzw. eine Reihe von Aktionen die auf Artefakten ausgeführt wird und neue Artefakte produziert, *Agenten* sind Entitäten, die im Kontext eines Prozesses auf diesen (z.B. durch das Kontrollieren oder Auslösen) einwirken. Abhängigkeiten zwischen diesen Entitäten werden in einem gerichteten Graphen durch Kanten ausgedrückt. OPM unterstützt außerdem das Abbilden diverser weiterer Eigenschaften dieser Knoten durch das Hinzufügen weiterer Metadaten und Beschreibungsregeln, die in einer quelloffenen Spezifikation erläutert werden. (Moreau u. a. 2011)

Um also die vollständige Provenienz von Datenworkflows einheitlich abbilden zu können, ist es notwendig Metadaten über die Artefakte und vor allem die Verarbeitungsschritte zu erheben. Für diese Aufgabe wird das OPM eingesetzt. Ein Artefakt in OPM beschreibt im TDS ein Datenartefakt (repräsentiert durch ein Topic), ein Prozess kann genutzt werden, um Metadaten über Verarbeitungsschritte zu erheben und ein Agent repräsentiert den ausführenden Nutzer eines Verarbeitungsschrittes. Ein *Workflow* ergibt sich also aus der Gesamtheit aller Topics in einer Datenverarbeitungskette und den zu den Verarbeitungsschritten zugehörigen Metadaten. Schließlich ergeben sich drei Nutzungsprofile der Datenverarbeitungsschicht mit der vorgestellten Architektur im Testfeld (siehe vereinfachte Darstellung in Abbildung 33):

Reihenkonfiguration von Workflows. In dieser Konfiguration wird das FEDMS als Middleware zwischen verschiedenen Datenverarbeitungsschritten genutzt. Jeder Verarbeitungsschritt bezieht die benötigten Datenartefakte aus einem (oder mehreren) Topic(s) und schreibt resultierende Artefakte ebenfalls wieder auf Topics. Eine direkte Kommunikation zwischen Systemen die Datenverarbeitungsschritte ausführen findet nicht statt. Diese Konfiguration kann bevorzugt in Situationen eingesetzt werden, in denen kein endgültiges System getestet wird. Die zentrale Nutzung des FEDMS ermöglicht hier eine einfache und schnell umsetzbare Kommunikationsmöglichkeit für verschiedene dezentral verwaltete Teilsysteme. Außerdem kann diese Konfiguration Anwendung in Fällen finden, in denen eine vertrauenswürdige, zentrale Instanz zum Austauschen der Daten benötigt wird (vorausgesetzt die involvierten Akteure vertrauen der Bereitstellung des FEDMS). Da das FEDMS hier aber als Teil der zu testenden Teilsysteme genutzt wird, ist eine Nutzung für die Evaluation finaler Systeme nicht empfehlenswert (vgl. z.B. *Phase III*).

Parallele Workflowkonfiguration. In dieser Konfiguration wird das FEDMS hauptsächlich in seiner Funktion der Überwachung und Analyse von Datenworkflows genutzt. Systeme, die Verarbeitungsschritte in einem Workflow ausführen kommunizieren direkt miteinander und schreiben lediglich eine Kopie von resultierenden Datenartefakten auf Topics. In den eigentlichen Datenverarbeitungsprozess wird somit nicht durch die FEDMS-Infrastruktur eingegriffen. Die Datenartefakte und Metadaten auf dem FEDMS können später analysiert werden, um beispielsweise einen Fehler im Datenworkflow zu lokalisieren oder um erweiterte Monitoring-Funktionalitäten zu implementieren. Diese Konfiguration eignet sich daher besonders für die Nutzung in *Phase II* und *III*.

Exklusiver Datenabruf. In dieser Konfiguration wird lediglich ein Topic genutzt, um angefragte Daten einem dezentral verwalteten (Teil-)System bereitzustellen. Nicht immer ist es notwendig Datenworkflows zu überwachen und nachzuverfolgen. Für das reine Abrufen von Daten aus dem TDS ist diese Konfiguration mit dem wenigsten Implementierungsaufwand verbunden.

Schließlich sind auch hybride Varianten möglich, in denen zu testende Teilsysteme Reihenkonfigurationen nutzen, während andere Teilsysteme im selben Workflow parallel konfiguriert werden. Theoretisch ist es auch denkbar, dass keine Datenquelle aus der Quelldatenschicht genutzt wird, sondern nur Datenartefakte aus den zu testenden Systemen ausgetauscht werden. Weiterhin ist anzumerken, dass die Darstellung in Abbildung 33 vereinfacht ist: Es können natürlich auch mehrere Topics als eingehende Datenartefakte eines Verarbeitungsschrittes genutzt werden. Zur Verwaltung der Topics und der Workflows werden zwei Services eingeführt (siehe auch Abschnitte 5.4 und 5.5), die auf Basis von Nutzeranfragen und unter Berücksichtigung entsprechender Rechte die Topics und Workflowmetadaten erstellen, ändern und entfernen können (vgl. Abbildung 25).

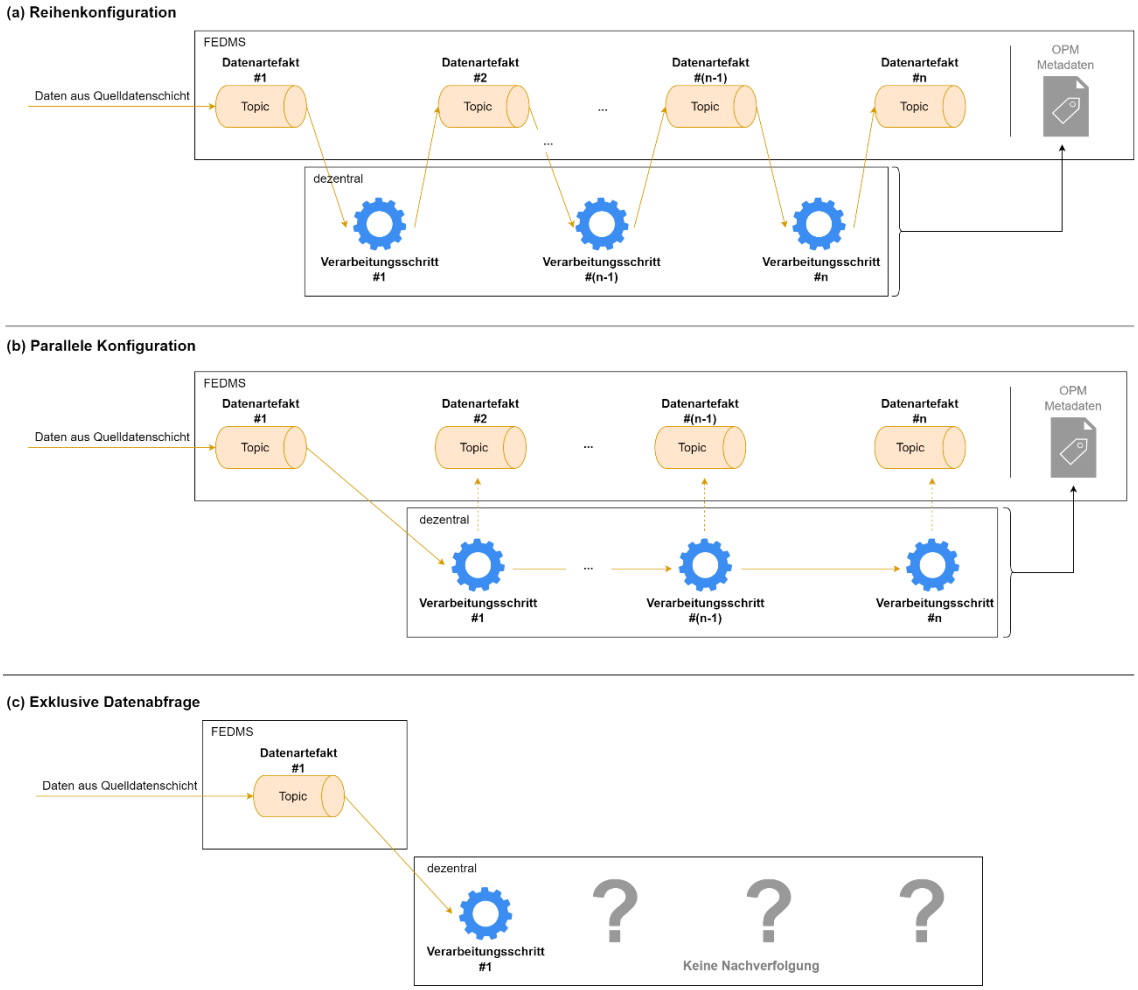

Abbildung 33: Workflowkonfigurationen im TDS mit Unterstützung des FEDMS: (a) Reihenkonfiguration, (b) Parallele Konfiguration und (c) Exklusive Datenabfrage.

4.4.3 Szenario-orientiertes Datenmanagement

In Abschnitt 2.2.3 wurde das Szenario-basierte Testen als wichtiger Bestandteil von *Phase II* des Testfelddatenmanagements vorgestellt. Im Folgenden wird insbesondere auf diesen Abschnitt und Abbildung 5 Bezug genommen, um ein Konzept zum Szenario-orientierten Datenmanagement abzuleiten.

Beginnend mit der Szenariogenerierung ist anzumerken, dass diese bereits bei der Auswahl der Datengrundlage abhängig vom zu testenden System ist. Hierzu muss festgelegt werden, welche Teilmengen vorhandener Daten genutzt werden, und nach welchem Maß relevante Situationen aus den Daten extrahiert werden sollen. Ein Beispiel aus dem maritimen Bereich liefern Lamm und Hahn (2018) mit der Extraktion von Beinaheunfällen aus maritimen Beobachtungsdaten wie AIS- und RADAR-Daten (siehe Abbildung 34). Zunächst lässt sich erkennen, dass sich der Prozess durch einen klassischen, linearen Datenworkflow mit Verarbeitungsschritten,

Datenartefakten und somit durch das Modell aus Abschnitt 4.4.2 abbilden lässt. Bis auf Abschnitt II. (vgl. Abbildung 34) sind die Verarbeitungsschritte allerdings spezifisch auf das Erkennen von Situationen mit hohem Kollisionsrisiko ausgerichtet, um damit bspw. Assistenzsysteme testen zu können, die im Bereich der Kollisionsvermeidung angesiedelt sind (Lamm und Hahn 2018). Dazu werden wie von Neurohr u. a. (2020) beschrieben, anwendungsfallspezifische Kritikalitätsmaße eingesetzt. Diese Verarbeitungsschritte sind also nicht ohne Weiteres allgemein für die Generierung beliebiger Szenarien aus Testfelddaten einsetzbar. Die Vorverarbeitung der Daten beinhaltet jedoch allgemein anwendbare Verarbeitungsschritte auf üblichen maritimen Datentypen, wie das Herausfiltern von Ausreißern oder das Interpolieren von fehlenden oder falschen Werten (Lamm und Hahn 2018). Für die Unterstützung von Szenario-Generierung und auch für die Anwendung in anderen Anwendungsfällen des Testfeldes werden in Abschnitt 4.4.4 FEDMS-interne Verarbeitungsschritte eingeführt und in Abschnitt 4.4.5 konkrete maritime Datenverarbeitungsschritte diskutiert, die bei einer größeren Anzahl von Anwendungsfällen eingesetzt werden können. Der Workflow zum Generieren von Szenarien aus historischen Daten kann also durch das FEDMS unterstützt, aber nicht vollständig übernommen werden.

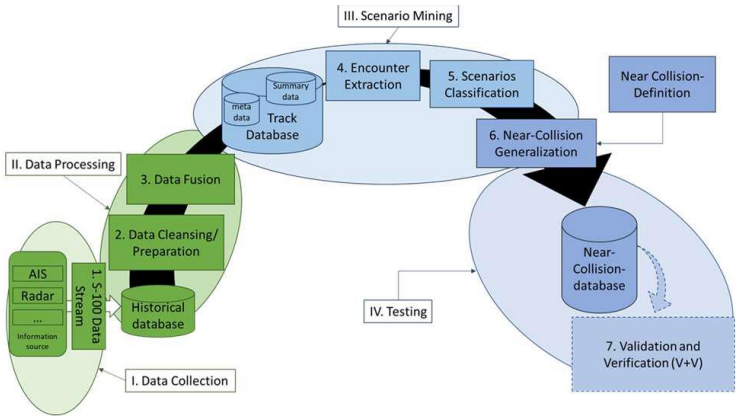

Abbildung 34: Workflow – Erkennung von Beinaheunfällen von Schiffen (Lamm und Hahn 2018).

Weitere relevante Aufgaben im Zusammenhang des Szenario-basierten Testens, die durch ein Testfeld-FEDMS unterstützt werden können, sind die Testdurchführung und die anschließende Auswertung von aufgezeichneten Daten (vgl. Anforderung A11). Durch die in Abschnitt 4.4.1 vorgestellte Kappa-Architektur ist ein Einsatz von FEDMS-basierten Funktionalitäten auch zur Laufzeit eines Testes bzw. der Durchführung eines Szenarios im Testfeld möglich. Im Folgenden wird dazu die Durchführung und Auswertung Szenario-basierter Testfälle aus der Datenperspektive betrachtet. Hierfür wird angenommen, dass sich der Datenverarbeitungsprozess (von der Entgegennahme von Sensordaten bis zur Ausgabe von resultierenden Daten) eines zu testenden Systems als Workflow modelliert lässt. Nach Ulbrich u. a. (2015) ist es von hoher Relevanz ein standardisiertes Verständnis der Begriffe *Szenario, Situation* und *Szene* zu haben, um Kontextmodellierung und Verhaltensplanung von automatisierten System zur Steuerung von

Fahrzeugen einheitlich beschreiben zu können. Die von Ulbrich u.a. vorgeschlagenen Definitionen dienen hier als Grundlage für die weitere Modellierung aus der Datenmanagement-Perspektive im Testfeld. So wird eine Szene als *„eine Momentaufnahme des Umfelds, welche die Szenerie, dynamische Elemente, die Selbstrepräsentation aller Akteure und Beobachter wie auch die Verknüpfung dieser Entitäten umfasst. (...)*“ (Ulbrich u. a. 2015, 2) beschrieben. Eine Situation hingegen ist „*die Gesamtheit der Umstände, die für die Auswahl geeigneter Verhaltensmuster zu einem bestimmten Zeitpunkt zu berücksichtigen sind. (...)*“ (Ulbrich u. a. 2015, 6). Die Situation ist somit subjektiv aus der Perspektive eines Systems definiert, welches seine Umgebung wahrnimmt und nach definierten Zielen mögliche Aktionen bewertet. Schließlich lässt sich ein Szenario als „*die zeitliche Entwicklung von Szenenelementen innerhalb einer Folge von Szenen, welche mit einer Startszene beginnt (...)*“ (Ulbrich u. a. 2015, 10) beschreiben.

Zunächst lässt sich feststellen, dass eine Situation die Ziele und internen Repräsentationen von Verhaltensmustern eines zu testenden Systems enthält. Solch eine subjektive Einordnung einer Szene lässt sich durch das Beobachten interner Zustände eines Systems im Testfeld analysieren, ist daher für eine allgemeine Modellierung von Szenario-basierten Testdurchläufen aus Datenmanagementsicht aber nicht relevant, da das Datenmanagement innerhalb dieser Systeme nicht Aufgabe der Testfeldinfrastruktur ist. Die im Modell von Neurohr u. a. (2020) beschriebenen Szenarien lassen sich durch die Definition von Ulbrich u.a. nun in kleinere Bestandteile - die Szenen - aufteilen. In einer diskreten Variante lässt sich ein Szenario also als Abfolge von Szenen beschreiben.

Bei der Durchführung von Testszenarien wird aus einer existierenden Szenariorepräsentation bzw. Szenariobeschreibung (aus dem Prozess der Szenariogenerierung) eine neue *Instanz* dieses Szenarios erstellt. Aus Datensicht entsteht also etwa aus einer Ablaufbeschreibung eines Testes eine Instanziierung eines Workflows durch das Erzeugen von Datenartefakten. Wird die Datensicht auf ein SuT durch einen Workflow modelliert (vgl. Konfigurationsmöglichkeiten in Abschnitt 4.4.2), so beschreibt eine Szenarioinstanz also die Menge an Datenartefakten (und Metadaten), die während der Durchführung eines Szenarios erhoben werden. Dies ist in Abbildung 35 dargestellt: Die genaue Art der Datenartefakte ist von einer Testspezifikation abhängig. Dabei muss die Abbildung interner Zustände im SuT (schematische Darstellung in Abbildung 35: graue Boxen und Pfeile als Komponenten und Datenflüsse) so stattfinden, dass die benötigten Daten zur Auswertung eines Testes und zur Feststellung eines Testergebnisses dokumentiert werden. Hierbei werden Systemkomponenten oder Prozessen im SuT Datenartefakte im FEDMS zugeordnet. Weitere Faktoren können benötigte Informationen sein, die eine bessere Nachvollziehbarkeit des Systemverhaltens ermöglichen. Dafür können nicht nur durch das SuT erzeugte Datenartefakte eine Rolle spielen, sondern auch Beobachtungsdaten aus

der Systemumgebung, um z.B. die korrekte Reaktion des Systems auf bestimmte Umgebungsbedingungen zu überwachen.

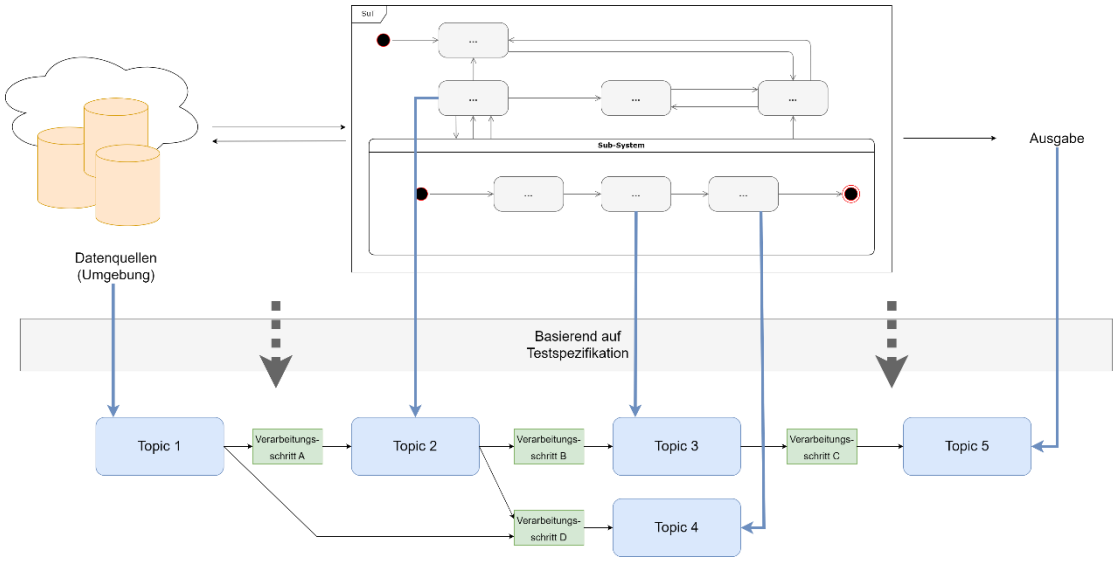

Abbildung 35: Modellierung eines Datenworkflows für ein SuT auf Basis einer Testspezifikation (schematische Darstellung) (vgl. Möller u. a. 2022).

Die durchgehende Aktivität eines zu testenden Systems während der Durchführung einer Reihe von Testläufen stellt die Herausforderung eines kontinuierlichen Datenstroms dar, in dem Testszenarien nicht klar voneinander getrennt werden. Das nachträgliche Extrahieren von Testszenarien aus einer Aufzeichnung eines solchen Datenstromes ist meist sehr aufwendig und erfordert eine genaue Analyse. Durch die Abbildung des SuT-Verhaltens auf einen Datenworkflow innerhalb des FEDMS ist jedem Datenartefakt ein separates Topic zugeordnet. Dies erleichtert zwar die Aufgabe der Analyse durch das Trennen unterschiedlicher Datenverarbeitungszustände, vereinfacht aber nicht die Aufteilung der Datenströme in einzelne Testfälle, bzw. Szenarioinstanzen. Zur Lösung dieses Problems ist es notwendig zusätzliche Metadaten bzgl. der Durchführung eines Testszenarios zur Testlaufzeit zu erheben. Das Einteilen eines Datenstroms in verschiedene Szenarien wird im Folgenden durch die Nutzung von Markern realisiert. Zu Beginn einer Durchführung eines Testszenarios sendet der Testleiter ein Signal an das FEDMS, welches den Start eines bestimmten Testszenarios kennzeichnet. Nach Vollendung eines Testes wird ebenfalls ein Signal an das System übermittelt, welches das Ende eines Szenariodurchlaufes markiert (dargestellt in Abbildung 36): Diese Marker werden auf jedes Datenartefakt eines Workflows angewendet, indem für das zugehörige Topic eine Referenz auf das, zu diesem Zeitpunkt aktuelle, Tupel gespeichert wird. So bleibt der eigentliche Aufbau (etwa im Gegensatz zu einer Neuzuweisung von Topics pro Szenario) unbeeinflusst und die Zuordnung von Datenartefakten zu Topics konsistent. Wichtig ist hier, dass die Daten in solchen Topics persistent gespeichert, und nicht nach der Zustellung an Consumer gelöscht werden. Für eine

spätere Abfrage der Daten einer Szenarioinstanz ist eine Referenz außerdem effizienter nutzbar als bspw. ein Zeitstempel, nach dem das gesamte Topic erst durchsucht werden müsste.

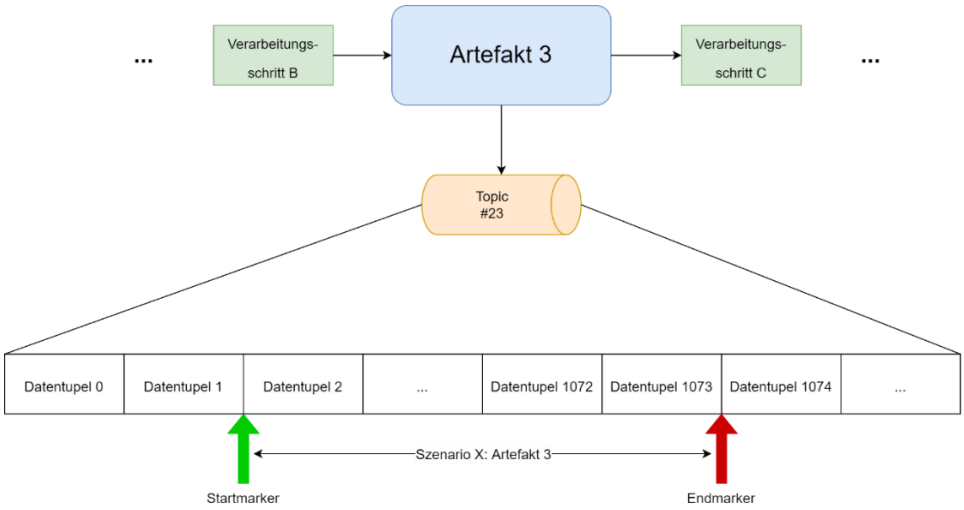

Abbildung 36: Hinzufügen von Szenariometadaten zu Datenartefakten: Start- und Endmarker.

Schließlich ergibt sich eine Szenarioinstanz also als eine Reihe von Start- und Endmarkerpaaren, die auf Daten in den Topics der Datenartefakte zeigen. Durch diese kann effizient und zielgerichtet auf verschiedene Datenartefakte zugegriffen werden und es können gewünschte Daten zu einer Szenarioinstanz extrahiert und exportiert werden. Ein erneutes Abspielen einer Szenarioinstanz inkl. einer Reproduktion des SuT-Verhaltens ist in der Reihenkonfiguration ebenfalls möglich, indem die assoziierten Topics nicht von der aktuellen Nachricht an (also „live“) gelesen werden, sondern beginnend beim Startmarker und die originale Zeitdifferenz zweier Nachrichten beachtet wird.

Eine einzelne Szene stellt per Definition eine Momentaufnahme eines Szenarios zu einem bestimmten Zeitpunkt dar (siehe oben). Diese kann durch das Durchsuchen der Topics zwischen den Markern extrahiert werden, indem aus jedem Topic das Datentupel abgerufen wird, welches die geringste zeitliche Differenz zur gewünschten, zeitlich definierten Momentaufnahme aufweist. Die Gesamtheit der Tupel aus den verschiedenen Topics aus einem Workflow stellt dann die gewünschte Szene dar.

4.4.4 FEDMS-gestützte Datenverarbeitung

Im Abschnitt 2.4 wurden typische Datenquellen eines maritimen Testfeldes betrachtet, und es wurde dargelegt, dass AIS-Daten einen Großteil der in der datengetriebenen maritimen Forschung verwendeten Daten ausmachen. Wie auch das Beispiel von Lamm und Hahn (2018) aus dem vorigen Abschnitt zeigt, sind die Einsatzbereiche solcher Daten zwar vielfältig, bestimmte Verarbeitungsschritte, wie die Datenvorverarbeitung, aber auch das Bewerten von Datenqualität,

oder etwa das Augmentieren von lückenhaften Datensätzen (beispielsweise durch Interpolation (vgl. z.B. D. Zhang u. a. 2017)) kommen dabei allerdings häufiger zum Einsatz. Der Bedarf für wiederverwendbare Lösungen in diesen Bereichen spiegelt sich auch in Anforderung A16 wider. Damit solche Verarbeitungsschritte nicht für jeden Workflow im Testfeld neu implementiert werden müssen, soll das FEDMS eine Schnittstelle bereitstellen, mit der vordefinierte Verarbeitungsschritte instanziiert und in einen Workflow integriert werden können. Um die Wiederverwertbarkeit solcher allgemeinen Verarbeitungsschritte zu gewährleisten, sollten Parameter von verwendeten Algorithmen für die Nutzer eines Verarbeitungsschrittes konfigurierbar sein. Ansonsten erfüllt ein durch das FEDMS bereitgestellter Verarbeitungsschritt die gleichen Anforderungen wie ein externer Verarbeitungsschritt und unterscheidet sich nur durch die Tatsache, dass dieser auf der FEDMS-Infrastruktur ausgeführt wird. In Abbildung 37 ist dargestellt, wie dies innerhalb des FEDMS ermöglicht werden kann: Dazu wird ein Verarbeitungsschritt durch die Datenverteilungsschicht automatisiert bereitgestellt, und kann dann durch den Nutzer instanziiert gestartet und gestoppt werden. Nach dem Starten beginnt der Verarbeitungsschritt auf den vorkonfigurierten Topics Daten zu lesen und schreibt das Resultat nach Ausführung des parametrisierten Verarbeitungsschrittes auf ein neues Topic, welches vom Nutzer ausgelesen werden kann. Durch das Stopp-Signal, oder nachdem alle Daten ausgelesen wurden, terminiert der interne Verarbeitungsschritt. Aufgrund der dynamischen Nutzbarkeit des Streaming Clusters können Daten während der Laufzeit des Verarbeitungsschrittes als Input hinzugefügt werden, und Teilresultate bereits vor Fertigstellung gelesen werden. So lassen sich entsprechende interne Verarbeitungsschritte auch live einsetzen. Es ist ebenfalls denkbar, dass Dritte ähnliche Konzepte implementieren, um auf ihrer Infrastruktur proprietäre Verarbeitungsschritte anzubieten.

Die Erforschung allgemeiner Verarbeitungsschritte für beliebige maritime Datenarten ist nicht Teil dieser Arbeit, da die verwendeten Algorithmen unabhängig der hier entworfenen Datenmanagementstrategie implementiert werden können. Trotz dessen beschäftigt sich der folgende Abschnitt mit dem Entwurf von FEDMS-gestützten Verarbeitungsschritten für den speziellen Bereich der AIS-Daten, da diese am häufigsten in der maritimen, datengetriebenen Forschung genutzt werden (siehe Abschnitt 2.4). Dies gilt der Verdeutlichung des Aufbaus von den Verarbeitungsschritten aus den Bereichen der Datenqualitätsbewertung und Datenvorverarbeitung. Da Methoden aus diesen Bereichen immer kontextabhängig sind, ist es notwendig hier einen konkreten Aufbau zu beschreiben.

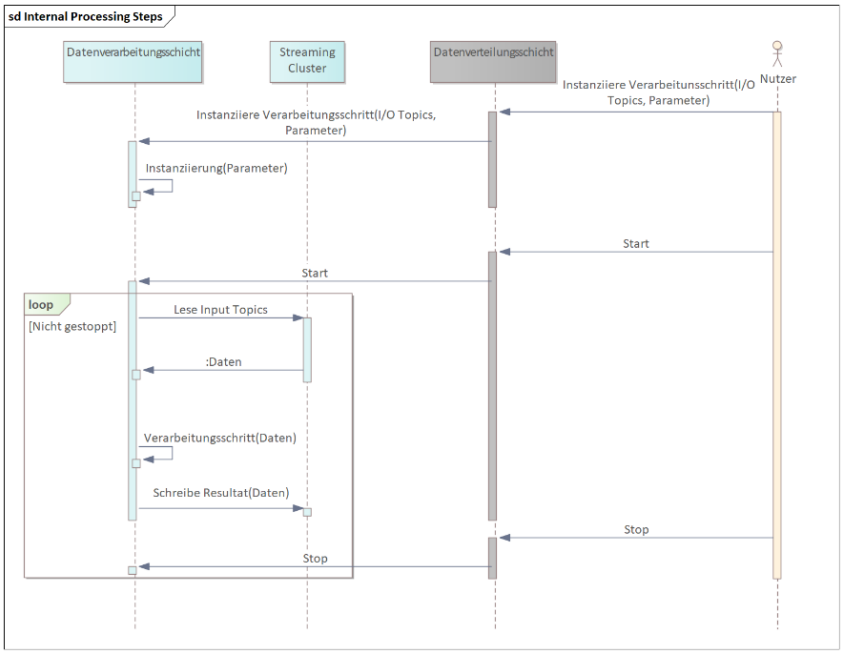

Abbildung 37: Instanziierung interner Verarbeitungsschritte auf der FEDMS-Infrastruktur.

4.4.5 Unterstützung maritimer Datenverarbeitungsprozesse

Im Bereich der maritimen Datenverarbeitung existiert eine Vielzahl von Methoden und Algorithmen, die für verschiedenste Zwecke eingesetzt werden. Für die Veranschaulichung der Unterstützung einer solchen Verarbeitungskette (vgl. Anforderung A16) wird erneut das Beispiel der AIS-Datenverarbeitung herangezogen. Im Folgen werden dazu zwei exemplarische Verarbeitungsschritte zur Datenqualitätsbewertung und zur Datenvorverarbeitung von AIS-Daten entworfen, und ihre Einbettung über das FEDMS in beliebige Workflows diskutiert. Außerdem wird das Problem von schlechten Datenverbindungen auf See, während der Durchführung von Feldtests betrachtet.

Messungen in AIS-Nachrichten werden jeweils selbstständig von den einzelnen AIS-Teilnehmern erhoben. Fehler in den Messinstrumenten dieser Teilnehmer können daher grundsätzlich nicht ausgeschlossen werden. Somit sind Bereinigung und Verifikation der AIS-Daten empfehlenswert. Zudem sollten Positionierungsdaten beispielweise mit RADAR-Aufzeichnungen abgeglichen werden, um eine sichere Datengrundlage zu gewährleisten. (Guerriero u. a. 2008)

Als exemplarischer Verarbeitungsschritt zur AIS-Datenqualitätsbewertung, wird die Datenqualität für dynamische AIS-Nachrichten betrachtet. Als grundlegende Kriterien sind hier die Vollständigkeit und Konsistenz relevant: Da AIS-Nachrichten viele Felder enthalten, die oft nicht alle mit Werten belegt werden, stellt die Vollständigkeit einen sinnvollen Kennwert dar, mit dem beurteilt werden kann, ob die Nachrichtenqualität eine ausreichende Grundlage darstellt, um

bestehende Fragestellungen zu untersuchen. Die Vollständigkeit wird nach Taleb u. a. (2016) mit folgender Formel berechnet:

$$Comp = \frac{N_{mv}}{N}$$

Hierbei steht N_{mv} für die Anzahl der vollständigen Nachrichten (also Nachrichten, die alle vorgesehenen Felder enthalten) in einem betrachteten Datensatz und N für die Anzahl aller Nachrichten in diesem Datensatz. Weiterhin ist es nicht unüblich, dass unplausible Werte in den AIS-Nachrichten vorzufinden sind. Um die Konsistenz einer Nachricht zu bewerten wird hier nach DIN EN ISO 19157 (ISO 2014) der Teilaspekt der Einhaltung des Wertebereichs überprüft. Hierzu werden die offiziellen Definitionen der Wertebereiche im AIS-Standard genutzt (siehe Tabelle 1).

Feld in dyn. AIS-Nachricht	**Definierter Wertebereich**
Schiffsposition (inkl. Genauigkeit)	Position nach WGS 84
Zeitstempel	UTC Zeitstempel
Kurs über Grund (COG)	0° ≤ COG < 360°
Geschwindigkeit über Grund (SOG)	SOG ≤ 102,2 Knoten
Heading	0° ≤ heading < 360°
Navigationsstatus	Bestandteil vorgegebener Status
Rate of turn (ROT)	-708° ≤ ROT ≤ 708°

Tabelle 1: Definierte Wertebereiche dynamischer AIS-Nachrichten (vgl. ITU-R 2014, 107-110)

Schließlich wird wie oben die Konsistenz des Datensatzes nach Taleb u. a. (2016) berechnet: $Cons = \frac{N_{vrc}}{N}$. Hierbei ist N_{vrc} die Anzahl der vollständig konsistenten Daten im betrachteten Datensatz. Für weiterführende Ansätze sind hier auch deutlich komplexere Prozesse denkbar, die vorherige und nachfolgende Nachrichten als Kontextinformationen für die Bewertung weiterer Konsistenzaspekte mit einbeziehen.

Als weiterer interner Verarbeitungsschritt soll ein (Vor-)Verarbeitungsschritt umgesetzt werden, der dynamische AIS-Datensätze aufbereitet. Das Ziel dieses Verarbeitungsschrittes ist es Nachrichten aus einem Datensatz zu entfernen, die die oben definierten Datenqualitätskriterien nicht erfüllen, und somit die Datenqualität des Datensatzes zu erhöhen. Da der Vorverarbeitungsschritt flexibel eingesetzt werden können soll, und das Entfernen aller nicht vollständigen oder konsistenten Nachrichten den Datensatz stark verkleinern könnte, wird eine Parametrisierung realisiert. So ist es einem Nutzer möglich den Verarbeitungsschritt so zu konfigurieren, dass bspw. nur Nachrichten mit fehlendem COG-Wert entfernt werden.

Die beiden vorgestellten Verarbeitungsschritte können dann so umgesetzt werden, dass sie Daten aus einem beliebigen Topic einlesen. Der Verarbeitungsschritt zur Datenqualitätsbewertung kann die Ergebnisse der Datenqualitätsbewertung in einem Log veröffentlichen, während der

Verarbeitungsschritt zur Aufbereitung der Daten jede Nachricht prüft und nur die Nachrichten an ein Output-Topic weiterleitet, die den durch die Parameter definierten Kriterien entsprechen. Die Verarbeitungsschritte müssen für beliebige Nutzer dann mehrfach dynamisch unter Angabe des Topics und gewünschter Parameter (z.B. welche Felder betrachtet werden sollen) auf der Infrastruktur des FEDMS instanziierbar sein.

Ein weiteres Problem ergibt sich bei der Unterstützung maritimer Datenverarbeitungsketten in Feldexperimenten: Das Erheben und Verarbeiten von AIS-Daten oder anderen maritimen Daten im Rahmen von Feldtests kann insbesondere dann eine Herausforderung darstellen, wenn mobile Testträger verwendet werden, die Daten über Verbindungen mit geringer Bandbreite oder kostenintensive Verbindungen übermitteln müssen (vgl. Anforderung A17). Um den Stakeholdern hier eine Lösung anzubieten, wird speziell für die Situationen ein effizientes Serialisierungsformat eingeführt, welches durch alle Komponenten des FEDMS, die Daten verarbeiten oder bereitstellen (Connectoren, Datenverarbeitungsschicht, Client-Bibliothek) unterstützt wird. In der Literatur wurden bereits diverse Serialisierungsmethoden evaluiert (Sumaray und Makki 2012), die verschiedene Charakteristiken bzgl. der Speichereffizienz aufweisen, bei der sich Protobuf als gute Alternative zu klassischen Serialisierungsmethoden zeigte. Weitere Details zur Integration von Protobuf werden in Abschnitt 5.2 diskutiert.

4.5 Datenverteilungsschicht

Eine allgemeine Absicherung der vorgestellten Architektur, wie sie in Anforderung A14 gefordert wird, soll durch die Datenverteilungsschicht gewährleistet werden. Diese Schicht separiert die Nutzerinteraktionen mit dem FEDMS von den restlichen Schichten und bietet somit eine zusätzliche Absicherung der Schichten, die sensible Daten verwalten. Zudem sind die Authentifizierung und Autorisierung die zentralen Aufgaben dieser Schicht. Eine Ausnahme bilden die Connectoren, die zwar ebenfalls authentifiziert werden, aber aufgrund ihrer direkten Verbindung zur Connector Zugriffsschicht nicht den Umweg über die Datenverteilungsschicht gehen. Zudem führen die Connectoren keine komplexen Interaktionen mit dem FEDMS durch, sondern registrieren sich lediglich zu Beginn und nehmen dann Datenanfragen entgegen. Die folgenden Abschnitte beschreiben die Mechanismen, die in der Datenverteilungsschicht zur Authentifizierung und zur Autorisierung von Nutzerzugriffen auf das FEDMS verwendet werden.

4.5.1 Authentifizierung

Um die Authentizität eines Nutzers im Testfeld sicherstellen zu können, muss diese im Rahmen einer Authentifizierung geprüft werden. Dies ist notwendig, damit die verschiedenen Services des

FEDMS nur solchen Nutzern zugänglich gemacht werden, die entsprechende Berechtigungen dazu besitzen. Diese Berechtigungen werden durch den Bereitsteller der Datenverteilungsschicht im Interesse der Stakeholder im Testfeld verwaltet. Authentizität ist „*die Echtheit und Glaubwürdigkeit des Objekts bzw. Subjekts, die anhand einer eindeutigen Identität und charakteristischen Eigenschaften überprüfbar ist*“ (Eckert 2018, 8). Dieses Prinzip wird typischerweise durch die Vergabe eines Benutzernamens und das Setzen eines Passwortes umgesetzt. Durch das Kennen des Passworts kann dieser Nutzer seine Authentizität nachweisen (Eckert 2018, 8). So sollen auch Nutzer des Testfeldes durch eine Kombination von Nutzername und Passwort authentifiziert werden. Aufgrund des modularen Aufbaus des FEDMS ist eine globale Authentifizierung jedoch nicht umzusetzen, da verschiedene Schnittstellen (Programmierbibliothek, Web-Interface, Streaming-Cluster und öffentliche Schnittstelle der Datenverteilungsschicht) unabhängig voneinander genutzt werden können. Um in dieser Situation die Nutzer mit Zugriffsrechten zu verwalten wird ein Identitätsprovider (IdP) für das Testfeld benötigt (vgl. auch Abschnitt 4.6.1). Dieser übernimmt im Testfeld die Aufgabe der Authentifizierung von Nutzern und stellt diesen Service für die verschiedenen Teilkomponenten zur Verfügung (vgl. Begriff des Single-Sign-On (Eckert 2018, 492)). Protokolle für einen solchen Dienst sind typischerweise standardisiert (vgl. etwa OAuth2 (Hardt 2012) oder SAML (Armando u. a. 2008)) und tragen somit zur Erweiterbarkeit des TDS bei: Zusätzliche Services (z.B. zur direkten Kommunikation mit Testträgern) können einen zentralen Identitätsprovider im Testfeld ebenfalls nutzen, um Nutzer zu authentifizieren.

Nach De Clercq (2002, 41) ist ein Identitätsprovider (auch: Authentifizierungsautorität) ein logisches Konzept, welches eine Entität beschreibt, „*der durch das Vertrauen ihrer Nutzer die Fähigkeit zugesprochen wird verlässliche Authentifizierungsmechanismen bereitzustellen*“. Die Testfeldnutzer müssen dem Testfeld-IdP also vertrauen. Probleme mit dieser Annahme werden in Abschnitt 4.6 diskutiert. Ein Prozess zur Authentifizierung im Testfeld ist in Abbildung 38 dargestellt: Ist ein Identitätsprovider vorhanden, kann sich ein Nutzer bei diesem authentifizieren (1) und erhält als Beleg für seine Authentizität einen Auth-Token (2), der typischerweise für einen bestimmten Zeitraum gültig ist (andere Beschränkungen sind ebenfalls möglich). Dieser Token kann dann verwendet werden, um sich bei einem Service im Testfeld zu authentifizieren (3). Um sicherzustellen, dass der Auth-Token gültig ist, wird dieser durch den Service mithilfe des Identitätsproviders validiert (4, 5). (De Clercq 2002)

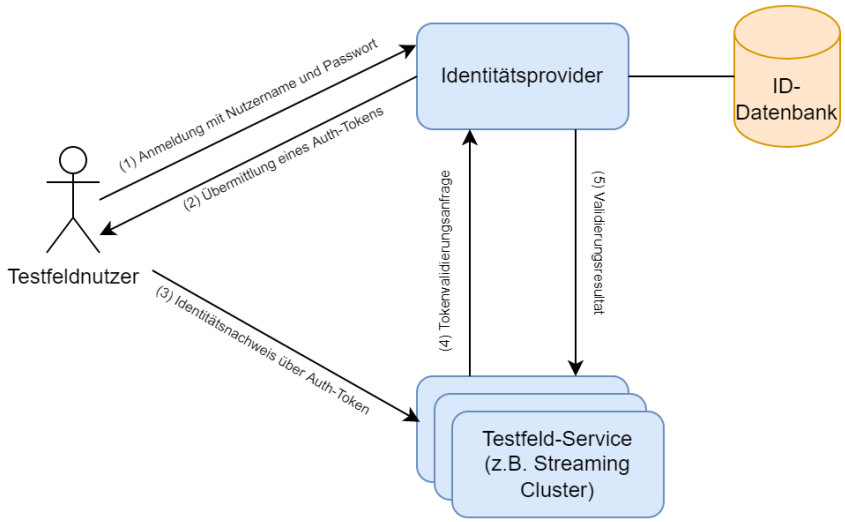

Abbildung 38: Single-Sign-On im Testfeld mithilfe eines Identitätsproviders (vgl. De Clercq 2002).

4.5.2 Autorisierung

Wurde ein Nutzer authentifiziert, muss bei der Nutzung einer FEDMS-Funktionalität in einem zweiten Schritt geprüft werden, ob der Nutzer berechtigt (also *autorisiert*, (vgl. Eckert 2018, 5)) ist auf eine angefragt Ressource (z.B. ein Topic oder die Dokumentation eines Workflows) zuzugreifen. Aus der bisherigen Modellierung des TDS ergeben sich drei Arten von Ressourcen, für welche die Zugriffsrechte verwaltet werden müssen:

- **Datenquellen** (repräsentiert durch die Connectoren): Hier muss durch den Datenprovider festlegt werden, welche Nutzer Daten aus der Datenquelle abfragen dürfen.

- **Workflows:** Workflows (inkl. Dokumentation) sind die zentrale Ressource des FEDMS und können durch Nutzer erstellt und mit anderen Nutzern geteilt werden.

- **Topics:** Topics sind zum einen nicht zwingend Teil von Workflows (vgl. Abbildung 33 (c)) und sind auf technischer Ebene durch die Abkapselung von Streaming Cluster und Datenverarbeitungsschicht getrennt. Somit ist eine eigene Betrachtung bei der Autorisierung notwendig. Beim Verändern von den Berechtigungen eines Workflows müssen diese mit den Berechtigungen für die referenzierten Topics synchronisiert werden.

Für die Autorisierung bzgl. der Datenquellen kann ein rollenbasiertes Modell (Role Based Access Control, (vgl. Sandhu und Samarati 1994)) eingesetzt werden: Für jede Datenquelle existiert eine entsprechende Rolle. Nutzer, denen diese Rolle zugeordnet ist, können Daten über das FEDMS abfragen. Die Rolle wird ausschließlich durch die Datenprovider und den

Testfeldinfrastrukturbetreiber vergeben. Für Topics und Workflows ist dies nicht sinnvoll, da diese durch Benutzer verwaltete Ressourcen sind und dynamisch erstellt, verändert und gelöscht werden. Hier ergibt sich der Bedarf für eine benutzerbestimmbare Zugriffskontrolle (vgl. Discretionary Access Control (Sandhu und Samarati 1994)), bei welcher der Zugriff auf Ressourcen auf Basis der Identität (also pro Nutzer) gewährt wird. Von der Datenverteilungsschicht wird das Workflow- & Szenariorepository genutzt, um die Zugriffsinformationen in Kombination mit den Metadaten zu speichern, damit ein schnellerer Zugriff ermöglicht wird. Für interne Verwaltungsaufgaben (z.B. das Registrieren eines Connectors beim FEDMS) können weitere interne Rollen eingesetzt werden.

4.6 Kontrollschicht

Das dezentrale Setup des TDS und die gegensätzlichen Interessen seiner Nutzer in Kombination mit den teilweise aufwendig erhobenen Daten kann dazu führen, dass ein grundsätzliches Misstrauen gegenüber anderen Teilnehmern entsteht. Die vorgestellte mehrschichtige Architektur kann durch das Zugriffsmanagement der Datenverteilungsschicht zwar sicherstellen, dass keine unautorisierten Zugriffe auf die Ressourcen des FEDMS stattfinden, eine genaue Nachverfolgung der einzelnen Lese- und Schreibaktionen auf den Topics, oder die sichere Zuordenbarkeit von externen Verarbeitungsschritten zu den FEDMS-Nutzern und die Garantie der Korrektheit ihrer Dokumentation werden jedoch nicht gewährleistet (vgl. Anforderung A14). Anfragen an die Schnittstellen der Datenverteilungsschicht können hierzu durch eine Kontrollschicht geloggt werden. Bezüglich der eigentlichen Datenübertragungen ist dies jedoch problematisch: Das Tracking atomarer Schreib- und Leseoperationen im Cluster ist durch das FEDMS zwar denkbar, würde aber die Performance durch das Überprüfen jeder einzelnen Transaktion enorm beeinträchtigen. Zudem müsste ein misstrauender Nutzer sich darauf verlassen (und dies gilt ebenfalls für die Dokumentation von Verarbeitungsschritten mittels des OPM), dass die Informationen des FEDMS nicht manipuliert wurden. Eine weitaus flexiblere Methode bietet das digitale Signieren von Informationen durch die Teilnehmer des TDS mittels kryptografischer Schlüsselpaare, die durch ein Zertifikat von einer Zertifizierungsstelle einer physischen Identität zuzuordnen sind.

Um höheren Sicherheitsanforderungen (beispielsweise in Zertifizierungsprozessen) gerecht zu werden, wird in Abschnitt 4.6.1 zunächst die Möglichkeit eingeführt auch externe Identitätsprovider einzubinden. Abschnitt 4.6.2 zeigt dann das Verfahren zum Signieren der Daten und in Abschnitt 4.6.3 wird die Maritime Connectivity Platform als mögliche Lösung für einen externen IdP vorgeschlagen.

4.6.1 Identitätsmanagement

Das sogenannte Identitätsmanagement beschreibt eine Menge an Funktionen die verwendet werden, um die Identität und zugehörige Informationen einer Entität sicherzustellen, und damit die Nutzung bestimmter Anwendungen zu ermöglichen (International Telecommunications Union 2009, i). Wie in Abschnitt 4.5.1 diskutiert, sieht die Datenverteilungsschicht einen Identitätsprovider vor, der durch den Testfeldbetreiber bereitgestellt werden soll. Im Allgemeinen wird aber eine Vielzahl an Sicherheits- und Vertrauensanforderungen an Identitätsprovider gestellt (Jøsang u. a. 2005). Somit bietet es sich an eine Möglichkeit zu schaffen, um Testfeldnutzer auch mit Identitäten von externen IdPs im Testfeld authentifizieren zu können und das Identitätsmanagement damit weiter zu dezentralisieren.

Damit nicht jeder IdP einzeln integriert werden muss, kann das Konzept eines Identity Brokers genutzt werden (dargestellt in Abbildung 39), welcher die Nutzung von Identitäten externer IdPs vereinfacht (Grassi, Lefkovitz, und Mangold 2015). Es wird Token eines externen IdPs genutzt (1, 2), um sich damit beim Broker zu authentifizieren (3, 4), welcher daraufhin einen lokalen Token erstellt (5, 6) (Reimer, Abraham, und Tan 2013), mit dem sich Nutzer dann bei den Services im TDS authentifizieren können (7, 8). Der Broker mappt also eine externe Identität auf eine Identität, die innerhalb des Testfeldes verwendet werden kann und tauscht einen externen Token gegen einen internen Token aus (Token-Exchange).

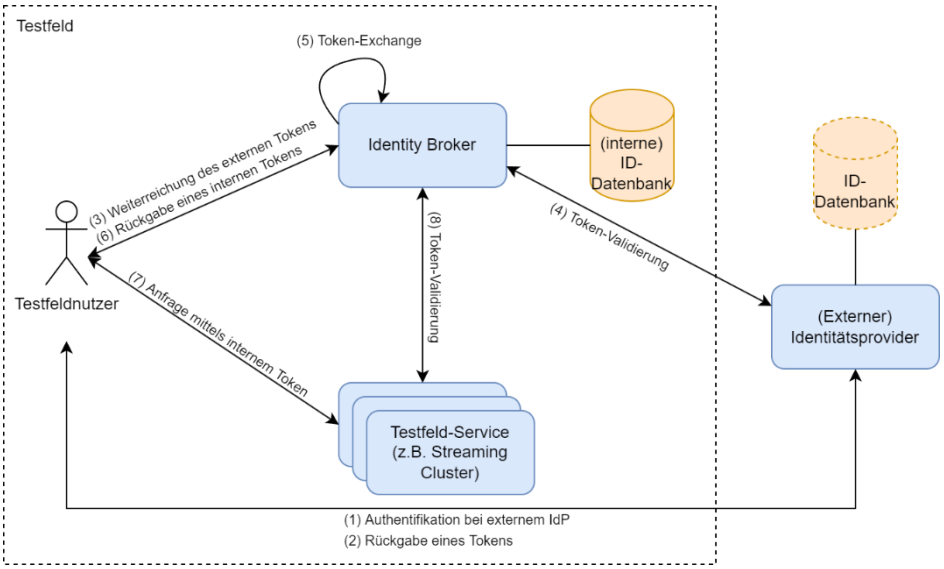

Abbildung 39: Identity Broker im Testfeld zur Nutzung externer IdPs (vgl. Reimer, Abraham, und Tan 2013).

Je nach Konfiguration der Testfeldnutzer und umgesetzter Workflows – etwa in einem Forschungsprojekt – ist es beispielsweise denkbar, dass besonders sensible Daten nur zwischen Teilnehmern ausgetauscht werden, wenn diese eine durch einen bestimmten externen IdP

verifizierbare Identität besitzen. Gleichzeitig ist hierzu aber ein gewisses Vertrauen in den Identity Broker vorauszusetzen, da dieser die internen Tokens des Testfeldes verwaltet und die des externen IdPs einsehen kann.

4.6.2 Signieren von Datenartefakten und Dokumentation

Weiterhin besteht das Problem der genauen Nachverfolgung von Verarbeitungsschritten auf der Ebene der Datentupel (vgl. Anforderung A13): In der bisher vorgestellten Konfiguration ist es zwar möglich durch die Autorisierungsmechanismen genau festzulegen, welche Akteure im Testfeld Zugriff auf ein Datenartefakt eines Workflows bzw. Topic haben, eine genaue Nachverfolgung der produzierten Daten auf Tupeln mit mehreren kollaborierenden Nutzern ist jedoch nicht möglich. Im Nachhinein ist nicht mehr sicher feststellbar, welche Datentupel von welchem Benutzer auf ein Topic geschrieben wurden. Um unabhängig von der Analyse von Transaktionslogs (welche durch das FEDMS verwaltet werden) eine sichere Möglichkeit zu bieten, welche Datentupel valide im Sinne eines Nutzers sind, wird die Möglichkeit eingeführt Daten zu signieren. In kritischen Konfigurationen, in denen sich die Teilnehmer vollständig misstrauen ist es empfehlenswert jede Nachricht einzeln zu signieren. Dies ist jedoch nicht immer notwendig, und gerade, wenn der Aspekt einer sicheren Dokumentation im Vordergrund steht, sollten effizientere Verfahren zur Verfügung stehen, die den eigentlichen Datenverarbeitungsprozess nicht beeinträchtigen.

Ähnliche Ansätze zum sicheren Tracking von Datenprovenienz finden sich bereits in der wissenschaftlichen Literatur. Eine aus dieser Arbeit hervorgegangene Publikation beschäftigt sich mit einer verwandten Situation, bei der jedoch keine Partei vorhanden ist, die die Rolle eines FEDMS-Bereitstellers einnehmen kann (Möller, Fröschle, und Hahn 2021). Im Testfeld kann wie folgt vorgegangen werden (Abbildung 40):

1. Der validierende Nutzer wählt ein Topic aus und ruft die Datentupel für einen spezifizierten Bereich ab.

2. Der Nutzer prüft die abgerufenen Datentupel auf Validität (z.B. entsprechen die Datentupeln exakt denen, die vom validierenden Nutzer zuvor übermittelt wurden).

3. Die validierten Datentupel werden zusammengefügt und ein Hash-Wert dieser Daten wird berechnet.

4. Der Hash-Wert wird mit dem Schlüssel des validierenden Nutzers signiert und mit weiteren Metadaten (Zeitpunkt der Validierung, validierender Nutzer, validierter Bereich (Topic und Offsets)) an das Workflow & Scenario Repository des FEDMS übermittelt.

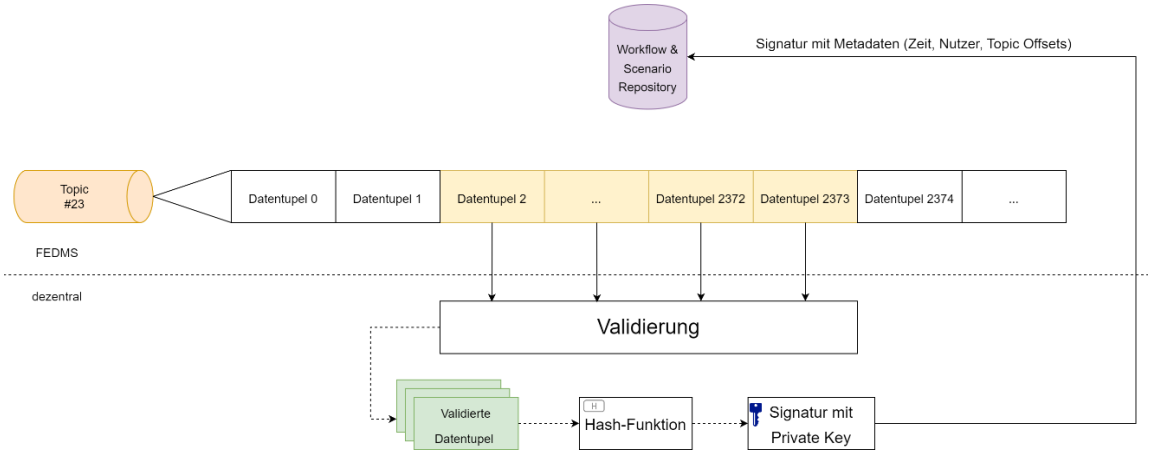

Abbildung 40: Validierung und Signatur von Abschnitten eines Topics (Datenartefakt).

Der validierende Nutzer signiert somit die Validität der Daten aus seiner Sicht als Urheber. Andere Nutzer mit Zugriff auf das referenzierte Topic können diese Signaturdaten abrufen, die Signatur überprüfen und den Hash ebenfalls berechnen und vergleichen. Durch die Verwendung einer Public-Key Infrastruktur (PKI) kann die Authentizität einer solchen Signatur mit Hilfe eines durch den IdP ausgestellten Zertifikates überprüft werden.

Auf diese Art und Weise können die eigentlichen Datentupel validiert und entsprechende Resultate verfügbar gemacht werden. Da die Transformation der Datentupel zumeist dezentral erfolgt, kann eine Signatur des Verarbeitenden bei Bedarf ebenfalls genutzt werden, um mit der eigenen Identität dafür zu bürgen, dass ein bestimmter Verarbeitungsschritt erfolgt ist. Hierfür wird aus den Metadaten eines Verarbeitungsschrittes (siehe Abschnitt 4.4.2) ebenfalls ein Hash generiert und signiert, der als weiteres Metadatenfeld angehängt wird.

4.6.3 Maritime Connectivity Platform

Die Maritime Connectivity Platform (MCP) ist ein Framework, welches es ermöglicht moderne und sichere, digitale Services in einem maritimen Umfeld operativ zu betreiben. Neben einer Lösung, um solche Services standardisiert auffindbar zu machen und einem Dienst, der für die Übermittlung von Daten genutzt werden kann, werden Instanzen des MCP-Frameworks vor allem als Identitätsprovider eingesetzt. Insgesamt wird das Ziel verfolgt eine dezentral organisierte Struktur aus verschiedenen Instanzen aufzubauen, zwischen denen benutzerdefinierte Vertrauensverhältnisse bestehen können. Hierbei muss von den Betreibern eine Reihe an Anforderungen erfüllt werden, um als MCP-Instanz anerkannt zu werden, die sich etwa auf die Prüfungsprozesse für das Ausstellen einer Identität beziehen. (Maritime Connectivity Platform Consortium 2021b)

Zusätzlich ermöglicht die Identitätsprovider-Komponente einer MCP-Instanz den Aufbau einer Public-Key Infrastruktur und das Authentifizieren für den Einsatz in Web-Services mittels des OAuth2/OpenID Connect Protokolls (Maritime Connectivity Platform Consortium 2021a). Aufgrund des dezentralen Aufbaus und der Möglichkeit eigene MCP-Instanzen zu etablieren, so wie dem Vorhandensein einer PKI bietet sich die MCP als standardmäßiger Identitätsprovider für die FEDMS-Architektur an. Die Web-Authentifizierung kann für das Testfeld Single-Sign-On System verwendet werden, und die PKI-Funktionalität kann von den Benutzern des Testfeldes zum Erstellen und Überprüfen von signierten Daten verwendet werden. Zusatzfunktionalitäten wie die Ressourcenverwaltung (z.B. von Schiffen, Geräten, oder weiteren Services) und das Zuordnen eindeutiger Ressourcenidentifikatoren (Maritime Resource Names) können bei Bedarf ebenfalls in das Testfeld integriert werden (Maritime Connectivity Platform Consortium 2021a). Zudem können Daten auch vor der Übertragung an das FEDMS durch die Teilnehmer mit ihren Schlüsselpaaren aus der MCP verschlüsselt werden. Somit wird eine komplett unabhängig von der FEDMS-Architektur verwaltete PKI genutzt, um eine zusätzliche Sicherheitsschicht einzuführen.

4.7 Rollenmodell und Bereitstellung der Schichten

Durch das Auslagern der Kontrolle über die Connectoren an die Datenprovider und die anfrageorientierte Datenarchitektur werden bereits Souveränitätsanforderungen der Datenprovider berücksichtigt und eine testfeldweite, zentrale Speicherung der Daten vermieden. Es kann jedoch nicht in jeder Testfeldkonfiguration davon ausgegangen werden, dass alle Datenprovider und -konsumenten einer einzigen, durch einen Testfeldbetreiber bereitgestellten Instanz des FEDMS vertrauen. Die Datenprovider und -konsumenten müssen zwar die Datenspeicherung nicht auf das FEDMS auslagern, aber darauf vertrauen, dass der Betrieb des Streaming-Clusters, die Überwachung der Bearbeitungshistorie und Durchsetzung der Datenzugriffsrechte, sowie das Identitäts- und Zugriffsmanagement ordnungsgemäß vom Testfeldbetreiber durchgeführt werden können. Aufgrund der modularen Architektur des FEDMS ist es jedoch möglich die Bereitstellung (Deployment) flexibel an verschiedene Vertrauensbedingungen anzupassen. Hierbei wird davon ausgegangen, dass die einzelnen Komponenten bzw. Schichten der Architektur durch unabhängige Parteien bereitgestellt werden. Je nach Architekturelement wird diesen Stakeholdern eine entsprechende Rolle zugewiesen, in deren Verantwortungsbereich es liegt das entsprechende Element unter Maximierung des Vertrauens zu betreiben. Abbildung 41 zeigt eine Übersicht über solch eine Dezentralisierung der Bereitstellung des FEDMS: Dies ermöglicht auch die dynamische Instanziierung eines FEDMS oder bestimmter Teile, durch Testfeldnutzer. So könnte etwa für ein größeres Projekt eine eigene Instanz eines FEDMS vollständig auf der Infrastruktur eines Projektpartners ausgeführt werden.

Datenprovider könnten dann Connectoren instanziieren, die nur Daten für diese Instanz bereitstellen. Die folgenden unabhängigen Rollen können auf Basis der vorgestellten Architektur definiert werden:

Datenprovider. Die Datenprovider erfüllen weiterhin die klassische Rolle des Bereitstellens von Daten über einen Connector. Dabei können mehrere vollständig unabhängig voneinander agierende Datenprovider dieselben Komponenten nutzen, um ihre Datenquellen über Connectoren verfügbar zu machen.

Datenkonsument. Auch die Rolle der Datenkonsumenten bleibt unverändert. Über die Client-Bibliothek (oder die Schnittstelle der Datenverteilungsschicht, vgl. Abschnitt 4.2) fragen die Datenkonsumenten Daten über das FEDMS ab. Für die Bibliothek ist dabei frei konfigurierbar, welche Instanz einer Datenverteilungsschicht angesprochen werden soll. Es ist auch möglich Daten aus verschiedenen Instanzen abzurufen und lokal zusammenzuführen.

Datentreuhänder. Der Datentreuhänder erfüllt die in Abschnitt 2.3.4 vorgestellte Rolle des Datentreuhänders und verwaltet als Intermediär die Daten, die zwischen Provider und Konsument ausgetauscht werden. Dies schließt den Betrieb des Ingestion Tools (bzw. Streaming-Clusters) und der Datenverarbeitungsschicht inkl. des Workflow- & Szenario Repositories ein.

Identitätsprovider. Wie bereits in Abschnitt 4.6 diskutiert kann auch die Rolle des Identitätsproviders durch Externe Services ergänzt werden. Es ist im Rahmen des dezentralisierten Deployments auch möglich den testfeldeigenen Identitätsprovider/-broker durch eine unabhängige Partei zu betreiben.

FEDMS-Provider. Der FEDMS-Provider stellt mit der Datenverteilungsschicht den zentralen Zugriffspunkt für die Testfeldressourcen bereit. Zudem kann er im dezentralisierten Deployment auch die Rolle einer Vermittlungsstelle (vgl. IDS-Referenzarchitektur) übernehmen, da er die Zugriffe der Nutzer auf die Ressourcen dokumentieren kann, um eine Grundlage für das Auflösen möglicher Konflikte bereitstellen zu können.

Broker. Der Broker stellt eine Connector-Zugriffsschicht bereit und indexiert somit die Datenquellen, die für eine Instanz des FEDMS bereitgestellt werden sollen und ordnet die Datenfragen auf diese zu.

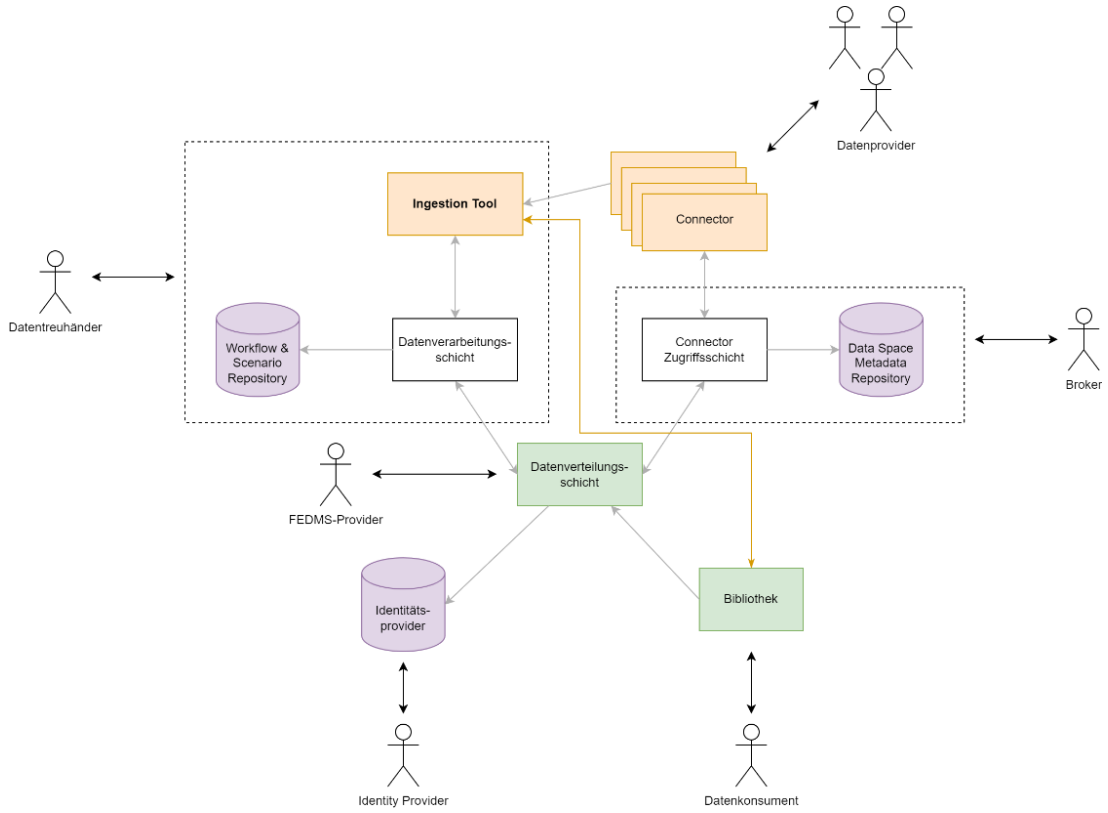

Abbildung 41: Dezentrales Deployment der verschiedenen Architekturschichten und Komponenten mit den jeweiligen Rollen.

Das dezentralisierte Deployment des FEDMS liefert ein stärker dezentralisiertes Modell eines Data Spaces, ähnlich wie es im International Data Space (vgl. Abschnitt 3.2) umgesetzt wird. Auf diese Weise kann insbesondere Anforderung A5 umgesetzt werden.

5 FEDMS-Prototyp

Nachdem im vorigen Kapitel ein Konzept zur Beantwortung der Forschungsfrage entwickelt wurde, soll dieses nun im folgenden Abschnitt prototypisch umgesetzt werden. Dies dient als Grundlage zur Demonstration der Anwendbarkeit des Konzeptes im Rahmen der Evaluation in Kapitel 6. Dieses Kapitel teilt sich in sechs Abschnitte auf, die die grundlegenden Komponenten der schichtenbasierten Architektur (5.1), die konkrete Umsetzung von Connectoren für maritime Datenquellen aus dem Testfeld (5.2), die Zuordnung von Datenanfragen (5.3), die Umsetzung der anfrageorientierten Kappa-Architektur (5.4), die Komponenten zum Management von Szenario und Workflowmetadaten (5.5) und die Absicherung des Systems diskutieren (5.6).

Die prototypische Implementierung baut auf dem in Abschnitt 3.3.1 vorgestellten eMIR-Testfeld auf. Die Prozesse zur Datenerhebung und Speicherung bzw. Bereitstellung wurden bis zu den Endpunkten der Datenquellen übernommen. Weitere Prozesse, wie die bestehende Infrastruktur zur Übermittlung der Daten oder das Data-Warehousing können die in Abschnitt 2.6 definierten Anforderungen nicht erfüllen (vgl. auch Abschnitt 3.4) und werden somit nicht berücksichtigt. Aus der Architektursicht werden also lediglich die maritimen Datenquellen als Grundlage für die prototypische Umsetzung verwendet. Weitere Details zur Implementierung und zu Benutzeroberflächen befinden sich in Anhang 8C und 8D.

5.1 Realisierung der schichtenbasierten Architektur

In Abschnitt 4.2 wurde eine schichtenbasierte Architektur vorgestellt, die verschiedene Aufgaben auf logisch voneinander getrennte Architekturschichten aufteilt und somit die Gesamtkomplexität des Systems reduziert. Um eine logische Trennung auch in der Implementierung zu gewährleisten, und um das in Abschnitt 4.7 vorgestellte, unabhängige Deployment von Architekturschichten gewährleisten zu können, werden die einzelnen Schichten vollständig voneinander abgekapselt als eigenstehende Microservices entwickelt. Die Implementierung der einzelnen Schichten nutzt grundsätzlich die Programmiersprache Java, die plattformunabhängig eingesetzt werden kann und für die zahlreiche bereits existierende Programmierbibliotheken für benötigte Teilfunktionen der zu implementierenden Komponenten verwendet werden können. Durch die Abkapselung ist es notwendig, dass jede Architekturschicht über eine standardisierte Kommunikationsschnittstelle verfügt, die genutzt werden kann, um mit anderen Schichten Daten auszutauschen. Hierzu wird für jede der Architekturschichten eine REST-API implementiert, und

von einem zentralen *RequestHandler* Modul verwaltet. Spring Boot[8] wird als Bibliothek zur Bereitstellung der API und als Framework für die Injektion von Abhängigkeiten im Java-Code verwendet. Darüber hinaus werden die API-Endpunkte in Übereinstimmung mit der OpenAPI-Spezifikation (Miller u. a. 2021) bereitgestellt. Der *RequestHandler* leitet eingehenden Anfragen an die entsprechenden Services in der Schicht weiter, die diese bearbeiten und einen entsprechenden Rückgabewert an den *RequestHandler* zurückliefern. Basierend auf diesem Prinzip implementieren die verschiedenen Schichten die benötigten Funktionalitäten, die mit dem *RequestHandler* kommunizieren. Selbes gilt auch für die Datenverteilungsschicht (*Data Distribution Layer*), die die an den *RequestHandler* angebundene REST-API öffentlich im Testfeld zur Verfügung stellt.

Da eine Kommunikation zwischen den Schnittstellen aller Schichten notwendig ist, bietet es sich an die Kommunikationsmodule in einer separaten Bibliothek bereitzustellen und diese jeweils in die einzelnen Architekturschichten einzubinden (siehe Anhang 8C: Abbildung 70). Diese *Shared Library* beinhaltet weitere Hilfsfunktionen, die in mehreren Schichten verwendet werden.

Für das dynamische Deployment im Testfeld werden die einzelnen Java-Applikationen, die die Layer des FEDMS darstellen in containerisierter Version im Open Container Initiative (OCI)-Image Format bereitgestellt. So können die einzelnen Applikationen weitestgehend unabhängig vom Host-System eines Servers in einer Container Runtime (z.B. Docker, containerd oder CRI-O) als Microservices betrieben werden. Dies abstrahiert das Konfigurieren diverser Abhängigkeiten, wie z.B. das Vorhandensein eines Java Runtime Environments (JRE) und erleichtert das flexible Deployment auf verschiedenen dezentral betriebenen Infrastrukturen der Stakeholder im Testfeld. Die Containerisierung der Java-Applikationen wird automatisiert als Teil des Build-Prozesses mithilfe des Automatisierungswerkzeuges Maven und dem Google Jib Plug-in (Goundan und Qingyang 2018) vorgenommen. Für die Orchestrierung der Container-Instanzen wird ein Kubernetes[9] Cluster verwendet (Abbildung 42). Dies dient der Überwachung der Verfügbarkeit der Container und ermöglicht eine dynamische horizontale Skalierung des FEDMS. Für die prototypische Implementierung wird ein Single-Node Cluster verwendet, da im Rahmen dieser Arbeit keine speziellen Anforderungen an eine hohe Verfügbarkeit der Services gestellt wurde. Hierbei sind die Microservices in vier verschiedenen Kubernetes-Namensräumen organisiert und stellen nur die markierten Schnittstellen (dargestellt als weiße Punkte) öffentlich im Testfeld zur Verfügung. Weiterhin wird die konfigurierbare Netzwerkschicht von Kubernetes genutzt, um eine Isolierung der nicht öffentlich zugängigen Endpunkte (Data Processing Layer,

[8] https://spring.io/projects/spring-boot, abgerufen am 31.05.2022.
[9] https://kubernetes.io/de/, abgerufen am 15.12.2022

Connectoren und Connector Access Layer) zu gewährleisten, während eine interne Kommunikation aber weiterhin möglich ist. Außerdem wird die Annahme getroffen, dass im Testfeld eine vertrauenswürdige, zentrale Bereitstellung des FEDMS durch den Testfeldbetreiber stattfindet. Das verteilte Deployment der Schichten erfordert ein anderes Netzwerk-Setup mit mehreren Nodes bzw. Clustern (oder alternativen Konfigurationen).

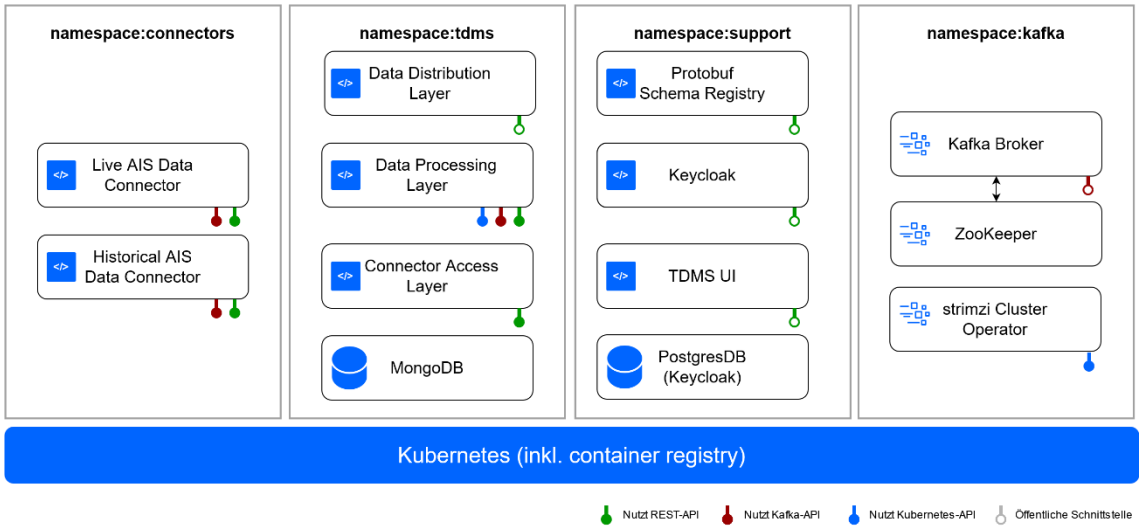

Abbildung 42: Übersicht – Deployment der einzelnen Architekturschichten innerhalb eines Kubernetes-Clusters (vgl. Möller u. a. 2022).

Die Microservices können durch Kubernetes yaml-Konfigurationsdaten konfiguriert werden. Entsprechende Parameter (z.B. geöffnete Ports, Hostnames der anderen Schichten oder Verbindungsinformationen zu Datenbanken) werden über das Setzen von Umgebungsvariablen durch Kubernetes an die Architekturschichten weitergegeben.

5.2 Connectoren für maritime Datenquellen

In Abschnitt 4.3.1 wurde die Architektur der Connectoren vorgestellt und dargelegt, dass im serviceorientierten Testfeld eine Vielzahl an Connectoren durch verschiedene Stakeholder unabhängig voneinander betrieben werden können. Weiterhin gilt es bei der Realisierung der Connectoren zu beachten, dass eine Standardisierte Schnittstelle zur Bereitstellung der Daten verwendet wird. Um nicht für jede Datenquelle einen vollständig neuen Connector entwickeln zu müssen, wird im Testfeld eine weitere Bibliothek (*Connector Core*, vgl. auch Anhang 8C: Abbildung 70) als Blaupause für den Entwurf eines Connector bereitgestellt.

Für das Formulieren von Datenanfragen mithilfe einer allgemeinen Anfragesprache wird GraphQL verwendet. Es ist gegenüber REST deutlich flexibler in der Strukturierung von Queries und reduziert die Komplexität (Vázquez-Ingelmo, Cruz-Benito, und García-Peñalvo 2017). Die

GraphQL-Queries werden an die REST-Schnittstelle von der Zugriffsschicht als Parameter übergeben und dann später durch den *GraphQLService* des Connectors interpretiert. Dieser enthält die Logik zur Transformation mit Hilfe eines benutzerdefinierten Schemas zwischen GraphQL und der nativen Technologie der Datenquelle. Zur Unterstützung bestimmter im vorliegenden Testfeld häufig verwendeter nativer Technologien werden in der Bibliothek bereits abstrakte *QueryRepository*-Klassen mit Hilfsfunktionen für das Mapping zu SQL, REST oder RabbitMQ-Datenquellen implementiert.

Weiterhin müssen die Connectoren eine effiziente Bandbreitennutzung gewährleisten (vgl. Anforderung A17). Hierfür wird das Protocol Buffers (Protobuf) Format von Google verwendet. Dabei handelt es sich um ein effizientes Serialisierungsformat, welches mithilfe von Schemata beliebige Datenstrukturen abbilden kann (Popić u. a. 2016). Im Vergleich zu anderen Serialisierungsformaten wie JSON, XML oder Apache Avro ist Protobuf besonders effizient im Hinblick auf Speichernutzung und Performance (Vanura und Kriz 2018). Allerdings kann Protobuf jedoch nicht von einem Datenkonsumenten verarbeitet werden, ohne das zugrundeliegende Datenschema zu kennen. Aus diesem Grund muss für die Bereitstellung dynamisch generierter Protobuf Schemata, eine sogenannte *Schema Registry* verwendet werden, die das Schema für ein Datenartefakt vom Datenprovider entgegennimmt und für einen Datenkonsumenten bereitstellt. Dieser Prozess wird durch die Verwendung eines entsprechend angepassten Kafka-Producers (im *Connector Core*) und -Consumer (in der Client-Bibliothek) Paares automatisiert und ist in der Entwicklung neuer Connectoren für Datenprovider nicht weiter beachtenswert (vgl. auch Abschnitt 5.4 zur Verwendung von Apache Kafka). Protobuf stellt hier nur eine standardisierte Lösung dar: Natürlich ist auch die Verwendung anderer Serialisierungsformate durch einen Connector möglich. Somit kann nun in Abbildung 43 der gesamte standardmäßige Prozess der Beantwortung von GraphQL-Datenanfragen innerhalb der Connectoren zusammengefasst werden: Nachdem der Connector eine Datenanfrage entgegengenommen hat, wird diese durch den GraphQL-Service analysiert und mittels eines internen Modells der GraphQL-Syntax repräsentiert. Aus dieser Repräsentation wird ein nativer Query abgeleitet, der durch die Datenquelle des Connectors beantwortet kann. Die native Datenantwort wird dann in Form einer Objektrepräsentation (ebenfalls aus dem GraphQL-Schema abgeleitet) gehalten und genutzt, um ein passendes Protobuf-Schema zu generieren. Dieses wird bei der Schema Registry registriert und gleichzeitig zur Serialisierung der Daten verwendet, die dann an den Kafka-Cluster übermittelt werden. Eine Verwendung von Protobuf außerhalb eines Connectors funktioniert analog mit Hilfe der Client-Bibliothek. Das Schema wird in diesem Fall direkt aus einem Java-Objekt generiert.

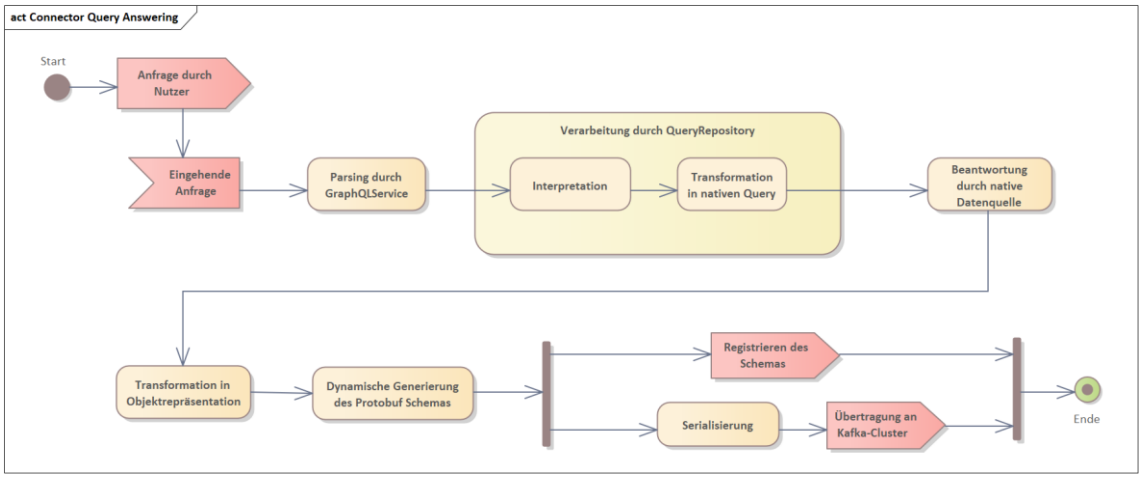

Abbildung 43: Beantwortung von GraphQL-Datenanfragen im Connector.

Im Folgenden werden nun Auszüge aus den konkreten Implementierungen der Connectoren für die maritimen Datenquellen des eMIR-Testfeldes vorgestellt.

5.2.1 AIS und RADAR Connector für historische Daten

Zunächst wurde ein Connector zur Bereitstellung einer umfangreichen Datengrundlage bzgl. historischer AIS- und RADAR-Daten im eMIR-Testfeld entwickelt. Die entsprechenden Daten werden in einer PostgreSQL Datenbank mit PostGIS-Erweiterung gehalten und befinden sich in einer einzelnen Tabelle. Wie in Listing 1 exemplarisch dargestellt, wurde zunächst ein GraphQL-Schema für die Datenquelle entwickelt, welches sich an dem Datenbankschema der zugrundeliegenden Datenbank orientiert. Hier ist ebenfalls erkennbar in welcher Granularität Datenabfragen eingeschränkt werden können: Der Datenprovider kann vordefinierte Anfragen als Query definieren, die dann durch bestimmte, frei definierbare Filter und Limits durch den Nutzer abgefragt werden können. Attribute können flexibel selektiert werden.

Für die weitere Realisierung wurden drei *QueryRepositories* vom Typ *QuerySQLRepository* (siehe Anhang 8C: Abbildung 68) abgeleitet, die jeweils statische und dynamische AIS-Nachrichten, sowie RADAR-Datenpunkte aus der PostgreSQL-Datenbank verarbeiten können. Wo es sinnvoll war, wurden Anfragefilter für das Finden exakter Übereinstimmungen in der Datenbank, das Finden größerer oder kleinerer Werte und Reichweitenfilter implementiert. Für die Verbindung zur PostgreSQL-Datenbank wurde das JDBC-Framework verwendet, um volle Kontrolle über das Ausführen der SQL-Anfragen zu erhalten. Diese werden abhängig von der eingehenden Anfrage in GraphQL durch das Konkatenieren vordefinierter Strings generiert.

```
type Query {
    RadarTargetState(first: Int, after: ID, filter: String): [RadarTargetState]
    AISDynamicMessage(first: Int, after: ID, filter: String): [AISDynamicMessage]
    AISStaticMessage(first: Int, after: ID, filter: String): [AISStaticMessage]
}
type AISDynamicMessage {
    id: ID!                 @filter(availableFilterMethods: "[\"equal\"]")
    referenceid: String     @filter(availableFilterMethods: "[\"equal\"]")
    mmsi: String            @filter(availableFilterMethods: "[\"equal\"]")
    latitude: Float!        @filter(availableFilterMethods: "[\"minmax\", \"gte\", \"lte\"]")
    ...
}
...
```

Listing 1: Auszug aus dem GraphQL-Schema des Connectors für historische AIS- und RADAR-Daten aus dem eMIR-Testfeld.

Abbildung 44 (1) zeigt eine exemplarische Anfrage, die auf Basis des vorgestellten Schemas ausgeführt werden kann. Mit der Anweisung *first: 1000* kann angegeben werden, dass die ersten 1000 Datenpunkte aus der Datenbank abgerufen werden sollen. *__datapoint_* definiert, dass jeder *AISDynamicMessage*-Datenpunkt als eigene Nachricht serialisiert und versendet werden soll. Die in (2) dargestellte SQL-Anfrage wird dann durch den Connector auf Basis von (1) abgeleitet und ausgeführt. Für die Serialisierung wird ebenfalls aus (1) ein Protobuf-Schema zur Serialisierung (3) dynamisch generiert und bei der *SchemaRegistry* registriert. Der *BatchEnd*-Nachrichtentyp mit einer standardisierten Konstante *magicNum* markiert das Ende von einem Datenbatch, der zu einer Anfrage zugehörig ist, damit ein Datenkonsument weiß, wann alle Nachrichten zur Beantwortung seiner Anfrage gestreamt worden sind (etwa, wenn kein Limit gesetzt wird).

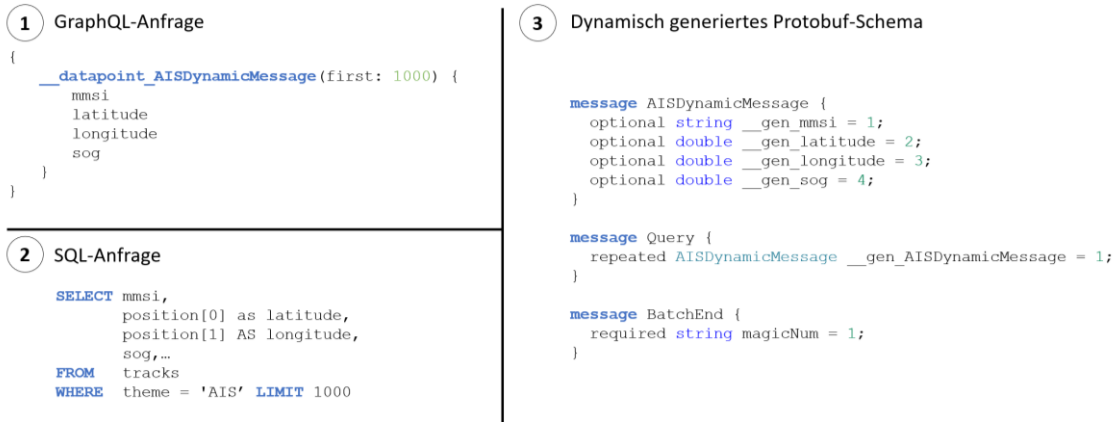

Abbildung 44: Anfragetransformation von GraphQL zu SQL (1 und 2) und dynamische Generierung eines Protobuf-Schemas zur Serialisierung (3).

5.2.2 AIS und RADAR Connector für Live-Daten

Die im eMIR-Testfeld vorhandene Infrastruktur stellt Live-Daten der Referenzwasserstraße in Form von AIS- und RADAR-Daten über einen RabbitMQ[10]-Server bereit. Der Datenstrom liegt in Form eines internen Datenmodells vor, welches an den IHO S-100 Standard angelehnt ist. Wie in Listing 2 zu erkennen ist, werden durch den Connector keine normalen *Queries* angeboten, es ist lediglich möglich eine *Subscription* bestimmter Daten durch das Abonnieren eines Datenstroms anzufragen.

```
type Query {}

type Subscription {
    RadarTargetState(filter: String): RadarTargetState
    AISDynamicMessage(filter: String): AISDynamicMessage
    AISStaticMessage(filter: String): AISStaticMessage
}

type AISDynamicMessage {
    referenceid: String     @filter(availableFilterMethods: "[\"equal\"]")
    mmsi: String            @filter(availableFilterMethods: "[\"equal\"]")
    latitude: Float!        @filter(availableFilterMethods: "[\"minmax\", \"gte\", \"lte\"]")
    ...
}
...
```

Listing 2: Auszug aus dem GraphQL-Schema des Connectors für AIS- und RADAR-Live-Daten aus dem eMIR-Testfeld.

Da RabbitMQ payload-agnostisch ist, existiert keine native Anfragesprache, in der die Daten nach den im Schema definierten Filtern und Attributen gefiltert werden könnten. Der „native Query“ (vgl. Abbildung 43) besteht hier also durch das Abonnieren des gesamten Datenstroms. Nach der Transformation in die interne Objektrepräsentation werden ggf. vorhandene Filter angewandt und ein Datenpunkt nur an den Datenkonsumenten übermittelt, wenn dieser den definierten Filterkriterien entspricht (sich also beispielsweise in einem bestimmten geographischen Bereich befindet). Die Generierung des Protobuf-Schemas funktioniert dann analog, wie im vorigen Abschnitt erläutert. Der implementierte Connector kann aus dem eingehenden Datenstrom multiple durch Filter modifizierte Datenströme ableiten, sodass auch eine parallele Beantwortung mehrerer Anfragen möglich ist. Unter Angabe der ID eines Datenstroms kann dieser durch den Konsumenten mithilfe der API-Funktion *stopStreaming* beendet werden.

[10] https://www.rabbitmq.com/, abgerufen am 17.12.2022

5.2.3 AIS-Connector für externe historische Dateien

Gegebenenfalls ist es notwendig eine beschränkte Datengrundlage durch das Einbeziehen externer Daten zu erweitern, um Anforderungen sehr datenintensiver Prozesse erfüllen zu können. Im Fall von AIS-Daten im eMIR-Testfeld ist die Datenerhebung auf die deutsche Bucht beschränkt. Aus diesem Grund wurde ein Connector implementiert, der weitere AIS-Daten aus externen Quellen bereitstellen und somit auch Daten für andere geografische Bereiche liefern kann. Frei verfügbare, historische AIS-Daten werden häufig im CSV-Format bereitgestellt, so etwa von der Danish Maritime Authority (Danish Maritime Authority o. J.) für Dänemark oder von MarineCadastre für die USA (NOAA Office for Coastal Management 2022).

Ausgehend von der Annahme, dass das Überführen der Daten aus den CSV-Dateien in eine Datenbanklösung nicht gewünscht oder nicht möglich ist, wurde exemplarisch ein Connector entwickelt, der auf der Grundlage von Dateien arbeitet. Für das direkte Anfrage bestimmter AIS-Datenpunkte mittels Filterkriterien innerhalb von CSV-Dateien wurde das Apache Drill Framework (vgl. Hausenblas und Nadeau 2013) verwendet, welches es ermöglicht in einer auf SQL basierenden Anfragesprache Anfragen auf CSV-Dateien auszuführen. Somit wird der eigentliche Zugriff auf die Daten abstrahiert und die Implementierung ähnelt dem in Abschnitt 5.2.1 vorgestellten Connector für historische AIS und RADAR-Daten. Um die Abfragezeit durch Apache Drill weiter zu optimieren, wurden die CSV-Dateien in das ebenfalls von Apache entwickelte parquet-Format (vgl. Vohra 2016b) konvertiert.

5.3 Anfragezuordnung von Datenanfragen

Die in Abschnitt 4.3.2 vorgestellte Connector Zugriffsschicht implementiert die wesentlichen Funktionalitäten der Registrierung von Connectoren und der Verwaltung ihrer Selbstbeschreibungen, der Entgegennahme und Beantwortung von Suchanfragen auf Connector-Metadaten und der Zuordnung von Datenanfragen. Weiterhin ermöglicht die Zugriffsschicht durch den Aufruf der API-Funktion *findSchemesBySearchTerm* die Stichwortsuche nach Connectoren, in Abhängigkeit von den durch sie bereitgestellten Daten. Connector können sich bei der Zugriffsschicht mithilfe der in vorigen Abschnitt diskutierten Selbstbeschreibung registrieren (*registerConnector)* und ebenfalls abmelden (*unregisterConnector*), wenn sie keine Anfragen mehr erhalten wollen. Die Connector Zugriffsschicht kapselt somit die gesamte Kommunikation mit den Connectoren ab, sodass diese nur Traffic aus einer Quelle entgegennehmen müssen und so mit entsprechenden Firewall-Policies einfacher abgesichert werden können.

Abbildung 45 zeigt einen Ausschnitt der der Meta-Informationen im Data Space Metadata Repository (vgl. Abschnitt 4.3.2): Für die Speicherung wird MongoDB als Datenbanklösung

eingesetzt. Diese dokumentenorientierte NoSQL-Datenbank ist flexibel erweiterbar und kann somit gewährleisten, dass spätere Anpassungen der Selbstbeschreibungsschemata im Testfeld Data Space einfach vorgenommen werden und komplexere hierarchische Zusammenhänge einfach abgebildet werden können (vgl. Győrödi u. a. 2015).


```
{
    "_id": {
        "$oid": "6216196a0c85ec313e4dd592"
    },
    "hostData": {
        "host": "aispsql-service.mirapla.svc.cluster.local",
        "port": 8001,
        "connectionTypeAPI": "BATCH",
        "userId": "a6cf150e-5b95-4520-96c0-ddd7ebfc20ab"
    },
    "scheme": {
        "name": "aispostgresql",
        "roleName": "aispostgresql",
        "owner": "OFFIS e. V.",
        "description": "This connector contains AIS and RADAR from (...)",
        "creationTS": {
            "$numberLong": "1612569600000"
        },
        "modificationTS": {
            "$numberLong": "1625961600000"
        },
        "graphQLSchema": "(...)",
        "exampleData": "(...)",
        "dictionary": [(...)]
    }
  ...
}
```


Abbildung 45: Auszug eines Datenbankeintrages im Data Space Schema Repository, der einen Connector beschreibt.

Wird nun eine Datenanfrage gestellt, kann durch die vorhandenen Einträge in der MongoDB iteriert werden und ein Matching der GraphQL-Anfrage und den existierenden GraphQL-Schemas durchgeführt werden. In der prototypischen Implementierung wird die Anfrage dem ersten Connector zugeordnet, welcher ein passendes Schema bereitstellt. Hier sind komplexere Prozesse denkbar, bei denen der Nutzer eine explizite Datenquelle angeben kann, oder aus einer Liste kompatibler Datenquellen wählen kann. Danach baut die Zugriffsschicht auf Basis der gespeicherten Selbstbeschreibungen eine Verbindung auf und leitet die validierte Datenanfrage weiter. So können fehlerhafte Anfragen direkt durch die Zugriffsschicht abgefangen werden. Der Connector antwortet wie im vorigen Abschnitt beschrieben auf die Datenanfrage oder lehnt diese ggf. ab, wenn eine Zugriffsrichtlinie verletzt wird.

5.4 Umsetzung der Datenstromverarbeitung

Als zentrale Komponente der Kappa-Architektur wurde in Abschnitt 4.4.1 das Ingestion Tool vorgestellt, welches in Form eines Streaming-Clusters zur Übermittlung der Daten zwischen den verschiedenen Datenquellen und Stakeholdern im serviceorientierten Testfeld dient. Als Industriestandard werden für nachrichtenorientierte Infrastrukturen, wie sie das Ingestion Tool auszeichnet, neben weiteren Alternativen hauptsächlich Apache Kafka oder RabbitMQ

eingesetzt, vgl. (Dubuc, Stahl, und Roesch 2020), (Auger, Exposito, und Lochin 2017), (Íñiguez und Galar 2022). Apache Kafka bietet jedoch einen deutlich größeren Nachrichtendurchsatz als RabbitMQ (Auger, Exposito, und Lochin 2017) bei einer vergleichbaren Latenz (Dobbelaere und Esmaili 2017). Kafka unterstützt zwar insgesamt weniger Programmiersprachen als RabbitMQ (Sharvari und Sowmya Nag 2019, Preprint), ist mit circa 17 jedoch ebenfalls durch eine hohe Kompatibilität ausgezeichnet. Dies ist insofern wichtig, da das FEDMS möglicherweise von einer Vielzahl verschiedener dezentral organisierter Infrastrukturen erreichbar seien muss (bspw. durch SuTs für die Dokumentation interner Systemzustände).

Für das FEDMS wurde konkret strimzi[11] eingesetzt, welches eine containerisierte Variante von Kafka bereitstellt und bei dem der Cluster über die native Kubernetes API verwaltet werden kann. So können über Kubernetes yaml-Konfigurationsdateien ein vollständiger Cluster aufgesetzt werden und gewünschte Verbindungsmechanismen und Authentifizierungs- und Autorisierungslösungen konfiguriert werden. Strimzi instanziiert dann die benötigten Container als Kubernetes-Services und bietet über die Kubernetes-API eine Schnittstelle für die Verwaltung von Kafka-Ressourcen (z.B. Topics) an. Weiterhin bietet strimzi als einzige Lösung auf dem Markt eine standardmäßige Lösung für die Authentifizierung und Autorisierung mittels OAuth2 und Keycloak (siehe Abschnitt 5.6).

In Abschnitt 4.4.2 wurde die dynamische Instanziierung von Topics durch die Nutzer des FEDMS beschrieben. Für diese Funktionalität muss eine programmatische Möglichkeit bestehen, um Topics im Kafka-Cluster zu erstellen, modifizieren und zu löschen und diese mit den Metadaten zu verknüpfen, die genutzt werden, um Workflows oder Szenarien zu beschreiben. Dies beschreibt eine wesentliche Aufgabe des Data Processing Layers und wird durch den *Topic Manager* umgesetzt (vgl. auch Abbildung 25). Abbildung 46 zeigt eine Übersicht über die Methoden des *Topic Manager*. Neben dem Erstellen und Löschen von Topics durch *deleteTopic* und *createTopic* können auch die Retention Time (siehe unten) modifiziert werden, oder das aktuelle Schema bzw. das aktuelle Offset (siehe Abschnitt 5.5) abgefragt werden. So kann, wie u.a. in Anforderung A5 gefordert, zusätzlich bestimmt werden, wie lange Daten im FEDMS (beim Datentreuhänder) gehalten werden dürfen.

Kafka Topics basieren auf mehreren Partitionen, die eine horizontale Skalierung von Topics ermöglichen, indem sie auf verschiedene Kafka-Server (Broker) aufgeteilt werden. Somit wird auch eine Redundanz geschaffen, falls einzelne Broker ausfallen sollten. Für Partitionen wird garantiert, dass die Nachrichten exakt in der Reihenfolge des Eintreffens gespeichert werden.

[11] https://strimzi.io/, abgerufen am 16.06.2022

Einzelne Nachrichten in einem Topic können nach dem Schreibprozess nicht mehr modifiziert werden. Außerdem wird pro Topic eine Retention Time festgelegt, nach deren Ablauf eine Nachricht aus dem Topic entfernt wird. (Apache Software Foundation 2017)

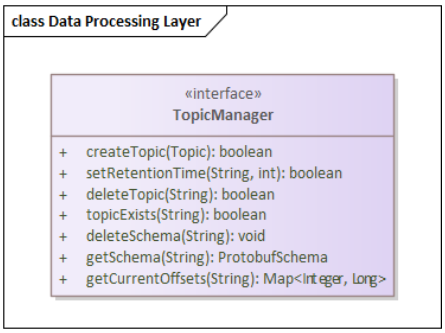

Abbildung 46: Topic Manager für das dynamische Verwaltung von Kafka-Topic zur Laufzeit des FEDMS.

Für den Einsatz im Testfeld wird für die prototypische Implementierung, die mit einem Single-Node Cluster und einem einzelnen Broker arbeitet, standardmäßig nur eine Partition pro Topic verwendet. Außerdem wird eine standardmäßige Retention Time von 7 Tagen gewählt, die das kurzfristige Speichern von Daten aus dem Testfeld innerhalb des FEDMS ermöglicht. Eine dauerhafte Speicherung ist innerhalb der Kappa-Architektur nicht vorgesehen und kann beispielsweise durch den Datentreuhänder erfolgen. Exemplarisch wurde hierzu ein konventionelles CKAN-Datenkatalogsystem (vgl. Abschnitt 3.1) im Testfeld aufgesetzt, welches über einen internen Verarbeitungsschritt des FEDMS verwendet wird, um Workflow oder Szenariodaten zu exportieren (siehe auch Anhang 8D). Hierzu wird mit Hilfe des Protobuf-Schemas der Inhalt jedes in einem Workflow referenzierten Topics exportiert und im json-Format gespeichert. Beim Export von Szenarien wird ebenfalls das Topic-Offset (entspricht den Szenario Markern aus Abbildung 36) berücksichtigt, und nur der markierte Bereich exportiert. Entsprechend der Retention Time in den Kafka-Topics wurde ein CleanUp-Service implementiert, der Workflow- und Szenariometadaten ebenfalls nach 7 Tagen entfernt. Beide Ablaufzeiten können ebenfalls auch manuell so konfiguriert werden, dass keine automatische Entfernung der Daten stattfindet.

Wie in Abschnitt 4.4.1 erläutert, findet die Implementierung des Speed-Layers der Kappa-Architektur anwendungsfallspezifisch und dezentral durch die verschiedenen Stakeholder des Testfeldes statt. Hierfür werden die entsprechenden REST-APIs des FEDMS genutzt, um den Kafka-Cluster zu verwalten und Kafka-Client-Libraries, um die eigentlichen Daten auf dem Cluster zu schreiben oder zu lesen. Neben dem Aufrufen der eigentlichen Funktionalitäten des FEDMS über diese Schnittstellen müssen auch Prozesse zur Authentifizierung und Autorisierung im Speed-Layer berücksichtigt werden. Um die Komplexität in der Interaktion mit dem FEDMS für die Nutzer zu verringern, wurde eine Java Client-Bibliothek implementiert, in der die

wichtigsten Schnittstellenaufrufe abstrahiert werden. Wie in Abbildung 47 dargestellt, wurde hierzu eine Session-basierte Modellierung vorgenommen, bei dem der Nutzer sich zunächst beim FEDMS authentifiziert und dann entsprechende Workflow-, Szenario-, Verarbeitungsschritt- und Topic-Sessions aus einer *parentSession* ableiten kann.

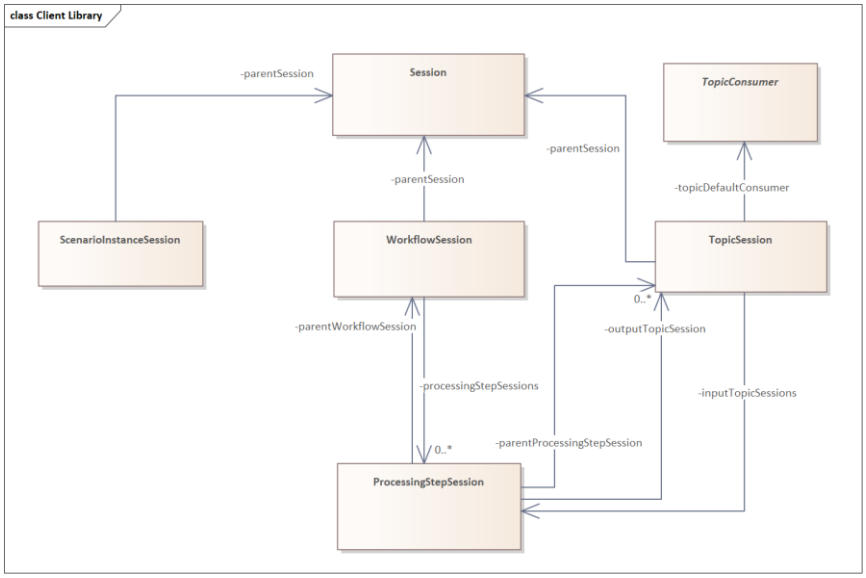

Abbildung 47: Session-basierte Abstraktion der FEDMS-Schnittstellen in der Java Client-Bibliothek.

Zudem verfügt die Bibliothek über eine spezielle Erweiterung eines Kafka-Consumers, die neben dem direkten Konsumieren von Daten weitere Funktionen ermöglicht: So können ältere Daten aus Topics in Originalgeschwindigkeit erneut konsumiert werden, um so etwa das Wiederholen eines Testdurchlaufes zu emulieren oder ein Topic ab einem bestimmten Offset zu konsumieren. Listing 3 zeigt einen Code-Ausschnitt, in dem mittels eines Session-Objekts eine Verbindung zum FEDMS hergestellt wird, und ein AIS-Datenstrom über einen GraphQL-Query instanziiert wird.

```
        //Authentifizierung des Nutzers beim FEDMS (nach vorheriger Konfiguration der Login-Daten):
        Session session = Session.loginWithPassword();
        //Definition einer GraphQL-Anfrage von Live-AIS Daten nördlich des 50. Breitengrades:
        String query = "subscription sub {AISDynamicMessage(filter: \"{\\\"latitude\\\" : { \\\"gte\\\" :
50 }}\"){ timestamp latitude longitude sog }}";
        //Ausführen der Anfrage:
        TopicSession inputTopic = session.executeQuery(query);
        //Verarbeiten der Daten:
        inputTopic.addListener(new TopicConsumer() {...});
```

Listing 3: Erstellen einer FEDMS-Session und Abfragen von Live AIS-Daten mittels der Java Client-Library.

Über das Listener-Pattern können dann wie im obigen Beispiel Datenströme abonniert und weiterverarbeitet werden. Ebenso können Datenartefakte (etwa aus Data Mining Workflows oder SuTs) über die Client-Bibliothek an den Kafka-Cluster übermittelt werden. Für die prototypische

Umsetzung wurde die Client-Bibliothek zwar ausschließlich in Java implementiert, es ist jedoch aufgrund der Verfügbarkeit von Kafka-Client Bibliotheken und der Verwendung standardisierter Technologien wie REST oder Protobuf genauso möglich für andere Plattformen entsprechende Bibliotheken zu entwickeln und bereitzustellen.

Als weiterer, nicht dezentralisierter Teil des Speed-Layers wurde in Abschnitt 4.4.4 die Möglichkeit vorgestellt durch das FEDMS unterstützte Datenverarbeitungsschritte bereitzustellen. Auf Implementierungsebene stellen solche vordefinierten Verarbeitungsschritte benutzerdefinierte Software dar, die auf der Infrastruktur des FEDMS ausgeführt wird. Um hier nicht die Technologieflexibilität zu verlieren, können Verarbeitungsschritte im OCI-Format (Open Container Initiative 2022) auf beliebigen Plattformen implementiert und in einer Container-Registry hochgeladen werden. Über die API des FEDMS kann dann im Workflow ein normaler Verarbeitungsschritt durch eine Referenz auf einen internen Verarbeitungsschritt erweitert werden. Wird der Workflow dann durch den Nutzer gestartet, wird über die Kubernetes-API ein Job erstellt, der die Ausführung des zugeordneten Container-Images auslöst. Zudem werden die zum Verarbeitungsschritt zugehörigen Topic-Identifier, und zusätzliche benutzerdefinierte Parameter als Umgebungsvariablen übergeben. Nach Terminierung des Containers oder Stoppen des Workflows durch einen Benutzer, wird der Job entfernt.

5.5 Verwaltung von Workflow- und Szenariometadaten

Für die Verwaltung von Workflow- und Szenariometadaten wird das in Abbildung 48 dargestellte Datenmodell verwendet: Das am OPM-orientierte Modell unterscheidet hier, wie in Abschnitt 4.4.2 erwähnt, zwischen Verarbeitungsschritten und Datenartefakten, die im Fall der prototypischen Umsetzung durch die Topics gegeben sind. Die Informationen zum Agenten werden als Attribute innerhalb der Klassen gespeichert. Die einzelnen *ProcessingSteps* einer Datenverarbeitungskette werden in der *Workflow* Klasse aggregiert. Zusätzlich kann eine Dokumentation hinzugefügt werden. Die in Anforderung A5 geforderten, benutzerdefinierten Metadaten zur Dokumentation der Nutzungsbedingungen von Datenartefakten können sowohl auf Workflow-, als auch auf Topicebene im Metadatenattribut *description* hinterlegt werden. Außerdem beinhaltet die *Documentation*-Klasse ein Attribut *signature* zum Halten von Signaturdaten für die Metadaten (siehe auch Abschnitt 4.6.2).

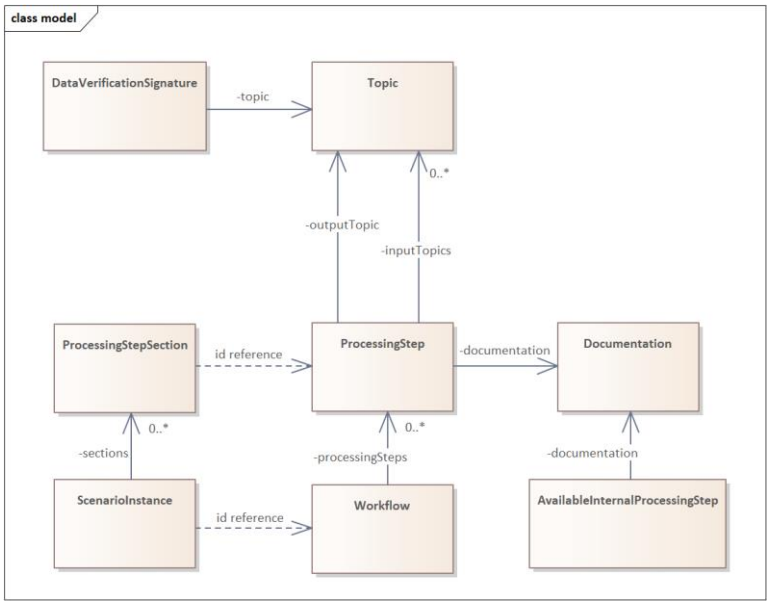

Abbildung 48: Datenmodell zur Verwaltung von Workflow- und Szenariometadaten in der Datenverarbeitungsschicht.

Szenariometadaten werden in der *ScenarioInstance*-Klasse gespeichert, die die Instanziierung eines Workflows repräsentiert. Analog zu den Verarbeitungsschritten eines Workflows werden in der *ProcessingStepSection* für die zu einem Verarbeitungsschritt gehörenden Teilbereiche der referenzierten Topics durch die Szenariomarker gespeichert. Dies passiert durch das Speichern von Topic-Offsets im Kafka-Cluster zur Start- und Endzeit eines Szenarios. Dasselbe Prinzip wird für die Aufzeichnung von Signaturen eingesetzt, bei der Daten aus einem durch Topic-Offsets markierten Bereich durch Nutzer des FEDMS signiert werden können.

Die Daten werden zur Verwaltung der Ressourcen des Data Processing Layers ähnlich zum Data Space Metadata Repository in einer MongoDB-Instanz gespeichert. Um Kompatibilität mit dem OPM zu gewährleisten, wurde in der *SharedLibrary* eine Transformation des internen Workflow-Datenmodells zu den standardisierten OPM-Datenformaten, insbesondere dem RDF-Format implementiert. Diese können dann etwa für einen Export genutzt werden, um die Metadaten für eine Archivierung von Workflowdaten zu speichern.

5.6 Absicherung

Zur Absicherung wurden auf Implementierungsebene mehrere Maßnahmen getroffen. Diese fokussieren sich auf die Authentifizierung von Clients, der Autorisierung von Zugriffsanfragen auf Daten und Metadaten, und der Verschlüsselung der genutzten Kommunikationskanäle.

Als grundlegende Lösung zur Absicherung des FEDMS wurde Keycloak eingesetzt. Keycloak ist eine Plattform für das Verwalten von Identitäten und Zugriffsrechten und lässt sich als Single-

Sign-On Lösung mittels standardisierter Schnittstellen wie OpenID Connect, OAuth2 und SAML in beliebige Systeme integrieren. Weiterhin existiert eine grafische Benutzeroberfläche, welche Möglichkeiten zur Verwaltung von Identitäten und Zugriffsrechten bietet. Zudem kann Keycloak auch als Identity Broker eingesetzt werden, der es Nutzern ermöglicht sich beim SSO-Prozess auch mit Identitäten anderer IdPs zu authentifizieren. Somit erfüllt Keycloak alle notwendigen Voraussetzungen, um als Identity Provider bzw. Broker für das in Abschnitt 4 ausgearbeitete Konzept zu dienen (vgl. insbesondere Abschnitte 4.5.1 und 4.6.1) und bietet im Vergleich zu anderen Lösungen eine einfache Bedienbarkeit und einen angemessenen Umfang an Features. (Divyabharathi und Cholli 2020)

Die für Dritte aus dem Testfeld erreichbaren Komponenten des FEDMS schließen die REST-Schnittstelle der Datenverteilungsschicht, den Kafka-Cluster (inkl. Schema-Registry), und (im allgemeinen Fall) die Schnittstelle der Connector-Zugriffsschicht ein. Keycloak stellt ebenfalls eine Schnittstelle (API und grafische Benutzeroberfläche) bereit, die aber bereits standardmäßig abgesichert ist. Alle anderen Schnittstellen, die durch die Datenverteilungsschicht bereitgestellt werden, einschließlich der verwendeten MongoDB-Instanz sind nur innerhalb des Kubernetes-Clusters erreichbar. Die Absicherung der einzelnen Schnittstellen wird im Folgenden vorgestellt. Mit Hilfe von Keycloak werden OAuth2 und das darauf aufbauende OpenID Connect Protokoll für Authentifizierung und Autorisierung verwendet.

OAuth2 verfolgt ein Modell, in welchem ein *Resource Owner* bestimmte Artefakte besitzt und diese bestimmten Endnutzern zur Verfügung stellt, die mit Hilfe eines Clients über einen *Authorization Server* beim *Resource Owner* Zugriff auf das gewünschte Artefakt erhalten. Dies funktioniert mittels eines Tokens, den der Client vom *Authorization Server* erhält und dem *Resource Owner* übergibt, um nachzuweisen, dass eine entsprechende Zugriffsberechtigung existiert. Um festzulegen, zu welchen Artefakten ein Nutzer Zugriff hat, kann im Token zusätzlich ein *Scope* festgelegt werden. In der prototypischen Implementierung wird ein Testfeldweiter *Scope* gewählt, sodass Tokens bei allen Diensten im Testfeld gültig sind. Für ein stärker dezentralisiertes Setup ist es auch denkbar, dass *Scopes* für jeden Service existieren. Da ein solcher OAuth2-Token jedoch keine Aussage über die Authentifizierung eines Nutzers trifft, wird das OAuth2-Protokoll durch OpenID Connect (OIDC) ergänzt, welches einen weiteren Token, den ID-Token bereitstellt. Dieser Token wird im JSON Web-Token (JWT) Format serialisiert, durch den IdP signiert und enthält u.a. eine User ID und eine Verfallszeit. (Richer und Sanso 2017)

Connector Zugriffsschicht. Die REST-Schnittstelle der Connector Zugriffsschicht dient ausschließlich zur Kommunikation mit den Connectoren und der Entgegennahme von Datenanfragen aus der Datenverteilungsschicht. Für die Datenverteilungsschicht ist diese Schnittstelle über das interne Netz der Kubernetes Netzwerkschicht erreichbar. Für das

prototypische Deployment wurden die Connectoren ebenfalls im Kubernetes-Cluster aufgesetzt. In einem Produktionssetup kann eine sichere Verbindung zwischen den Connectoren und der Zugriffsschicht hergestellt werden, bei der durch eine Firewall nur der Traffic registrierter Connectoren zugelassen wird. Bei der Registrierung eines Connectors muss dieser sich mittels OIDC-Token authentifizieren. Hierbei muss der verwendeten Identität die *connector*-Rolle zugewiesen sein. Außerdem teilt der Connector als Teil der Selbstbeschreibung bei der Registrierung den Namen der Rolle mit, die ein Nutzer benötigt, um Daten dieses Connectors anzufragen. Wird von einem Nutzer eine Datenanfrage gestellt, wird nach der Authentifizierung des Nutzers überprüft, ob dieser die entsprechende Rolle besitzt, um Daten des angefragten Connectors abzurufen.

Kafka-Cluster. Kafka besitzt standardmäßig ebenfalls die Möglichkeit Nutzer via OAuth2 zu authentifizieren, bevor sie auf die Ressourcen eines Clusters zugreifen können. Dafür kommt der sogenannte *OAUTHBEARER*-Mechanismus zum Einsatz. Hierbei wird den Producern und Consumern der Kafka Client-Bibliothek jeweils ein kurzlebiger OAuth2-Token übergeben, der dann vor dem Datenaustausch durch den Broker verifiziert wird und nach 15 Minuten abläuft. So kann sichergestellt werden, dass Nutzer sich regelmäßig bei Keycloak authentifizieren müssen. Damit eine dynamische Verwaltung der Zugriffsrechte für die Topics auf dem Cluster vorgenommen werden kann, besitzt die Datenverteilungsschicht Admin-Rechte für die Verwaltung des Clusters und kann somit benutzerdefinierte Modifikationen von Topic-Zugriffsrechten direkt umsetzen. Jegliche Kommunikation mit dem Cluster ist außerdem über das SSL-Protokoll verschlüsselt.

Datenverteilungsschicht. Neben dem Kafka-Cluster ist die Datenverteilungsschicht die von Testfeldnutzern verwendete Schnittstelle, um mit dem FEDMS zu interagieren. Die Datenverteilungsschicht ist über OIDC abgesichert. Bevor ein Nutzer die Schnittstelle nutzen kann, muss er sich bei Keycloak authentifizieren und einen OIDC-Token vorweisen. Dabei wird verifiziert, dass der Nutzer die Standardrolle zur Verwendung des FEDMS besitzt. Bestimmte Funktionen der REST-API können ebenfalls nur durch Nutzer mit bestimmten Rollen durchgeführt werden. So kann beispielsweise nur ein Nutzer mit der Rolle *processingstep_admin* neue interne Verarbeitungsschritte (siehe Abschnitt 4.4.4) für andere Nutzer freischalten. Die Kommunikation mit der Datenverteilungsschicht erfolgt über HTTPS und ist somit via TLS vollständig verschlüsselt.

Datenverarbeitungsschicht. Die Datenverarbeitungsschicht bietet (abgesehen vom Kafka-Cluster) keine öffentlich verfügbare Schnittstelle im Testfeld an und wird lediglich von der Connector-Zugriffsschicht und der Datenverteilungsschicht angesprochen. Im prototypischen Setup dieser Arbeit ist die Datenverarbeitungsschicht lediglich aus dem Netzwerk des Kubernetes-Clusters erreichbar. In einem Produktionssetup können Firewall-Konfigurationen

verwendet werden, um nur den vorgesehenen Traffic zwischen den Schichten zu erlauben. Die Datenverteilungsschicht verfügt über Admin-Rechte für das Zuweisen von Berechtigungen zum Zugriff auf Topics des Kafka-Clusters, da sich die Nutzer bei der Datenverteilungsschicht direkt authentifizieren. In der vorliegenden prototypischen Implementierung erfordert dies ein Vertrauensverhältnis zwischen den Betreibern der Datenverarbeitungsschicht und der Datenverteilungsschicht. Im vorliegenden Aufbau werden die entworfenen Schichten allerdings durch denselben Stakeholder (Testfeldbetreiber) zur Verfügung gestellt. Für den allgemeinen Fall ist es denkbar, dass Betreiber einer Datenverarbeitungsschicht (inkl. Kafka-Cluster) oder einer Connector-Zugriffsschicht ihre eigene Instanz einer Datenverteilungsschicht nutzen. Weiterhin hat die Datenverarbeitungsschicht exklusiven Zugriff auf die MongoDB zur Verwaltung der Workflow- und Szenariometadaten, die ebenfalls die Zugriffsrechte für die Ressourcen pro Nutzer speichern. Für diese wurde ein einfaches Konzept implementiert, in dem ein Nutzer Schreibrechte, Leserechte und Adminrechte (ermöglicht das Verteilen der Rechte) besitzen kann. Standardmäßig erhält der Ersteller einer Ressource alle diese Rechte und kann dann als Admin weiteren Nutzern Zugriff gewähren.

Maritime Connectivity Platform. Um die MCP als zusätzlichen Identitätsprovider zu verwenden, wurde in Keycloak ein neuer OIDC-IdP angelegt. Neben dem normalen lokalen Login kann sich ein Nutzer dann auch mit einem OIDC-Token der MCP bei Keycloak authentifizieren. Dabei wird ein neuer lokaler Benutzeraccount angelegt, der auf eine föderierte Identität bei der MCP verweist. Der MCP-Token wird dann bei der Authentifizierung gegen einen lokalen Token ausgetauscht (*Token Exchange*). Ein Testfeld-Service, der die Bestätigung benötigt, dass ein Nutzer über eine MCP-Identität verfügt, kann dies über einen API-Endpunkt von Keycloak prüfen. Zudem stellt die MCP eine Public-Key Infrastruktur bereit, bei welcher jede Identität an ein ECDSA-Schlüsselpaar gekoppelt ist. Dieses Schlüsselpaar kann in der FEDMS-Client Bibliothek importiert werden, und genutzt werden, um Daten auf einem Topic zu signieren. Dazu wird eine Standardimplementierung eines *Verifying Consumers* bereitgestellt (siehe Abschnitt 4.6.2), die benutzerdefinierte Teilbereiche von Daten aus einem Topic abruft und diese mit dem privaten Schlüssel signiert. Die Signatur wird dann in der MongoDB-Instanz gespeichert und kann durch andere Stakeholder abgerufen und verifiziert werden. Der öffentliche MCP-Schlüssel kann in Keycloak im Benutzeraccount hinterlegt und durch die Datenverteilungsschicht anderen Nutzern zur Verifizierung von Signaturen bereitgestellt werden.

Verwaltung mittels Keycloak. Nutzer, Rollen und die Zugriffsrechte für den Kafka-Cluster können mit dem Keycloak-Admin Interface auf einer grafischen Benutzeroberfläche verwaltet werden. Neben dem Anlegen neuer Benutzer können auch Rollen erstellt und zugewiesen werden (z.B. für den Zugriff auf einzelne Connectoren). Weiterhin können die Berechtigungen für den Kafka-Cluster modifiziert werden und weitere externe Identitätsprovider hinzugefügt werden.

6 Evaluation der FEDMS-Architektur

Ausgehend von der eingangs definierten Forschungsfrage, wie dezentral organisierte maritime Daten für einen datengetriebenen Forschungs- und Entwicklungsprozess im Testfeldkontext bereitgestellt werden können, wurden im vorangegangenen Teil dieser Arbeit Anforderungen erhoben und mit verwandten Arbeiten verglichen. Weiter wurde ein Konzept für ein Forschungs- und Entwicklungsdatenmanagementsystem entwickelt, dessen prototypische Umsetzung dann in Kapitel 5 diskutiert wurde. Um nun zu prüfen, ob das Lösungskonzept aus Kapitel 4 die Forschungsfrage beantworten kann, wird folgende Evaluationshypothese untersucht:

Das Forschungs- und Entwicklungsdatenmanagementsystem ist für die Bereitstellung von dezentral organisierten maritimen Daten im Testfeldkontext geeignet und kann den identifizierten Handlungsbedarf in Bezug auf die verwandten Arbeiten abdecken.

Dafür wird zunächst in Abschnitt 6.1 ein strukturiertes Vorgehen vorgestellt. Weiter werden in Abschnitt 6.2 im Rahmen dieses Vorgehens Nutzungsszenarien für das FEDMS hergeleitet, die die Basis für drei Fallstudien darstellen. Aus der Durchführung der Fallstudien soll dann der Erfüllungsgrad der Ziele und Anforderungen der Arbeit ermittelt werden. Fallstudie I (Abschnitt 6.3) beschäftigt sich mit dem Entwurf eines Prädiktionsmodell für den Schiffsverkehr, Fallstudie II (Abschnitt 6.4) mit der Untersuchung eines Assistenzsystems zur Kollisionsvermeidung und Fallstudie II (Abschnitt 6.5) mit der Contract-basierten Zertifizierung eines Systems im Testfeld. Die in den Anforderungen A17 und A18 definierten Performanz-Kriterien sind für alle Nutzungsszenarien relevant und werden deshalb separat in Abschnitt 6.6 untersucht. Abschließend folgt eine Diskussion der Ergebnisse in Abschnitt 6.7 und eine Analyse bzgl. der Abdeckung der Anforderungen und Ziele dieser Arbeit in Abschnitt 6.8.

6.1 Vorgehen

Als zentrales Lösungskonzept zur Beantwortung der Forschungsfrage wurde in Kapitel 4 eine Systemarchitektur entwickelt. Der zu evaluierende Kern der Arbeit ist also zunächst durch das abstrakte Konzept dieser Systemarchitektur gegeben.

Für die Evaluation von komplexen System- und Softwarearchitekturen gibt es in der wissenschaftlichen Literatur bereits zahlreiche Methoden, die sich in vier verschiedene Bereiche aufteilen lassen (Santos u. a. 2022):

- Bei der **erfahrungsbasierten Evaluation** wird das Wissen von Experten genutzt, um zu entscheiden, ob eine Architektur den Erwartungen der Stakeholder entspricht und ihre Anforderungen erfüllt.

- Architekturen können in bestimmten Fällen auch durch das Untersuchen **mathematischer Modelle** evaluiert werden, wenn sich bestimmte Eigenschaften formal beweisen lassen.

- Die **simulationsbasierte Evaluation** nutzt Simulationsmodelle um bestimmte Eigenschaften wie Performance oder Zuverlässigkeit zu untersuchen.

- Die **Szenario-basierte Evaluation** setzt bestimmte Szenarien ein, um eine qualitative Untersuchung der Architektur durchzuführen. Dadurch kann eine ganze Reihe an Kriterien abgedeckt werden, und zusätzlich können Risiken oder Schwachstellen identifiziert werden.

Nach Santos u. a. (2022) beschäftigen sich erfahrungsbasierte Ansätze eher mit den subjektiven Eindrücken der Experten und weniger mit expliziten Attributen. Weiter fokussieren sich mathematische Modelle und simulationsbasierte Evaluationsmethoden tendenziell eher auf spezifische Attribute, während Szenario-basierte Ansätze eine Vielzahl von Kriterien abdecken können (Santos u. a. 2022). Da es sich bei der entwickelten Systemarchitektur um eine flexibel einsetzbare Lösung handelt, die einer Vielzahl von Anforderungen gerecht werden muss (vgl. Abschnitt 2.6) ergibt es hier also am meisten Sinn einen Szenario-basierten Ansatz zu verwenden. Eine erfahrungsbasierte Evaluation käme zwar ebenfalls in Frage, birgt aber aufgrund der multiplen Stakeholdergruppen die Gefahr, dass die Experten der einzelnen Bereiche keine gemeinsame Gesamtsicht auf die Bewertung der Architektur etablieren können, da ggf. konfliktäre Interessen bestehen (vgl. Abschnitt 2.6) und, dass nicht immer vollständig ersichtlich ist, auf welchen objektiven Kriterien erfahrungsbasierte Evaluationsergebnisse aufbauen.

Eine populäre Methodik in der Szenario-basierten Evaluation von Architekturen ist das von Kazman u. a. (1994) entwickelte SAAM-Framework (Software Architecture Analysis Method). SAAM ist eine fünf-Schritte Methode, bei der Funktionalität und Struktur der Architektur in Zusammenhang gebracht werden, Qualitätsattribute ausgewählt und anhand von abgeleiteten Nutzungsszenarien evaluiert werden. Dabei wird ein besonderer Fokus auf Struktur, Funktionalität und den durch die Architektur hergestellten Zusammenhang zwischen diesen beiden Elementen gelegt. (Kazman u. a. 1994)

Molter (1999) erweiterte SAAM um den Aspekt der Wiederverwendbarkeit einer Architektur (das Verfahren wird im Folgenden als ESAAM bezeichnet). Hierbei wird die Fragestellung evaluiert, inwiefern Architekturen, die für einen bestimmten Use-Case eingesetzt wurden, in einem anderen Use-Case derselben Domäne wiederverwendet werden können. Diese Frage stellt sich insbesondere für Architekturen im serviceorientierten Testfeld, da hier kontinuierlich neue Testaufbauten durch die Stakeholder vorgenommen werden. Die wesentliche Erweiterung von SAAM in ESAAM ist die Verwendung sogenannter *Protoszenarien*. Dabei handelt es sich um

„generische Beschreibungen von Wiederverwendungssituationen oder Interaktionen mit einem System“ (Molter 1999, 5). Diese werden einerseits verwendet, um daraus konkrete Szenarien abzuleiten und zum anderen, um allgemeine Betrachtungen durchzuführen, welche strukturellen Elemente der Architektur für diese Szenarien genutzt werden. Abbildung 49 zeigt eine Übersicht über die wichtigsten Schritte in ESAAM: Nachdem die Protoszenarien erhoben und zu konkreten Szenarien erweitert wurden, können noch weitere normale Szenarien hinzugefügt werden, um eine Selektion aus Kandidaten-Szenarios zusammenzustellen, aus denen die eigentlichen Testszenarien ausgewählt werden. Die einzelnen Szenarien werden dann als *direkt* oder *indirekt* klassifiziert: Direkte Szenarien können ohne eine Veränderung der Architektur durchgeführt werden, indirekte nur mit einer Anpassung oder Erweiterung der Architektur. Bei der Evaluation können dann die konkreten Szenarien durchgeführt und ausgewertet werden, Protoszenarien können zusätzlich auf einer abstrakteren Ebene durchgespielt werden. Weiterhin lassen sich Zusammenhänge zwischen einzelnen Szenarien analysieren, sodass typische Nutzungsmuster oder ungewünschte Zustände erkannt werden können. Schließlich können auf der Basis dieses Verfahrens Vergleiche mit anderen Architekturen durchgeführt werden. (Molter 1999)

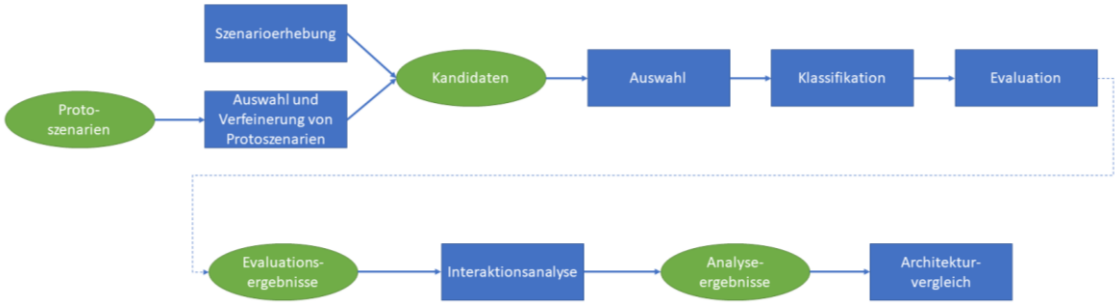

Abbildung 49: Erweiterung von SAAM, durch Berücksichtigung von Wiederverwendbarkeitsaspekten (Ausschnitt, vgl. Molter 1999).

Zur Erhebung der Protoszenarien kann zur Evaluation des Lösungskonzeptes das 3-Phasenmodell aus Abschnitt 2.2.3 dienen: Die drei Phasen beschreiben wiederkehrende Tätigkeiten bei der Nutzung eines serviceorientierten Testfeldes.

6.2 Herleitung der zu testenden Szenarien

Basierend auf dem 3-Phasenmodell aus Abschnitt 2.2.3 und den in Abschnitt 2 diskutierten Grundlagen, wurden insgesamt 22 Protoszenarien identifiziert, die in Abbildung 50 mit ihrer Zugehörigkeit zu der entsprechenden Phase dargestellt sind. Die Klassifizierung in direkte (dargestellt in schwarz) und indirekte Szenarien (dargestellt in blau) wurde hier ebenfalls bereits durchgeführt, und wird in den folgenden Unterabschnitten genauer erläutert. Die Auswahl und Konkretisierung der Szenarien finden durch das Definieren von drei Fallstudien in den folgenden

Abschnitten statt. Die Fallstudien sollen für ein FEDMS relevante Teile eines Systementwicklungsprozesses und mit diesen einhergehende Testfeldnutzungsmuster abbilden.

In Fallstudie I wird ein Prädiktionsmodell für Schiffsverkehr entwickelt, welches live eingesetzt werden soll, um Schiffsverhalten in der Elbe vorherzusagen. Die Fallstudien II und III zentrieren sich um die Entwicklung eines maritimen Assistenzsystems zur Kollisionsrisikobewertung, dessen Verhalten in Fallstudie II in einem V+V-Prozess untersucht wird und welches in Fallstudie III einen Teil eines exemplarischen Zertifizierungsprozess durchlaufen soll. Dieses Kollisionsrisikobewertungssystem steht zwar nicht im direkten Zusammenhang mit dem Modell aus Fallstudie I, es ist jedoch denkbar, dass eine solche Modellentwicklung als Vorarbeit stattgefunden hat. Für die Durchführung der Fallstudien wird ein reales Testfeld benötigt. Hierzu wird die grundlegende Infrastruktur des in Abschnitt 3.3.1 vorgestellten Testfelds eMIR verwendet, auf dessen Datenquellen bereits die prototypische Implementierung des FEDMS aufbaut.

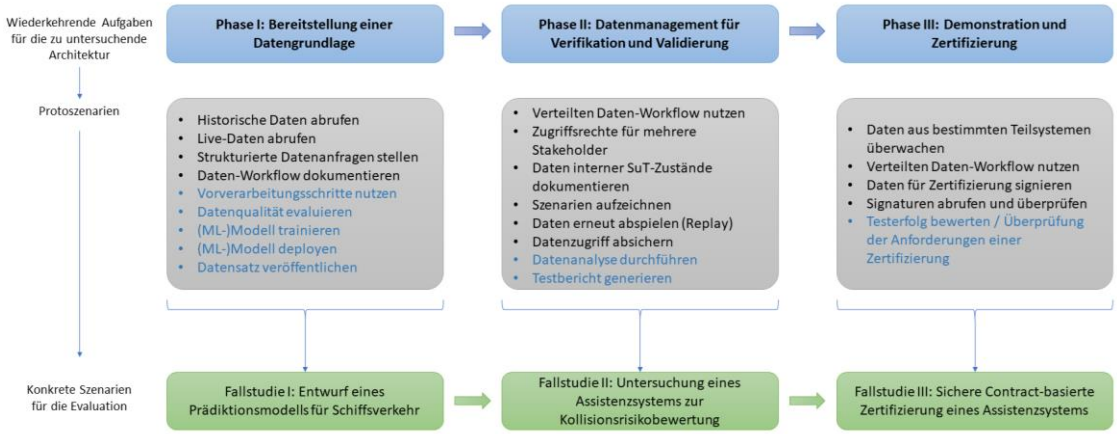

Abbildung 50: Übersicht - ESAAM-basiertes Vorgehen zur Evaluation des FEDMS.

6.3 Fallstudie I: Entwurf eines Prädiktionsmodells für Schiffsverkehr

Zuerst soll eine Fallstudie zur Repräsentation der Aktivitäten in *Phase I* des Testfelddatenmanagements durchgeführt werden. Wie in Abschnitt 2.4 erläutert, stellen AIS-Daten die mit am häufigsten verwendete Datenquelle in der datengetriebenen maritimen Forschung dar. Häufig werden AIS-Daten als Grundlage für die Entwicklung von (ML-)Modellen zur Verwendung in intelligenten Navigationssystemen eingesetzt (Tu u. a. 2018). Typischerweise werden solche Modelle auf historischen Datensätzen trainiert, und dann z.B. als Teil eines intelligenten Navigationssystems verwendet. Die Entwicklung eines solchen Modells findet klassischerweise mittels des KDD-Prozesses (vgl. Abschnitt 2.1 und 2.4) statt und beinhaltet entsprechende Vorverarbeitungsschritte und eine Evaluation der Datenqualität. Somit eignet sich

der Entwurf eines ML-Modells zur Verwendung in einem intelligenten maritimen Navigationssystem auf der Grundlage von AIS-Daten als konkrete Umsetzung der Proto-Szenarien.

6.3.1 Ausgewählte Szenarien

Das betrachtete Problem dieser Fallstudie bezieht sich auf die Vorhersage von Schiffsverhalten im dichten Verkehr einer Wasserstraße (siehe Abbildung 51): Zur Anwendung im Verkehrsmanagement, oder als Prädiktion für eine Kollisionsverhütung soll eine binäre Entscheidung vorhergesagt werden. Im Folgenden ist diese binäre Entscheidung gegeben durch die Fragestellung, ob ein sich in die Elbe einfahrendes Schiff auf der Höhe von Brunsbüttel entscheidet in den Nord-Ostsee-Kanal einzufahren, oder die Fahrt auf der Elbe in Richtung Hamburg fortsetzt.

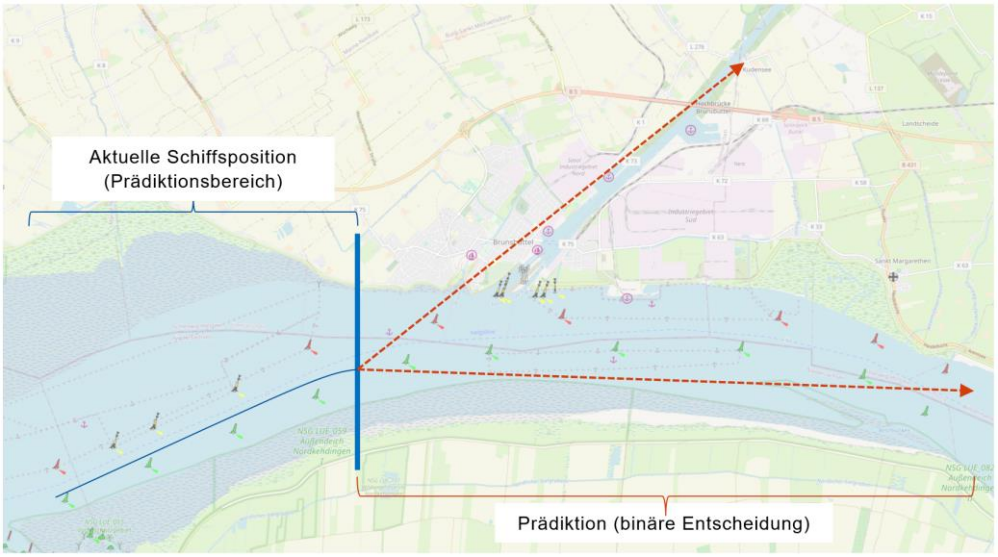

Abbildung 51: Prädiktion der Navigationsentscheidung eines Schiffes basierend auf aktuellen Schiffsdaten (Kartendaten: OpenSeaMap).

Dazu werden nun die in Abbildung 50 definierten Protoszenarien konkretisiert:

(S01) Historische Daten abrufen. Für das Training des ML-Klassifikators muss ein Trainingsdatensatz erstellt werden, der AIS-Daten aus dem Bereich um Brunsbüttel enthält. Das FEDMS wird genutzt, um diese Daten aus einer verfügbaren Quelle abzurufen.

(S02) Live-Daten abrufen. Das Ziel der Entwicklung des beschriebenen Prädiktionsmodells ist die spätere Einbettung in ein produktives System. Trotzdem sollte es möglich sein das Modell als einzelne Komponente mit Hilfe eines Testfeldes separat zu evaluieren. Über das FEDMS werden also Live AIS-Daten abgerufen und in das Modell eingespeist, welches so unter realistischen Testbedingungen evaluiert werden kann.

(S03) Strukturierte Datenabfragen stellen. Für das Abrufen der Trainingsdaten bietet es sich an nur Daten abzufragen, die im geografischen Bereich der Entscheidungsfindung vorliegen. Zudem sollte die Größe des Trainingsdatensatzes begrenzt werden. Neben der Anzahl der Datenpunkte sind bestimmte Attribute wie die Namen der betrachteten Schiffe möglicherweise nicht relevant und können direkt herausgefiltert werden. Ähnliches gilt auch für die Live-Daten: Hier ergibt es wenig Sinn eine Prädiktion für eine Entscheidung durchzuführen, wenn ein Schiff sich außerhalb der Elbe befindet.

(S04) Daten-Workflow dokumentieren. Da es sich bei einem KDD-Prozess um eine komplexe Datenverarbeitungskette handelt, sollte hier mit Hilfe des FEDMS eine entsprechende Dokumentation durchgeführt werden, sodass die Ergebnisse später nachvollziehbar sind und durch andere Teilnehmer reproduziert werden können.

(S05) Datenqualität evaluieren. Wie in den vorangegangenen Kapiteln dargelegt wurde, sind AIS-Daten häufig von geringer Datenqualität. Um hier eine grobe Einschätzung zu haben, mit welcher Datenqualität ein Trainingsdatensatz erzeugt wurde, soll eine Evaluation der Datenqualität der verwendeten AIS-Daten stattfinden. Aufgrund der Kontextabhängigkeit von Datenqualitätsmaßen ist hier ein spezielles Modul notwendig, welche die eigentliche FEDMS-Architektur mit Hilfe eines internen Verarbeitungsschrittes (siehe Abschnitt 4.4.4) erweitert.

(S06) Vorverarbeitungsschritte nutzen. Eine reine Datenqualitätsbewertung ist zwar hilfreich für die Analyse eines Datensatzes, trägt aber nicht zur Verbesserung der Datenqualität des Trainingsdatensatzes bei. Für das Training des Modells sollen nur AIS-Datenpunkte verwendet werden, die vollständig sind. Hierzu wird wie bei S05 eine Erweiterung der FEDMS-Architektur durch einen internen Verarbeitungsschritt vorgenommen.

(S07) ML-Modell trainieren. Schließlich wird das ML-Modell mit den durch das FEDMS bereitgestellten und vorverarbeiteten Daten trainiert. Das Entwerfen und Trainieren von ML-Modellen befindet sich außerhalb der vorgesehenen Einsatzmöglichkeiten des FEDMS. Die Architekturerweiterung dieses indirekten Szenarios umfasst also nicht das Modell selbst, sondern nur die Schnittstelle zwischen FEDMS und Modell, um die Trainingsdaten abzurufen.

(S08) ML-Modell deployen. Ähnlich wie beim Training des ML-Modells geht es beim Deployment des ML-Modells um das Herstellen einer Verbindung zwischen FEDMS und dem Modell selbst. In diesem Szenario soll eine kontinuierliche Verbindung aufgebaut werden, die es ermöglicht Live AIS-Daten aus dem eMIR-Testfeld abzurufen und zur direkten Klassifikation in das ML-Modell einzuspeisen. Die Ausgabe des Modells soll über einen längeren Zeitraum überwacht werden können.

(S09) Veröffentlichen eines Datensatzes. Für die Reproduzierbarkeit durch andere Nutzer des Testfeldes soll der Datensatz, welcher zum Training des Modells verwendet wurde, außerhalb des

FEDMS veröffentlicht werden. Hierbei soll der Datensatz so anonymisiert werden, dass kein Zusammenhang zu personenbezogenen Daten hergestellt werden kann.

6.3.2 Testaufbau und -durchführung

Das betrachtete Problem der Prädiktion einer Entscheidung ist ein klassisches Klassifikationsproblem. Gegeben ein Trainingsdatensatz $(x_1, y_1), \dots, (x_n, y_n)$ mit einer unbekannten Abbildung f mit $f(x) = y$, soll eine Funktion h gefunden werden, die f möglichst gut approximiert. Im Fall, dass nur eine endliche Anzahl von möglichen Werten (Klassen) für y existieren, handelt es sich um ein Klassifikationsproblem. (vgl. Russell und Norvig 2010, 695f.)

Im beschriebenen Fall gilt also $y = \{\text{Elbe}, \text{Nord-Ostsee-Kanal}\}$. Weniger trivial ist die Auswahl der Eingabeparameter x: Hierzu gibt es in der Literatur verschiedenste Ansätze, was die Auswahl der in den AIS-Daten vorhandenen Attribute gilt, wenn es um die Vorhersage von Verkehrsverhalten geht, vgl. z.B. (Murray und Perera 2020) und (Hexeberg u. a. 2017). Zur Demonstration beschränkt sich das betrachtete Modell jeweils auf Informationen aus der letzten bekannten AIS-Nachricht eines Schiffes mit dynamischen Informationen. Dabei gilt $x = (\phi, \lambda, v, \alpha, \omega, S)$ mit Breitengrad ϕ und Längengrad λ der aktuellen Position, Geschwindigkeit über Grund (SOG) v, Kurs über Grund (COG) α, Rotationsgeschwindigkeit ω und dem AIS-Navigationsstatus S.

Abbildung 52 zeigt einen Überblick über die verwendete FEDMS-Konfiguration: Zunächst werden historische AIS-Daten abgerufen (S01). Hierzu wird der Connector für die historische AIS PostgreSQL Datenbank des eMIR-Testfeldes verwendet (vgl. Abschnitt 5.2.1). Dafür wurde ein Query geschrieben (siehe Listing 4), der die Daten nur in einem definierten Bereich um Brunsbüttel abruft, die Attribute auf $(\phi, \lambda, v, \alpha, \omega, S)$ beschränkt und insgesamt 1.000.000 Datenpunkte abfragt (S03). Ein ähnlicher Query wurde im späteren Verlauf ebenfalls für die Live-Daten verwendet.

Für die Evaluierung der Datenqualität (S05) wurde eine python-Applikation eingesetzt, die an das Konzept aus Abschnitt 4.4.5 angelehnt ist und die Konsistenz und Vollständigkeit der Daten im Bezug zur AIS-Spezifikation bewertet und auf einer Skala von 0-100% einordnet. Die python-Applikation wird in die Container-Registry des FEDMS übertragen und kann dann nach einer Registrierung als Verarbeitungsschritt in einem Workflow ausgewählt werden. Die ermittelten Werte zur Datenqualität werden dann schließlich in einem Bericht ausgegeben und können vom Nutzer über die Logs eingesehen werden. Um den Nutzen des zweiten internen Verarbeitungsschritts zur Entfernung unvollständiger Datenpunkte zu untersuchen, wird der Verarbeitungsschritt zur Datenqualitätsbewertung zwei Mal im Workflow eingesetzt, sodass durch den Vergleich der beiden Berichte überprüft werden kann, ob eine tatsächliche Aufwertung

der Datenqualität stattfindet. Der eingesetzte interne Vorverarbeitungsschritt liest den AIS-Rohdatensatz ein, und entfernt zur Verbesserung der Datenqualität die Datenpunkte, bei denen nicht alle Attribute vollständig vorhanden sind (S06). Dieser interne Verarbeitungsschritt wurde ebenfalls mittels python implementiert und als Container-Image in die Container-Registry des FEDMS übermittelt.

```
{  __datapoint_AISDynamicMessage(first: 1000000, after: 5434661631, filter: "{
        \"latitude\" : {\"gte\" : 53.63068, \"lte\" : 54.23068 },
        \"longitude\" : {\"gte\" : 8.77185, \"lte\" : 9.57185 }
    }")
    {
        mmsi
        latitude
        longitude
        sog
        cog
        rot
        navstatus
    }
}
```

Listing 4: GraphQL-Query zur Abfrage der Rohdaten für den Trainingsdatensatz.

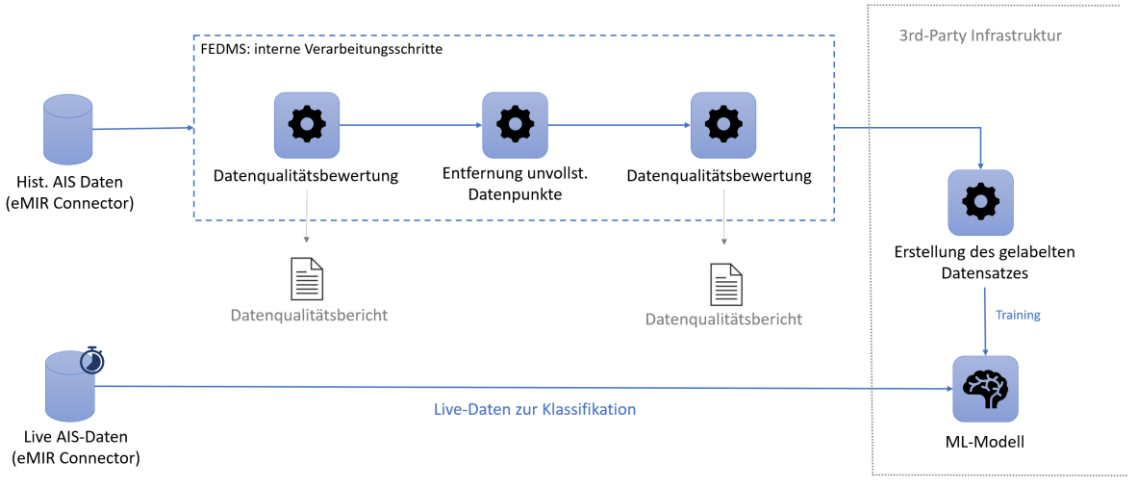

Abbildung 52: Testaufbau für Fallstudie I – Training eines ML-Modells auf Basis historischer Daten und Klassifikation von Live-Daten.

Als ML-Modell wird ein Random Forest Classifier (vgl. Breiman 2001) mit einer maximalen Baumtiefe von 10 und eine Gesamtanzahl von 10 Entscheidungsbäumen eingesetzt. Für die Implementierung wurde die Umsetzung des Random Forest Modells der Java-Bibliothek WEKA (Witten, Frank, und Hall 2011) verwendet. Zudem wurde eine Java Applikation entwickelt, die durch Verwendung der FEDMS Client-Bibliothek die Workflowmetadaten für das FEDMS erstellt, und die Ausführung der Rohdatenabfrage triggert. Nachdem die vorverarbeiteten Daten aus einem durch das FEDMS bereitgestellten Topic entnommen wurden, wird der

Trainingsdatensatz erstellt. Hierzu werden zunächst anhand des Kurses α eines Schiffes alle Datenpunkte mit landauswärts fahrenden Schiffen entfernt. Danach wird eine Gruppierung von Datenpunkten pro Schiff durchgeführt. Dies passiert anhand der Maritime Mobile Service Identity (MMSI) die als eindeutiger Schiffsidentifikator ebenfalls in den Daten vorhanden ist (vgl. Listing 4). Die gruppierten Datenpunkte ergeben dann einen Track $\tau = \{x_1, \dots, x_m\}$, der eine historische Schiffstrajektorie über die Zeit repräsentiert. Da die Trajektorie aus den historischen Datensätzen bereits in der Vergangenheit liegt, kann die Navigationsentscheidung eines Schiffes direkt aus den Daten abgeleitet werden. Hierzu wird ein geografischer Filter eingesetzt, der überprüft, ob der Track Datenpunkte in der Elbe (hinter Brunsbüttel) oder dem Nord-Ostsee-Kanal enthält. Nur Tracks, bei denen diese Klassifikation eindeutig vorgenommen werden kann, werden zum Trainingsdatensatz hinzugefügt. Für einen Track τ mit der Entscheidung $\tilde{y} \in \{\text{Elbe, Nord-Ostsee-Kanal}\}$, gilt dann: $\forall x_j \in \tau: f(x_j) = \tilde{y}$. So kann aus den historischen Daten ein Trainingsdatensatz erstellt werden, indem alle Datenpunkte aller Tracks mit dem entsprechenden Label des zugehörigen Tracks annotiert werden. Mit diesem Trainingsdatensatz erfolgt das Training des Random Forests (S07). Schließlich werden die zu klassifizierenden Daten in das trainierte Modell eingespeist. Hierzu wird ein Live AIS-Datenstrom des eMIR-Testfeldes verwendet, der den betrachteten Bereich abdeckt und über RabbitMQ bereitgestellt wird (S02, vgl. Abschnitt 5.2.2). Eine langfristige Verwendung dieses Modells ist nun auch auf einer lokalen Infrastruktur möglich, indem ein solcher kontinuierlicher Datenstrom (ebenfalls über das FEDMS) geliefert wird (S08). Die Ausgabe des Modells wurde ebenfalls in einer Java-Applikation mit der ArcGIS Java API[12] dargestellt.

Damit andere Nutzer des Testfeldes (bspw. Mitglieder eines Forschungsprojektes) eine Basis haben, auf der verschiedene Modelle verglichen werden können, soll der Datensatz veröffentlicht werden (S09). Weil die im Datensatz enthaltene MMSI jedoch eine indirekt personenbezogene Information darstellt (vgl. Abschnitt 2.4), wurde der verwendete Connector so erweitert, dass jede MMSI durch einen Hashwert also ein Pseudonym ersetzt wird. Der so entstandene Datensatz wurde auf der CKAN-Instanz des Testfeldes veröffentlicht (siehe Abschnitt 5.4). Auf diese Weise ist eine De-Anonymisierung/Pseudonymisierung nur möglich, wenn einem Angreifer bereits ein ausführlicherer Datensatz (mit originaler MMSI) vorliegen würde. Bei erweiterten Prädiktionsmodellen, die die MMSI als Parameter verwenden, müsste zusätzlich eine Kommunikation mit dem Live-AIS Connector hergestellt werden, sodass die Pseudonymisierung der beiden Datenquellen abgestimmt werden.

[12] https://developers.arcgis.com/java/, abgerufen am 24.07.2022

6.3.3 Auswertung

Der Aufbau und die Durchführung der beschriebenen Testszenarien hat an einem realen Prädiktionsproblem verdeutlicht, wie das FEDMS dazu genutzt werden kann eine typische Modellentwicklung im Testfeld voranzutreiben. Als Ergebnis wurde ein Random Forest Classifier Modell trainiert und eine Klassifikationsgenauigkeit von 89.9% erreicht. Hierzu wurde der erstellte Trainingsdatensatz in 80% Trainingsdaten und 20% Testdaten für die Performance-Bewertung aufgeteilt. Mit Hilfe des internen Verarbeitungsschrittes zur Datenqualitätsbewertung konnte festgestellt werden, dass das Entfernen von unvollständigen Datenpunkten die Datenqualität in den beiden Dimensionen Vollständigkeit und Konsistenz messbar verbessert hat (jeweils von 99,0% auf 100%). Die Anwendung des trainierten Modells ist in Abbildung 53 dargestellt: Hier werden die Live AIS-Daten aus dem RabbitMQ-Connector für den geografischen Bereich der Elbe und des Nord-Ostsee-Kanals abgefragt, und als einzelne Datenpunkte in das trainierte ML-Modell eingespeist. Die Daten werden in der entwickelten Java-Applikation alle 500ms aktualisiert und mit einer neuen Prädiktion des Modells versehen. Ein Dreieck in der Abbildung stellt jeweils ein Schiff dar. Die aktuell vorhergesagte Entscheidung wird als Farbe des Dreiecks dargestellt (Rot für den Nord-Ostsee-Kanal und Blau für die Elbe). Es ist zu erkennen, dass die Prädiktion nicht vollständig korrekt funktioniert, was hier ggf. durch eine zu kleine Trainingsdatensatzgröße im Bereich kurz hinter der Abzweigung zu erklären ist.

Abbildung 53: Live-Prädiktion von Navigationsentscheidungen für Schiffe im überwachten Bereich des eMIR-Testfeldes in Elbe und Nord-Ostsee-Kanal (Kartendaten: ArcGIS).

Durch die durch das FEDMS bereitgestellten Schnittstellen war es möglich das Ziel der Fallstudie zu erreichen und auf Basis der Testfelddatenressourcen ein ML-Modell zur Prädiktion einer Verkehrsentscheidung zu entwickeln. Dieses Vorhaben basierte auf der grundlegenden Möglichkeit verschiedene dezentral verwaltete Datenquellen an das FEDMS anzubinden und Datenanfragen für diese entgegenzunehmen und zuzuordnen (vgl. Anforderungen A1, A2 und A3). Nur so konnten in den betrachteten Szenarien die Rohdaten abgerufen werden, um den

Trainingsdatensatz zu erstellen. Die Angabe des gewünschten Datenbereiches im verwendeten GraphQL-Query zeigt zudem, dass eine feingranulare Abfrage von bestimmten Datenbereichen möglich ist (vgl. Anforderung A3). Dabei wurde die Funktionsfähigkeit der verwendeten maritimen Connectoren demonstriert, wie sie in Anforderung A15 definiert wurde. Insbesondere wurden Metadaten über die Datenquellen durch das FEDMS genutzt, um die benutzerdefinierten Anfragen den richtigen Datenquellen zuordnen zu können (vgl. Anforderung A4). Weiterhin wurde gezeigt, dass durch die flexible Nutzbarkeit des Kafka-Clusters und der internen Verarbeitungsschritte, Daten-Workflows beliebig auf FEDMS-Infrastruktur und durch Stakeholder verwaltete lokale Infrastruktur verteilt werden können (vgl. Anforderung A5 und A9). Außerdem wurde durch die Fallstudie demonstriert, dass auch Anforderung A10 (Live-Datenverarbeitung) erfüllt werden konnte und, dass in einem Projekt neben historischen Daten auch Live-Daten mittels derselben FEDMS-Infrastruktur verwendet werden können. Zudem war es durch eine Erweiterung des Connectors möglich die MMSI, die eine indirekt personenbezogene Information darstellt, zu pseudonymisieren und so den Datenschutz aus der Datenproviderperspektive zu gewährleisten (vgl. Anforderung A7). Schließlich wurde der gesamte Prozess als Workflow vom FEDMS dokumentiert und konnte in einem standardisierten Format für die Archivierung exportiert werden (vgl. A12, siehe dazu analog auch Abbildung 56 in Fallstudie II für ein Beispiel). Wie in Anforderung A16 gefordert wurde das FEDMS mittels der internen Verarbeitungsschritte dazu genutzt die Datenqualität zu bewerten und nachweislich zu verbessern. Mit der Implementierung der Verarbeitungsschritte zur Datenqualitätsbewertung und Verbesserung mittels python, wurde ebenfalls gezeigt, dass es neben der standardmäßig bereitgestellten Java-Bibliothek auch möglich ist weitere Sprachen für die Implementierung zu nutzen.

6.4 Fallstudie II: Untersuchung eines Assistenzsystems zur Kollisionsrisikobewertung

Als zweite Fallstudie soll ein weiterer klassischer Anwendungsfall eines serviceorientierten Testfelds betrachtet werden. Hierbei kommt das Testfeld bei der Verifikation und Validierung eines maritimen Assistenzsystems zum Einsatz. Im Speziellen wird dafür das Assistenzsystem Maritime Traffic Collision Avoidance System (MTCAS) als SuT verwendet. MTCAS ist ein Assistenzsystem, welches kritische Verkehrssituationen erkennen, Navigationsverhalten vorhersagen und kritische Situationen durch die Verhandlung von Ausweichmanövern auflösen kann (Steidel und Hahn 2019). Für diese Fallstudie wird ein Teilsystem von MTCAS untersucht, welches mittels sogenannten *Escalation States* eine Kollisionsrisikobewertung von maritimen Verkehrssituationen durchführt, und Nutzer so vor kritischen Situationen warnen kann. Escalation States werden pro von MTCAS wahrgenommenen Schiff berechnet und geben die Kritikalität

einer Verkehrssituation in fünf Klassen an: 1 – Clear State, 2 – Recommendation State, 3 – Danger State, 4 – Last Minute Maneuver (LMM) State, 5 – Accident State. Um diese Kritikalitätsbewertung durchzuführen nutzt MTCAS einen Datenverarbeitungsprozess (siehe Abbildung 54), in dem zunächst rohe Sensordaten (z.B. von GPS, AIS und RADAR) im NMEA0183-Format (vgl. Langley 1995) eingelesen werden. Daraufhin erfolgen eine Transformation in ein internes Sensordatenmodell, die Interpretation der Sensordaten mittels eines Verkehrssituationsmodells, eine Verhaltensprädiktion und schließlich die Klassifizierung des Kollisionsrisikos pro Schiff als Escalation State. Für die Fallstudie liegt der Quellcode von MTCAS vor.

Damit im folgenden Abschnitt keine Verwechslungen auftreten, werden Szenarien als Teil der V+V von MTCAS als *Testszenarien* bezeichnet, während (Proto-)Szenarien als Teil des ESAAM-Prozesses weiterhin als *Szenarien* bezeichnet werden.

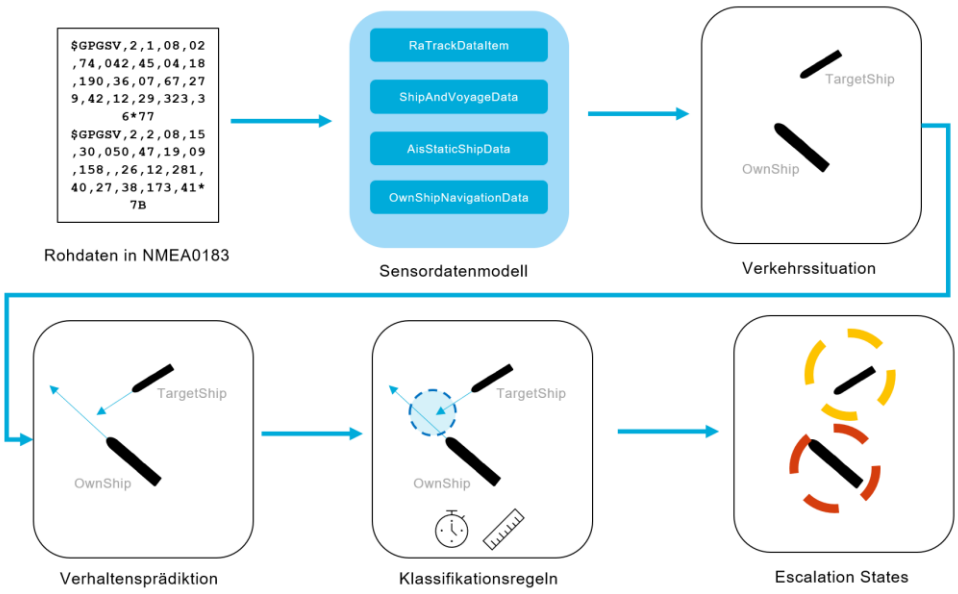

Abbildung 54: Schematische Darstellung des Datenverarbeitungsprozesses von MTCAS zur Kritikalitätsbewertung einer maritimen Verkehrssituation.

6.4.1 Ausgewählte Szenarien

Die Fallstudie orientiert sich an der Durchführung eines Szenario-basierten Testverfahrens (vgl. Abschnitt 2.2.3). Da eine Szenario-basierte V+V eines solchen komplexen Systems mit einer vollständigen Menge von Testszenarien keine neuen Erkenntnisse über die Anwendbarkeit des FEDMS auf die untersuchte Problemstellung liefern würde (es würden lediglich unterschiedliche Daten in den entworfenen Workflows verarbeitet) wird das Testverfahren repräsentativ mit einem Testszenario durchgeführt. Weitere Tests funktionieren dann analog. Zudem wird davon ausgegangen, dass drei unabhängige Stakeholder das Testverfahren kollaborativ durchführen, und

ein Interesse haben die eigenen Daten nur für Teilnehmer zur Verfügung zu stellen, die entsprechende Berechtigungen besitzen (vgl. auch Abschnitt 2.3.4): Die Tests werden mit Hilfe einer maritimen Verkehrssimulation eines Simulationsdatenproviders durchgeführt, die mit realen Live-Daten vom Testfeldbetreiber angereichert wird. Der dritte Stakeholder ist durch den Entwickler des MTCAS gegeben.

(S10) Verteilten Daten-Workflow nutzen. Die Anwesenheit mehrerer Stakeholder im Testverfahren erfordert die Nutzung von verteilten Infrastrukturen, die unter der Kontrolle der einzelnen Parteien stehen. Sofern keine Einigung auf die Verwendung einer zentralen Infrastruktur im Testfeld möglich ist, muss ein verteilter Daten-Workflow verwendet werden, der die im Testverfahren verwendete Datenverarbeitungskette repräsentiert. Der Workflow im beschriebenen Setup wird zwischen Simulationsprovider, Systementwickler des MTCAS und dem Testfeldbetreiber als Provider für die Live-Verkehrsdaten geteilt.

(S11) Zugriffsrechte für mehrere Stakeholder. Teile des Workflows müssen für mehrere der involvierten Stakeholder erreichbar sein. Nur so können Daten zwischen den Parteien ausgetauscht werden. Auf diese Weise kann etwa das Bereitstellen der benötigten Sensordaten von einem Stakeholder durchgeführt werden, während der MTCAS-Entwickler Daten aus dem System selbst als weiteren Teil des Workflows bereitstellt.

(S12) Daten interner SuT-Zustände dokumentieren. Damit eine spätere Auswertung der Testszenarien stattfinden kann, müssen bestimmte Datenartefakte innerhalb von MTCAS extrahiert und dokumentiert werden können, sodass später geprüft werden kann, ob die gegeben Akzeptanzkriterien für einen erfolgreichen Test erfüllt worden sind.

(S13) Testszenarien aufzeichnen. Nicht nur die einzelnen internen Systemzustände von MTCAS müssen für eine vollständige Auswertung eines Tests dokumentiert werden, sondern alle als Teil des Test-Setups durchgeführten Verarbeitungsprozesse, sowie die erhobenen Datenartefakte. Dies schließt hier auch die Aufzeichnung der durch die Simulation bereitgestellten Daten, sowie die Live-Verkehrsdaten aus dem Testfeld und die von MTCAS berechneten Escalation States mit ein. Aufgezeichnete Daten eines definierten Testszenarios müssen zusammenhängend und klar von anderen Daten unterscheidbar sein.

(S14) Daten erneut abspielen (Replay). Für die Reproduzierbarkeit eines Testszenarios sollen die aufgezeichneten Daten so wieder in den Testaufbau eingespeist werden, dass dasselbe Verhalten von MTCAS für weitere Analysen erneut ausgelöst werden kann. Das ist insbesondere sinnvoll, da Live-Daten bei einer einfachen Wiederholung eines Testszenarios nicht in gleicher Art und Weise reproduzierbar sind. Denkbar ist auch, dass in einem solchen Testaufbau die Verwendung der Verkehrssimulationsumgebung besonders kostenintensiv ist und durch das

erneute Einspielen der Daten (ohne erneute Erzeugung durch die Simulation) Kosten eingespart werden können.

(S15) Datenzugriff absichern. Durch die gemeinsame Nutzung eines Workflows stellen etwa der Simulationsprovider oder der Testfeldbetreiber Daten bereit, die möglicherweise nicht für andere beliebige Nutzer des Testfeldes einsehbar sein dürfen. Hier muss der Transportweg entsprechend abgesichert werden und es bedarf der Autorisierung und Authentifizierung involvierter Parteien beim Abruf der Daten.

(S16) Datenanalyse durchführen. Schließlich muss eine Datenanalyse der aufgezeichneten Testszenarien stattfinden. Dabei werden die aufgezeichneten Daten aus dem FEDMS exportiert und in einer benutzerdefinierten Umgebung vom MTCAS-Entwickler analysiert. Dieser Prozess fokussiert sich auf die Aufbereitung der exportierten Daten und der Vorbereitung des Testberichtes. Es handelt es sich um ein indirektes Szenario, da die Auswertung von Testszenariodaten größtenteils anwendungsspezifisch (in Bezug auf den Testaufbau und das SuT) ist.

(S17) Testbericht generieren. Auch die Generierung des finalen Testberichtes ist ein indirektes Szenario. Der Testbericht legt die Grundlage für die Bewertung des Testerfolges. Im letzten Schritt kann dies dann durch die Auswertung des Berichts unter dem Einsatz von Expertenwissen erfolgen.

6.4.2 Testaufbau und -durchführung

Im verwendeten Testaufbau (Abbildung 55) kollaborieren drei verschiedene Stakeholder im Testfeld (S10). Der Simulationsprovider simuliert das gewünschte Testszenario, generiert dabei für alle beteiligten Schiffe dynamische Schiffsinformationen im NMEA0183-Format und stellt diese dem MTCAS als Datenartefakt über ein Topic des Kafka-Clusters bereit. Gleichzeitig kann der Testfeldbetreiber dasselbe Topic nutzen, um Live-Testfelddaten (ebenfalls im NMEA0183-Format) für erweiterte Tests bereitzustellen. So können simulative und reale Daten (etwa um die Tests realistischer zu machen) kombiniert werden. Durch die Verwendung des nachrichtenbasierten Consumer/Producer Pattern in Kafka (vgl. Abschnitt 4.4.1) können NMEA0183-Daten aus mehreren Quellen gleichzeitig auf dasselbe Topic geschrieben werden. Für das MTCAS ist es dann nicht mehr ersichtlich, dass die Daten aus zwei Datenquellen kombiniert wurden. Der Datenstrom aus dem entsprechenden Topic kann, wie ein Datenstrom aus einer realen Datenquelle, behandelt werden (in der Regel als Datenstrom eines schiffsinternen Sensorsystems zum Empfang und der Verarbeitung von AIS-Nachrichten). Lediglich die Anbindung an Kafka muss als Erweiterung des Systems für den Test betrachtet werden. Hierzu wurde ein externer Adapter entwickelt, sodass das SuT selbst nicht modifiziert werden musste.

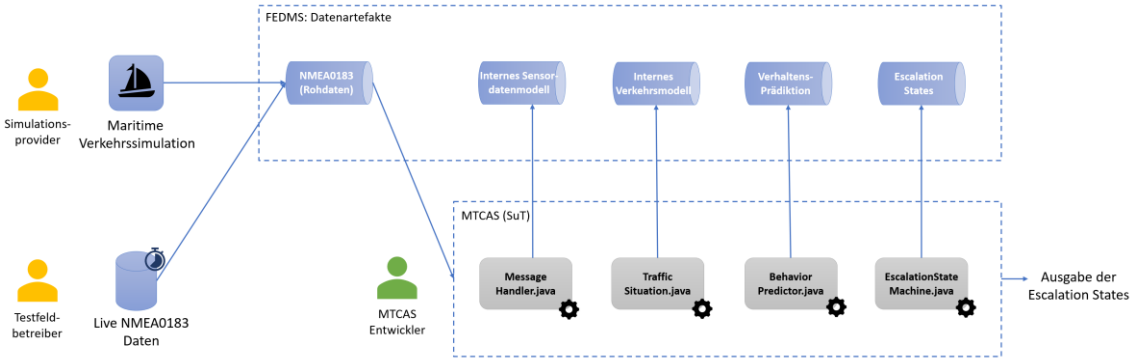

Abbildung 55: Testaufbau für das Szenario-basierte Testverfahren der Kollisionsrisikobewertung von MTCAS (vgl. Möller u. a. 2022).

Für die Simulation des Testszenarios wurde das Software-Werkzeug HAGGIS verwendet, das eine Modellierungs- und Co-Simulationsumgebung implementiert und einen Teil des eMIR Testfeldes darstellt (Rüssmeier, Lamm, und Hahn 2019).

Im Unterschied zu Fallstudie I wird das FEDMS (wie in Abbildung 55 zu erkennen) in einer hybriden Konfiguration (siehe Abschnitt 4.4.2) verwendet: Die Eingabedaten werden zwar über das FEDMS bereitgestellt, die restlichen Datenartefakte werden allerdings parallel zum SuT verwaltet. Dies minimiert die notwendigen Modifikationen des SuT für die Integration mit dem FEDMS. Da MTCAS vollständig in Java entwickelt wurde, kann wieder die Java Client-Bibliothek verwendet werden, um eine Verbindung zum FEDMS herzustellen, sodass interne Datenverarbeitungszustände als Datenartefakt übermittelt werden und über die Zeit überwacht werden können (S12). Die Übermittlung der Daten an das FEDMS sind die einzigen Modifikationen, die am SuT vorgenommen worden sind, sodass möglichst realistische Testbedingungen vorhanden sind. Zudem wurde die Kommunikation mit dem FEDMS an kritischen Stellen auf einen separaten Thread ausgelagert, sodass keine Wechselwirkungen mit dem eigentlichen Systemverhalten entstehen können. Die Ein- und Ausgabedaten von MTCAS wurden kontinuierlich übermittelt, während die internen Datenmodelle alle 500ms aufgezeichnet wurden (siehe Listing 5 als Beispiel für die Aufzeichnung des internen Verkehrsmodells).

Für die Umsetzung des Daten-Replays (S14) wurde die Java Client-Bibliothek verwendet, um die Inhalte des NMEA0183-Datenstroms bei Bedarf in Originalgeschwindigkeit wieder in das MTCAS einspeisen zu können (siehe Listing 6). Außerdem wurde ein interner Verarbeitungsschritt verwendet, der die Daten aus allen Topics im json-Format exportiert und in die CKAN-Instanz (siehe Abschnitt 5.4) überträgt. So können die Analyse und Auswertung der Testdaten (S16, S17 – siehe Abschnitt 6.4.3) vom eigentlichen Test abgekoppelt werden und die Daten auf Dauer persistent und auffindbar gespeichert werden.

```
Timer t = new Timer(); //Timer erstellen, der periodisch das int. Datenmodell ausliest
t.scheduleAtFixedRate(new TimerTask() {
      @Override
      public void run() {
          TrafficSituation tsit = TrafficSituation.INSTANCE; //MTCAS Verkehrsmodell
          //Serialisierung vorbereiten:
          Gson gson = new GsonBuilder().serializeSpecialFloatingPointValues().create();
          JsonElement traffic = gson.toJsonTree(TrafficSituation.INSTANCE);
          HashMap<String, Object> map = gson.fromJson(traffic, HashMap.class);
          //Protobuf-Schema aus den Daten ableiten:
          FEDMSHelper.getInstance().getTrafficSituationTopic().defineSchema(map, "TrafficSituation");
          //Daten an das FEDMS übermitteln:
          FEDMSHelper.getInstance().getTrafficSituationTopic().publishFromCustomSchema(map,
"TrafficSituation", true);
      }
  }, 0, 500);
```

Listing 5: Auszug aus dem angepassten MTCAS Java-Code: Periodisches Sampling des internen Datenmodells und Übermittlung an das FEDMS.

```
// Existierendes Topic mit Daten laden:
TopicSession topicSession = session.loadTopic("621de75226766039cb85175d");
// Replay der Daten von Beginn an, in Originalgeschwindigkeit:
topicSession.consumeReplay("earliest", true, data -> {
    //Daten extrahieren und an MTCAS übergeben:
    String nmeaSentence = (String) data.get("data");
    handler.handle(nmeaSentence);
    return null;
});
```

Listing 6: Auszug aus dem Code des Kafka-Adapters von MTCAS – Replay von existierenden Daten aus dem Cluster.

Für die Zugriffsrechte auf die einzelnen Datenartefakte des in Abbildung 55 dargestellten Workflows wird die in Abschnitt 4.5.2 vorgestellte diskrete Zugriffskontrolle eingesetzt. Tabelle 2 zeigt eine Übersicht über die vergebenen Zugriffsrechte (S11): Testfeldbetreiber und Simulationsprovider können nur auf das NMEA0183-Topic schreiben, aber nicht lesen. So kann sichergestellt werden, dass sie die Daten der jeweils anderen Partei nicht einsehen können. Nur der MTCAS-Entwickler bzw. das SuT selbst hat die entsprechende Berechtigung. Um die intern erzeugten Daten von MTCAS zu schützen, haben Testfeldbetreiber und Simulationsprovider keine Zugriffsrechte auf die MTCAS-Topics und Metadaten: Sie befinden sich im dargestellten Setup in einer Datenproviderrolle.

Die Transportwege sind standardmäßig über TLS abgesichert (siehe Abschnitt 5.6) und nur authentifizierte Nutzer haben Zugriff auf das FEDMS (vgl. Abschnitt 4.5.1) (S15). Für eine

zusätzliche Absicherung können zudem benutzerdefinierte Verschlüsselungsverfahren angewandt werden, bevor die Daten übermittelt werden.

	NMEA0183 Topic	**MTCAS Topics**	**Workflow Metadaten**
Testfeldbetreiber	- / Schreiben	- / -	- / -
Simulationsprovider	- / Schreiben	- / -	- / -
MTCAS (Entwickler)	Lesen / (Schreiben)	Lesen / Schreiben	Lesen / Schreiben

Tabelle 2: Zugriffskontrolle für die im Aufbau von Fallstudie erhobenen Daten.

Als repräsentatives Testszenario wurden Daten aus einem realen Kollisionsunfall auf See genutzt und basierend auf dem Unfallbericht ein Abbild der Situation in der Simulation erstellt. Da hier also bereits klar ist, dass eine gefährliche Situation – bis hin zu einem schweren Unfall – vorhanden ist, ist die Ausgabe von entsprechend hohen Kollisionsrisikowarnungen als erwartetes Verhalten des SuTs zu beschreiben. Konkret wird hier die Kollision zwischen der FU SHAN HAI und der GDYNIA betrachtet, die sich im Jahr 2003 auf der Ostsee, nördlich von Bornholm ereignete (Danish Maritime Authority 2003). Zur Durchführung des Tests wird zunächst MTCAS gestartet und mit Hilfe des Web-UI des FEDMS eine neue Testszenarioinstanz für den Workflow erstellt (vgl. Abschnitt 5.5). Daraufhin wird das Testszenario in der Simulation gestartet. Nachdem das Testszenario in der Simulation vollständig abgespielt ist, wird die Aufzeichnung über das Web-UI beendet (S13).

6.4.3 Auswertung

Der Workflow-basierte Ansatz, der hier eingesetzt wurde, um das Systemverhalten von MTCAS zu analysieren, erlaubt es einzelne Schritte in der Datenverarbeitungskette zu betrachten und bezüglich des Testerfolges auszuwerten, bzw. Rückschlüsse auf Fehlverhalten zu ziehen. Um dies im Einzelnen durchzuführen, wurden die im CKAN bereitgestellten Datensätze heruntergeladen und werden im Folgenden exemplarisch ausgewertet. Zur Dokumentation des Workflows wurden die PROV-DM konformen Metadaten durch die Workflow- & Szenario Managementkomponente des FEDMS bereitgestellt. Abbildung 56 zeigt einen Ausschnitt der persistierten Provenienzinformationen. Um die Analyse zu erleichtern und die Ergebnisse übersichtlicher präsentieren zu können, wird im Folgenden ein Testlauf ausgewertet, bei dem auf das zusätzliche Einspeisen von Live-Daten verzichtet wurde.

```
1  …
2  <rdf:Description rdf:about="https://testbed.emaritime.de/provenance
       /MTCASInternalModel">
3      <rdf:type rdf:resource="https://www.w3.org/ns/prov#Entity"/>
4      <prov:wasGeneratedBy rdf:resource="https://testbed.emaritime.de
           /provenance/transformToMTCASModel_ProcessingStep"/>
5  </rdf:Description>
6  <rdf:Description rdf:about="https://testbed.emaritime.de/provenance
       /mtcas_engineer">
7      <rdf:type rdf:resource="https://www.w3.org/ns/prov#Agent"/>
8  </rdf:Description>
9  <rdf:Description rdf:about="https://testbed.emaritime.de/provenance
       /transformToMTCASTrafficModel_ProcessingStep">
10     <rdf:type rdf:resource="https://www.w3.org/ns/prov#Activity"/>
11     <prov:wasAssociatedWith rdf:resource="https://testbed.emaritime
           .de/provenance/ mtcas_engineer"/>
12     <prov:used rdf:resource="https://testbed.emaritime.de/provenance
           /MTCASInternalModel"/>
13 </rdf:Description>
14 …
```

Abbildung 56: Auszug aus den PROV-DM Provenienzdaten des eingesetzten Workflows (Möller u. a. 2022).

Basierend auf den exportierten Daten kann nun die eigentliche Analyse des Testszenarios durchgeführt werden. Dies passiert mit klassischen Werkzeugen aus dem Data Science Bereich. Im durchgeführten Szenario wurden die json-Exports der einzelnen Topics in python mit den Bibliotheken *pandas*, *numpy* und *matplotlib* ausgewertet. Im ersten Schritt wurden zunächst die Rohdaten aus dem NMEA0183-Topic mit dem Testszenario aus der Simulation abgeglichen. So ließ sich validieren, dass die Simulation die Anforderungen an die NMEA0183-Daten erfüllt, die MTCAS an diese Daten stellt. Abbildung 57 zeigt einen Auszug aus dem Abgleich, bei dem die Korrektheit der Positionsinformationen (Längen- und Breitengrad) abgeglichen wurden: Der linke Teil zeigt das Szenario in der Simulation, der rechte Teil ist eine Darstellung basierend auf den aus dem NMEA0183-Topic exportieren Daten. Durch diesen Abgleich wird in einem ersten Schritt die Validität des Testaufbaus sichergestellt.

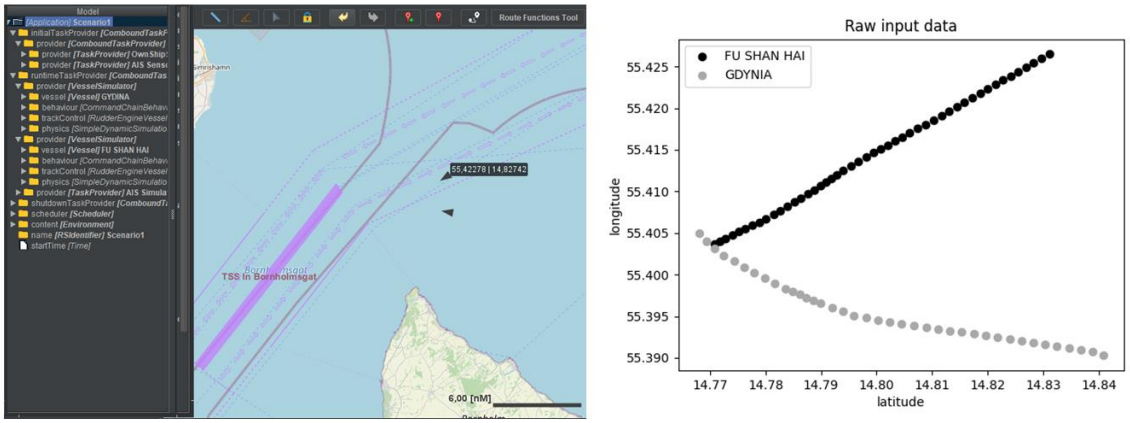

Abbildung 57: Maritime Verkehrssimulation HAGGIS (links, Kartendaten: OpenSeaMap) und Plot der simulierten Positionsdaten der FU SHAN HAI und GDYNIA (rechts, Möller u. a. 2022).

Nachdem auf ähnliche Weise auch festgestellt wurde, dass das interne Sensordatenmodell von MTCAS die empfangenen Daten korrekt abbildet, wurde die Berechnung des Kollisionsrisikos

analysiert. Da dieses im Wesentlichen auf der vorhergesagten minimalen Begegnungsdistanz der beiden Schiffe basiert (Closest Point of Approach distance (CPAd)), wurde dieser Wert aus dem Datenmodell der BehaviorPredictor.java Klasse extrahiert. Die Vorhersage basiert unter anderem auf der aktuellen Geschwindigkeit, der Position und dem Kurs des jeweiligen Schiffs. Hierbei (siehe Abbildung 58, links) ist auffällig, dass der CPAd-Wert ab Minute 7 des Testszenarios stark ansteigt. Aufgrund der immer kleiner werdenden Distanz entsprach dies zunächst nicht dem erwarteten Systemverhalten. Weitere Untersuchungen der Daten aus dem internen Verkehrsmodell zeigten jedoch, dass die Schiffe zu Minute 7 jeweils eine leichte Kursänderung in entgegengesetzte Richtungen ausführen. Dies ist in Abbildung 58 rechts zu erkennen und lässt sich vermutlich auf das verwendete Simulationsmodell zurückführen, in dem die Schiffe ihren Kurs automatisch anpassen, um einer vordefinierten Route zu folgen.

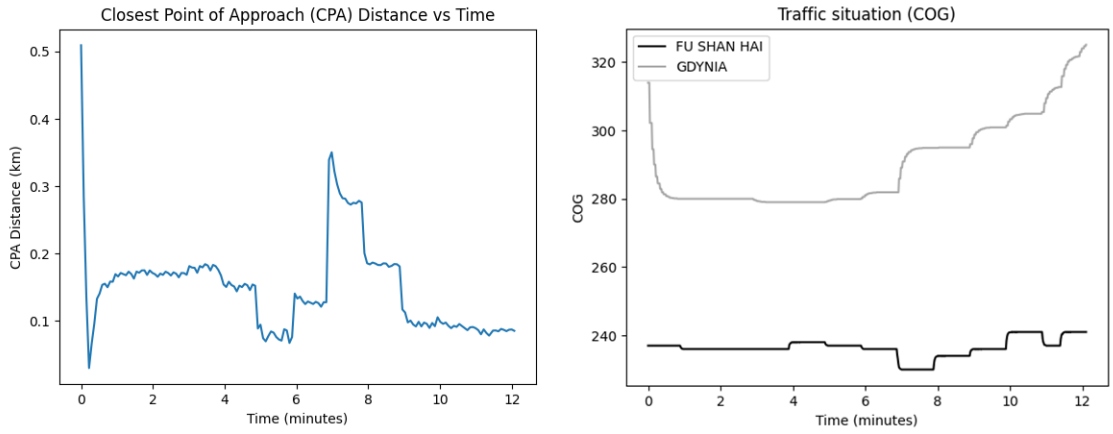

Abbildung 58: Zeitlicher Verlauf der Berechnung des CPAd-Werts (links) und Kurs der Schiffe über Grund (rechts) (Möller u. a. 2022).

MTCAS projiziert bei der Vorhersage der minimalen Begegnungsdistanz die Schiffe in die Zukunft, zum besagten Punkt der minimalen Begegnungsdistanz. Hier lässt sich das obige Phänomen ebenfalls nachvollziehen (dargestellt in Abbildung 59). Schließlich lässt sich also feststellen, dass MTCAS mit Einschränkungen das erwartete Systemverhalten zeigt. Da aber im Test auch die ausgegebenen Escalation States größere Schwankungen bei der Kursänderung von unter 15° pro Schiff zeigen, ist in weiteren Tests oder Validierungsprozessen zu klären, ob dies auch insgesamt ein angemessenes Systemverhalten darstellt. Des Weiteren hat sich gezeigt, dass das gewählte Simulationsmodell möglicherweise nicht angemessen für den beschriebenen Testaufbau ist.

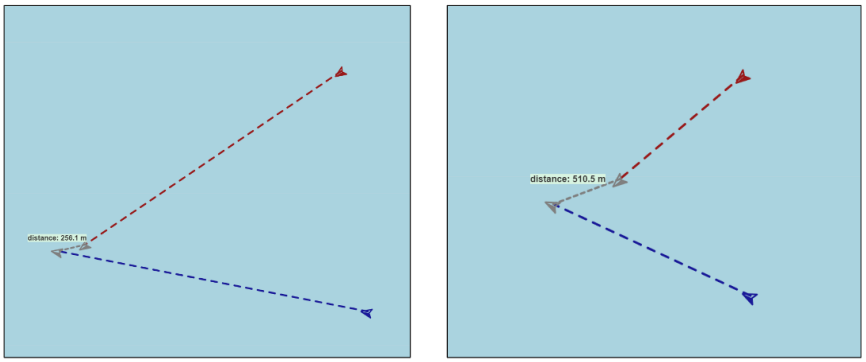

Abbildung 59: Projektion der GDYNIA und FU SHAN HAI zum Zeitpunkt der minimalen Begegnungsdistanz: Vor (links) und nach dem Manöver (rechts) bei Minute 7 (Möller u. a. 2022).

An diesem Prozess wurden die Einsatzmöglichkeiten des in Kapitel 4 vorgestellten Datenmanagementkonzepts mit Hilfe des FEDMS demonstriert. Da der eigentliche V+V Prozess von MTCAS nicht im Fokus dieser Arbeit steht, wurde das Experiment jedoch nicht mit korrigierten Bedingungen erneut ausgeführt.

Wie der Testaufbau gezeigt hat, ist es unter Verwendung des FEDMS problemlos möglich mit verschiedenen Stakeholdern einen verteilten Datenworkflow zum Test eines maritimen Assistenzsystems zu verwenden. Hierbei konnten die Simulation, eine live AIS-Datenquelle aus dem Testfeld und ein SuT auf verschiedenen, voneinander unabhängigen, lokal verwalteten Infrastrukturen gemeinsam genutzt werden, ohne dass ein erheblicher Integrationsaufwand notwendig war (vgl. Anforderung A9). Wie in Anforderung A10 gefordert, wurde demonstriert, dass das FEDMS über den Kafka-Cluster problemlos Daten aus Live-Testdurchläufen verwalten kann und diese in Form von Testszenarioinstanzen zwischenspeichern und exportieren kann (A11). Das in Tabelle 2 definierte, feingranulare Zugriffsschema für die verwalteten Daten, wurde erfolgreich genutzt, um das Rollenmodell der einen Datenprovider (Simulationsprovider und Testfeldbetreiber) und des Datenkonsumenten (MTCAS-Entwickler) und die damit verbundenen Zugriffsrechte abzubilden (vgl. Anforderung A8). Die Sicherheit der Datentransportwege (A14) wurde wie in Fallstudie I durch die Verwendung der Java Client-Bibliothek und der dort implementierten, verschlüsselten Kommunikation mit dem FEDMS sichergestellt. Durch die Bibliothek konnte ebenfalls eine einfache Injektion (vgl. Listing 5) in den MTCAS-Code stattfinden, um mit minimalen Änderungen interne Systemzustände zu überwachen, die bei der Auswertung des durchgeführten Testszenarios hilfreich waren. Nachdem der benötigte Workflow im FEDMS konfiguriert wurde, konnte über die FEDMS Weboberfläche per Klick eine Datenaufzeichnung für das Testszenario gestartet und nach Ende des Tests gestoppt werden. Um die Reproduzierbarkeit eines Tests zu gewährleisten, wurde zudem für das NMEA0183-Topic die Replay-Funktionalität der FEDMS Client-Bibliothek genutzt. Schließlich wurde die Architektur im indirekten Szenario S16 erweitert, um eine Datenanalyse durchzuführen: Die Daten wurden in

einem Standardformat (json) exportiert und konnten direkt in der Analyse in einem python-Skript weiterverarbeitet werden. Da die Analyseergebnisse in diesem Fall nur für einen Stakeholder (MTCAS-Entwickler) relevant sind, bietet es sich hier an die Daten aus den Tests zu exportieren und auf der lokalen Infrastruktur zu verarbeiten (vgl. auch Anforderung A5). Bei mehreren involvierten Teilnehmern könnte eine Erweiterung des Daten-Workflows zur Auswertung der Daten erstellt und mit entsprechenden Zugriffsbeschränkungen versehen werden. In einer solchen Variante wäre es auch möglich Verarbeitungsschritte zu entwickeln, die die oben durchgeführten Auswertungen live durchführen und schon zur Laufzeit des Tests das Systemverhalten analysieren können. Weiter ist noch anzumerken, dass die Datenquellen in diesem Test nicht als Connector eingebunden wurden, sondern die Daten direkt über die Client-Bibliothek eingespeist wurden. Für einen einmaligen Test ergibt es also Sinn Daten ohne Connector direkt auf den Kafka-Cluster einzuspeisen. Dies zeigt erneut die Flexibilität der entworfenen Architektur und ermöglicht eine einfachere Verwendung des FEDMS.

6.5 Fallstudie III: Contract-basierte Zertifizierung im Testfeld

Die letzte Phase des Testfelddatenmanagements beschäftigt sich mit der Demonstration und Zertifizierung. Beide Aktivitäten haben gemeinsam, dass hier bestimmte Aspekte zu testender Systeme beobachtet werden. Bei der Zertifizierung wird zusätzlich geprüft, ob die beobachteten Systeme abschließend ein spezifiziertes Verhalten aufzeigen. In der folgenden Fallstudie soll das FEDMS genutzt werden, um einen Teil eines Zertifizierungsverfahrens zu unterstützen. Hierbei wird die Annahme getroffen, dass erneut mehrere Stakeholder an diesem Verfahren beteiligt sind, und verteilt entwickelte Teilsysteme bereitstellen, die über gemeinsame Schnittstellen kommunizieren und somit ein Gesamtsystem bilden. Dies stellt einen klassischen Anwendungsfall im *Model-based Systems Engineering* Prozess dar (D'Ambrosio und Soremekun 2017). Für die Zertifizierung eines Assistenzsystems sind diverse Anforderungen zu erfüllen, die durch jeweilige Standards, wie in der Automobilindustrie etwa ISO 26262 (ISO 2018), ISO 21448 (ISO 2022) oder ANSI/UL 4600 definiert sind. So müssen beispielsweise nach ANSI/UL 4600 (ANSI 2022), Abschnitt 5 Begründungen für den sicheren Betrieb eines Systems „*eine strukturierte Erklärung in Form von Behauptungen [sein], die durch Argumente und Beweise gestützt werden und die belegen, dass [das System] (…) für die Operational Design Domain annehmbar sicher ist und die den Lebenszyklus [des Systems] (…) abdecken.*" (ANSI 2022, 23). Eine strukturierte Methode, auf der solche Argumente aufgebaut werden können, ist das Contract-basierte Design. Insbesondere für die verteilte Entwicklung eines Systems eignet sich ein solches Vorgehen, bei dem Verantwortlichkeiten aufgeteilt werden können, verschiedene Hersteller an dem Entwurf eines Gesamtsystems voneinander unabhängig teilnehmen können und basierend

auf Annahmen über die anderen Teilsysteme und die Umgebung bestimmte Garantien für die eigenen Teilsysteme geben (Benvenuti u. a. 2008).

Annahmen und Garantien werden als Contracts formalisiert, die als Tupel $C = (A, G)$ dargestellt werden können, wobei A eine Menge von Annahmen und G eine Menge von Garantien ist. Eine Implementierung eines Systems erfüllt einen Contract, wenn unter der Annahme von A, G eingehalten werden kann. Hierbei beziehen sich die Annahmen auf die Umgebung und die Garantien auf die Schnittstellen des Systems. Für die Kombination mehrerer Teilsysteme lassen sich auch Operatoren für die Kombination von mehreren Contracts definieren, sodass Annahmen und Garantien für ein Gesamtsystem auf Basis seiner Komponenten abgeleitet werden können. In der Verifikation wird dann geprüft, ob die Komposition der Teilsysteme die Gesamtspezifikation des Systems erfüllt. (Sangiovanni-Vincentelli, Damm, und Passerone 2012)

Dieser Prozess stellt ein formales Verfahren dar, um auf Basis der Contracts *offline* eine Aussage darüber treffen zu können, ob ein zusammengesetztes System eine Spezifikation erfüllt. Um die Korrektheit der Systemfunktionalität eines Systems sicherstellen zu können ist zudem auch die Beobachtung des Systems zur Laufzeit (*online*) durch einen sogenannten Monitor sinnvoll, bei der festgestellt werden kann, inwiefern die Annahmen und Garantien auf verschiedenen Ebenen in einem operationellen Setting auch eingehalten werden (Ehmen u. a. 2020). Hierfür müssen entsprechende Daten über ein beobachtetes System erhoben und ausgewertet werden. Für eine vollständige Zertifizierung eines Systems kann eine ganze Reihe Monitore benötigt werden, die durch die verschiedenen Entwickler der Teilsysteme bereitgestellt werden müssen. Um in solchen komplexen Aufbauten ein Monitoring kollaborativ durchzuführen, und sicherstellen zu können, dass Daten aus einem bestimmten Teilsystem eines Entwicklers stammen, ist das Tracking eines Daten-Workflows notwendig. Zusätzlich sollte aber auch garantiert werden können, dass die für eine offizielle Auswertung genutzten Daten vom Entwickler eines Systems als solche auch zur Verfügung gestellt wurden (Anforderung A13). Dieses Szenario bildet die Grundlage der dritten Fallstudie.

Im Speziellen sollen Time-Contracts für das bereits in Fallstudie II vorgestellte MTCAS betrachtet werden. Hierbei beziehen sich die Annahmen und Garantien auf das Zeitverhalten der Subsysteme. Da es sich bei MTCAS um ein sicherheitskritisches System handelt, muss sichergestellt werden können, dass die einzelnen Subsysteme auch bestimmte Zeit-Constraints bei der Risikobewertung einer Situation einhalten. Im Folgenden wird davon ausgegangen, dass die Kollisionsrisikobewertungsfunktion von MTCAS durch das Zusammenspiel von drei Subsystemen erreicht wird, die jeweils von verschiedenen Entwicklern bereitgestellt werden. Vor der Entwicklung der Systeme wurden Zeit-Constraints festgelegt, die im Rahmen der Zertifizierung eingehalten werden müssen. Das erste Subsystem verarbeitet hierbei die Sensordaten und stellt das bereits in Abschnitt 6.4 vorgestellte *TrafficSituation*-Modell zur

Verfügung. Dieses wird vom zweiten Subsystem verwendet, um eine CPA-Berechnung durchzuführen. Schließlich nutzt das dritte Subsystem die Resultate des zweiten Subsystems für das Ermitteln eines Escalation States.

6.5.1 Ausgewählte Szenarien

In dieser Fallstudie wird die Einhaltung konkreter Time-Contracts zur Laufzeit des MTCAS untersucht. Für den Testaufbau der Fallstudie wird erneut mit der Simulation gearbeitet und eine Überwachung des Systems mit dem FEDMS durchgeführt.

(S18) Daten aus bestimmten Teilsystemen überwachen. Zur Überwachung der Timing-Contracts in MTCAS müssen bestimmte Inputs- und Outputs der Subsysteme überwacht werden. Dazu müssen nach dem Recorder-Monitor Pattern (vgl. Ehmen u. a. 2020) sogenannte Recorder an bestimmten Stellen in das System injiziert werden. Diese zeichnen die gewünschten Daten auf und übermitteln sie an das FEDMS. Das eigentliche Systemverhalten darf dadurch im Wesentlichen nicht beeinflusst werden.

(S19) Verteilten Daten-Workflow nutzen. Auch bei dieser Fallstudie sind mehrere Stakeholder involviert, die jeweils eigene Teilsysteme bereitstellen. Um hierbei die Abhängigkeiten zwischen den Subsystemen und den Timing-Contracts zu verwalten, soll das FEDMS verwendet werden, um mit Hilfe des Workflow-Modells eine kollaborativ nutzbare Datenstruktur zur Organisation und Zugriffsbeschränkung für die Daten bereitzustellen.

(S20) Daten für Zertifizierung signieren. In einem offiziellen Zertifizierungsverfahren muss sichergestellt werden, dass die zur Analyse verwendeten Daten in einem zusammengesetzten System von mehreren Entwicklern dem richtigen System zugeordnet werden können. Eine Signatur der Daten durch den entsprechenden Entwickler zielt hier auf die Non-Repudiation ab (vgl. Eckert 2018, 12), sodass der Zertifizierer sich darauf verlassen kann die korrekten Daten des Systems erhalten zu haben, und dass nicht im Nachhinein behauptet werden kann, dass fehlerhafte oder manipulierte Daten die Grundlage eines Zertifizierungsverfahrens gebildet haben. So kann auch sichergestellt werden, dass keine Veränderung der Daten stattgefunden hat, nachdem sie auf dem FEDMS zwischengespeichert wurden. Weiterhin ist es auf diese Weise möglich sicher nachzuweisen, welche Parteien Garantien der Contracts ihrer Teilsysteme verletzt haben.

(S21) Signaturen abrufen und überprüfen. Die aufgezeichneten Signaturen sollten zentral über das FEDMS verfügbar sein, sodass diese durch den Zertifizierer abgerufen werden können. In einem zweiten Schritt sollte es durch das Abrufen der (Schlüssel-)Zertifikate der einzelnen Parteien möglich sein diese Signaturen auch zu verifizieren.

(S22) Testerfolg bewerten / Überprüfung der Anforderungen. Basierend auf den signierten Daten kann der Zertifizierer eine Auswertung durchführen. Hierfür wird nach dem Recorder-Monitor Pattern (vgl. Ehmen u. a. 2020) ein Monitor als Erweiterung implementiert, der die Timing-Contracts aller Systeme kennt und auf dem signierten Datenstrom überprüfen kann, ob diese zu jedem Zeitpunkt eingehalten wurden. So lässt sich bereits zur Laufzeit verifizieren, dass das Gesamtsystem die Spezifikationen der Timing-Contracts erfüllt.

6.5.2 Testaufbau und -durchführung

Um die beschriebenen Szenarien durchzuführen ist zunächst eine formale Spezifikation der Time-Contracts für das zu testende System MTCAS notwendig. Hierzu wird das MULTIC-Framework verwendet, welches eine Werkzeugkette zur Definition von Time-Contracts und zum virtuellen Integrationstest zusammengesetzter Systeme bietet, sodass die Komposition unter Berücksichtigung eines Gesamtsystem Timing-Contracts formal verifiziert werden kann (Böde u. a. 2019). Die Contracts wurden hierbei alle in MTSL (MULTIC Timing Specification Language) verfasst (vgl. Böde u. a. 2019). Abbildung 60 zeigt eine Übersicht über die für MTCAS definierten Contracts: Zunächst wird für das Gesamtsystem definiert, dass dieses unter der Annahme eines kontinuierlichen Inputs, der alle 100 bis 5000 ms auftritt, einen Output produzieren kann, der alle 100 ms bis 1500 ms mit einer Abweichung von 50 ms auftritt. Das Subsystem zur Verarbeitung der Sensorinformationen garantiert bei regelmäßigem Input eine Reaktionszeit von maximal 100 ms. Die CPA-Berechnung im zweiten Subsystem wird unabhängig von den Inputs periodisch in einem Intervall von 100 ms und 1500 ms durchgeführt. Damit aber eine sinnvolle und aktuelle Berechnung möglich ist, wird ebenfalls ein regelmäßiger Input angenommen. Die Berechnung der Escalation-States benötigt keine kontinuierliche Datengrundlage und garantiert daher ohne Voraussetzung eine Reaktion innerhalb von 50 ms. Dies ist dadurch zu begründen, dass es sich hier um ein einfaches regelbasiertes System handelt, welches allein auf der Berechnung des aktuellen CPA einen Escalation State bestimmt. Für dieses Experiment wurde die MTCAS-Verarbeitungskette von AIS-Nachrichten mit Positionsdaten eines fremden Schiffes betrachtet.

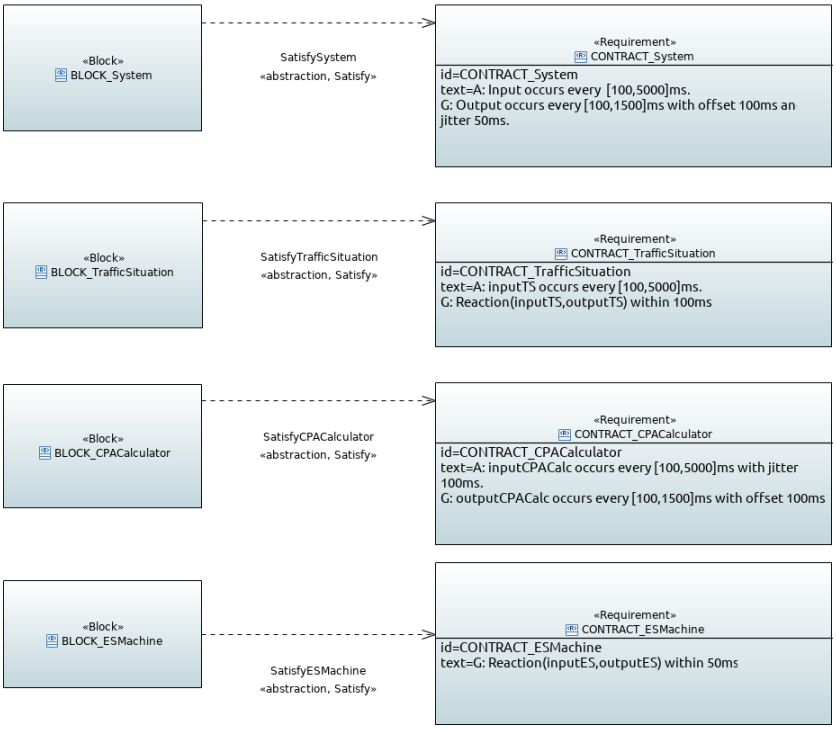

Abbildung 60: Time-Contracts für MTCAS-Gesamtsystem und Teilsysteme.

Nachdem ein virtueller Integrationstest (offline) erfolgreich abgeschlossen ist, kann der online Test des Systems stattfinden, der im Fokus dieser Fallstudie liegt: In diesem Fall garantiert ein erfolgreicher virtueller Integrationstest, dass das Erfüllen der Contracts der einzelnen Subsysteme dazu führt, dass auch der Contract für das Gesamtsystem erfüllt werden kann. Die Erfüllung der Contracts der Teilsysteme liegt jedoch in der Verantwortung der Systementwickler und ist von der spezifischen Implementierung abhängig. Da das Zeitverhalten solcher Implementierungen aufgrund ihrer Komplexität oder anderen Gründen oft nicht formal analysiert werden kann, ist eine online-Verifikation notwendig, bei der das System zur Laufzeit auf die Einhaltung der Contracts geprüft wird. Abbildung 61 zeigt eine Übersicht über den Aufbau einer solchen online-Verifikation mit Hilfe des FEDMS: Der grundsätzliche Aufbau ähnelt dem aus Fallstudie II, in der das MTCAS mit Simulationsdaten betrieben wird. Das System ist nun aber aus den drei vorgestellten Subsystemen zusammengesetzt, die jeweils von den verschiedenen Entwicklern betrieben werden. Die Timing-Contracts beziehen sich lediglich auf das Zeitverhalten der Systeme, die Korrektheit der übermittelten Daten müsste also separat verifiziert werden. Um die Einhaltung der Timing-Contracts zu überwachen, wird das Recorder-Monitor Pattern (siehe S18) eingesetzt: Jedes Teilsystem wird um eine Software-Komponente (Recorder) erweitert, die aus den systeminternen Datenverarbeitungsprozessen Events generiert, diese mit einem Zeitstempel versieht, und mit Hilfe der Client-Bibliothek an das FEDMS übermittelt. So wird etwa ein Event generiert, wenn die Sensordaten aus der Simulation eingegangen sind, und ein weiteres Event, wenn das Teilsystem die TrafficSituation basierend auf den Eingabedaten aktualisiert hat. Die

Events werden jeweils in einem eigenen Datenartefakt gespeichert. Der Zertifizierer erhält Lesezugriff auf alle Datenartefakte und implementiert einen Monitor, der zur Laufzeit (oder auch nach der Laufzeit des Tests) überprüft, ob die Annahmen und Garantien der Timing-Contracts eingehalten werden. Je nach Anwendungsfall kann der Zugriff so eingeschränkt werden, dass lediglich der Zertifizierer Zugriff auf alle Datenartefakte hat.

Als zusätzliche Sicherheitsmaßnahme (vgl. S20) werden die Events mit den Identitäten ihrer Entwickler signiert. Da hierzu eine PKI benötigt wird, wird im Rahmen dieser Fallstudie die öffentliche Test-Instanz der Maritime Connectivity Platform verwendet, um die Public-Private-Schlüsselpaare für die Nutzer zu verwalten. Die öffentlichen Schlüssel der Teilnehmer können dann durch den Zertifizierer verwendet werden, um ihre Signaturen zu verifizieren (S21). Aufgrund der somit entstehenden Non-Repudiationseigenschaft in Kombination mit der temporären Speicherung der Daten durch die Datenverarbeitungsschicht können Aufzeichnungen von Testläufen ohne Weiteres auch nach der Testlaufzeit erneut abgespielt werden und als Eingabe für einen geänderten Monitor (beispielsweise bei nachträglicher Änderung der Timing-Contracts) verwendet werden, ohne einen Testlauf zu wiederholen.

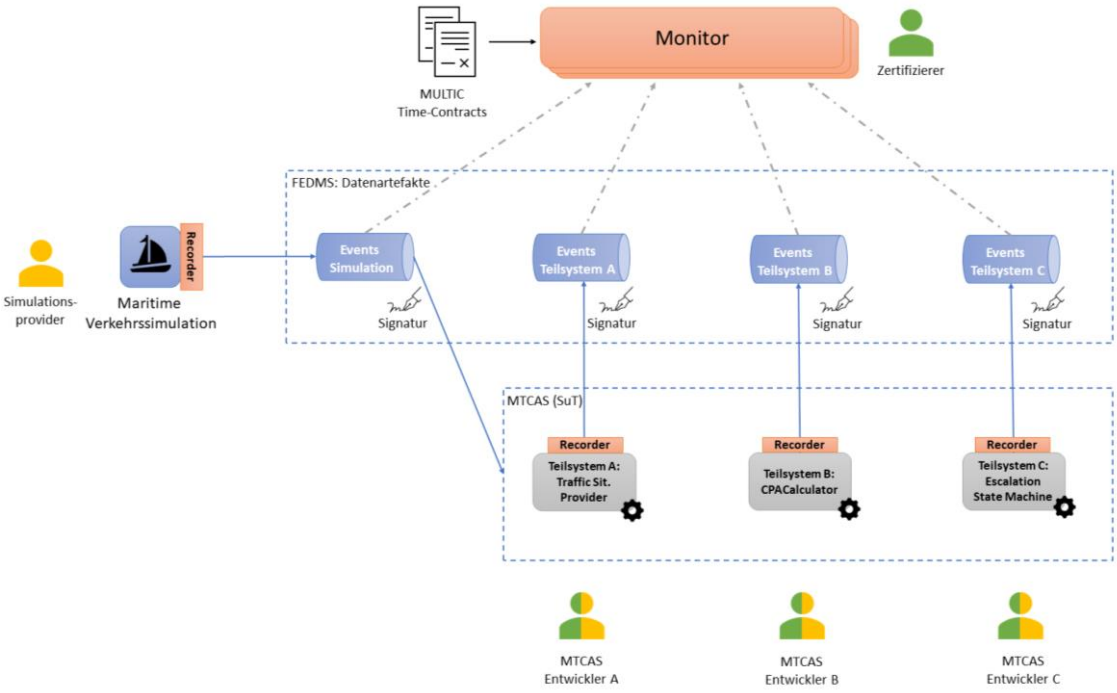

Abbildung 61: Testaufbau für die Zertifizierung des Zeitverhaltens von MTCAS mittels Time-Contracts.

Listing 7 zeigt einen Ausschnitt aus dem MTCAS-Code, bei dem der in Abschnitt 5.6 vorgestellte *Verifying Consumer* eingesetzt wird: Bei der Initialisierung des Consumers muss neben einem Public-Private-Schlüsselpaar auch eine Methode zur Verifizierung der Daten angegeben werden. In der Implementierung von MTCAS werden sämtliche übermittelte Events mit Zeitstempel lokal

zwischengespeichert, um so in der Verifizierung zu überprüfen, dass die Daten in einem FEDMS-Datenartefakt auch tatsächlich durch das entsprechende Teilsystem erzeugt wurden. Nur diese Events werden mit dem privaten Schlüssel signiert. Um übermäßigen Netzwerkverkehr zu vermeiden ist es frei konfigurierbar wie viele Nachrichten für eine Signatur zusammengefasst werden. Im vorliegenden Beispiel wurden jeweils 10 Nachrichten zusammengefasst und mit einer Signatur versehen. Dies kann je nach Anwendungsfall (geringe verfügbare Bandbreite, Zeitconstraints bei der Signaturverifikation, etc.) angepasst werden.

```
//Erstellen eines verifizierenden Consumers mit einem MCP-Schlüsselpaar
VerifyingConsumer<DynamicMessage> consumer = new VerifyingConsumer<>(this::verifyMessage, keycloakClient,
consumerProps, session.getTopic().getId(), "keypair/keystorepfad.ks", "keystore_password");

//Starten der Verifizierung für einen bestimmten Teilbereich eines Topics
new Thread(() -> consumer.startVerification(offset - 10, offset)).start();
...
//Methode zum verifizieren gesendeter Nachrichten:
private boolean verifyMessage(ConsumerRecord<String, DynamicMessage> message) {
    //Einlesen der einkommenden Nachricht:
    Map<String, ?> dataMap = DynamicProtobufBuilder.buildDataFromDynamicProtobufMessage(message.value());
    String typeName = (String) dataMap.get("__typename");
    Object data = dataMap.get("data");
    //Prüfen, ob diese Nachricht gesendet wurde:
    synchronized (this.records) {
        if (this.records.stream().anyMatch(r -> r.getTypeName().equals(typeName) &&
r.getTheData().equals(data)))
            return true;
        else
            return false;
    }
}
```

Listing 7: Auszug aus dem angepassten MTCAS Java-Code: Verifizierender Consumer zum Signieren der übermittelten Daten.

Über die REST-Schnittstelle können dann auf der Seite des Zertifizierers die hinterlegten Public-Keys der Teilnehmer vom FEDMS abgerufen werden und die Signaturdaten mit den entsprechenden Daten aus der Datenverarbeitungsschicht verifiziert werden.

Zur Durchführung der Zertifizierung des Zeitverhaltens vom MTCAS wurde erneut das in Abschnitt 6.4 historische Testszenario eingesetzt.

6.5.3 Auswertung

Der dargestellte Testaufbau und die Testdurchführung demonstrieren in einem realen Szenario, wie ein zusammengesetztes maritimes Assistenzsystem, welches durch mehrere unabhängige

Entwickler bereitgestellt wurde, sicher zertifiziert werden kann. Hierbei wurde insbesondere gezeigt, wie Timing-Contracts mit der FEDMS-Architektur in einem online-Test verifiziert werden können. Für die Auswertung wurden im Monitor Log-Ausgaben implementiert, die über den Status der Einhaltung der Annahmen und Garantien der Timing-Contracts informieren. In einem Zertifizierungsverfahren könnten diese dann durch den Zertifizierer ausgewertet werden, um den Erfolg der Tests für die Zertifizierung zu bewerten. Die Testläufe wurden auf einer Windows 10 Maschine mit einer AMD Ryzen 7 5800X CPU (3,8 GHz Basis-Takt) und 48GB RAM ausgeführt.

Zunächst einmal fiel bei der Auswertung auf, dass die Garantie der Escalation State Machine teilweise verletzt wurde (siehe Abbildung 62): So dauert das Reagieren teilweise über 200 ms (statt maximal 50 ms).

```
2022-11-28 09:45:09.158  WARN 30204 --- d.u.e.c.monitors.EscalationStateMonitor  : Detected
ReactionGuarantee violation (at event #117). Reaction should occur in interval [0ms,50ms] but
reaction time was 262ms

2022-11-28 09:45:09.158  WARN 30204 --- d.u.e.c.monitors.EscalationStateMonitor  : Detected
ReactionGuarantee violation (at event #119). Reaction should occur in interval [0ms,50ms] but
reaction time was 52ms

2022-11-28 09:45:10.073  WARN 30204 --- d.u.e.c.m.TrafficSituationMonitor        : Detected
ReactionGuarantee violation (at event #9). Reaction should occur in interval [0ms,100ms] but
reaction time was 641ms
```

Abbildung 62: Auszug der Log-Nachrichten während des Betriebs von MTCAS: Verletzung mehrerer Garantien in der Traffic Situation und der Escalation-State Klassifikation.

Neben den Verletzungen der Garantien der Escalation-State Klassifikation wurden auch vereinzelte Verletzungen beim Update der Traffic Situation beobachtet. Hier wurde die Reaktionszeit von 100ms mit über 600ms deutlich überschritten. Die Ursache für diese Anomalie konnte durch eine Code-Analyse auf die Implementierung des verwendeten, internen Event-Bus zurückgeführt werden. Weitere Verletzungen von Garantien oder Verletzungen der Annahmen der Timing-Contracts wurden nicht beobachtet.

Neben der reinen Bewertung der Timing-Daten verifiziert der Monitor wie oben beschrieben auch die Signaturen der übermittelten Daten. Für die Fallstudie wurden valide Schlüsselpaare der MCP genutzt, die durch den Monitor entsprechend verifiziert werden konnten (siehe Abbildung 63, oben). Zum Testen der korrekten Funktionsweise des Monitors wurde außerdem eine Signatur modifiziert. In diesem Fall schlug die Verifikation – wie erwartet – fehl und die Daten aus dem entsprechenden Teilsystem (identifiziert durch das Topic) wurden durch den Zertifizierer nicht anerkannt (siehe Abbildung 63, unten).

```
2022-10-27 08:49:27.640  INFO 3544 --- d.u.e.c.tdms.VerificationService  : Accessed all data
for topic. Starting Verification of Signatures.

2022-10-27 08:49:27.640   INFO 3544 --- d.u.e.c.tdms.VerificationService   : Verification
Successful!

_____________________________________________________________________________________

2022-10-27 08:50:01.320  INFO 3544 --- d.u.e.c.tdms.VerificationService  : Accessed all data
for topic. Starting Verification of Signatures.

2022-10-27 08:50:01.320   WARN 3544 --- d.u.e.c.tdms.VerificationService   : Found illegal
signature in topic 6357d2e946e83b0a0b083e13 at offsets 110 to 120

2022-10-27 08:50:01.320   ERROR 3544 --- d.u.e.c.tdms.VerificationService    : Verification
Failed!
```

Abbildung 63: Auszug der Log-Nachrichten während der Signaturverifikation: Erfolgreiche (oben) und fehlgeschlagene (unten) Verifikation.

Abschließend lässt sich also sagen, dass das MTCAS die online-Verifikation der definierten Timing-Contracts aufgrund der Verletzungen der Reaktionszeitgarantien von 50ms in der Escalation-State Klassifikation und 100ms in der Traffic Situation Auswertung nicht bestanden hat.

Insgesamt konnte mit der Architektur des FEDMS ein Zertifizierungsverfahren unterstützt werden, welches Timing-Daten bzgl. der Ausführung bestimmter Verarbeitungsschritte innerhalb des SuT benötigt. Hierzu wurde das vorgestellte Recorder-Monitor Pattern implementiert, wobei sowohl Monitor als auch die Recorder Implementierungen die Client-Bibliothek des FEDMS nutzten, um Daten im Testfeld auszutauschen (vgl. Szenario S18). Auch in diesem Szenario wurde ein kollaborativer Workflow als Grundlage genutzt, um Daten aus den Systemen verschiedener Entwickler zusammenzuführen (S19). Dabei wurden die Zugriffsrechte so eingeschränkt, dass nur der Zertifizierer uneingeschränkte Leserechte auf die erhobenen Daten hatte, und die Entwickler keine Daten für die jeweils anderen Entwickler offenlegen mussten. Der auf den in MTSL spezifizierten Contracts basierende Monitor konnte diese Daten auslesen und die Einhaltung der entsprechenden Garantien und Annahmen zur Laufzeit analysieren und bot somit eine Möglichkeit zur Bewertung eines Testerfolgs (S22). Um hierbei die Authentizität der Daten sicherzustellen und auch nach Abschluss des Tests eine legitimierte Datengrundlage zu besitzen, wurden die übermittelten Timing-Daten jeweils durch die Entwickler signiert (S20) und anschließend vom Zertifizierer abgerufen und verifiziert (S21). Die Verwaltung der Signaturdaten wurde ebenfalls durch das FEDMS übernommen. Diese ließen sich als Metadatum für bestimmte Teilmengen eines Datenartefakts speichern. Neben den bereits in Fallstudie I und II überprüften Anforderungen A1, A3, A5, A8, A9, A10 und A14, die hier in einem weiteren Use-Case verifiziert und validiert werden konnten, sind insbesondere noch zwei weitere Anforderungen

hervorzuheben: Die souveräne Datenintegration (A6) ist im vorliegenden Zertifizierungsprozess durch das Verwenden der quelloffenen Client-Bibliothek gegeben, sodass die einzelnen Entwickler weiterhin die volle Kontrolle über ihren Quellcode haben. Die Timing-Daten wurden über die Datenverarbeitungsschicht (bereitgestellt durch einen vertrauenswürdigen Datentreuhänder) ausgetauscht und konnten, neben ihrem Urheber, nur durch den Zertifizierer gelesen werden. Des Weiteren ist mit der beschriebenen Prozedur zur Erhebung und Validierung der Signaturen auch Anforderung A13 erfüllt. Durch die Verwendung eines Schlüsselpaares, das durch einen etablierten Identitätsprovider an die Identität eines Entwicklers gebunden ist, lässt sich kryptografisch sicher der Ursprung der Daten feststellen.

6.6 Latenz- und Durchsatzmessungen

Für die praktische Verwendung der entworfenen Architektur im Testfeld sind Latenzen und Übertragungsgeschwindigkeiten ausschlaggebende Kriterien. Da es sich bei der Umsetzung des vorgestellten Konzeptes um eine prototypische Umsetzung handelt, ist die Messung von Latenzen und Datenübertragungsraten jedoch nicht zwingend auf zukünftige, alternative Implementierungen zu verallgemeinern. Unter anderem beschränkt sich die vorliegende Implementierung auf die Verwendung einer einzelnen Partition pro Kafka-Topic (siehe Abschnitt 5.4), und schöpft somit nicht die Performanzvorteile von Kafka gegenüber anderen Nachrichten-Brokern aus. Daher soll im Folgenden ein allgemeiner Überblick über die ungefähr zu erwartenden Kennzahlen gegeben werden.

Wie in Abschnitt 4 beschrieben, verfügt die Architektur über zwei wesentliche Schnittstellen, die von Testfeldnutzern eingesetzt werden. Die REST-Schnittstellen werden jedoch ausschließlich zur Konfiguration des FEDMS und zur Verwaltung von Metadaten eingesetzt, nicht jedoch für die Übermittlung der eigentlichen Daten. In den untersuchten Use-Cases (vgl. Abschnitt 2.6) stellt die Konfiguration des FEDMS keine Performanz-kritische Komponente dar, da sie überwiegend vor oder nach der Ausführung von Datenverarbeitungsketten stattfindet. Des Weiteren ist aufgrund der Möglichkeit der unabhängigen und dezentralen Instanziierung mehrerer FEDMS-Komponenten (siehe Abschnitt 4.7) nicht zu erwarten, dass in einem Testfeld eine überproportional große Anzahl an Anfragen an die REST-Schnittstellen einer Instanz übermittelt wird. Im Gegensatz dazu wird der Kafka-Cluster zur Übermittlung der eigentlichen Daten eingesetzt und stellt damit eine Performanz-kritische Komponente dar, die beispielsweise dafür sorgen muss, dass Sensordaten mit entsprechend geringer Latenz und ausreichend hohem Durchsatz an ein SuT übermittelt werden können. Hier ist es auch denkbar, dass größere Datenmengen in möglichst kurzer Zeit übermittelt werden müssen, etwa bei der Verarbeitung großer Datensätze, die für das Training von ML-Modellen verwendet werden. Aus diesem Grund

beschränken sich die nachfolgenden Betrachtungen auf die Übermittlung der eigentlichen Daten durch das FEDMS, und nicht auf das Management der zugehörigen Metadatenstrukturen.

Für die Experimente wird die *Latenz* als die Zeit (in Millisekunden) angenommen, die eine Nachricht benötigt, um von einem Producer zu einem Consumer eines Kafka-Topics benötigt (vgl. Abbildung 31). Der *Durchsatz* ist die maximale Datenmenge, die pro Zeiteinheit über das System übermittelt werden kann und wird in MB/s gemessen. Beide Größen werden in einem lokalen Testaufbau unter der Verwendung einer einzelnen Maschine verwendet. Dies ist dadurch zu begründen, dass so möglichst maximale Werte ermittelt werden können, die die Grenzen des Systems abbilden. Die Erweiterbarkeit der Architektur durch dezentrale Komponenten auf unterschiedlichen Infrastrukturen kann dazu führen, dass etwa durch die Beschränkung von Netzwerkbandbreite schlechtere Werte erreicht werden. Schließlich wird noch die Effizienz der dynamischen Serialisierung von Daten mit Protobuf (vgl. Abschnitt 4.4.5) untersucht, und mit anderen Standardformaten verglichen. Die Tests wurden mit einem microk8s Single-Node Cluster durchgeführt, der auf einer WSL2-VM (Windows Subsystem für Linux 2) mit 16 GB RAM und 8 CPU-Kernen aufgesetzt wurde. Beim verwendeten Speichermedium handelt es sich um eine SSD mit einer maximalen Lesegeschwindigkeit von ~560 MB/s und einer maximalen Schreibgeschwindigkeit von ~530 MB/s. Alle Performanztests wurden mit den Standardbibliotheken von Kafka in Java implementiert.

Für den Durchsatztest wurden auf derselben Maschine Producer und Consumer instanziiert, die jeweils in einem eigenen Thread gestartet wurden. Topics aus vorherigen Tests wurden nicht weiterverwendet, sondern vor jedem Test neu erstellt. Die übermittelten Nachrichten bestanden aus zufällig generierten Strings und hatten jeweils eine Größe von 4kb. Insgesamt wurden zwei Testserien durchgeführt. Die erste Serie stellt einen kürzeren Burst von 30.000 Nachrichten pro Producer dar, während bei der zweiten Serie 150.000 Nachrichten eine längere Auslastung des Systems simulieren sollen. Aufgrund der Verwendung von nur einer Partition pro Topic, wurden pro Testserie Daten gleichzeitig an jeweils 1, 2, 4, 5 und 10 Topics übermittelt. Die Datenpunkte in Abbildung 64 stellen den Durchschnitt der Datenübertragungsrate während der Übermittlung aller Nachrichten dar. Da während der Tests keine weiteren Datenübertragungen stattfanden, stellt dies den Gesamtdurchsatz des Systems dar.

Die Ergebnisse in Abbildung 64 zeigen, dass mit einer zunehmenden Parallelisierung (Erhöhen der parallel verwendeten Topics) die Datenrate in Serie 1 etwa linear ansteigt. Ähnliches ist zunächst auch in Serie 2 zu beobachten, wobei jedoch eine etwas geringere Steigerung der Datenrate erkennbar ist. Es ist also zu erkennen, dass das System bei kürzeren Bursts einer großen Menge von Nachrichten höhere Durchsätze als bei einem kontinuierlichen Datenstrom erreicht. Bei der Verwendung von 10 Topics bricht der Durchsatz in Serie 2 jedoch ein. Hier ist zu vermuten, dass die dem Container im Kubernetes-Cluster zur Verfügung stehenden Ressourcen

(RAM oder CPU-Kerne) ausgeschöpft werden, und ein Aufrechterhalten des erwarteten Durchsatzes nicht mehr möglich ist. Dies zeigt die Grenzen des Systems im gewählten Testaufbau. Abgesehen davon zeigen die Testserien, dass selbst die prototypische Konfiguration bereits recht hohe Datenraten von knapp 180 MB/s unterstützt. Es ist zu erwarten, dass die Datenrate pro Topic (siehe jeweils erster Datenpunkt der Serien) mit der Verwendung mehrerer Partitionen noch einmal deutlich gesteigert werden kann (Wu, Shang, und Wolter 2019). Geht man von den ~100 MB/s des Prototyps aus, so könnte ein typischer Data Mining-AIS-Datensatz der Größe von 500 GB (vgl. z.B. AbuAlhaol u. a. 2018) in ca. 83 Minuten übertragen werden (vgl. auch Anforderung A18).

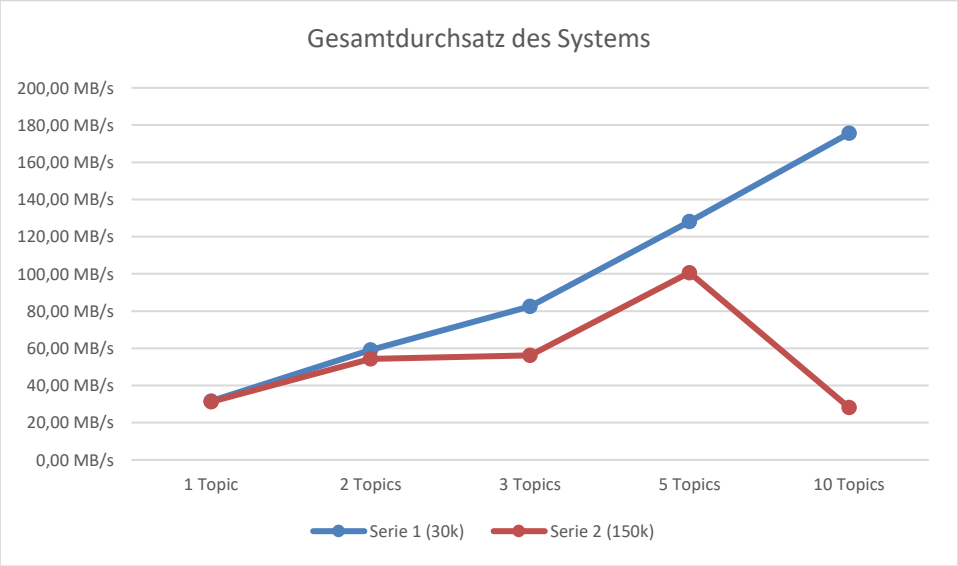

Abbildung 64: Gesamtdurchsatz des Systems – Testserien mit 30.000 und 150.000 Nachrichten.

Für die Latenzmessung wurde ein Sample aus 100.000 Nachrichten mit einer Größe von jeweils 6 Byte betrachtet. Um Randeffekte bei der Initialisierung und dem Beenden der Clients auszuschließen, wurden insgesamt 300.000 Nachrichten gesendet und nur die 100.000 Nachrichten in der Mitte betrachtet. Gemessen wurde die Zeit vom Start der Übermittlung einer Nachricht, bis zum vollständigen Empfang der Nachricht. Hierbei wurden jeweils ein Consumer und Producer verwendet. Der Consumer wurde vor dem Producer instanziiert, sodass keine Verzögerungen durch Initialisierungsprozeduren entstehen konnten. Abbildung 65 zeigt einen Plot der Nachrichtenlatenzen (mit logarithmischer Skalierung der y-Achse): Die mittlere Latenz betrug 4.31 ms, der Median 3.62 ms, die Standardabweichung 2.27 ms. Die maximal gemessene Latenz betrug 18.03 ms. Dies entspricht etwa den zu erwartenden Werten aus der Literatur (Le Noac'H, Costan, und Bougé 2017). Anzumerken ist hier, dass es sich um theoretische Werte in einem optimalen Setup handelt. Bei praktischen Testaufbauten muss auf diese Werte etwa noch die Zeit addiert werden, die eine Nachricht benötigt, um von der entsprechenden Infrastruktur eines Stakeholders zum FEDMS zu gelangen. Im Verhältnis zur kritischen Latenz bei

ferngesteuerten Schiffen, die laut Yim und Park (2021) auf minimal 0.2 Minuten geschätzt wird, ist die hier gemessene Latenz ein zu vernachlässigender Bruchteil (vgl. Anforderung A18), selbst wenn der Kafka-Cluster als direkte Middleware für ein solches System genutzt werden sollte, was im Allgemeinen nicht notwendig ist (vgl. Abbildung 33). Je nach Anwendungsfall könnte das System sogar den Anforderungen im Automobilbereich (100ms Reaktionszeit, vgl. Anforderung A18) gerecht werden.

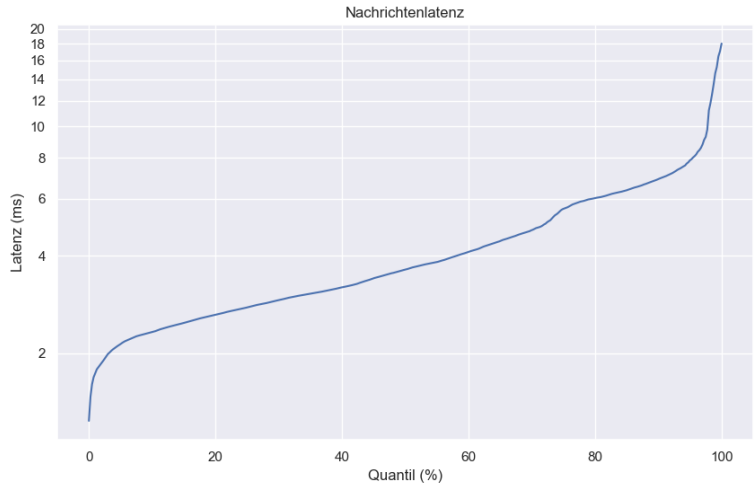

Abbildung 65: Quantil-Plot der Latenzen des Kafka-Clusters.

Für die Latenz- und Durchsatzmessungen wurde die reine Übermittlung von Daten, unabhängig von der Art des Payloads betrachtet. In der Praxis müssen jedoch oft komplexere Datenstrukturen für die Übertragung serialisiert werden. Wie in den Abschnitten 4 und 5 vorgestellt, nutzt das FEDMS als Standardlösung die Serialisierung mittels Protobuf. Da hier die Speichereffizienz als ausschlaggebendes Kriterium für die Wahl der Serialisierungsmethode angeführt wurde, wurde in einem weiteren Test evaluiert, wie effizient die Verwendung von Protobuf mittels dynamisch generierter Schemas (vgl. auch Abschnitt 5.2) ist. Hierzu wurde ein Testaufbau gewählt, bei dem eine verschachtelte Datenstruktur mit zufälligen Daten instanziiert, und mittels verschiedener Serialisierungsmethoden via FEDMS übertragen wurde. Die verwendete Datenstruktur enthielt zwei String-Attribute, ein Integer-Attribut, ein Boolean-Attribut, ein Long-Attribut, eine Liste von Strings und eine Sub-Struktur mit 3 String-Attributen und zwei Integer-Attributen und könnte etwa eine typische Kontrollnachricht für ein Schiffssteuerungssystem darstellen. Für die Vergleichbarkeit wurden die Liste und die Strings je mit fest definierten Größen instanziiert. Als Vergleichsmethoden für die Serialisierung wurden JSON und XML gewählt. Als zusätzliche Maßnahme wurde eine GZIP-Kompression auf die serialisierten Daten in JSON und XML angewandt. Insgesamt wurden pro Serialisierungsmethode je 10.000 Instanzen übertragen. Neben der Gesamtgröße der übermittelten, serialisierten Daten wurde auch der Nachrichtendurchsatz pro Verfahren gemessen. Hierbei wurde von einem Szenario ausgegangen, in welchem die Daten

nicht vollständig vorliegen, sondern erst während der Laufzeit erhoben werden. Somit musste jede Nachricht einzeln vor der Übermittlung serialisiert werden. Die Serialisierung wurde auf einer Windows-Maschine mit 48 GB RAM und einer 8-Kern CPU mit einem Basistakt von 3,8 GHz ausgeführt. Es wurden je ein Producer und Consumer verwendet und keine Parallelisierung implementiert. Die resultierenden Messdaten sind in Abbildung 66 dargestellt: Zunächst einmal ist zu erkennen, dass XML mit etwa 6.500 kB mit Abstand die ineffizienteste Serialisierungsmethode ist. Mit etwa 3.100 kB bewegt sich JSON im mittleren Bereich. Die Anwendung der GZIP-Kompression führt bei JSON nur zu einer geringfügigen Verbesserung, bei XML aber zu einer deutlich höheren Effizienz (bei beiden Methoden etwa 2900 kB). Protobuf kann alle Daten mit etwa 1650 kB mit Abstand am effizientesten serialisieren. So könnte ein Schiff mit einer GPRS-Verbindung theoretisch noch etwa 40 der verwendeten Testnachrichten pro Sekunde empfangen. Beim theoretisch maximalen Nachrichtendurchsatz zeigt sich ein anderes Bild: Hier sind JSON und XML trotz der größeren Datenmenge deutlich schneller als Protobuf. Auch die Anwendung der GZIP-Kompression führt hier nicht zu einem geringeren Nachrichtendurchsatz als Protobuf. Weiter ist zu bedenken, dass XML und JSON die Informationen über das Schema bereits mit in der Nachricht übermitteln. Bei der Serialisierung mittels Protobuf muss dieses zunächst über eine Schema-Registry übermittelt und durch den Consumer abgerufen werden, damit die Daten deserialisiert werden können.

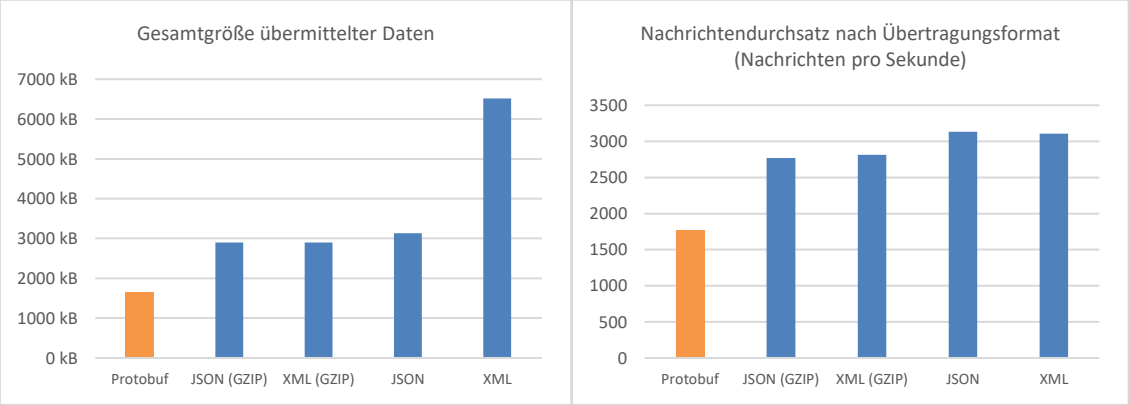

Abbildung 66: Messergebnisse bzgl. der Speichereffizienz (links) und Geschwindigkeit (rechts) der Protobuf-Serialisierung.

Es zeigt sich also, dass Protobuf sich insbesondere anbietet, wenn nur eine geringe Bandbreite vorhanden ist, und sich das Schema der Daten nicht besonders häufig ändert. Für einen hohen Nachrichtendurchsatz sind andere Serialisierungsmethoden vorteilhafter, wenn die Nachrichtengröße nur eine untergeordnete Rolle spielt. So ist Protobuf etwa sinnvoll einsetzbar, wenn Daten in einem Feldtest über eine Mobilfunkverbindung oder Satellit übermittelt werden müssen (vgl. Anforderung A17), die mit geringen Bandbreiten oder hohen kosten pro Dateneinheit verbunden sind. Da der Kafka-Cluster unabhängig vom Payload verwendet werden kann, ist beides mit der FEDMS-Architektur möglich. Die Standardimplementierung der

Connectoren und der Client-Bibliothek müsste hierzu allerdings angepasst werden, sodass nicht nur mit Protobuf, sondern auch JSON oder XML serialisiert werden kann.

6.7 Diskussion

Als Teil der Interaktionsanalyse (vgl. Abschnitt 6.1) zwischen verschiedenen Szenarien in den Fallstudien ließen sich keine unerwarteten Seiteneffekte feststellen. Im Folgenden werden die festgestellten Nutzungsmuster analysiert.

Es ist zu erwarten, dass die direkten Szenarien aus der ersten Fallstudie eine häufig verwendete Grundlage für diverse KDD-Prozesse darstellen, die das Abfragen von Daten aus verschiedenen Quellen beinhalten. Die indirekten Szenarien erfordern hier mehr Aufwand, um die FEDMS-Architektur an neue Anwendungsfälle anzupassen. Dies wurde jedoch schon beim Entwurf der Architektur berücksichtigt, sodass austauschbare Module wie die Verarbeitungsschritte zur Datenqualitätsbewertung flexibel über die Container-Registry nachgeladen werden können, ohne die grundlegende Architektur des FEDMS zu verändern. Auf ähnliche Weise hätte auch das Training des ML-Modells auf FEDMS-Infrastruktur ausgeführt werden können. In Fallstudie I wurde aber die Möglichkeit genutzt auch externe Erweiterungen der Architektur zuzulassen, sodass die Souveränität einzelner Nutzer nicht beeinträchtigt wird. Auch hier ist mit der Java Client-Bibliothek ein Framework vorhanden, welches für eine einfache Integration lokal verwalteter Infrastrukturen als dezentralen Teil des FEDMS zur Verfügung steht.

Ähnliches lässt sich auch in der zweiten Fallstudie feststellen, in der das kollaborative Testen eines Systems als eine Hauptaufgabe betrachtet werden kann, die durch die Szenarien S10 bis S16 definiert wird. Auffällig ist hier die Ähnlichkeit zu S04, die auch die Verwendung eines kollaborativen Daten-Workflows auf dem FEDMS voraussetzt. In Fallstudie II mussten die größten Anpassungen, bzw. Erweiterungen bei den Schritten vorgenommen werden, die spezifisch auf die verarbeiteten Daten zugeschnitten sind (insbesondere also die Szenarien S16 und S17). Diese Unterscheidung ist ebenfalls in der dritten Fallstudie zu beobachten, bei der ebenfalls als Grundlage ein kollaborativer Workflow genutzt wird (S19) und ein datenspezifisches Monitoring-Konzept implementiert wird (S22). Zudem wurde die Funktionalität des FEDMS Daten über die Client-Bibliothek zu extrahieren und Signaturen zu verwalten (Szenarien S18, S20, S21) genutzt.

Dies zeigt, dass der wichtigste Teil des FEDMS das sichere kollaborative Arbeiten mit Daten-Workflows darstellt. Diese Funktionalität wurde in allen Fallstudien genutzt. Darauf aufbauend kommen „Zusatzfunktionalitäten“ zum Einsatz, wie die Nutzung von Verarbeitungsschritten auf FEDMS-Architektur (Fallstudie I), die Verwendung der Client-Bibliothek, um zu testende Systeme zu überwachen (Fallstudien II & III) und die Absicherung der Dokumentation durch die

Verwaltung von Signaturen (Fallstudie III). Schließlich haben alle Fallstudien Szenarien, die eine Erweiterung der Architektur erfordern, die speziell auf die zu verarbeiteten Daten zugeschnitten ist (ML-Modell in Fallstudie I, Testauswertung in Fallstudie II, Monitor in Fallstudie III). Insgesamt zeigt sich also, dass die entworfene FEDMS-Architektur generisch genug ist, um für eine Reihe von Use-Cases eingesetzt zu werden. Durch die Zusatzfunktionalitäten ist sie jedoch nicht so generisch, dass für jeden neuen Use-Case aufwendige Weiterentwicklungsarbeiten erforderlich sind.

Wie im nächsten Abschnitt (6.8) im Detail erläutert wird, erfüllt das FEDMS alle Anforderungen und befriedigt somit den Handlungsbedarf, der am Ende von Kapitel 3 aufgezeigt wurde. Mit dieser Information liefert Kapitel 3 den in ESAAM geforderten Architekturvergleich.

Da während des Zeitraumes, in dem diese Arbeit verfasst wurde, besonders im europäischen Raum die verwandten Konzepte (siehe auch Abschnitt 3.2) GAIA-X und International Data Space an immer größerer Popularität gewonnen haben, ist mit ebenso großer Wahrscheinlichkeit davon auszugehen, dass entsprechende Weiterentwicklungen dieser Architekturen (oder bestimmter Komponenten) in Zukunft als Standard für die Realisierung von Data Spaces eingesetzt werden. Da zum Zeitpunkt des Verfassens dieser Arbeit noch nicht alle gestellten Anforderungen an die einzelnen Komponenten erfüllt werden konnten oder für die Architekturkonzepte noch keine vollständige Implementierung vorhanden war, wurde in dieser Arbeit eine eigene prototypische Implementierung verfolgt. In zukünftigen Weiterentwicklungen könnten FEDMS-Architekturelemente durch ihre Gegenstücke im IDS bzw. der GAIA-X Architektur ausgetauscht werden und das FEDMS somit in einen globalen Kontext von Data Spaces eingebunden werden. Tabelle 3 zeigt eine Gegenüberstellung der verschiedenen Architekturelemente des entworfenen FEDMS, dem IDS und GAIA-X. Architekturelemente, die sich überschneidenden Anforderungen gerecht werden, sind in derselben Zeile dargestellt. Eine 1:1-Beziehung zwischen den Anforderungsclustern ist jedoch nicht zwangsläufig vorhanden. Auf einen technischen Detailvergleich wird an dieser Stelle verzichtet, da sich die beiden Projekte zum Zeitpunkt des Verfassens dieser Arbeit noch in aktiver Entwicklung befinden. Für den Großteil der Architekturelemente würde eine Migration zum IDS oder GAIA-X größtenteils eine Anpassung auf Implementierungsebene bedeuten, da die grundlegenden Prinzipien aller Architekturen alle auf dem Konzept dezentraler Datenökosysteme aufbauen. Die FEDMS-Architektur unterscheidet sich auf konzeptioneller Ebene durch eine etwas stärkere Kopplung der Architekturschichten (vgl. auch Abschnitt 4.7), die durch die Anpassung auf die speziellen Anforderungen des Testfeldes zurückzuführen ist, und in Kapitel 4 als direkte Konsequenz aus den Anforderungen hergleitet wurde. Es ist jedoch zu erwarten, dass diese Kopplung durch das Implementieren zusätzlicher DataApps (im IDS) oder GAIA-X-konformer Services auflösbar ist.

FEDMS	International Data Space[13]	GAIA-X[14]
FEDMS Connector	IDS Connector	Eclipse Data Space Connector / GAIA-X FS Data Exchange Services
Connector Zugriffsschicht	Broker / Vocabulary Hub	GAIA-X FS Federated Catalogue
Datenverteilungsschicht	Clearing House (Zugriffsdokumentation)	GAIA-X FS Data Exchange Services
Verarbeitungsschritte	App Store / Data Apps	GAIA-X-konforme Services zur Datenverarbeitung
Datenverarbeitungsschicht (Kappa-Architektur + Workflows)	-	-
Kontrollschicht	PKI / Dynamic Attribute Provisioning Service / Dynamic Trust Monitoring / Clearing House	GAIA-X FS Identity & Trust / GAIA-X FS for Notarization and Credential storage

Tabelle 3: Gegenüberstellung der Architekturelemente des FEDMS, IDS und GAIA-X.

Auffallend ist, dass die Datenverarbeitungsschicht momentan keine vollständig äquivalenten Architekturelemente in GAIA-X oder dem IDS hat. Sie ist speziell auf die Anwendung in einem serviceorientierten Testfeld zugeschnitten und stellt einen wesentlichen Beitrag dieser Arbeit dar (siehe auch Abschnitt 7.1).

6.8 Abdeckung der Anforderungen und Ziele

Im Folgenden soll nun basierend auf den Auswertungen der Fallstudien bewertet werden, ob die Evaluationshypothese widerlegt werden konnte. Dies ist der Fall, wenn nicht alle in Abschnitt 2.6 definierten Anforderungen erfüllt werden konnten, insbesondere jene, die den in Abschnitt 3.5 identifizierten Handlungsbedarf abdecken. Weiterhin sind die aus der Forschungsfrage abgeleiteten Ziele (vgl. Abschnitt 1.3) zu prüfen, deren Erreichung sich aus der Erfüllung der jeweils zugeordneten Anforderungen ergibt.

[13] Siehe (Otto, Steinbuß, u.a. 2019)
[14] Siehe (GAIA-X 2022).

Architekturelle Kernanforderungen (Teilziel 1)

Für die Zusammenführung mehrerer Datenquellen des Testfeldes zu einer kohärenten Einheit wurden mehrere Architekturkomponenten entworfen: Speziell der FEDMS-Connector und die Connector-Zugriffsschicht ermöglichen das strukturierte Anfragen von Daten durch eine standardisierte Schnittstelle (A2). Durch die dynamische Funktionsweise der Connector-Zugriffsschicht ist ebenfalls eine Möglichkeit zur Erweiterung zur Laufzeit des FEDMS gegeben (vgl. A1). Die Integration dezentraler Testfelddatenquellen und die Verwaltung der Zugriffe auf diese Daten (A3) wurde in allen Fallstudien demonstriert. In Fallstudie I wurden hierzu Connectoren eingesetzt und zusätzlich die Fähigkeit des FEDMS geprüft Metadaten der Datenquellen zu verwalten (A4) und strukturierte Datenanfragen in entsprechender Komplexität zu beantworten. In den Fallstudien II und III wurden Workflows zum Organisieren und Abfragen der Daten über die Datenverarbeitungsschicht verwendet. Somit wurde weiter erfolgreich demonstriert, dass Daten je nach Nutzergruppe und Use-Case entweder über einen Connector mit der flexiblen und strukturierten Abfragesprache GraphQL oder als Teil eines Datenworkflows über das Auslesen von einzelnen Topics (ggf. mit entsprechenden Offsets) abgerufen werden können.

Anforderungen an das Teilen von Daten (Teilziel 2)

Auch für die Bereitstellung von Daten im Testfeld durch die verschiedenen Stakeholder wurden die vorhandenen Anforderungen umgesetzt. So bietet das in Abschnitt 4.7 vorgestellte Rollenmodell eine Möglichkeit die Interessenkonflikte aufzulösen und die Architekturelemente des FEDMS durch verschiedene Stakeholder und ohne zentrale Instanz bereitzustellen (A8). Dies wurde etwa in Fallstudie III demonstriert, indem die MCP als externer Identitätsprovider für das Signieren der Daten eingesetzt wurde. Zudem wurde mit dem Connector ein Software-Artefakt umgesetzt, welches ein souveränes Datenmanagement durch die Betreiber der dezentral organisierten Datenquellen im Testfeld Data Space ermöglicht (A6). Fallstudie I zeigte hier anhand einer historischen und einer Live-Datenquelle auf, wie Connectoren eingesetzt werden können, um Daten über das FEDMS und unter der Kontrolle des Datenproviders bereitzustellen. So konnte exemplarisch eine Pseudonymisierung von AIS-Daten implementiert werden, die zeigt, dass auch Problemstellungen des Datenschutzes durch eine Erweiterung der FEDMS-Architektur gehandhabt werden können (A7). Insbesondere die Erweiterbarkeit des quelloffenen Connectors trägt hierzu bei, da dieser eine Abkapselung zwischen einer nativen Datenquelle und dem FEDMS bzw. dem Rest des Testfeld Data Spaces gewährleistet. In Fallstudie II und III konnte die Souveränität der Nutzer bzgl. des Codes von zu testenden Systemen und der übermittelten Systemdaten durch die Verwendung der für Testfeldnutzer quelloffenen Client-Bibliothek gewährleistet werden. Schließlich führt der Einsatz eines Datentreuhänders (siehe Abschnitt 4.7)

und die benutzerdefinierbaren Speicherzeiten (vgl. Abschnitt 5.4) dazu, dass die Nutzer flexibel darüber entscheiden können, wie Daten in einem Daten-Workflow verwaltet werden (A5).

Anforderungen bezüglich der Datenweiterverarbeitung (Teilziel 3)

Zur Unterstützung der dezentralen Datenverarbeitungsprozesse im Testfeld wurden in der FEDMS-Architektur mehrere Anforderungen berücksichtigt. Zunächst einmal ist es möglich über die Schnittstellen des Systems (bzw. mit der Client-Bibliothek) beliebige externe und dezentral verwaltete Verarbeitungsschritte in einen Datenworkflow zu integrieren (A9). Dies stellt die Grundlage für die Unterstützung der dezentralen Datenverarbeitungsketten im Testfeld dar und wurde insbesondere in Fallstudie I dargestellt. Des Weiteren wird es durch die anfrageorientierte Kappa-Architektur (siehe Abschnitt 4.4.1) ermöglicht auch Live- bzw. Streaming-Daten mit Hilfe des FEDMS zu verarbeiten (A10). Alle drei Fallstudien haben dies erfolgreich demonstriert und zeigen, dass dies auch innerhalb komplexer Testaufbauten möglich ist. Schließlich ist durch das Szenario-orientierte Datenmanagementkonzept aus Abschnitt 4.4.3 eine Möglichkeit vorhanden, um Daten aus mehreren strukturierten Testdurchläufen innerhalb des FEDMS zu organisieren (A11). In Fallstudie II wurde dies anhand eines Testszenarios basierend auf historischen Daten zur Kollision zweier Schiffe validiert. Somit wurde also insgesamt eine Architektur entwickelt, die generische, dezentrale Verarbeitungsschritte abbilden kann, aber auch testfeldspezifische Anforderungen wie die Live-Datenverarbeitung und das Szenario-orientierte Datenmanagement abbilden kann.

Anforderungen bezüglich Vertrauens, Sicherheit und Nachvollziehbarkeit (Teilziel 4)

Auch die Gewährleistung von Sicherheit und Nachverfolgbarkeit der Datenverarbeitungsschritte in einem datengetriebenen Forschungsprozess wird als zentrales Element in der vorgestellten FEDMS-Architektur berücksichtigt. Zunächst wird es für die kollaborativen Daten-Workflows ermöglicht weitere Metadaten beim Datentreuhänder zu hinterlegen, die die Provenienz der Daten abbilden (A12) und diese (wie in Fallstudie I und II gezeigt) in einem standardisierten Format zu exportieren. Weiterhin ist es möglich die Daten in der Datenverarbeitungsschicht durch die Teilnehmer eines Workflows signieren zu lassen und somit eine weitere Maßnahme zur Sicherung des Vertrauens zwischen den Teilnehmern zu treffen (vgl. A13). Dies wurde ausführlich in Fallstudie III überprüft und diskutiert. Schließlich sind allgemeine Sicherheitsaspekte in der Implementierung berücksichtigt worden (A14, siehe dazu auch Abschnitt 5.6). Zudem kann durch eine Trennung der Rollen beim Deployment des FEDMS (siehe Abschnitt 4.7) durch den FEDMS-Provider eine Protokollierung der Interaktion mit dem System angefertigt werden, sodass Konflikte im Nachhinein durch die Analyse dieser Protokolle aufgelöst werden könnten. Außerdem können durch die Datenprovider weitere beliebige Nutzungsbeschränkungen in den

Connectoren implementiert werden, und der Broker hat die Kontrolle darüber welche Connectoren für das FEDMS zugelassen werden (vgl. A6 und A14).

Anforderungen der maritimen Domäne (Teilziel 5)

Das FEDMS wurde im Kontext eines maritimen Testfeldes entwickelt und bietet somit speziell auch Unterstützung für die Verwaltung maritimer Daten und berücksichtigt dabei gewisse Umweltbedingungen in maritimen Umgebungen. In Abschnitt 5.2 wurde im Detail vorgestellt, wie drei Connectoren zur Bereitstellung von AIS und RADAR-Daten entworfen wurden (vgl. A15). Diese wurden in Fallstudie I erfolgreich eingesetzt, um die Rohdaten für einen Workflow zur Erstellung eines ML-Datensatzes zu erzeugen. Als Teil dieses Workflows wurden ebenfalls die Verarbeitungsschritte zur AIS-Datenqualitätsbewertung und Vorverarbeitung eingesetzt, die in Abschnitt 4.4.5 vorgestellt wurden (A16). So konnte die Datenqualität entlang des Workflows messbar gemacht und aufgewertet werden. Weitere Datenverarbeitungsschritte für maritime Daten können in Zukunft analog implementiert werden. Schließlich wurde mit einer standmäßigen Unterstützung für Protobuf eine Möglichkeit geschaffen, um auch in Tests in Gebieten mit beschränkter Internetbandbreite durchzuführen (A17, beispielsweise eine Realtest-Version von Fallstudie II). Der letzte Teil der Evaluation zeigte in Abschnitt 6.6, dass durch die dynamische Nutzung von Protobuf entscheidende Vorteile in Bezug auf die Speichereffizienz möglicher Serialisierungsformate entstehen und, dass typische Anwendungsfälle des FEDMS im maritimen Testfeld bzgl. der Latenz und dem Datendurchsatz des Prototyps umsetzbar sind (A18). Somit wird also insgesamt eine Vereinfachung des datengetriebenen Forschung- und Entwicklungsprozesses für maritime Testfelddaten erreicht, die über die dazu vorgesehen Architekturelemente (Connectoren und (interne) Verarbeitungsschritte) beliebig erweitert werden kann.

Insgesamt wurde gezeigt, dass alle Ziele und zugehörigen Anforderungen erreicht wurden. Somit konnte die Evaluationshypothese nicht widerlegt werden und mit der entworfenen FEDMS-Architektur besteht ein Lösungskonzept zur Beantwortung der Forschungsfrage.

7 Fazit

Im folgenden Kapitel werden die Inhalte dieser Arbeit zusammengefasst und die Forschungsergebnisse eingeordnet und diskutiert. Weiterhin wird dargestellt welche Limitierungen das ausgearbeitete Konzept aufweist und welche anschließenden Forschungsaktivitäten die Erkenntnisse dieser Arbeit in Zukunft festigen und erweitern könnten.

7.1 Zusammenfassung

Trends der letzten Jahre in zahlreichen Bereichen von Forschung und Wirtschaft zeigen eine klare Entwicklung hin zu datenintensiven Verfahren und Systemen die immer größere Mengen von Daten verarbeiten. Diese Entwicklungen sind auch in der maritimen Domäne zu beobachten, insbesondere bei neuartigen Assistenzsystemen, die eine Vielzahl von Daten verarbeiten. Durch das Zusammenfügen verschiedenster Datenquellen ergeben sich in diesen Systemen neue Möglichkeiten für die Unterstützung und Autonomisierung einer sichereren und effizienteren Navigation. Allerdings erfordern komplexere Datenverarbeitungsprozesse auch neue Architekturen und damit ebenso neue Testverfahren, die wiederum Anforderungen an Teststände und Testfelder stellen. Jene sind insbesondere wichtig, weil sie die Möglichkeit bieten etablierte Testverfahren methodisch sinnvoll durchzuführen und Sensorinfrastruktur zur Überwachung von Tests bereitstellen. Je komplexer zu testende Systeme werden, desto ausgeprägtere Testverfahren werden benötigt, um diese Komplexität abzubilden. Zudem kommt die zunehmende Kollaboration verschiedener Stakeholder bei der voneinander unabhängig entwickelte Teilsysteme zu einem Gesamtsystem zusammengesetzt werden. Dies führt dazu, dass moderne Testfeldarchitekturen häufig generisch und unabhängig von den zu testenden Systemen entwickelt und als eigene Services zur Verfügung gestellt werden. So entsteht in serviceorientierten Testfeldern ein dezentral organisiertes Netz aus verschiedenen Stakeholdern, Datenquellen und zu testenden datenverarbeitenden (Teil-)Systemen, für welches eine Strategie zum souveränen Management von Testfelddaten benötigt wird. Daher wurde im Rahmen dieser Arbeit die Forschungsfrage *„Wie können dezentral organisierte maritime Daten für einen datengetriebenen Forschungs- und Entwicklungsprozess im Testfeld bereitgestellt werden?“* beantwortet.

Dazu wurde zunächst der aktuelle Stand der Forschung vorgestellt: Hierbei wurden aktuelle Erkenntnisse zum Forschungsdatenmanagement diskutiert und diese insbesondere auf maritime Testfelder bezogen. Basierend darauf wurde ein 3-Phasen Modell für das Testfelddatenmanagement hergeleitet, welches als Grundlage für die weiteren Kapitel der Arbeit diente. Ebenso wurden die Herausforderungen und Lösungsansätze für Datenmanagement in dezentral organisierten Strukturen, wie dem serviceorientierten Testfeld vorgestellt. Es wurde

dabei gezeigt, dass eine Data Space Architektur die Anforderungen der vorhandenen Stakeholder abbilden kann, ohne die grundlegenden Bedingungen im Testfeld wesentlich zu verändern. Nach einer Use-Case-Analyse wurden schließlich Anforderungen aus fünf Themenclustern betrachtet, die an ein Lösungskonzept zur Beantwortung der Forschungsfrage gestellt werden müssen. Weiter wurden in Kapitel 3 verwandte Arbeiten untersucht: Nachdem die Verwendung von klassischen Forschungsdatenmanagementsystemen aufgrund ihrer Zentralisiertheit als Lösungskonzept ausgeschlossen werden konnte, wurde die Recherche auf Datenmanagementsysteme fokussiert, die speziell auf dezentral organisierte Datenökosysteme ausgerichtet sind. Ergänzt wurde diese Betrachtung durch die Analyse von Datenmanagementarchitekturen in bestehenden Testfeldern der maritimen Domäne und der Automobildomäne. Die Analyse der Arbeiten ergab, dass speziell für kollaborative Systems-Engineering Prozesse in Testfeldern noch keine allgemeine Methodik zum Management von Daten existiert, die gleichzeitig die Dezentralität und die speziellen Anforderungen der Stakeholder eines serviceorientierten, maritimen Testfeldes abbilden könnte.

Um diese Forschungslücke zu schließen, wurde in Kapitel 4 ein Konzept für eine FEDMS-Architektur für maritime Testfelder vorgestellt. Ausgehend von der grundlegenden Struktur eines Testfeld Data Spaces wurde eine schichtenbasierte Architektur entworfen, die insbesondere das kollaborative Entwickeln und Testen von Systemen innerhalb des serviceorientieren Testfeldes unterstützt. Mit Hilfe des 3-Phasen Modells für das Testfelddatenmanagement wurde hierzu eine anfrageorientierte Kappa-Architektur mit Datenquellen-Connectoren verwendet, und ein dezentrales Workflow-Modell entworfen, welches in allen Phasen die Anforderungen der Stakeholder abbilden kann. Für die explorative Datenanalyse und die Modellbildung unterstützen die Connectoren komplexe Datenanfragen über eine standardisierte Schnittstelle, und gewährleisten gleichzeitig die Souveränität der Datenprovider. Zudem können insbesondere Szenario-basierte Testverfahren durch das FEDMS unterstützt werden, indem verteilte Daten-Workflows mit entsprechenden Metadaten annotiert werden. Schließlich ist das FEDMS mit einer Kontroll- und Datenverteilungsschicht abgesichert, die Zugriffe auf alle anderen Komponenten kontrollieren und externe Identitätsprovider integrieren kann, sodass besonders kritische Datentransaktionen durch die Beteiligten signiert und nachverfolgt werden können. Die prototypische Implementierung dieser Architektur unter der Berücksichtigung spezieller maritimer Datenarten wurde in Kapitel 5 beschrieben.

Die Architektur und die parallel dazu entwickelte holistische Datenmanagementmethodik für serviceorientierte maritime Testfelder wurde schließlich in Kapitel 6 evaluiert. Zunächst wurde eine Evaluation basierend auf der ESAAM-Methodik durchgeführt. Hierbei fand ebenfalls eine Orientierung am 3-Phasen Modell aus Kapitel 2 statt. Insgesamt wurden 22 Nutzungsszenarien der FEDMS-Architektur in drei Fallstudien untersucht, die sich mit dem Entwurf eines ML-

Modells zur maritimen Verkehrsprädiktion, der Untersuchung des maritimen Assistenzsystems MTCAS und einer Contract-basierten Zertifizierung eines Systems im Testfeld beschäftigten. Weiterhin wurde anhand von Performanz-Messungen überprüft, ob die verwendeten Technologien für den definierten Einsatzbereich des FEDMS geeignet sind. Es folgte eine Diskussion der Einsetzbarkeit und Erweiterung der Architektur und eine Einordnung der Ergebnisse in den Kontext relevanter Standardisierungsaktivitäten wie GAIA-X und dem IDS. Abschließend wurde die Abdeckung der Ziele und Anforderungen dieser Arbeit analysiert.

7.2 Ergebnisse

Anhand der Auswertungen der einzelnen Teile der Evaluation lassen sich die Ergebnisse der Arbeit beschreiben. Zunächst konnte anhand von 22 Szenarien und basierend auf dem 3-Phasen Modell in den drei Fallstudien demonstriert werden, dass alle definierten Anforderungen erfüllt wurden und eine Anwendbarkeit der entworfenen FEDMS-Architektur für serviceorientiere maritime Testfelder gegeben ist.

In der ersten Fallstudie wurde gezeigt, dass die entworfene FEDMS-Architektur eine Ende-zu-Ende Datenverarbeitung im Testfeld unterstützen kann. In der Datenverarbeitungskette war es dabei möglich vordefinierte und wiederverwendbare Verarbeitungsschritte mit einer externen Datenverarbeitungskette zu kombinieren und somit unter Berücksichtigung der Datensouveränitätprinzipien einen ML-Datensatz zu erstellen. Schließlich konnten strukturierte Datenabfragen an das FEDMS gestellt werden, welches durch automatisches Routing der Abfrage unter Berücksichtigung von nutzerspezifischen Zugriffsberechtigungen sowohl historische Daten für das Training des Modells als auch Live-Daten für ein Test-Deployment des Modells bereitstellen konnte. Zudem konnten Daten pseudonymisiert werden und für die weitere und langfristige Archivierung exportiert werden. Somit wurde gezeigt, dass die FEDMS-Architektur die Modellbildung und Wissensgenerierung im Testfeld (*Phase I*) erheblich unterstützt.

Die zweite Fallstudie zeigte die Anwendbarkeit des FEDMS für die detaillierte Untersuchung eines maritimen Assistenzsystems zur Kollisionsvermeidung im Rahmen der V+V. Dabei wurde demonstriert, dass das Konzept der kollaborativen Workflows für das kollaborative Testen eines zusammengesetzten Systems geeignet ist, insbesondere wenn in einem serviceorientierten Testfeld geistiges Eigentum einzelner Entwickler geschützt werden muss und nicht alle (Diagnose-)Daten mit anderen Nutzern geteilt werden sollen. Mit der Erweiterung eines Workflows zur Szenario-orientierten Datenverwaltung war es zudem möglich einzelne Testläufe klar voneinander abzugrenzen. Eine FEMDS-unterstützte Auswertung der gesammelten Daten lieferte dabei relevante, neue Erkenntnisse über das Systemverhalten des SuT und konnte durch

die standardisierte Erhebung von Metadaten zur Provenienz wissenschaftlich reproduzierbar begründet werden.

In der dritten Fallstudie war es möglich ein Zertifizierungsverfahren auf Datenebene durch das FEDMS zu unterstützen und in einem dezentralen Setup für ein zusammengesetztes System zu ermöglichen. Mit Hilfe der entworfenen Client-Bibliothek konnten sogenannte Recorder in das SuT injiziert werden. Diese verwalteten Events aus den Teilsystemen in einem kollaborativen Workflow, der die Systemarchitektur spiegelte. Ein Monitor konnte dann über die FEDMS-Architektur auf diesen Workflow zugreifen und zur Laufzeit des Systems verifizieren, ob die definierten Contracts durch die Teilsysteme eingehalten wurden. Für eine Legitimation des für die Zertifizierung verwendeten Datensatzes war es den Entwicklern möglich die Daten ihrer Teilsysteme zu signieren und diese Signaturen durch das FEDMS verwalten zu lassen.

Schließlich konnte bei den Performanz-Messungen festgestellt werden, dass die verwendeten Technologien etwa für das Streaming größerer Sensordaten geeignet sind, aber auch in maritimen Umgebungen mit geringer Bandbreite eingesetzt werden können, und dass typische Anwendungsfälle des FEDMS im maritimen Testfeld bzgl. der Latenz und dem Datendurchsatz des Prototyps umsetzbar sind.

Der wesentliche Beitrag dieser Forschungsarbeit lässt sich zudem beim Vergleich mit Arbeiten erkennen, die die Data Space Architektur standardisieren. In Tabelle 3 ist zu sehen, dass viele Architekturelemente, die für die Umsetzung eines Data Spaces notwendig sind, auch in den allgemeineren Modellen von GAIA-X und dem IDS wiederzufinden sind. Den Besonderheiten serviceorientierter, maritimer Testfelder und den speziellen Anforderungen ihrer Stakeholder können solche Standardmodelle aber im Allgemeinen nicht gerecht werden. Erst der Entwurf des kollaborativen Workflowmodells und die Kopplung mit einer anfrageorientierten Kappa-Architektur ermöglichen hier überhaupt erst die effiziente Nutzung einer Data Space Architektur in Testfeldern. Weitere Anpassungen und Architekturelemente berücksichtigen dabei spezielle Besonderheiten und Technologien der maritimen Domäne und häufig vorkommender Datenarten. Des Weiteren trägt diese Arbeit auf theoretischer Ebene zu einem besseren Verständnis von Datenmanagementprozessen in Testfeldern bei und zeigt eine neue Methode zur kollaborativen Evaluation von Systemen auf, die große Mengen von Daten verarbeiten. Zudem unterstützt das in dieser Arbeit entworfene und evaluierte Konzept den zunehmenden Trend der Datensouveränität und hilft beim Umgang mit Daten aus Tests von zunehmend komplexer werdenden maritimen Assistenzsystemen. Schließlich trägt dieses erweiterbare Konzept zur Entwicklung generischer und serviceorientierter Testfelder bei, die eine meist deutlich effizientere und kostengünstigere Variante von Use-Case spezifischen Testfeldern darstellen.

7.3 Limitierungen und Ausblick

Das in Kapitel 4 vorgestellte Konzept wurde auf Basis der Forschungsfrage, der Ziele und der Anforderungen aus den Kapiteln 1 und 2 entwickelt. In der Evaluation wurde mit Hilfe der prototypischen Implementierung gezeigt, dass die Anforderungen und Ziele erfüllt und die Forschungsfrage beantwortet werden konnten. Darüber hinaus ist ein Einsatz der FEDMS-Architektur zwar denkbar, wurde jedoch nicht in der Entwicklung des Lösungskonzeptes berücksichtigt. So bietet das Metadatenmodell der Datenworkflows zwar die Möglichkeit Informationen zu Nutzungslizenzen für die ausgetauschten Daten zu speichern, verwendet hier aber kein standardisiertes Modell oder vordefinierte Nutzungslizenzen. Weiter ist es nicht möglich diese durchzusetzen, indem etwa Daten mit zeitlich beschränkten Lizenzen nach Ablauf der erlaubten Nutzungszeit nicht mehr verfügbar sind. Verwandte Arbeiten bieten hier bereits Ansätze (vgl. Otto, Steinbuß, u.a. 2019), die in Zukunft in die FEDMS-Architektur integriert werden könnten. Auf derselben Ebene könnte die Verhandlung von Verträgen zum Datenaustausch implementiert werden, die ebenfalls nicht Teil der Ziele dieser Arbeit war, aber insbesondere bei der Integration eines Testfeldes in größere Infrastrukturen wie GAIA-X relevant sein könnte. In dieser Hinsicht ist erneut darauf hinzuweisen, dass (nach Fertigstellung entsprechender Standards) eine Harmonisierung der FEDMS-Architektur mit Data Space Standards anzustreben ist, wie bereits in Abschnitt 6.7 diskutiert wurde.

Weiterhin wurden im Rahmen dieser Arbeit nur einige Connectoren und interne Verarbeitungsschritte exemplarisch konzeptioniert und umgesetzt. Für den Einsatz in einem generischen serviceorientierten Testfeld mit zahlreichen Datenquellen ist für eine effektive Verwendung des FEDMS natürlich eine größere Anzahl von Connectoren und Verarbeitungsschritten notwendig. So könnten etwa noch weitere Connectoren zur Bereitstellung von Wetterdaten, Schiffsdaten, Karten- oder Kameradaten entwickelt werden, und zusätzliche parametrisierbare Verarbeitungsschritte, die eine Vorverarbeitung oder Fusion dieser Daten übernehmen könnten, erforscht werden.

Auch die Connectoren selbst könnten für bestimmte Anwendungsfälle noch erweitert werden. Fallstudie I demonstrierte eine einfache Pseudonymisierung der MMSIs in einem AIS-Datensatz durch das Berechnen von Hashes. Hier sind aber deutlich komplexere und sicherere Verfahren denkbar (vgl. z.B. Kohlmayer u. a. 2014), um eine Pseudonymisierung bzw. Anonymisierung durchzuführen. Davon abgesehen können auch weitere Maßnahmen zum Datenschutz oder der Zugangskontrolle über die Connectoren und ihre Anwendungen in der maritimen Domäne Teil weiterer Forschungsaktivitäten sein.

Abschließend kann die Untersuchung der Übertragbarkeit dieser Arbeit auf Testfelder anderer Verkehrsdomänen (Automobilverkehr, Luft- und Raumfahrt und Zugverkehr) als weitere

anschließende Forschungsaktivität gesehen werden. Es ist zu vermuten, dass gerade in anderen Domänen bei denen physikalische Testfelder in vergleichbarer Weise genutzt werden ein Großteil der Architekturkomponenten, die nicht speziell auf maritime Daten ausgerichtet wurden (wie das Workflow-Modell und die anfrageorientiere Kappa-Architektur), wiederverwertbar ist.

8 Literaturverzeichnis

Abbaspour Asadollah, Sara, Rafia Inam, und Hans Hansson. 2015. „A Survey on Testing for Cyber Physical System“. In *Testing Software and Systems*, herausgegeben von Khaled El-Fakih, Gerassimos Barlas, und Nina Yevtushenko, 194–207. Lecture Notes in Computer Science. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-25945-1_12.

Abdel-Aal, R. E., M. A. Elhadidy, und S. M. Shaahid. 2009. „Modeling and Forecasting the Mean Hourly Wind Speed Time Series Using GMDH-Based Abductive Networks“. *Renewable Energy* 34 (7): 1686–99. https://doi.org/10.1016/j.renene.2009.01.001.

AbuAlhaol, Ibrahim, Rafael Falcon, Rami Abielmona, und Emil Petriu. 2018. „Mining Port Congestion Indicators from Big AIS Data“. In *2018 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN.2018.8489187.

Adland, Roar, Haiying Jia, und Siri Strandenes. 2017. „Are AIS-based trade volume estimates reliable? The case of crude oil exports“. *Maritime Policy & Management*, März, 1–9. https://doi.org/10.1080/03088839.2017.1309470.

Agostinho, Carlos, Yves Ducq, Gregory Zacharewicz, João Sarraipa, Fenareti Lampathaki, Raul Poler, und Ricardo Jardim-Goncalves. 2016. „Towards a Sustainable Interoperability in Networked Enterprise Information Systems: Trends of Knowledge and Model-Driven Technology“. *Computers in Industry*, Special Issue on Future Perspectives On Next Generation Enterprise Information Systems, 79 (Juni): 64–76. https://doi.org/10.1016/j.compind.2015.07.001.

Aho, T., O. Sievi-Korte, T. Kilamo, S. Yaman, und T. Mikkonen. 2020. „Demystifying Data Science Projects: A Look on the People and Process of Data Science Today“. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 12562 LNCS: 153–67. https://doi.org/10.1007/978-3-030-64148-1_10.

Ameixieira, Carlos, André Cardote, F. Neves, Rui Meireles, S. Sargento, Luís Coelho, João Afonso, u. a. 2014. „Harbornet: a real-world testbed for vehicular networks“. *IEEE Communications Magazine*. https://doi.org/10.1109/MCOM.2014.6894460.

Amorim, Ricardo, João Aguiar Castro, João Rocha, und Cristina Ribeiro. 2017. „A comparison of research data management platforms: architecture, flexible metadata and interoperability“. *Universal Access in the Information Society* 16 (November). https://doi.org/10.1007/s10209-016-0475-y.

Ando, Hideyuki. 2017. „Activities of smart ship application platform 2 project (SSAP2)“. *International Marine Purchasing Association (IMPA), London, UK*.

Andreasson, Björn, Mikael Olofsson, Anders Johannesson, Cajsa Jersler Fransson, Peter Bergljung, Håkan Heurlin, Tuomas Martikainen, u. a. 2019. „STM Validation Deliverables 2.6, 2.10 & 2.12 - Voyage Management Testbed Report“. https://stm-stmvalidation.s3.eu-west-1.amazonaws.com/uploads/20200225090150/STMVal_D2.6-D2.10-D2.12-Voyage-management-testbed-report-1.pdf.

ANSI. 2022. „UL 4600 Standard for Safety: Evaluation of Autonomous Products“. American National Standards Institute. https://www.shopulstandards.com/ProductDetail.aspx?productid=UL4600.

Apache Software Foundation. 2017. „Apache Kafka - Documentation“. Apache Kafka. 2017. https://kafka.apache.org/documentation/.

Armando, Alessandro, Roberto Carbone, Luca Compagna, Jorge Cuellar, und Llanos Tobarra. 2008. „Formal analysis of SAML 2.0 web browser single sign-on: breaking the SAML-based single sign-on for google apps“. In *Proceedings of the 6th ACM workshop on Formal methods in security engineering*, 1–10.

Auger, Antoine, Ernesto Exposito, und Emmanuel Lochin. 2017. „Sensor observation streams within cloud-based IoT platforms: Challenges and directions“. In *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, 177–84. IEEE.

Bader, Sebastian, Jaroslav Pullmann, Christian Mader, Sebastian Tramp, Christoph Quix, Andreas W. Müller, Haydar Akyürek, u. a. 2020. „The International Data Spaces Information Model – An Ontology for Sovereign Exchange of Digital Content“. In *The Semantic Web – ISWC 2020*, herausgegeben von Jeff Z. Pan, Valentina Tamma, Claudia d’Amato, Krzysztof Janowicz, Bo Fu,

Axel Polleres, Oshani Seneviratne, und Lalana Kagal, 176–92. Lecture Notes in Computer Science. Cham: Springer International Publishing.

Ball, Alex. 2012. *Review of data management lifecycle models*. University of Bath, IDMRC.

Benvenuti, Luca, Alberto Ferrari, Emanuele Mazzi, und A. L. Vincentelli. 2008. „Contract-based design for computation and verification of a closed-loop hybrid system“. In *International Workshop on Hybrid Systems: Computation and Control*, 58–71. Springer.

Berbić, Jadran, Eva Ocvirk, Dalibor Carević, und Goran Lončar. 2017. „Application of Neural Networks and Support Vector Machine for Significant Wave Height Prediction“. *Oceanologia* 59 (3): 331–49. https://doi.org/10.1016/j.oceano.2017.03.007.

Bertino, Elisa, Gabriel Ghinita, Murat Kantarcioglu, Dang Nguyen, Jae Park, Ravi Sandhu, Salmin Sultana, Bhavani Thuraisingham, und Shouhuai Xu. 2014. „A Roadmap for Privacy-Enhanced Secure Data Provenance“. *Journal of Intelligent Information Systems* 43 (3): 481–501. https://doi.org/10.1007/s10844-014-0322-7.

BMWi. 2020. „GAIA-X: Das europäische Projekt startet in die nächste Phase“. Bundesministerium für Wirtschaft und Energie (BMWi). https://www.bmwk.de/Redaktion/DE/Publikationen/Digitale-Welt/gaia-x-das-europaeische-projekt-startet-in-die-naechste-phase.pdf?__blob=publicationFile&v=8.

Böde, Eckard, Werner Damm, Günter Ehmen, Martin Fränzle, Kim Grüttner, Philipp Ittershagen, Bernhard Josko, u. a. 2019. „FAT-Schriftenreihe 316: MULTIC Tooling“. Verband der Automobilindustrie. https://www.vda.de/vda/de/aktuelles/publikationen/publication/fat-schriftenreihe-316.

Breiman, Leo. 2001. „Random Forests“. *Machine Learning* 45 (1): 5–32. https://doi.org/10.1023/A:1010933404324.

Brinkmann, Marius. 2018. „Physikalische Testfeld-Architektur für die Unterstützung der Entwicklung von automatisierten Schiffsführungssystemen“. Dissertation, Universität Oldenburg. http://oops.uni-oldenburg.de/3725/.

Brinkmann, Marius, Mohamed Abdelaal, und Axel Hahn. 2018. „Vessel-in-the-Loop Architecture for Testing Highly Automated Maritime Systems“. In *17th Conference on Computer and IT Applications in the Maritime Industries (COMPIT 18)*, 308–21. Pavone, Italien. http://data.hiper-conf.info/compit2018_pavone.pdf.

Brinkmann, Marius, und Axel Hahn. 2017. „Testbed Architecture for Maritime Cyber Physical Systems“. In *2017 IEEE 15th International Conference on Industrial Informatics*, 923–28. IEEE.

Brinkmann, Marius, Axel Hahn, und Bjørn Åge Hjøllo. 2017. „Physical Testbed for Highly Automated and Autonomous Vessels“. In *16th International Conference on Computer and IT Applications in the Maritime Industries*.

Brinkmann, Marius, Arne Stasch, und Axel Hahn. 2016. „Testbeds for Verification and Validation of Maritime Safety“. In *12th International Symposium on Integrated Ship's Information Systems & Marine Traffic Engineering Conference*, 11.

Brost, G.S., M. Huber, M. Wei, M. Protsenko, J. Schütte, und S. Wessel. 2018. „An ecosystem and IoT device architecture for building trust in the industrial data space“. In *CPSS 2018 - Proceedings of the 4th ACM Workshop on Cyber-Physical System Security, Co-located with ASIA CCS 2018*, 39–50. https://doi.org/10.1145/3198458.3198459.

Brouer, Berit Dangaard, Christian Vad Karsten, und David Pisinger. 2016. „Big Data Optimization in Maritime Logistics“. In *Big Data Optimization: Recent Developments and Challenges*, herausgegeben von Ali Emrouznejad, 319–44. Studies in Big Data. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-30265-2_14.

Brouer, Berit, Christian Vad Karsten, und David Pisinger. 2017. „Optimization in liner shipping“. *4OR* 15 (März). https://doi.org/10.1007/s10288-017-0342-6.

Cariou, Pierre. 2011. „Is slow steaming a sustainable means of reducing CO 2 emissions from container shipping?“ *Transportation Research Part D-transport and Environment - TRANSP RES PT D-TRANSP ENVIRO* 16 (Mai): 260–64. https://doi.org/10.1016/j.trd.2010.12.005.

Cavanillas, Jose Maria, Edward Curry, und Wolfgang Wahlster, Hrsg. 2016. *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*. Springer International Publishing. https://doi.org/10.1007/978-3-319-21569-3.

Chen, Min, Shiwen Mao, und Yunhao Liu. 2014. „Big Data: A Survey“. *Mobile Networks and Applications* 19 (2): 171–209. https://doi.org/10.1007/s11036-013-0489-0.

Cheng, Liang, Zhaojin Yan, Yijia Xiao, Yanming Chen, Fangli Zhang, und Manchun Li. 2018. „Using big data to track marine oil transportation along the 21st-century Maritime Silk Road“. *Science China Technological Sciences* 62 (Dezember). https://doi.org/10.1007/s11431-018-9335-1.

Chowdhury, Mashrur, Mizanur Rahman, Anjan Rayamajhi, Sakib Khan, Mhafuzul Islam, Zadid Khan, und James Martin. 2018. „Lessons Learned from the Real-World Deployment of a Connected Vehicle Testbed“. *Transportation Research Record Journal of the Transportation Research Board*, Oktober. https://doi.org/10.1177/0361198118799034.

Chung, P.W.H., und Z. Liao. 2008. „Cross-organisation dataspace (COD) - Architecture and implementation“. In *Proceedings - International Conference on Computer Science and Software Engineering, CSSE 2008*, 6:448–51. https://doi.org/10.1109/CSSE.2008.1638.

Cindy, K.N., W. Ning, F.T.G. Narcisse, X. De, und S. François. 2009. „Building semantic relationships incrementally in dataspace“. In *2009 1st International Conference on Information Science and Engineering, ICISE 2009*, 2288–91. https://doi.org/10.1109/ICISE.2009.370.

Claramunt, Christophe, Cyril Ray, Elena Camossi, Anne-Laure Jousselme, Melita Hadzagic, Gennady Andrienko, Natalia Andrienko, Yannis Theodoridis, George Vouros, und Loic Salmon. 2017. *Maritime data integration and analysis: recent progress and research challenges*.

Contarinis, Stilianos, Athanasios Pallikaris, und Byron Nakos. 2020. „The Value of Marine Spatial Open Data Infrastructures—Potentials of IHO S-100 Standard To Become the Universal Marine Data Model“. *Journal of Marine Science and Engineering* 8 (8): 564. https://doi.org/10.3390/jmse8080564.

Cooper, Brian F., Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, und Russell Sears. 2010. „Benchmarking cloud serving systems with YCSB“. In *Proceedings of the 1st ACM symposium on Cloud computing*, 143–54. SoCC '10. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/1807128.1807152.

Coraddu, Andrea, Serena Lim, L. Oneto, Kayvan Pazouki, R. Norman, und A. Murphy. 2019. „A novelty detection approach to diagnosing hull and propeller fouling“. *Ocean Engineering* Volume 176: 65–73. https://doi.org/10.1016/J.OCEANENG.2019.01.054.

Coraddu, Andrea, Luca Oneto, Aessandro Ghio, Stefano Savio, Davide Anguita, und Massimo Figari. 2016. „Machine Learning Approaches for Improving Condition-Based Maintenance of Naval Propulsion Plants“. *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 230 (1): 136–53. https://doi.org/10.1177/1475090214540874.

Crowston, Kevin, und Jian Qin. 2011. „A capability maturity model for scientific data management: Evidence from the literature“. *Proceedings of the American Society for Information Science and Technology* 48 (1): 1–9.

Cuno, Silke, Lina Bruns, Nikolay Tcholtchev, Philipp Lämmel, und Ina Schieferdecker. 2019. „Data governance and sovereignty in urban data spaces based on standardized ICT reference architectures“. *Data* 4 (1): 16.

Curry, Edward. 2012. „System of systems information interoperability using a linked dataspace“. In *2012 7th International Conference on System of Systems Engineering (SoSE)*, 101–6. IEEE.

———. 2019. „Fundamentals of Real-time Linked Dataspaces“. In *Real-time Linked Dataspaces, Enabling Data Ecosystems for Intelligent Systems*, 63–80. https://doi.org/10.1007/978-3-030-29665-0_4.

———. 2020. „Dataspaces: Fundamentals, Principles, and Techniques“. In *Real-Time Linked Dataspaces: Enabling Data Ecosystems for Intelligent Systems*, herausgegeben von Edward Curry, 45–62. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-29665-0_3.

Czarnecki, Krzysztof. 2018. *Operational Design Domain for Automated Driving Systems - Taxonomy of Basic Terms*. https://doi.org/10.13140/RG.2.2.18037.88803.

D’Ambrosio, Joseph, und Grant Soremekun. 2017. „Systems engineering challenges and MBSE opportunities for automotive system design“. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2075–80. IEEE.

Danish Maritime Authority. 2003. „Collision between Chinese bulk carrier FU SHAN HAI and Cypriot container vessel GDYNIA“. Copenhagen, Denmark: MINISTRY OF ECONOMIC AND BUSINESS AFFAIRS. https://www.vragguiden.dk/FuShanHai.pdf.

———. o. J. „AIS Data“. Zugegriffen 12. Juni 2022. http://dma.dk/safety-at-sea/navigational-information/ais-data.

De Clercq, Jan. 2002. „Single sign-on architectures“. In *International Conference on Infrastructure Security*, 40–58. Springer.

Deelman, Ewa, und Ann Chervenak. 2008. „Data management challenges of data-intensive scientific workflows“. In *2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID)*, 687–92. IEEE.

Demchenko, Yuri, Cees De Laat, und Peter Membrey. 2014. „Defining architecture components of the Big Data Ecosystem“. In *2014 International Conference on Collaboration Technologies and Systems (CTS)*, 104–12. IEEE.

Demchenko, Yuri, Paola Grosso, Cees De Laat, und Peter Membrey. 2013. „Addressing big data issues in scientific data infrastructure“. In *2013 International Conference on Collaboration Technologies and Systems (CTS)*, 48–55. IEEE.

Dhar, Vasant. 2013. „Data science and prediction“. *Communications of the ACM* 56 (12): 64–73. https://doi.org/10.1145/2500499.

Diran, D., T. Hoppe, J. Ubacht, A. Slob, und K. Blok. 2020. „A data ecosystem for data-driven thermal energy transition: Reflection on current practice and suggestions for re-design“. *Energies* 13 (2). https://doi.org/10.3390/en13020444.

Divyabharathi, D. N., und Nagaraj G. Cholli. 2020. „A review on identity and access management server (keycloak)“. *International Journal of Security and Privacy in Pervasive Computing (IJSPPC)* 12 (3): 46–53.

Dixon, Charles S., und Russell G. Morrison. 2008. „A Pseudolite-Based Maritime Navigation System: Concept through to Demonstration“. *Positioning* 1 (13).

DNV GL AS. 2018. „DNVGL-CG-0264: Autonomous and remotely operated ships“. https://rules.dnv.com/docs/pdf/DNV/cg/2018-09/dnvgl-cg-0264.pdf.

Dobbelaere, Philippe, und Kyumars Sheykh Esmaili. 2017. „Kafka versus RabbitMQ: A comparative study of two industry reference publish/subscribe implementations: Industry Paper“. In *Proceedings of the 11th ACM international conference on distributed and event-based systems*, 227–38.

Duan, Yucong, Guohua Fu, Nianjun Zhou, Xiaobing Sun, Nanjangud C Narendra, und Bo Hu. 2015. „Everything as a service (XaaS) on the cloud: origins, current and future trends“. In *2015 IEEE 8th International Conference on Cloud Computing*, 621–28. IEEE.

Dubuc, Timothée, Frederic Stahl, und Etienne B. Roesch. 2020. „Mapping the big data landscape: technologies, platforms and paradigms for real-time analytics of data streams“. *IEEE Access* 9: 15351–74.

Eckert, Claudia. 2018. *IT-Sicherheit: Konzepte - Verfahren - Protokolle*. *IT-Sicherheit*. De Gruyter Oldenbourg. https://doi.org/10.1515/9783110563900.

Eggers, Günter, Bernd Fondermann, Berthold Maier, Klaus Ottradovetz, Julius Pfrommer, Ronny Reinhardt, Hannes Rollin, u. a. 2020. „GAIA-X: Technical Architecture“. Federal Ministry for Economic Affairs and Energy.

Ehmen, Günter, Björn Koopmann, Yosab Bebawy, und Philipp Ittershagen. 2020. „Measurement-based Online Verification of Timing Properties in Distributed Systems“. In *2020 International Conference on Omni-layer Intelligent Systems (COINS)*, 1–6. IEEE.

Eisner, D. A. 2018. „Reproducibility of Science: Fraud, Impact Factors and Carelessness“. *Journal of Molecular and Cellular Cardiology* 114 (Januar): 364–68. https://doi.org/10.1016/j.yjmcc.2017.10.009.

Elsayed, Ibrahim, und Peter Brezany. 2012. „Dataspace Support Platform For E-Science“. *Computer Science* 13 (Januar): 49. https://doi.org/10.7494/csci.2012.13.1.49.

Engel, Avner. 2010. „Introduction“. In *Verification, Validation, and Testing of Engineered Systems*, 1–59. John Wiley & Sons, Ltd. https://doi.org/10.1002/9780470618851.ch1.

European Union Agency for Cybersecurity, J Domingo-Ferrer, M Hansen, J Hoepman, D Le Métayer, R Tirtea, S Schiffner, und G Danezis. 2015. *Privacy and data protection by design : from policy to engineering*. https://doi.org/10.2824/38623.

Falorsi, P.D., B. Liseo, und M. Scannapieco. 2019. „Dealing with privacy issues in data integration: Scenarios for official statistics“. *Frontiers in Artificial Intelligence and Applications* 317: 272–79. https://doi.org/10.3233/FAIA190028.

Fayyad, Usama, Gregory Piatetsky-Shapiro, und Padhraic Smyth. 1996. „From Data Mining to Knowledge Discovery in Databases“, AI Magazine, , Nr. 17: 37–54.

Feick, Martin, Niko Kleer, und Marek Kohn. 2018. „Fundamentals of real-time data processing architectures lambda and kappa“. *SKILL 2018-Studierendenkonferenz Informatik*.

Fernández, Pablo, José Pablo Suárez, Agustín Trujillo, Conrado Domínguez, und José Miguel Santana. 2018. „3D-Monitoring Big Geo Data on a Seaport Infrastructure Based on FIWARE“. *Journal of Geographical Systems* 20 (2): 139–57. https://doi.org/10.1007/s10109-018-0269-2.

Fletcher, Tom, Vikram Garaniya, Shuhong Chai, Rouzbeh Abbassi, Hongyang Yu, Thuy Chu Van, Richard Brown, und Faisal Khan. 2018. „An application of machine learning to shipping emission inventory“. *The International Journal of Maritime Engineering* 160 (Dezember): A381–95. https://doi.org/10.3940/rina.ijme.2018.a4.500).

Franklin, Michael, Alon Halevy, und David Maier. 2005. „From databases to dataspaces: a new abstraction for information management“. *ACM Sigmod Record* 34 (4): 27–33.

Fraunhofer ISST. 2021. „Dataspace Connector“. Dataspace Connector. 2021. https://international-data-spaces-association.github.io/DataspaceConnector/.

Freitas, André, Seán O'Riáin, und Edward Curry. 2020. „Querying and Searching Heterogeneous Knowledge Graphs in Real-Time Linked Dataspaces“. In *Real-Time Linked Dataspaces: Enabling Data Ecosystems for Intelligent Systems*, herausgegeben von Edward Curry, 105–24. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-29665-0_7.

Funabiki, Nobuo, Dadet Pramadihanto, Riyadh Arridha, und Sritrusta Sukaridhoto. 2017. „Classification extension based on IoT-big data analytic for smart environment monitoring and analytic in real-time system“. *International Journal of Space-Based and Situated Computing* 7 (Januar): 82. https://doi.org/10.1504/IJSSC.2017.10008038.

GAIA-X. 2021. „Software Requirements Specification for Gaia-X Federation Service: Sovereign Data Exchange - Data Exchange Logging Service SDE.DELS“. eco – Association of the Internet Industry (eco – Verband der Internetwirtschaft e.V.). https://www.gxfs.eu/download/1734/.

———. 2022. „Gaia-x - Architecture Document - 22.04. Release“. https://gaia-x.eu/wp-content/uploads/2022/06/Gaia-x-Architecture-Document-22.04-Release.pdf.

Gallego, Antonio-Javier, Pablo Gil, Antonio Pertusa, und Robert B. Fisher. 2019. „Semantic Segmentation of SLAR Imagery with Convolutional LSTM Selectional AutoEncoders“. *Remote Sensing* 11 (12): 1402. https://doi.org/10.3390/rs11121402.

Gambardella, Luca Maria, Andrea-Emilio Rizzoli, und Marco Zaffalon. 1998. „Simulation and Planning of an Intermodal Container Terminal“. *Simulation* 71 (August): 107–16. https://doi.org/10.1177/003754979807100205.

Gaspar, W., R. Braga, und F. Campos. 2011. „SciProv: An architecture for semantic query in provenance metadata on e-Science context“. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6865 LNCS: 68–81.

Gedik, Buğra, Deepak S. Turaga, und Henrique C. M. Andrade, Hrsg. 2014. „Introduction to stream processing“. In *Fundamentals of Stream Processing: Application Design, Systems, and Analytics*, 33–74. Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9781139058940.004.

Gelhaar, Joshua, und Boris Otto. 2020. „Challenges in the Emergence of Data Ecosystems“. *PACIS 2020 Proceedings*, Juni. https://aisel.aisnet.org/pacis2020/175.

Goundan, Appu, und Chen Qingyang. 2018. „Introducing Jib — Build Java Docker Images Better“. *Google Cloud Platform Blog* (blog). 9. Juli 2018. https://cloudplatform.googleblog.com/2018/07/introducing-jib-build-java-docker-images-better.html.

Grassi, Paul, Naomi Lefkovitz, und Kevin Mangold. 2015. „Privacy-Enhanced Identity Brokers“. *NIST-NCCoE. October* 19.

Gray, Jim. 1996. „Evolution of data management“. *Computer* 29 (10): 38–46.

Grobe, Hannes, Michael Diepenbroek, Nicolas Dittert, Manfred Reinke, und Rainer Sieger. 2006. „Archiving and Distributing Earth-Science Data with the PANGAEA Information System“. In *Antarctica: Contributions to Global Earth Sciences*, herausgegeben von Dieter Karl Fütterer, Detlef Damaske, Georg Kleinschmidt, Hubert Miller, und Franz Tessensohn, 403–6. Berlin, Heidelberg: Springer. https://doi.org/10.1007/3-540-32934-X_51.

Grossman, R.L. 2018. „Progress Toward Cancer Data Ecosystems“. *Cancer Journal (United States)* 24 (3): 122–26. https://doi.org/10.1097/PPO.0000000000000318.

Guerriero, Marco, Peter Willett, Stefano Coraluppi, und Craig Carthel. 2008. „Radar/AIS data fusion and SAR tasking for maritime surveillance“. In *Proceedings of the 11th International Conference on Information Fusion, FUSION 2008*, 1–5. IEEE.

Győrödi, Cornelia, Robert Győrödi, George Pecherle, und Andrada Olah. 2015. „A comparative study: MongoDB vs. MySQL“. In *2015 13th International Conference on Engineering of Modern Electric Systems (EMES)*, 1–6. https://doi.org/10.1109/EMES.2015.7158433.

Hahn, A., und T. Noack. 2016. „EMaritime Integrated Reference Platform“. In *Deutscher Luft- Und Raumfahrtkongress*. Braunschweig: Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V., Bonn. https://publikationen.dglr.de/?tx_dglrpublications_pi1[document_id]=420297.

Hake, Georg, Sebastian Feuerstack, und Axel Hahn. 2020. „Towards Recertification of Modular Updates in Integrated Maritime Systems of Systems“. In *Computer Safety, Reliability, and Security*, herausgegeben von António Casimiro, Frank Ortmeier, Friedemann Bitsch, und Pedro Ferreira, 50–63. Lecture Notes in Computer Science. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-54549-9_4.

Handayani, Dini, und Wahju Sediono. 2015. „Anomaly Detection in Vessel Tracking: A Bayesian Networks (BNs) Approach“. *International Journal of Maritime Engineering (RINA Transactions Part A)* 157 (A3): 145–52.

Harati-Mokhtari, Abbas, Alan Wall, Philip Brooks, und Jin Wang. 2007. „Automatic Identification System (AIS): data reliability and human error implications“. *The Journal of Navigation* 60 (3): 373.

Hardt, Dick. 2012. „The OAuth 2.0 authorization framework“. RFC 6749, October.

Hassan, Umair ul, Adegboyega Ojo, und Edward Curry. 2020. „Catalog and Entity Management Service for Internet of Things-Based Smart Environments“. In *Real-Time Linked Dataspaces: Enabling Data Ecosystems for Intelligent Systems*, herausgegeben von Edward Curry, 89–103. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-29665-0_6.

Hausenblas, Michael, und Jacques Nadeau. 2013. „Apache Drill: Interactive Ad-Hoc Analysis at Scale“. *Big Data* 1 (2): 100–104. https://doi.org/10.1089/big.2013.0011.

Heilig, Leonard, Eduardo Lalla-Ruiz, und Stefan Voss. 2017. „Digital transformation in maritime ports: analysis and a game theoretic framework“. *NETNOMICS: Economic Research and Electronic Networking* 18 (Dezember). https://doi.org/10.1007/s11066-017-9122-x.

Herodotou, Herodotos, Sheraz Aslam, Henrik Holm, und Socrates Theodossiou. 2021. „Big Maritime Data Management“. In *Maritime Informatics*, 313–34. Springer.

Hexeberg, Simen, Andreas L. Flaten, Bjorn-Olav H. Eriksen, und Edmund F. Brekke. 2017. „AIS-based vessel trajectory prediction“. In *2017 20th International Conference on Information Fusion (Fusion)*, 1–8. Xi’an, China: IEEE. https://doi.org/10.23919/ICIF.2017.8009762.

Hey, Anthony JG, Stewart Tansley, und Kristin Michele Tolle. 2009. *The fourth paradigm: data-intensive scientific discovery*. Bd. 1. Microsoft research Redmond, WA.

Hey, Tony, und Anne Trefethen. 2003. „e-Science and its implications“. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 361 (1809): 1809–25.

Hoffmann Pham, Katherine, Jeremy Boy, und Miguel Luengo-Oroz. 2018. „Data Fusion to Describe and Quantify Search and Rescue Operations in the Mediterranean Sea“. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, 514–23. https://doi.org/10.1109/DSAA.2018.00066.

Hoque, Mohammad Aminul, und Ragib Hasan. 2020. „VFbed: An Architecture for Testbed-as-a-Service for Vehicular Fog-based Systems“. In *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, 1–6. https://doi.org/10.1109/WF-IoT48130.2020.9221384.

Hossain, Mahmud, Shahid Noor, Yasser Karim, und Ragib Hasan. 2017. „IoTbed: A Generic Architecture for Testbed as a Service for Internet of Things-Based Systems“. In *2017 IEEE International Congress on Internet of Things (ICIOT)*, 42–49. https://doi.org/10.1109/IEEE.ICIOT.2017.14.

Hugoson, Mats-Åke. 2009. „Centralized versus Decentralized Information Systems“. In *History of Nordic Computing 2*, herausgegeben von John Impagliazzo, Timo Järvi, und Petri Paju, 106–15. IFIP Advances in Information and Communication Technology. Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-642-03757-3_11.

Hummel, Patrik, Matthias Braun, Max Tretter, und Peter Dabrock. 2021. „Data Sovereignty: A Review“. *Big Data & Society* 8 (1). https://doi.org/10.1177/2053951720982012.

Humphrey, Chuck. 2006. „e-Science and the Life Cycle of Research“. https://doi.org/10.7939/R3NR4V.

IALA. 2020. „IALA Guideline 1157: Web Service Based S-100 Data Exchange“. International Association of Marine Aids to Navigation and Lighthouse Authorities. https://www.iala-aism.org/product/g1157/.

———. 2021. „IALA Guideline 1128: The Specification of e-Navigation Technical Services“. International Association of Marine Aids to Navigation and Lighthouse Authorities. https://www.iala-aism.org/product/g1128/.

———. o. J. „E-Navigation Testbeds and FAQ“. IALA AISM. Zugegriffen 26. August 2021. https://www.iala-aism.org/technical/e-nav-testbeds/.

IEC TC80. 2022. „IEC 63173-2:2022: Maritime navigation and radiocommunication equipment and systems - Data interfaces - Part 2: Secure communication between ship and shore (SECOM)“. IEC. https://webstore.iec.ch/publication/64543.

IEEE. 1990. „IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries“. *IEEE Std 610*, 1–217. https://doi.org/10.1109/IEEESTD.1991.106963.

IMO. 1974. „International Convention for the Safety of Life at Sea (SOLAS)“. International Maritime Organization. https://treaties.un.org/doc/Publication/UNTS/Volume%201184/volume-1184-I-18961-English.pdf.

Íñiguez, Luis, und Mikel Galar. 2022. „A Scalable and Flexible Open Source Big Data Architecture for Small and Medium-Sized Enterprises“. In *16th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2021)*, herausgegeben von Hugo Sanjurjo González, Iker Pastor López, Pablo García Bringas, Héctor Quintián, und Emilio Corchado, 273–82. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-87869-6_26.

International Telecommunications Union. 2009. „Recommendation ITU-T Y.2720“. ITU-T. https://www.itu.int/rec/T-REC-Y.2720-200901-I.

ISO. 2014. „DIN EN ISO 19157:2014-04, Geoinformation- Datenqualität (ISO_19157:2013); Englische Fassung EN_ISO_19157:2013“. Beuth Verlag GmbH. https://doi.org/10.31030/2090178.

———. 2018. „ISO 26262: Road vehicles - Functional Safety“. International Organization for Standardization. https://www.iso.org/standard/68383.html.

———. 2022. „ISO 21448: Road vehicles - Safety of the Intended Functionality“. International Organization for Standardization. https://www.iso.org/standard/68383.html.

ISO/IEC/IEEE. 2010. „ISO/IEC/IEEE International Standard - Systems and software engineering – Vocabulary“. *ISO/IEC/IEEE 24765:2010(E)*, Dezember, 1–418. https://doi.org/10.1109/IEEESTD.2010.5733835.

ITU-R. 2014. „Rec. ITU-R M.1371-5: Technical Characteristics for an Automatic Identification System Using Time Division Multiple Access in the VHF Maritime Mobile Frequency Band“. https://www.itu.int/dms_pubrec/itu-r/rec/m/R-REC-M.1371-5-201402-I!!PDF-E.pdf.

Ivanov, Todor, und Rekha Singhal. 2018. „ABench: Big Data Architecture Stack Benchmark“. In *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering*, 13–16. ICPE ’18. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3185768.3186300.

Jafarzadeh, Sepideh, und Ingrid Schjølberg. 2018. „Operational Profiles of Ships in Norwegian Waters: An Activity-Based Approach to Assess the Benefits of Hybrid and Electric Propulsion“. *Transportation Research Part D: Transport and Environment* 65 (Dezember): 500–523. https://doi.org/10.1016/j.trd.2018.09.021.

Jankowski, Nicholas W. 2007. „Exploring e-science: An introduction“. *Journal of Computer-Mediated Communication* 12 (2): 549–62.

Jarke, M., und C. Quix. 2017. „On Warehouses, Lakes, and Spaces: The Changing Role of Conceptual Modeling for Data Integration“. In *Conceptual Modeling Perspectives*. https://doi.org/10.1007/978-3-319-67271-7_16.

Ji, Shouling, Weiqing Li, Mudhakar Srivatsa, und Raheem Beyah. 2014. „Structural data de-anonymization: Quantification, practice, and implications“. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 1040–53.

Jo, Sung-Woong, und Woo-Seong Shim. 2019. „LTE-maritime: High-speed maritime wireless communication based on LTE technology“. *IEEE Access* 7: 53172–81.

Jøsang, Audun, John Fabre, Brian Hay, James Dalziel, und Simon Pope. 2005. „Trust requirements in identity management“. In *Proceedings of the 2005 Australasian workshop on Grid computing and e-research-Volume 44*, 99–108. Citeseer.

Jurczyk, Pawel, und Li Xiong. 2009. „Distributed anonymization: Achieving privacy for both data subjects and data providers“. In *Data and Applications Security XXIII*, 191–207. Springer.

Kaisler, Stephen, Frank Armour, J Alberto Espinosa, und William Money. 2013. „Big data: Issues and challenges moving forward“. In *2013 46th Hawaii International Conference on System Sciences*, 995–1004. IEEE.

Kalipe, Godson Koffi, und Rajat Kumar Behera. 2019. „Big Data Architectures: A detailed and application oriented review“. *Int. Journal Innov. Technol. Explor. Eng* 8: 2182–90.

Kazman, Rick, Len Bass, Gregory Abowd, und Mike Webb. 1994. „SAAM: A method for analyzing the properties of software architectures“. In *Proceedings of 16th International Conference on Software Engineering*, 81–90. IEEE.

Khatri, Vijay, und Carol V. Brown. 2010. „Designing Data Governance“. *Commun. ACM* 53 (1): 148–52. https://doi.org/10.1145/1629175.1629210.

Kim, Sungil, Heeyoung Kim, und Yongro Park. 2017. „Early Detection of Vessel Delays Using Combined Historical and Real-Time Information“. *Journal of the Operational Research Society* 68 (2): 182–91. https://doi.org/10.1057/s41274-016-0104-4.

Klitzke, Lars, C. Koch, Andreas Haja, und F. Köster. 2019. „Real-world Test Drive Vehicle Data Management System for Validation of Automated Driving Systems“. In *VEHITS*. https://doi.org/10.5220/0007720501710180.

Kohlmayer, Florian, Fabian Prasser, Claudia Eckert, und Klaus A. Kuhn. 2014. „A Flexible Approach to Distributed Data Anonymization“. *Journal of Biomedical Informatics*, Special Issue on Informatics Methods in Medical Privacy, 50 (August): 62–76. https://doi.org/10.1016/j.jbi.2013.12.002.

Koppe, Roland, und Angela Schäfer. 2015. „Enabling central access to marine data: Data portal German marine research“. In *OCEANS 2015 - Genova*, 1–5. https://doi.org/10.1109/OCEANS-Genova.2015.7271507.

Krishnan, S. P. T., und Jose L. Ugia Gonzalez. 2015. „Google Cloud Pub/Sub“. In *Building Your Next Big Thing with Google Cloud Platform: A Guide for Developers and Enterprise Architects*, herausgegeben von S. P. T. Krishnan und Jose L. Ugia Gonzalez, 277–92. Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-1004-8_12.

Lalla-Ruiz, Eduardo, Belén Melián-Batista, und J. Marcos Moreno-Vega. 2012. „Artificial Intelligence Hybrid Heuristic Based on Tabu Search for the Dynamic Berth Allocation Problem“. *Engineering Applications of Artificial Intelligence* 25 (6): 1132–41. https://doi.org/10.1016/j.engappai.2012.06.001.

Lam, Jasmine Siu Lee, und Xiunian Zhang. 2019. „Innovative Solutions for Enhancing Customer Value in Liner Shipping“. *Transport Policy* 82 (Oktober): 88–95. https://doi.org/10.1016/j.tranpol.2018.09.001.

Lamm, Arne, und Axel Hahn. 2018. „Towards Critical-Scenario Based Testing With Maritime Observation Data“. In *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO)*. https://doi.org/10.1109/OCEANSKOBE.2018.8559045.

———. 2019. „Statistical Maneuver Net Generation for Anomaly Detection in Navigational Waterways“. In *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*. https://doi.org/10.1109/CoDIT.2019.8820641.

Langley, Richard B. 1995. „NMEA 0183: A GPS receiver interface standard“. *GPS world* 6 (7).

Last, Philipp, Christian Bahlke, Martin Hering-Bertram, und Lars Linsen. 2014. „Comprehensive Analysis of Automatic Identification System (AIS) Data in Regard to Vessel Movement Prediction“. *Journal of Navigation* 67 (5): 791–809. https://doi.org/10.1017/S0373463314000253.

Le Dantec, C.A., K.E. Watkins, R. Clark, und E. Mynatt. 2015. „Cycle atlanta and OneBusAway: Driving innovation through the data ecosystems of civic computing“. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9171: 327–38. https://doi.org/10.1007/978-3-319-21006-3_32.

Le Noac'H, Paul, Alexandru Costan, und Luc Bougé. 2017. „A performance evaluation of Apache Kafka in support of big data streaming applications“. In *2017 IEEE international conference on big data (Big Data)*, 4803–6. IEEE.

Lee, Habin, Nursen Aydin, Youngseok Choi, Saowanit Lekhavat, und Zahir Irani. 2018. „A Decision Support System for Vessel Speed Decision in Maritime Logistics Using Weather Archive Big Data“. *Computers & Operations Research* 98 (Oktober): 330–42. https://doi.org/10.1016/j.cor.2017.06.005.

Lee, Paul Tae-Woo, Sung-Woo Lee, Zhi-Hua Hu, Kyoung-Suk Choi, Na Young Hwan Choi, und Sung-Ho Shin. 2018. „Promoting Korean international trade in the East Sea Economic Rim in the context of the Belt and Road Initiative“. *Journal of Korea Trade* 22 (3): 212–27. https://doi.org/10.1108/JKT-03-2018-0015.

Lenzerini, Maurizio. 2002. „Data integration: A theoretical perspective“. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 233–46.

Li, Xiaohuan, Xumin Huang, Chunhai Lia, und Lei Shu. 2019. „EdgeCare Leveraging Edge Computing for Collaborative Data Management in Mobile Healthcare Systems“. *IEEE Access* Volume 7 (Februar): 22011–25. https://doi.org/10.1109/ACCESS.2019.2898265.

Liang, Fan, Wei Yu, Dou An, Qingyu Yang, Xinwen Fu, und Wei Zhao. 2018. „A survey on big data market: Pricing, trading and protection“. *IEEE Access* 6: 15132–54.

Liang, X., J. Zhao, S. Shetty, J. Liu, und D. Li. 2017. „Integrating blockchain for data sharing and collaboration in mobile healthcare applications“. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 1–5. https://doi.org/10.1109/PIMRC.2017.8292361.

Lin, Shih-Chieh, Yunqi Zhang, Chang-Hong Hsu, Matt Skach, Md E. Haque, Lingjia Tang, und Jason Mars. 2018. „The Architectural Implications of Autonomous Driving: Constraints and

Acceleration“. In *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, 751–66. ASPLOS ’18. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3173162.3173191.

Lind, M., S. Haraldson, M. Karlsson, A. Zerem, E. Olsson, N. Mellegård, G. Eriksson, u. a. 2015. „STM Validation Deliverable 1.1 - Enabling port optimization by a digital collaborative platform“. https://s3-eu-west-1.amazonaws.com/stm-stmvalidation/uploads/20190402152729/STMVal_D1.1-Enabling-port-optimization-by-a-digital-collaborative-platform.pdf.

Liu, B., X. L. Yu, S. Chen, X. Xu, und L. Zhu. 2017. „Blockchain Based Data Integrity Service Framework for IoT Data“. In *2017 IEEE International Conference on Web Services (ICWS)*, 468–75. https://doi.org/10.1109/ICWS.2017.54.

Liu, Wen. 2018. „A robust GA/PSO-hybrid Algorithm in intelligent shipping route planning systems for maritime traffic networks“. *Journal of Internet Technology* 19 (Dezember). https://doi.org/10.3966/160792642018111906001.

Lutterotti, Paolo, Giovanni Pau, Daniel Jiang, Mario Gerla, und Luca Delgrossi. 2008. „C-vet, the UCLA vehicular testbed: An open platform for vehicular networking and urban sensing“. In *International Conference on Wireless Access for Vehicular Environments (WAVE 2008)*. Bd. 182.

Mao, R., H. Xu, W. Wu, J. Li, Y. Li, und M. Lu. 2015. „Overcoming the challenge of variety: big data abstraction, the next evolution of data management for AAL communication systems“. *IEEE Communications Magazine* 53 (1): 42–47. https://doi.org/10.1109/MCOM.2015.7010514.

Maritime Connectivity Platform Consortium. 2021a. „MCC Identity Management and Security: General Approach and Basic Requirements“. https://maritimeconnectivity.github.io/maritimeconnectivity.net/docs/MCP%20Gen4%20Requirements%20for%20MCP%20identity%20service%20providers.pdf.

———. 2021b. „Requirements for MCP identity service providers“. https://maritimeconnectivity.github.io/maritimeconnectivity.net/docs/MCP%20Gen4%20Requirements%20for%20MCP%20identity%20service%20providers.pdf.

Marquez-Barja, Johann, Bart Lannoo, Bart Braem, Carlos Donato, Vasilis Maglogiannis, Siegfried Mercelis, Raf Berkvens, u. a. 2019. „Smart Highway: ITS-G5 and C-V2X based testbed for vehicular communications in real environments enhanced by edge/cloud technologies“. In *28th European Conference on Networks and Communications (EuCNC)*. Valencia, Spain.

Martins, Ricardo, Joao Borges de Sousa, Renato Caldas, Chiara Petrioli, und John Potter. 2014. „SUNRISE project: Porto university testbed“. In *2014 Underwater Communications and Networking (UComms)*, 1–5. IEEE.

Mascaro, Steven, Ann E. Nicholso, und Kevin B. Korb. 2014. „Anomaly Detection in Vessel Tracks Using Bayesian Networks“. *International Journal of Approximate Reasoning*, Applications of Bayesian Networks, 55 (1, Part 1): 84–98. https://doi.org/10.1016/j.ijar.2013.03.012.

McAfee, Andrew, und Erik Brynjolfsson. 2012. „Big data: the management revolution“. *Harvard business review* 90 (10): 60–68.

Millefiori, Leonardo M., Dimitrios Zissis, Luca Cazzanti, und Gianfranco Arcieri. 2016. „A distributed approach to estimating sea port operational regions from lots of AIS data“. In *2016 IEEE International Conference on Big Data (Big Data)*, 1627–32. https://doi.org/10.1109/BigData.2016.7840774.

Miller, Darrel, Jeremy Whitlock, Marsh Gardiner, Mike Ralphson, Ron Ratovsky, Uri Sarid, Jason Harmon, und Tony Tam. 2021. „OpenAPI Specification v3.1.0 | Introduction, Definitions, & More“. 15. Februar 2021. https://spec.openapis.org/oas/latest.html.

Mirović, Maris, Mario Miličević, und Ines Obradović. 2018. „Big data in the maritime industry“. *NAŠE MORE: znanstveni časopis za more i pomorstvo* 65 (1): 56–62.

Möller, Julius, Sibylle Fröschle, und Axel Hahn. 2021. „Permissioned Blockchain for Data Provenance in Scientific Data Management“. In *Innovation Through Information Systems*, 22–38. Springer, Cham. https://doi.org/10.1007/978-3-030-86800-0_2.

Möller, Julius, Dennis Jankowski, und Axel Hahn. 2021. „Towards an Architecture to Support Data Access in Research Data Spaces“. In *2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI)*, 317. Las Vegas, NV, USA: IEEE. https://doi.org/10.1109/IRI51335.2021.00049.

Möller, Julius, Dennis Jankowski, Arne Lamm, und Axel Hahn. 2022. „Data Management Architecture for Service-Oriented Maritime Testbeds“. *IEEE Open Journal of Intelligent Transportation Systems* 3: 631–49. https://doi.org/10.1109/OJITS.2022.3207235.

Molter, Georg. 1999. „Integrating SAAM in domain-centric and reuse-based development processes“. In *Proceedings of the 2nd Nordic Workshop on Software Architecture, Ronneby*, 1–10. Citeseer.

Monino, Jean-Louis. 2021. „Data Value, Big Data Analytics, and Decision-Making“. *Journal of the Knowledge Economy* 12 (1): 256–67. https://doi.org/10.1007/s13132-016-0396-2.

Moreau, Luc, Ben Clifford, Juliana Freire, Joe Futrelle, Yolanda Gil, Paul Groth, Natalia Kwasnikowska, Simon Miles, Paolo Missier, und Jim Myers. 2011. „The open provenance model core specification (v1. 1)“. *Future generation computer systems* 27 (6): 743–56.

Mosley, Mark, Michael H Brackett, Susan Earley, und Deborah Henderson. 2009. *DAMA guide to the data management body of knowledge*. Technics Publications.

Munim, Ziaul Haque, Mariia Dushenko, Veronica Jaramillo Jimenez, Mohammad Hassan Shakil, und Marius Imset. 2020. „Big data and artificial intelligence in the maritime industry: a bibliometric review and future research directions“. *Maritime Policy & Management* 47 (5): 577–97. https://doi.org/10.1080/03088839.2020.1788731.

Munoz-Arcentales, A., S. Lopez-Pernas, A. Pozo, A. Alonso, J. Salvachua, und G. Huecas. 2019. „An architecture for providing data usage and access control in data sharing ecosystems“. In *Procedia Computer Science*, 160:590–97. Coimbra, Portugal. https://doi.org/10.1016/j.procs.2019.11.042.

Murray, Brian, und Lokukaluge Prasad Perera. 2020. „A dual linear autoencoder approach for vessel trajectory prediction using historical AIS data“. *Ocean Engineering* 209: 107478.

Nesheim, Dag Atle, Karin Bernsmed, Bjørn Zernichow, Ørnulf Rødseth, und Per Håkon Meland. 2021. „Secure, Trustworthy and Efficient Information Exchange - Enabling Added Value through The Maritime Data Space and Public Key Infrastructure“. In *20th International Conference on Computer and IT Applications in the Maritime Industries*. Mülheim, Germany.

Neurohr, Christian, Lukas Westhofen, Tabea Henning, Thies de Graaff, Eike Möhlmann, und Eckard Böde. 2020. *Fundamental Considerations around Scenario-Based Testing for Automated Driving*.

Nickovic, Dejan, Wolfgang Herzner, Eike Möhlmann, Sebastian Gerwinn, Gustavo Garcia Padilla, Fabian Diegmann, Sadri Hakki, und Colin Snook. 2018. „Enable S3: D3.2.2 v2 V&V Methodology“.

NOAA Office for Coastal Management. 2022. „MarineCadastre.gov | Vessel Traffic Data“. 12. Juni 2022. https://marinecadastre.gov/ais/.

Oliveira, Marcelo Iury S., und Bernadette Farias Lóscio. 2018. „What is a data ecosystem?“ In *Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age*, 1–9.

Oliveira, M.I.S., L.E.R. de Alencar Oliveira, G.F.A. Barros Lima, und B.F. Lóscio. 2017. „A platform for supporting open data ecosystems“. *Lecture Notes in Business Information Processing* 291: 313–37. https://doi.org/10.1007/978-3-319-62386-3_15.

Open Container Initiative. 2022. „Open Container Initiative - Image Specification“. 9. Mai 2022. https://github.com/opencontainers/image-spec/blob/main/spec.md.

Otto, Boris. 2022. „The Evolution of Data Spaces“. In *Designing Data Spaces : The Ecosystem Approach to Competitive Advantage*, herausgegeben von Boris Otto, Michael ten Hompel, und Stefan Wrobel, 3–15. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-93975-5_1.

Otto, Boris, Sebastian Steinbuß, und et al. 2019. „International Data Space - Reference Architecture Model“. International Data Spaces Association. https://www.internationaldataspaces.org/ressource-hub/publications-ids/.

Otto, Boris, und Kristin Weber. 2011. „Data governance“. In *Daten-und Informationsqualität*, 277–95. Springer.

Pampus, Julia. 2022. „Eclipse Dataspace Components“. Eclipse Dataspace Components. 29. August 2022. https://eclipse-dataspaceconnector.github.io/docs/#/README.

Pampus, Julia, Brian-Frederik Jahnke, und Ronja Quensel. 2022. „Evolving Data Space Technologies: Lessons Learned from an IDS Connector Reference Implementation“. In *Leveraging Applications of Formal Methods, Verification and Validation. Practice*, herausgegeben von Tiziana Margaria und Bernhard Steffen, 366–81. Lecture Notes in Computer Science. Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-19762-8_27.

Park, Jinwan, Jungsik Jeong, und Youngsoo Park. 2021. „Ship Trajectory Prediction Based on Bi-LSTM Using Spectral-Clustered AIS Data“. *Journal of Marine Science and Engineering* 9 (9): 1037. https://doi.org/10.3390/jmse9091037.

Parveen, Rizwan, und Tanishq Nandan. 2022. „Developing a Testing Framework for Cyber-Physical Systems using Gazebo“. In *2022 IEEE Workshop on Design Automation for CPS and IoT (DESTION)*, 50–56. https://doi.org/10.1109/DESTION56136.2022.00015.

Peffers, Ken, Tuure Tuunanen, Marcus A. Rothenberger, und Samir Chatterjee. 2007. „A Design Science Research Methodology for Information Systems Research“. *Journal of Management Information Systems* 24 (3): 45–77. https://doi.org/10.2753/MIS0742-1222240302.

PEGASUS. 2019. „Basics For Testing - Booth No. 15: Scenario Database“. *PEGASUS Abschlussveranstaltung & Symposium*, 5.

Perera, Lokukaluge, und Brage Mo. 2016. *Marine Engine Operating Regions under Principal Component Analysis to evaluate Ship Performance and Navigation Behavior*. *IFAC-PapersOnLine*. Bd. 49. https://doi.org/10.1016/j.ifacol.2016.10.487.

Peterson, Zachary, Mark Gondree, und Robert Beverly. 2011. „A Position Paper on Data Sovereignty: The Importance of Geolocating Data in the Cloud“. In *HotCloud'11: Proceedings of the 3rd USENIX conference on Hot topics in cloud computing*.

Pettenpohl, Heinrich, Markus Spiekermann, und Jan Ruben Both. 2022. „International Data Spaces in a Nutshell“. In *Designing Data Spaces : The Ecosystem Approach to Competitive Advantage*, herausgegeben von Boris Otto, Michael ten Hompel, und Stefan Wrobel, 29–40. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-93975-5_3.

Pietrzykowski, Zbigniew, und Janusz Uriasz. 2009. „The Ship Domain – A Criterion of Navigational Safety Assessment in an Open Sea Area“. *The Journal of Navigation* 62 (1): 93–108. https://doi.org/10.1017/S0373463308005018.

Popić, Sr\djan, Dražen Pezer, Bojan Mrazovac, und Nikola Teslić. 2016. „Performance evaluation of using Protocol Buffers in the Internet of Things communication“. In *2016 International Conference on Smart Systems and Technologies (SST)*, 261–65. IEEE.

Port of Rotterdam. o. J. „Consent for Use of AIS Data“. Port of Rotterdam. Zugegriffen 19. Juli 2021. https://www.portofrotterdam.com/en/inland-shipping/use-inland-vessel-ais-data/consent-use-ais-data.

Provost, Foster, und Tom Fawcett. 2013. „Data science and its relationship to big data and data-driven decision making“. *Big data* 1 (1): 51–59.

Pütz, Andreas, Adrian Zlocki, Julian Bock, und Lutz Eckstein. 2017. „System validation of highly automated vehicles with a database of relevant traffic scenarios“. In *12th ITS European Congress*, 1:E5. Strasbourg, France. https://www.pegasusprojekt.de/files/tmpl/pdf/12th%20ITS%20European%20Congress_Folien.pdf.

Qin, Jian, Alexander Ball, und Jane Greenberg. 2012. „Functional and Architectural Requirements for Metadata: Supporting Discovery and Management of Scientific Data“. *Proceedings of the International Conference on Dublin Core and Metadata Applications*, Januar.

Raj, Vinay, und Ravichandra Sadam. 2021. „Performance and complexity comparison of service oriented architecture and microservices architecture“. *International Journal of Communication Networks and Distributed Systems* 27 (1): 100–117. https://doi.org/10.1504/IJCNDS.2021.116463.

Rajasekar, A.K., und R.W. Moore. 2001. „Data and metadata collections for scientific applications“. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 2110: 72–80.

Reiher, David, und Axel Hahn. 2021. „Review on the Current State of Scenario-and Simulation-Based V&V in Application for Maritime Traffic Systems“. In *OCEANS 2021: San Diego–Porto*, 1–9. IEEE.

Reimer, Tim, Phil Abraham, und Qing Tan. 2013. „Federated identity access broker pattern for cloud computing“. In *2013 16th International Conference on Network-Based Information Systems*, 134–40. IEEE.

Richards, Mark. 2015. *Software Architecture Patterns: Understanding Common Architecture Patterns and When to Use Them*. Sebastopol, CA, USA: O'Reilly Media, Inc.

Richer, Justin, und Antonio Sanso. 2017. *OAuth 2 in Action*. Manning Publications.

Roh, Yuji, Geon Heo, und Steven Euijong Whang. 2021. „A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective“. *IEEE Transactions on Knowledge and Data Engineering* 33 (4): 1328–47. https://doi.org/10.1109/TKDE.2019.2946162.

Rousidis, Dimitris, Emmanouel Garoufallou, Panos Balatsoukas, und Miguel-Angel Sicilia. 2014. „Metadata for Big Data: a preliminary investigation of metadata quality issues in research data repositories“. *Information services & use* 34 (3–4): 279–86.

Rubin, Victoria, und Tatiana Lukoianova. 2013. „Veracity roadmap: Is big data objective, truthful and credible?“ *Advances in Classification Research Online* 24 (1): 4.

Ruiz, Alejandra, Mehrdad Sabetzadeh, und Paolo Panaroni. 2011. „Challenges for an open and evolutionary approach to safety assurance and certification of safety-critical systems“. In *2011 First International Workshop on Software Certification*, 1–6. IEEE.

Rusche, Christian. 2022. „Einführung in Gaia-X – Hintergrund, Ziele und Aufbau“. *Rusche IW-Report*, Nr. 10 (März). https://www.iwkoeln.de/studien/christian-rusche-einfuehrung-in-gaia-x-hintergrund-ziele-und-aufbau.html.

Russell, Stuart J, und Peter Norvig. 2010. *Artificial Intelligence : A Modern Approach*. 3. ed. Prentice-Hall Series in Artificial Intelligence. Upper Saddle River, New Jersey 07458.: Prentice Hall.

Rüssmeier, N., A. Lamm, und A. Hahn. 2019. „A Generic Testbed for Simulation and Physical-Based Testing of Maritime Cyber-Physical System of Systems“. *Journal of Physics: Conference Series* 1357 (1): 012025. https://doi.org/10.1088/1742-6596/1357/1/012025.

Saad, M.I.M., K.A. Jalil, und M. Manaf. 2014. „Achieving trust in cloud computing using secure data provenance“. In *2014 IEEE Conference on Open Systems (ICOS)*, 84–88. https://doi.org/10.1109/ICOS.2014.7042634.

Sala, Enric, Juan Mayorga, Christopher Costello, David Kroodsma, M L D Palomares, Daniel Pauly, Rashid Sumaila, und Dirk Zeller. 2018. „The economics of fishing the high seas“. *Science Advances* 4 (Juni): eaat2504. https://doi.org/10.1126/sciadv.aat2504.

Salheddine, Kabou, und Sidi Mohamed Benslimane. 2014. *A new distributed anonymization protocol to satisfy multiple data providers privacy requirements*. *CEUR Workshop Proceedings*. Bd. 1294.

Sanaei, M., M.S. Taslimi, M. AbdolhoseinZadeh, und M.H. Khani. 2019. „A study and analysis of the open government data ecosystem models“. *Iranian Journal of Information Processing Management* 34 (2): 609–36.

Sanchez-Gonzalez, Pedro-Luis, David Díaz-Gutiérrez, Teresa J. Leo, und Luis R. Núñez-Rivas. 2019. „Toward Digitalization of Maritime Transport?“ *Sensors* 19 (4): 926. https://doi.org/10.3390/s19040926.

Sandhu, R.S., und P. Samarati. 1994. „Access control: principle and practice“. *IEEE Communications Magazine* 32 (9): 40–48. https://doi.org/10.1109/35.312842.

Sangiovanni-Vincentelli, Alberto, Werner Damm, und Roberto Passerone. 2012. „Taming Dr. Frankenstein: Contract-based design for cyber-physical systems“. *European journal of control* 18 (3): 217–38.

Santos, Daniel S., Brauner R. N. Oliveira, Rick Kazman, und Elisa Y. Nakagawa. 2022. „Evaluation of Systems-of-Systems Software Architectures: State of the Art and Future Perspectives“. *ACM Computing Surveys*, Februar. https://doi.org/10.1145/3519020.

Sarabia-Jácome, D., C. E. Palau, M. Esteve, und F. Boronat. 2019. „Seaport Data Space for Improving Logistic Maritime Operations“. *IEEE Access* 8: 4372–82. https://doi.org/10.1109/ACCESS.2019.2963283.

Schäuffele, Jörg, und Thomas Zurawka. 2016. *Automotive Software Engineering*. Wiesbaden: Springer Fachmedien. https://doi.org/10.1007/978-3-658-11815-0.

Shahir, Hamed Yaghoubi, Uwe Glasser, Amir Yaghoubi Shahir, und Hans Wehn. 2015. „Maritime situation analysis framework: Vessel interaction classification and anomaly detection“. In *2015 IEEE International Conference on Big Data (Big Data)*, 1279–89. https://doi.org/10.1109/BigData.2015.7363883.

Shakhovska, Natalya, und Yurii Bolubash. 2015. *Dataspace architecture and manage its components class projection*. Bd. Volume 4. ECONTECHMOD.

Shams-ul-Arif, Qadeem Khan, und S. A. K. Gahyyur. 2009. „Requirements engineering processes, tools/technologies, & methodologies“. *International Journal of reviews in computing* 2 (6): 41–56.

Sharvari, T, und K Sowmya Nag. 2019. „A study on Modern Messaging Systems- Kafka, RabbitMQ and NATS Streaming“. arXiv. https://doi.org/10.48550/arXiv.1912.03715.

Shoshani, Arie, und Doron Rotem. 2009. *Scientific data management: challenges, technology, and deployment*. CRC Press.

Sidibé, Abdoulaye, und Gao Shu. 2017. „Study of Automatic Anomalous Behaviour Detection Techniques for Maritime Vessels“. *The Journal of Navigation* 70 (4): 847–58. https://doi.org/10.1017/S0373463317000066.

Silveira, PAM, AP Teixeira, und C Guedes Soares. 2013. „Use of AIS data to characterise marine traffic patterns and ship collision risk off the coast of Portugal“. *The Journal of Navigation* 66 (6): 879–98.

Simmhan, Yogesh L., Beth Plale, und Dennis Gannon. 2005. „A survey of data provenance in e-science“. *ACM Sigmod Record* 34 (3): 31–36.

S. Oliveira, M.I., G.F. Barros Lima, und B. Farias Lóscio. 2019. „Investigations into Data Ecosystems: a systematic mapping study“. *Knowledge and Information Systems* 61 (2): 589–630. https://doi.org/10.1007/s10115-018-1323-6.

Solmaz, Berkan, Erhan Gundogdu, Veysel Yucesoy, Aykut Koc, und A. Alatan. 2018. „Fine-Grained Recognition of Maritime Vessels and Land Vehicles by Deep Feature Embedding“. *IET Computer Vision* 12 (Juli). https://doi.org/10.1049/iet-cvi.2018.5187.

Steidel, Matthias, und Axel Hahn. 2019. „MTCAS -An Assistance System for Collision Avoidance at Sea“. In *18th Conference on Computer and IT Applications in the Maritime Industries (COMPIT)*. Tullamore.

Steidel, Matthias, Arne Lamm, Sebastian Feuerstack, und Axel Hahn. 2019. „Correcting the Destination Information in Automatic Identification System Messages“. In *Workshop on Information Systems and Applications in Maritime Domain (ISAMD 2019)*. Bd. Lecture Notes in Business Information Processing. Seville, Spain: Springer.

Strong, Diane M., Yang W. Lee, und Richard Y. Wang. 1997. „Data quality in context“. *Communications of the ACM* 40 (5): 103–10.

Sumaray, Audie, und S. Kami Makki. 2012. „A comparison of data serialization formats for optimal efficiency on a mobile platform“. In *Proceedings of the 6th international conference on ubiquitous information management and communication*, 1–6.

Sun, Yunchuan, Junsheng Zhang, Yongping Xiong, und Guangyu Zhu. 2014. „Data Security and Privacy in Cloud Computing“. *International Journal of Distributed Sensor Networks* 10 (7): 190903. https://doi.org/10.1155/2014/190903.

Swanson, Kevin S., Alexander A. Brown, Sean N. Brennan, und Cynthia M. LaJambe. 2013. „Extending driving simulator capabilities toward Hardware-in-the-Loop testbeds and remote vehicle interfaces“. In *2013 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*, 115–20. https://doi.org/10.1109/IVWorkshops.2013.6615236.

Sweeney, Latanya. 2002. „k-anonymity: A model for protecting privacy“. *International journal of uncertainty, fuzziness and knowledge-based systems* 10 (05): 557–70.

Taleb, Ikbal, Hadeel El Kassabi, Mohamed Serhani, Rachida Dssouli, und Chafik Bouhaddioui. 2016. *Big Data Quality: A Quality Dimensions Evaluation*. https://doi.org/10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0122.

Tansley, Robert, Mick Bass, David Stuve, Margret Branschofsky, Daniel Chudnov, Greg McClellan, und MacKenzie Smith. 2003. „The DSpace Institutional Digital Repository System: Current Functionality“. In *2003 Joint Conference on Digital Libraries, 2003. Proceedings.* Institute of Electrical and Electronic Engineers. https://doi.org/10.1109/JCDL.2003.1204846.

Tene, Omer, und Jules Polonetsky. 2012. „Privacy in the Age of Big Data: A Time for Big Decisions“. *Stanford Law Review* 64 (64): 63–69.

Terrizzano, Ignacio G, Peter M Schwarz, Mary Roth, und John E Colino. 2015. „Data Wrangling: The Challenging Journey from the Wild to the Lake.“ In *7th Conference on Innovative Data Systems Research 2015*. Asilomar, USA.

Teuscher, Andreas, Gerd Brost, Uwe Brettner, Martin Böhmer, Bastian Fraune, Christian Haas, Tina Hardt, u. a. 2020. „DIN SPEC 27070:2020-03, Anforderungen und Referenzarchitektur eines Security Gateways zum Austausch von Industriedaten und Diensten“. Beuth Verlag GmbH. https://doi.org/10.31030/3139499.

Thomas, Llewellyn DW, und Aija Leiponen. 2016. „Big data commercialization“. *IEEE Engineering Management Review* 44 (2): 74–90.

Tocco, Fabrice, und Laurent Lafaye. 2022. „Data Platform Solutions“. In *Designing Data Spaces : The Ecosystem Approach to Competitive Advantage*, herausgegeben von Boris Otto, Michael ten Hompel, und Stefan Wrobel, 383–93. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-93975-5_23.

Tsou, Ming-Cheng, und Chao-Kuang Hsueh. 2010. „The study of ship collision avoidance route planning by ant colony algorithm“. *Journal of Marine Science and Technology* 18 (Oktober).

Tu, Enmei, Guanghao Zhang, Lily Rachmawati, Eshan Rajabally, und Guang-Bin Huang. 2018. „Exploiting AIS Data for Intelligent Maritime Navigation: A Comprehensive Survey From Data to Methodology“. *IEEE Transactions on Intelligent Transportation Systems* 19 (5): 1559–82. https://doi.org/10.1109/TITS.2017.2724551.

Ulbrich, Simon, Till Menzel, Andreas Reschka, Fabian Schuldt, und Markus Maurer. 2015. „Definition der Begriffe Szene, Situation und Szenario für das automatisierte Fahren“. In *Fahrerassistenzworkshop Walting*. Bd. 10. Walting.

U.S. Department of Transportation. 2012. „Connected Vehicle Infrastructure & Components | Safety, Mobility, & Environmental Applications | In-Vehicle Devices“. https://www.its.dot.gov/research_archives/connected_vehicle/pdf/DOT_CVBrochure.pdf.

Valbi, Eleonora, Fabio Ricci, Samuela Capellacci, Silvia Casabianca, Michele Scardi, und Antonella Penna. 2019. „A model predicting the PSP toxic dinoflagellate Alexandrium minutum occurrence in the coastal waters of the NW Adriatic Sea“. *Scientific Reports* 9 (März). https://doi.org/10.1038/s41598-019-40664-w.

Vanura, Jan, und Pavel Kriz. 2018. „Perfomance Evaluation of Java, JavaScript and PHP Serialization Libraries for XML, JSON and Binary Formats“. In *Services Computing – SCC 2018*, herausgegeben von João Eduardo Ferreira, George Spanoudakis, Yutao Ma, und Liang-Jie Zhang, 166–75. Lecture Notes in Computer Science. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-94376-3_11.

Vázquez-Ingelmo, Andrea, Juan Cruz-Benito, und Francisco J García-Peñalvo. 2017. „Improving the OEEU’s data-driven technological ecosystem’s interoperability with GraphQL“. In *Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality*, 89. ACM.

Verykios, Vassilios S., Elisa Bertino, Igor Nai Fovino, Loredana Parasiliti Provenza, Yucel Saygin, und Yannis Theodoridis. 2004. „State-of-the-art in privacy preserving data mining“. *ACM SIGMOD Record* 33 (1): 50–57. https://doi.org/10.1145/974121.974131.

Vohra, Deepak. 2016a. „Apache Kafka“. In *Practical Hadoop Ecosystem: A Definitive Guide to Hadoop-Related Frameworks and Tools*, herausgegeben von Deepak Vohra, 339–47. Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-2199-0_9.

———. 2016b. „Apache parquet“. In *Practical Hadoop Ecosystem*, 325–35. Springer.

Volk, M., D. Staegemann, S. Bosse, R. Haüsler, und K. Turowski. 2020. „Approaching the (Big) data science engineering process“. In *IoTBDS 2020 - Proceedings of the 5th International Conference on Internet of Things, Big Data and Security*, 428–35.

Vouros, George A., Christos Doulkeridis, Georgios Santipantakis, und Akrivi Vlachou. 2018. „Taming Big Maritime Data to Support Analytics“. In *Information Fusion and Intelligent Geographic Information Systems (IF&IGIS'17)*, herausgegeben von Vasily Popovich, Manfred Schrenk, Jean-Claude Thill, Christophe Claramunt, und Tianzhen Wang, 15–27. Lecture Notes in Geoinformation and Cartography. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-59539-9_2.

Wallis, Jillian, Christine Borgman, Matthew Mayernik, Alberto Pepe, R.Anitha Nithya, und Mark Hansen. 2007. „Know Thy Sensor: Trust, Data Quality, and Data Integrity in Scientific Digital Libraries“. In *Research and Advanced Technology for Digital Libraries*, 4675:380–91. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-74851-9_32.

Wang, F., P. Liu, F. Azar, J. Pearson, und G. Madlmayr. 2006. „Experiment management with metadata-based integration for collaborative scientific research“. In *22nd International Conference on Data Engineering (ICDE'06)*, 2006:96. https://doi.org/10.1109/ICDE.2006.65.

Wang, Richard Y., und Diane M. Strong. 1996. „Beyond accuracy: What data quality means to data consumers“. *Journal of management information systems* 12 (4): 5–33.

Wang, Yihan, Shaoxu Song, und Lei Chen. 2016. „A Survey on Accessing Dataspaces“. *ACM SIGMOD Record* 45 (2): 33–44. https://doi.org/10.1145/3003665.3003672.

Ward, Robert, Lee Alexander, Barrie Greenslade, und Anthony Pharaoh. 2008. „IHO S-100: The New Hydrographic Geospatial Standard for Marine Data and Information“. *Canadian Hydrographic Conference*, Mai. https://scholars.unh.edu/ccom/425.

Wei, Te, Wei Feng, Yunfei Chen, Cheng-Xiang Wang, Ning Ge, und Jianhua Lu. 2021. „Hybrid satellite-terrestrial communication networks for the maritime Internet of Things: Key technologies, opportunities, and challenges“. *IEEE Internet of things journal* 8 (11): 8910–34.

Wende, Kristin. 2007. „A model for data governance-Organising accountabilities for data quality management“. *ACIS 2007 Proceedings*, 80.

Wiener, M., F.T. Sommer, Z.G. Ives, R.A. Poldrack, und B. Litt. 2016. „Enabling an Open Data Ecosystem for the Neurosciences“. *Neuron* 92 (3): 617–21. https://doi.org/10.1016/j.neuron.2016.10.037.

Wilkinson, Mark D., Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, und Philip E. Bourne. 2016. „The FAIR Guiding Principles for scientific data management and stewardship“. *Scientific data* 3 (1): 1–9.

Witten, Ian H., Eibe Frank, und Mark A. Hall, Hrsg. 2011. „The WEKA Data Mining Workbench“. In *Data Mining: Practical Machine Learning Tools and Techniques (Third Edition)*, 403–587. The Morgan Kaufmann Series in Data Management Systems. Boston: Morgan Kaufmann. https://doi.org/10.1016/B978-0-12-374856-0.00018-3.

Wu, Han, Zhihao Shang, und Katinka Wolter. 2019. „Performance prediction for the apache kafka messaging system“. In *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 154–61. IEEE.

Xu, Lei, Chunxiao Jiang, Jian Wang, Jian Yuan, und Yong Ren. 2014. „Information security in big data: privacy and data mining“. *IEEE Access* 2: 1149–76.

Yan, Xinping, Kai Wang, Yupeng Yuan, Xiaoli Jiang, und Rudy R. Negenborn. 2018. „Energy-Efficient Shipping: An Application of Big Data Analysis for Optimizing Engine Speed of Inland Ships Considering Multiple Environmental Factors“. *Ocean Engineering* 169 (Dezember): 457–68. https://doi.org/10.1016/j.oceaneng.2018.08.050.

Yang, Wang, Tian Ye, Li Tie-shan, Peng Dongcheng, und Zhou Yihua. 2018. „Visualization analysis of shipping recruitment information based on R“. In *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, 6–10. https://doi.org/10.1109/ICACI.2018.8377582.

Yeo, Gavin, Shiau Hong Lim, Laura Wynter, und Hifaz Hassan. 2019. „MPA-IBM Project SAFER: Sense-Making Analytics for Maritime Event Recognition“. *INFORMS Journal on Applied Analytics* 49 (4): 269–80. https://doi.org/10.1287/inte.2019.0997.

Yim, Jeong-Bin, und Deuk-Jin Park. 2021. „Estimating Critical Latency Affecting Ship's Collision in Re-Mote Maneuvering of Autonomous Ships“. *Applied Sciences* 11 (22): 10987. https://doi.org/10.3390/app112210987.

Zerbino, Pierluigi, Davide Aloini, Riccardo Dulmin, und Valeria Mininno. 2018. „Process-Mining-Enabled Audit of Information Systems: Methodology and an Application“. *Expert Systems with Applications* 110 (November): 80–92. https://doi.org/10.1016/j.eswa.2018.05.030.

———. 2019. „Towards Analytics-Enabled Efficiency Improvements in Maritime Transportation: A Case Study in a Mediterranean Port“. *Sustainability* 11 (August): 4473. https://doi.org/10.3390/su11164473.

Zhang, Daiyong, Jia Li, Qing Wu, Xinglong Liu, Xiumin Chu, und Wei He. 2017. „Enhance the AIS data availability by screening and interpolation“. In *2017 4th International Conference on Transportation Information and Safety (ICTIS)*, 981–86. https://doi.org/10.1109/ICTIS.2017.8047888.

Zhang, Wei, Suren Byna, Chenxu Niu, und Yong Chen. 2019. „Exploring metadata search essentials for scientific data management“. In *2019 IEEE 26th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, 83–92. IEEE.

Zhong, M., M. Liu, und Q. Chen. 2008. „Modeling heterogeneous data in dataspace“. In *2008 IEEE International Conference on Information Reuse and Integration*, 404–9. https://doi.org/10.1109/IRI.2008.4583065.

Zhou, Xin, Xiaodong Gou, Tingting Huang, und Shunkun Yang. 2018. „Review on testing of cyber physical systems: Methods and testbeds“. *IEEE Access* 6: 52179–94.

Zhuang, H., und H.-Y. Lee. 2016. „Data-as-a-service: A cloud-based federated platform to facilitate discovery of private sector datasets“. In *Proceedings - 2015 International Conference on Cloud Computing Research and Innovation, ICCCRI 2015*, 9–14. https://doi.org/10.1109/ICCCRI.2015.34.

Zichichi, Mirko, Michele Contu, Stefano Ferretti, und Víctor Rodríguez-Doncel. 2020. „Ensuring Personal Data Anonymity in Data Marketplaces through Sensing-as-a-Service and Distributed Ledger“. In *3rd Distributed Ledger Technology Workshop Co-located with ITASEC 2020*. Ancona, Italy.

Zrenner, Johannes, Frederik Oliver Möller, Christian Jung, Andreas Eitel, und Boris Otto. 2019. „Usage control architecture options for data sovereignty in business ecosystems“. *Journal of Enterprise Information Management* 32 (3): 477–95.

Zuiderwijk, Anneke, Marijn Janssen, und Chris Davis. 2014. „Innovation with Open Data: Essential Elements of Open Data Ecosystems“. *Information Polity* 19 (1–2): 17–33. https://doi.org/10.3233/IP-140329.

Anhang

A Glossar

3-Phasenmodell des Testfelddatenmanagements. Das 3-Phasenmodell des Testfelddatenmanagements beschreibt die Datenmanagementaufgaben in einem maritimen Testfeld. Die drei Phasen sind das Verwalten einer Datengrundlage für Modellbildung und Wissensgenerierung, das Verwalten von Daten während des Entwicklungsprozesses eines Systems im Rahmen der V+V und das Verwalten von Daten für die Demonstration und Zertifizierung von Systemen (vgl. Abschnitt 2.2.3).

Anfrageorientierte Kappa-Architektur. Die anfrageorientierte Kappa-Architektur ist eine abgewandelte Version der Kappa-Architektur (vgl. Kalipe und Behera 2019), bei der der eingehende Datenstrom nicht statisch erzeugt wird, sondern durch individuelle Daten-Anfragen definiert werden kann.

Connector. Im Rahmen dieser Arbeit wird ein Connector als Software-Artefakt definiert, welches es ermöglicht Datenquellen des Testfeldes über eine standardisierte Schnittstelle an das FEDMS anzubinden. Connectoren werden innerhalb der Infrastruktur eines Datenproviders betrieben und ermöglichen somit die Datensouveränität (siehe auch Abschnitt 4.3.1).

Data Space. Data Spaces sind ein Architekturkonzept zum Management von Daten. Sie können als die Koexistenz einer Menge von dezentral verwalteten Daten definiert werden, die durch ein unterstützendes System verbunden wird. Dieses erweiterbare System unterstützt dabei hauptsächlich die Integration der verschiedenen Datenquellen und verbessert deren Zugänglichkeit (siehe auch Abschnitt 2.3.5). (Franklin, Halevy, und Maier 2005)

Datenartefakt. Ein Datenartefakt ist eine endliche Menge oder ein Strom von Datentupeln (vgl. Gedik, Turaga, und Andrade 2014), die in einem gemeinsamen Zusammenhang genutzt werden.

Datenkonsument. Ein Datenkonsument ist ein Teilnehmer eines Datenökosystems, der Daten von einem oder mehreren Datenprovidern abruft und für seine Zwecke nutzt bzw. weiterverarbeitet (vgl. Abschnitt 2.3.4).

Datenprovider. Ein Datenprovider ist ein Teilnehmer eines Datenökosystems, der Daten für andere Teilnehmer des Datenökosystems generiert und/oder bereitstellt (vgl. Abschnitt 2.3.4).

Datensouveränität. Datensouveränität „kann als die Fähigkeit einer natürlichen oder juristischen Person definiert werden, die volle Kontrolle über ihre Daten zu haben“. (Otto u.a. 2019, 9)

Datenökosystem. Ein Datenökosystem ist ein sozio-technisches Netzwerk (Gelhaar und Otto 2020). Dieses besteht im Allgemeinen aus mehreren Akteuren, die verschiedene Interessen

Fähigkeiten und Anforderungen haben und bestehende Beziehungen nutzen, um Daten auszutauschen, und verteilte Infrastrukturen oder Services zur Datenverarbeitung nutzen (Oliveira und Lóscio 2018). Oft unterliegt dieser Austausch von Ressourcen bestimmten Lizenzen, Standards oder Qualitätsuntersuchungen (Oliveira und Lóscio 2018).

Dezentralität. Mit Dezentralität ist in dieser Arbeit die Eigenschaft gemeint, dass Teilnehmer eines Datenökosystems ihre Daten und Infrastruktur selbst verwalten können und keine zentrale Entität über diese bestimmt. Trotz dessen wird eine Kooperation im Datenökosystem angestrebt, sodass bestimmte Schnittstellen zum Zugriff auf Daten und Infrastruktur durch die Teilnehmer bereitgestellt werden. (Hugoson 2009)

Forschungs- und Entwicklungsdatenmanagementsystem (FEDMS). Das FEDMS ist das System, welches als Hauptteil des Lösungskonzeptes dieser Arbeit entworfen wurde. Es unterstützt die Nutzer eines maritimen Testfeldes bei den Aufgaben des Datenmanagements innerhalb des Testfeldes. Der wesentliche Unterschied zu einem klassischen Forschungsdatenmanagementsystem liegt in seiner Fähigkeit einen Systementwicklungsprozess im Testfeld zu begleiten und die Dezentralität eines Data Spaces abbilden zu können.

Intermediär. Ein Intermediär ist ein Teilnehmer eines Datenökosystems, der den Austausch von Daten zwischen Datenprovider und Datenkonsument vereinfacht. Dies kann z.B. in Form von Infrastrukturbereitstellung, Vertrauensbildung oder Vereinfachen der Auffindbarkeit geschehen (vgl. Abschnitt 2.3.4).

Microservices. Microservices sind ein Trend in der Weiterentwicklung von Service-orientierten Architekturen. Dabei werden Services in kleineren Einheiten und unabhängiger in voneinander separierten Prozessen bereitgestellt. Microservices kommunizieren meist mit leichtgewichtigen Kommunikationsprotokollen und ermöglichen es verschiedene Technologien für einzelne Funktionen einer Gesamtarchitektur zu verwenden. Meist wird Containerisierung eingesetzt, um diese Services flexibel und skalierbar bereitzustellen. (Raj und Sadam 2021)

Schichtenbasierte Architektur. Der Entwurf von schichtenbasierten Architekturen hilft bei der logischen Trennung von Softwareschichten. So entsteht eine klare Verteilung von Verantwortlichkeiten und komplexe Aufgaben können auf einfachere Aufgaben heruntergebrochen werden (Richards 2015). Im Rahmen des Konzeptes dieser Arbeit wurde eine schichtenbasierte Architektur für das FEDMS entwickelt, die aus einer Quelldatenschicht, einer Datenverarbeitungsschicht, einer Datenverteilungsschicht und einer Kontrollschicht besteht.

Szenario-orientiertes Datenmanagement. Das Szenario-orientierte Datenmanagement ist eine Methode, mit der Daten aus verschiedenen Testszenarien eines Systems auf dem FEDMS verwaltet werden können. Dies ermöglicht eine bessere Trennung von Daten einzelner Tests und

vereinfacht den Zugriff für die Datenanalyse, die für eine Testauswertung erforderlich ist (siehe auch Abschnitt 4.4.3).

Testfeld. Ein Testfeld ist „eine Umgebung, die Hardware, Messgeräte, Simulatoren, Software-Werkzeuge und andere unterstützende Elemente enthält, die für die Durchführung eines Tests benötigt werden.“ (ISO/IEC/IEEE 2010, 466).

Testfeld Data Space. Der Testfeld Data Space beschreibt eine Data Space Architektur in einem serviceorientierten maritimen Testfeld. Er umfasst die verschiedenen Stakeholder, ihre Datenquellen, Services, und die genutzte Infrastruktur (inkl. des FEMDS), sowie die Beziehungen zwischen diesen Entitäten und die Prozesse zum Übermitteln von Daten, die in dieser Arbeit diskutiert werden.

Verarbeitungsschritt. Ein Verarbeitungsschritt ist die Anwendung einer Transformation auf eine oder mehrere Datenartefakte durch einen klar definierten Stakeholder im Testfeld, die als Ergebnis ein neues Datenartefakt generiert.

Workflow. Ein Workflow ergibt sich also aus der Gesamtheit aller Datenartefakte in einer Datenverarbeitungskette und den zu den Verarbeitungsschritten zugehörigen Metadaten.

B Literaturanalyse „maritime datengetriebene Forschung“

Referenz	Ziel	Daten	Nutzungsart
(Abdel-Aal, Elhadidy, und Shaahid 2009)	Vorhersage von Winddaten	Stündliche Windgeschwindigkeit	Trainingsdaten
(Adland, Jia, und Strandenes 2017)	Schätzung des (Öl-)Handelsvolumens aus AIS	AIS-Daten, Daten zum Ölhandelsvolumen	Datenanalyse / KDD
(Funabiki u. a. 2017)	Klassifikation Wasserqualität	Wasser(-qualitäts)daten	Datenarchitektur / Trainingsdaten
(Berbić u. a. 2017)	Vorhersage der Significant Wave Height	Daten zur Wellenhöhe	Trainingsdaten
(B. D. Brouer, Karsten, und Pisinger 2016)	Optimierung von Planungsproblemen der Linienschifffahrt	Wirtschaftliche Hafendaten (Kosten etc.)	Evaluation
(B. Brouer, Karsten, und Pisinger 2017)	Optimierung von Planungsproblemen der Linienschifffahrt	Wirtschaftliche Hafendaten (Kosten etc.)	Evaluation
(Cariou 2011)	Analyse von CO2-Emissionen	Daten zum Kraftstoffverbrauch, Daten zu den Betriebskosten des Schiffes, Daten zur Schiffsreise (Kraftstoffverbrauch etc.)	Datenanalyse / KDD
(Cheng u. a. 2018)	Analyse von Öl-Transporten	AIS-Daten	Datenanalyse / KDD
(Coraddu u. a. 2016)	Evaluation von Schiffsantriebsverfahren	Daten aus der Simulation eines Antriebssystems	Traningsdaten

(Coraddu u. a. 2019)	Detektion von “Vessel Fouling”	Motordaten / Kraftstoffverbrauch, Wetterdaten: Wind und Wellen	Trainingsdaten
(Fernández u. a. 2018)	Überwachung von Daten aus Seehäfen	Wetterdaten: Wasserstand, Wellendaten, Windgeschwindigkeit, Niederschlag	Datenarchitektur
(Fletcher u. a. 2018)	Vorhersage gasförmiger Emissionen	Motorleistung, Drehzahlen, Emissionsdaten	Traningsdaten
(Gallego u. a. 2019)	Erkennung von “Oil Spills”	RADAR-Daten (durch Flugzeug erhoben)	Trainingsdaten
(Gambardella, Rizzoli, und Zaffalon 1998)	Containerterminal-Planungsprobleme (Ressourcenzuweisung)	Simulierte und reale Containerterminal-Daten	Evaluation
(Handayani und Sediono 2015)	Anomalieerkennung im Vessel Tracking	AIS-Daten	Trainingsdaten
(Heilig, Lalla-Ruiz, und Voss 2017)	Spieltheoretisches Framework für maritime Digitalisierung	Transportkosten-Daten im Hamburger Hafen (nur als Evaluation eines Modells)	Evaluation
(Jafarzadeh und Schjølberg 2018)	Identifizierung von Schiffstypen, die von Elektro- und Hybridantrieben profitieren können	AIS-Daten, zusätzliche Schiffsdaten, (See-)Karten	Datenanalyse / KDD
(Kim, Kim, und Park 2017)	Verspätungen von Schiffen vorhersagen	AIS-Daten (Echtzeit und historisch), Frachtbriefe	Trainingsdaten
(Lalla-Ruiz, Melián-Batista, und Marcos Moreno-Vega 2012)	Planungsproblem: Terminierung von Schiffsliegeplätzen	Generierte Daten für Schiffsanlegeprobleme	Evaluation
(Lam und Zhang 2019)	Kundenwerte in der Linienschifffahrt verbessern	Experteninterviews	Evaluation
(H. Lee u. a. 2018)	Verwendung von Wetterdaten zur Schätzung des tatsächlichen Kraftstoffverbrauchs für die Geschwindigkeitsoptimierung	Wellen-, Strömungs- und Winddaten, Reedereidaten (Treibstoffverbrauch, Geschwindigkeit, Entfernungen)	Data Architecture + KDD / Datenanalyse
(P. T.-W. Lee u. a. 2018)	Analyse der Schifffahrtsnetze in Korea	Hauptsächlich AIS-Daten, angereichert mit anderen Daten (nicht genauer beschrieben)	Datenanalyse / KDD
(W. Liu 2018)	Schiffsrouting	Kartendaten	Evaluation
(Mascaro, Nicholso, und Korb 2014)	Anomalieerkennung in AIS Daten	AIS-Daten, Wetterdaten (Temperatur, Bewölkung, Windgeschwindigkeit), Schiffsinformationen,	Trainingsdaten
(Millefiori u. a. 2016)	Schätzung der Betriebsregionen von Seehäfen	AIS-Daten	Datenanalyse / KDD
(Perera und Mo 2016)	Bewertung von Schiffsleistung und Navigationsverhalten	Schiffsleistungsdaten (Kraftstoff, Motor usw.), Navigationsinformationen	Datenanalyse / KDD
(Hoffmann Pham, Boy, und Luengo-Oroz 2018)	Analyse von Rettungseinsätzen im Mittelmeer	AIS, Broadcast-Warnungen (wie NAVTEX), Tweets, Berichte	Datenanalyse / KDD und Trainingsdaten
(Pietrzykowski und Uriasz 2009)	Herleitung von Ship Domain Modelen	Experteninterviews	Evaluation / Datenanalyse

(Sala u. a. 2018)	Analyse der Wirtschaftlichkeit der Hochseefischerei	Globale Fischereiüberwachungsdatenbank (basierend auf AIS- und Schiffsüberwachungsdaten), Satellitendaten	Datenanalyse / KDD + ML(?)
(Shahir u. a. 2015)	Anomalieerkennung	AIS-Daten	Trainingsdaten
(Solmaz u. a. 2018)	Erkennung von Seeschiffen und Landfahrzeugen in Bildern	Bilddaten einschließlich Metadaten zu Schiffen (+ Datensatz zu Fahrzeugen)	Trainingsdaten
(Tsou und Hsueh 2010)	Kollisionsvermeidende Routenplanung	Simulierte Begegnungssituationen	Evaluation
(Valbi u. a. 2019)	Vorhersage von Algenvorkommen	Molekularanalyse von Meerwasserproben, Wetterdaten	Trainingsdaten
(Yan u. a. 2018)	Optimierung der Motordrehzahl von Binnenschiffen	Motordaten, Schiffsdynamikdaten, Wetterdaten (Windgeschwindigkeit, Windrichtung, Wassertiefe und Wassergeschwindigkeit),	KDD / Trainingsdaten
(Yang u. a. 2018)	Visuelle Analyse von Informationen zur Einstellung von Seeleuten	Informationen zur Einstellung von Seeleuten	Datenanalyse / KDD
(Yeo u. a. 2019)	Generisches Analyse-Framework für den Seeverkehr	AIS-Daten, RADAR-systeme und operative Systeme, z. B. Hafenverkehrsmanagementsystem und Schiffsabfertigungssystem (Clearance of Vessel System)	Datenarchitektur / KDD / Datenanalyse
(Zerbino u. a. 2018)	Process-Mining-basierte Prüfung von Informationssystemen am Beispiel eines Port Community Systems	Prozessmodelldaten aus der Port Community	Evaluation
(Zerbino u. a. 2019)	Ermöglichung analytikgestützter Verbesserungen der Effizienz von Hafentransportprozessen mit Process Mining	Port Community Prozessdaten (Ereignisse, Status, Schiffskennung, Route)	Datenanalyse / KDD / Process-Mining

C UML-Diagramme der prototypischen Umsetzung

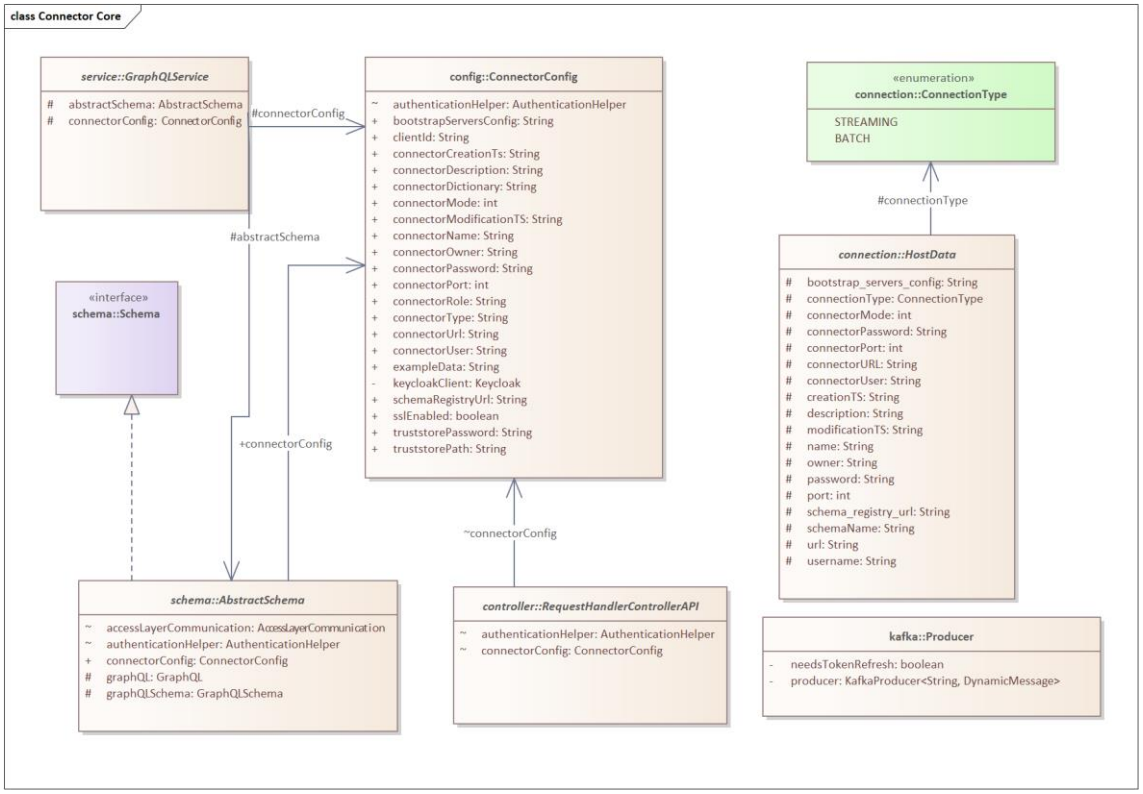

Abbildung 67: Klassendiagramm der Connector Core Bibliothek.

Beschreibung: Abbildung 67 zeigt das Klassendiagramm der Connector Core Bibliothek. Neben einer Konfigurationsklasse *ConnectorConfig*, um den Connector selbst zu konfigurieren (z.B. Name des Connectors oder Port, auf dem die REST-Schnittstelle bereitgestellt werden soll) existiert auch eine *HostData*-Klasse, die serialisiert als Selbstbeschreibung des Connectors in Form von Metadaten an die Connector-Zugriffsschicht geliefert wird. Außerdem wird die Schnittstelle für alle Connectoren in *RequestHandlerControllerAPI* definiert und eine Standardimplementierung eines Producers für die Kommunikation mit dem Kafka-Cluster (siehe Abschnitt 5.4) bereitgestellt.

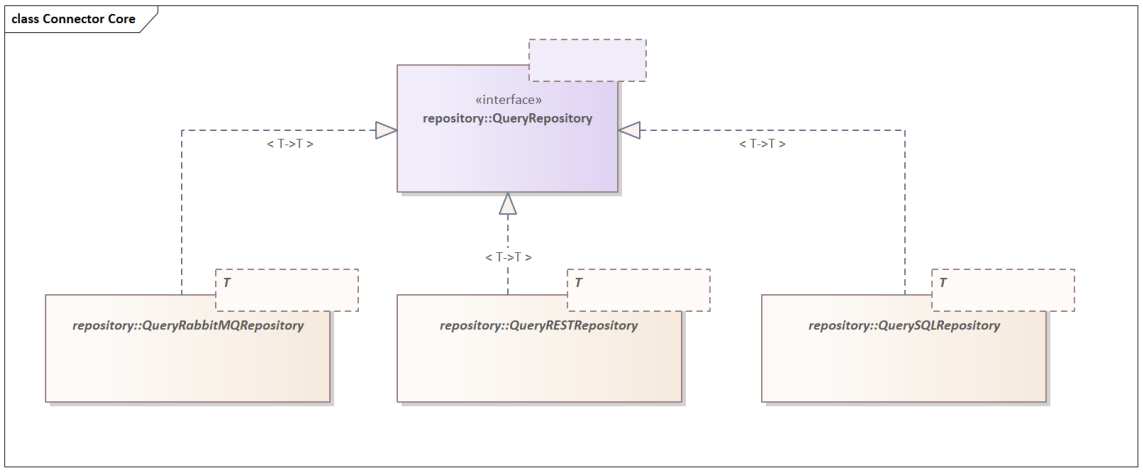

Abbildung 68: Hilfsklassen für häufig verwendete Technologien: SQL, RabbitMQ und REST.

Beschreibung: Abbildung 68 zeigt Hilfsklassen für die Implementierung der Connectoren zur Auflösung von GraphQL-Queries und der Generierung nativer Queries für die Datenquellen eines Connectors.

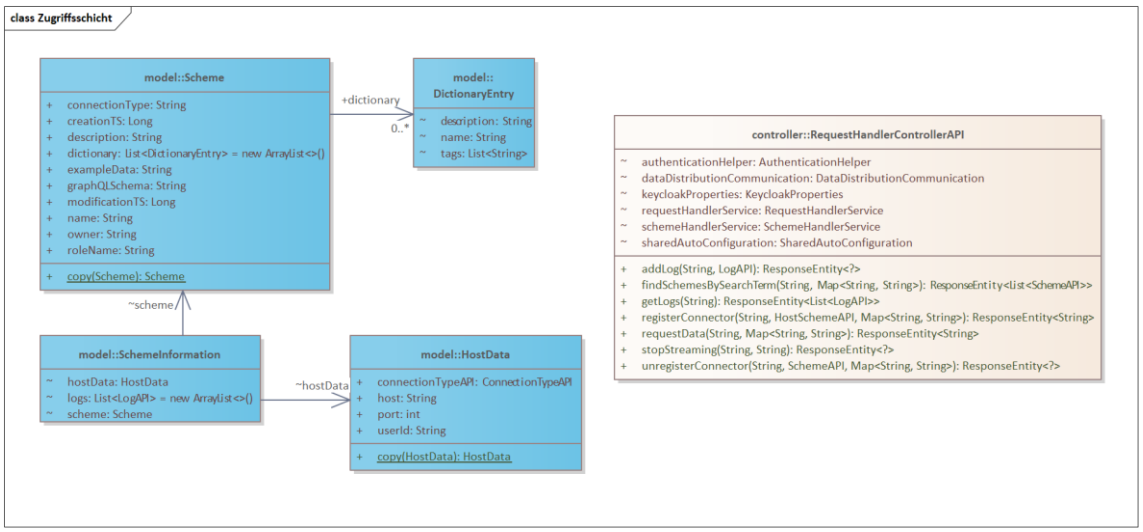

Abbildung 69: Ausschnitt des Klassendiagramms der Connector-Zugriffsschicht: Internes Datenmodell und *RequestHandler*.

Beschreibung: Abbildung 69 zeigt einen Ausschnitt des internen Datenmodells (in blau) und den *RequestHandler* der Connector Zugriffsschicht. Das Datenmodell zur Verwaltung der Connector-Metadaten (inkl. des GraphQL-Schemas) referenziert hier außerdem eine abgeleitete Klasse der in Abbildung 67 dargestellten *HostData*-Klasse. Datenanfragen können als GraphQL-Anfrage über *requestData* getätigt werden, und aktive Streams via *stopStreaming* angehalten werden.

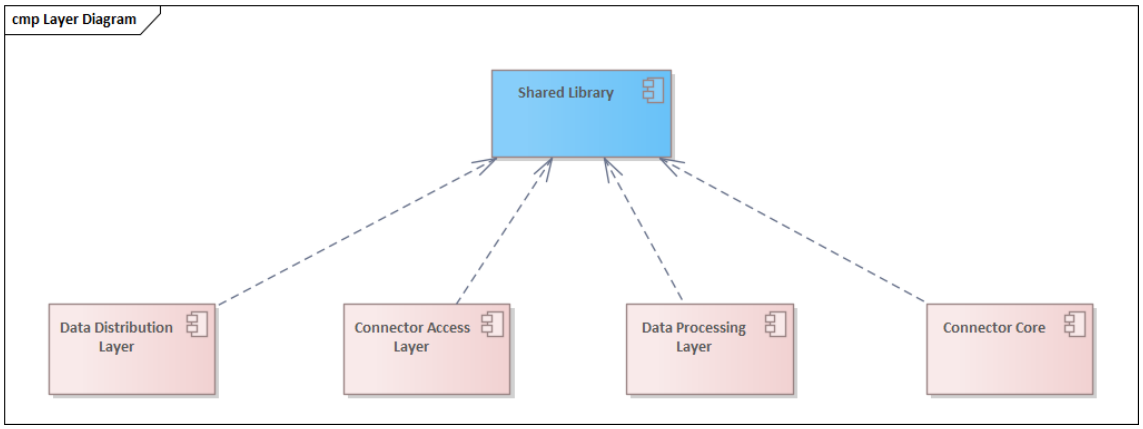

Abbildung 70: Shared Library für gemeinsame Funktionalitäten in verschiedenen Architekturschichten.

Beschreibung: Abbildung 70 zeigt die Verwendung einer Shared Library durch verschiedene Komponenten der FEDMS-Architektur. Die Shared Library enthält Hilfsfunktionen und Datenmodelle zur Kommunikation zwischen den einzelnen Architekturschichten.

D UI der prototypischen Umsetzung

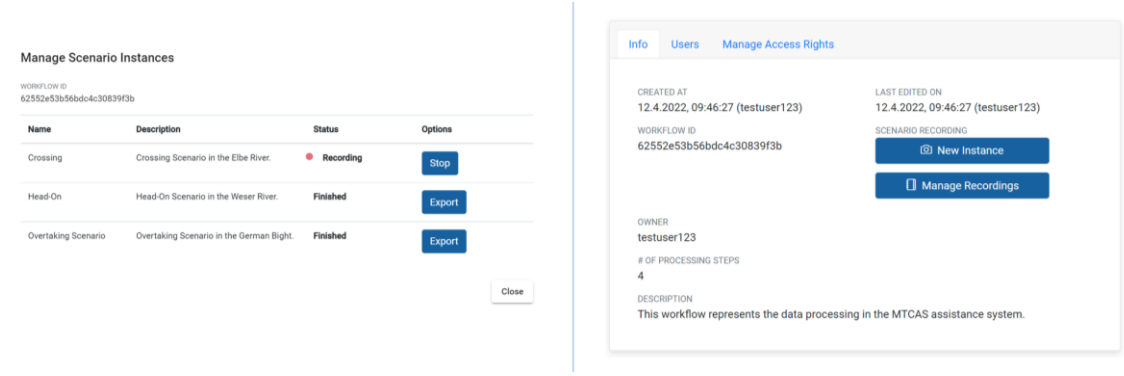

Abbildung 71: Teile der Web-Oberfläche für die REST-Schnittstelle der Datenverteilungsschicht zum Verwalten von Workflows und Szenario-Instanzen (vgl. Möller u. a. 2022).

Beschreibung: Als unterstützende Maßnahme für das Szenario-Management wurde eine Web-Oberfläche entwickelt (siehe Abbildung 71), die verwendet werden kann, um Szenarioaufzeichnungen und Workflows zu verwalten. Die Web-Oberfläche basiert auf der REST-Schnittstelle Datenverteilungsschicht. Die Web-Oberfläche wurde hauptsächlich während der Entwicklung zum Debugging der FEDMS-Funktionalitäten verwendet.

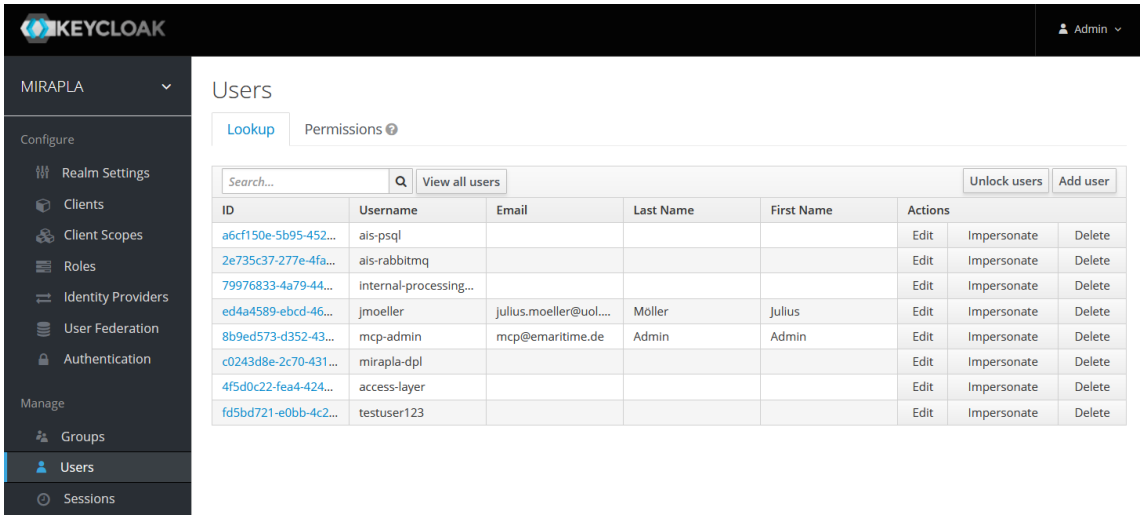

Abbildung 72: Grafische Benutzeroberfläche von Keycloak zur Nutzerverwaltung.

Beschreibung: Abbildung 72 zeigt die von Keycloak bereitgestellte Benutzeroberfläche für die Nutzerverwaltung im FEDMS.

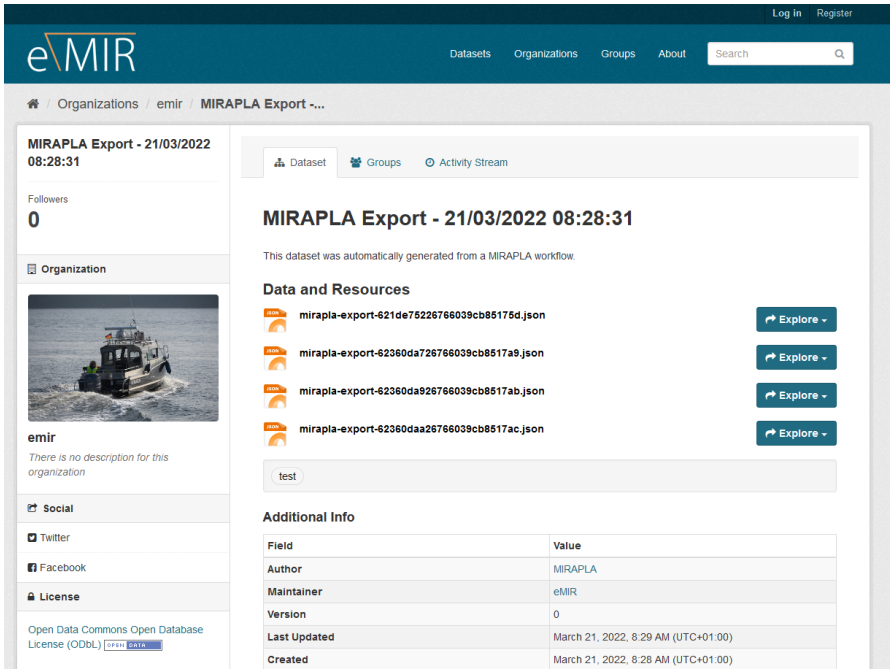

Abbildung 73: CKAN-System zur Persistierung von Szenario oder Workflowdaten.

Beschreibung: Abbildung 73 zeigt die CKAN-Instanz, die prototypisch im eMIR-Testfeld eingesetzt wurde, um Daten aus Workflows auch langfristig permanent zu speichern.