INF
Informatik

Masterthesis

Data sharing between heterogeneous data systems without compromising individual data autonomy and data sovereignty.

Luca Rommler
Matrikel-Nr.: 766788

First Examiner: Herr Prof. Dr.-Ing. Markus Schöller

Second Examiner: Herr Prof. Dr.-Ing. Christian Kücherer

Submission date: 30.04.2024

Hochschule Reutlingen
Reutlingen University
Alteburgstraße 150 • 72762 Reutlingen • Deutschland • T. +49 (0)7121 271-0 • F. +49 (0)7121 271-1101 • www.reutlingen-university.de

English Version

The exponential growth of data in the digital realm presents significant challenges in its management, sharing, and utilization. Conventional data systems often fail in upholding a users′ data sovereignty and autonomy, crucial for maintaining ownership. This underscores the need for innovative solutions, such as the "Secure Bilateral Federated Database System" (SBFDBS), which facilitates secure bilateral data exchange while preserving data governance.

This paper delineates the design, execution, and validation of the SBFDBS in the context of engineering data management. Employing the Arc42 template, this paper provides theoretical foundations with practical applications through iterative phases. It systematically examines SBFDBS′s capabilities in securely and bilaterally exchanging engineering data and rigorously assesses its performance and reliability, culminating in successful design validation.

The SBFDBS achieves enhanced data autonomy and sovereignty, fostering trust within data ecosystems. It provides a foundation for future research across domains like automotive, healthcare, and academia. By enabling secure and scalable data exchange, the SBFDBS catalyzes innovation, from predictive maintenance to medical diagnostics, indicating its broad applicability and transformative potential beyond its original context.

This paper comprehensively explores the SBFDBS′s design, execution, and validation within engineering data management, offering a robust solution to conventional data system challenges. The methodical approach yields insights into the system′s efficacy, positioning it as a pivotal tool for reshaping data management practices. Continued research and development efforts are warranted to further enhance the SBFDBS′s capabilities, ensuring its relevance in an evolving digital landscape.

Deutsche Version

Das exponentielle Wachstum von Daten im digitalen Bereich stellt eine beträchtliche Herausforderung bezüglich deren Verwaltung, Weitergabe und Nutzung. Konventionelle Datensysteme erweisen sich häufig als unzureichend im Hinblick auf die Wahrung der Datenautonomie und -hoheit der Nutzer, was für den Erhalt des Eigentums von entscheidender Bedeutung ist. Dies betont die Dringlichkeit innovativer Lösungsansätze, wie des „Sicheren Bilateralen Föderierten Datenbanksystems“ (SBFDBS), welches einen sicheren bilateralen Datenaustausch ermöglicht, während die Datenautonomie und -souveränität der Nutzer erhalten bleibt.

Diese Thesis präsentiert das Design, die Implementierung und die Validierung des SBFDBS im Kontext des Ingenieurdatenmanagements. Mittels der Arc42-Vorlage werden theoretische Grundlagen zusammen mit praktischen Anwendungen durch iterative Phasen geboten. Es erfolgt eine systematische Untersuchung der Leistungsfähigkeit des SBFDBS im sicheren und bilateralen Austausch von Ingenieurdaten sowie eine gründliche Bewertung dessen Leistung und Zuverlässigkeit, welche in einer erfolgreichen Systemvalidierung resultiert.

Das SBFDBS erzielt eine verbesserte Datenautonomie und -hoheit der Nutzer und stärkt das Vertrauen innerhalb von Datenökosystemen. Es dient als Grundlage für künftige Forschungsbemühungen in verschiedenen Bereichen wie der Automobilindustrie, dem Gesundheitswesen und der akademischen Welt. Durch die Ermöglichung eines sicheren und skalierbaren Datenaustauschs beschleunigt das SBFDBS Innovationen, von der prädiktiven Wartung bis hin zur medizinischen Diagnostik, was seine breite Anwendbarkeit und sein transformatorisches Potenzial über den ursprünglichen Anwendungskontext hinaus unterstreicht.

Dieses Dokument untersucht ausführlich das Design, die Implementierung und die Validierung von SBFDBS im Rahmen des Ingenieurdatenmanagements und bietet somit eine robuste Lösung für die Herausforderungen herkömmlicher Datensysteme. Der methodische Ansatz liefert Einblicke in die Wirksamkeit des Systems und positioniert es als entscheidendes Werkzeug zur Neugestaltung von Datenverwaltungspraktiken. Kontinuierliche Forschungs- und Entwicklungsanstrengungen sind erforderlich, um die Fähigkeiten von SBFDBS weiter zu verbessern und seine Relevanz in einer sich ständig wandelnden digitalen Landschaft sicherzustellen.

Table of Contents

1 Introduction

In the contemporary digital landscape, the exponential growth [Eri23, TE23] of data generation and collection has become a defining characteristic. These vast reservoirs of data serve as pivotal assets across diverse domains, spread throughout both personal and commercial spheres of activity. However, the burgeoning importance of data poses a challenge: how to effectively manage, share, and leverage this abundance of information. Traditional centralized data systems, while offering their own array of benefits, often confront inherent limitations, particularly regarding the preservation of data sovereignty and autonomy.

Central to this discourse is the notion of data sovereignty, defined as the principle asserting the rights of individuals and organizations to maintain control over their own data. In a landscape dominated by conventional centralized data systems, data sovereignty is frequently compromised as users give up control over their data to third-party entities. Such relinquishment engenders a host of security and privacy concerns, fundamentally undermining the autonomy of data owners.

Against this backdrop, the impetus driving the present investigation emerges from a recognition of the escalating significance of data sovereignty within an increasingly interconnected digital landscape. The contemporary digital ecosystem is replete with instances where users find themselves at the nexus of centralized data architectures, entrusting their valuable data assets to external service providers. Yet, this paradigm is not without its perils, as evidenced by the multitude of data breaches and privacy infringements [Pro23] that have underscored the vulnerabilities inherent in relinquishing data control.

In this context, the necessity to devise alternative frameworks that safeguard data sovereignty and facilitate seamless data exchange becomes increasingly apparent. It is within this context of necessity and opportunity that the conceptualization and development of a prototype system, known as the "Secure Bilateral Federated Database System" (SBFDBS), takes its foundation.

The core of this endeavor resides in the articulation of a paradigm — one grounded in the principles of secure and bilateral data exchange. By departing from

the centralized characteristic of traditional data architectures, the SBFDBS aims to democratize data governance, granting entities the autonomy to assert control over their data assets without succumbing to the influences of external intermediaries.

Although the SBFDBS is designed for versatile application across various contexts, this paper primarily concentrates on its implementation within the realm of engineering data. The intricate nature of engineering data underscores the need for a specialized focus in this regard.

Engineering data encompasses a wide array of information generated throughout the course of engineering projects, spanning design, analysis, manufacturing, and maintenance phases. It includes technical specifications, blueprints, Computer-Aided-Design (CAD) models, simulations, test results, maintenance logs, and more. Essentially, it encapsulates the quantitative and qualitative data integral to the planning, execution, and maintenance of engineering endeavors across various industries, such as aerospace, automotive, civil engineering, and beyond.

One of the foremost challenges in handling engineering data pertains to its sheer volume and complexity. Engineering projects generate massive datasets that consist of intricate technical details, which necessitate robust storage, management, and retrieval systems. Moreover, the interdisciplinary nature of engineering often involves collaboration among diverse teams, adding layers of complexity to data exchange and coordination.

Security is another critical concern in the realm of engineering data. Given the classified and sensitive nature of engineering information, ensuring data confidentiality, integrity, and availability is paramount. Unauthorized access, data breaches, or intellectual property theft can have severe repercussions, ranging from compromised competitive advantage to safety and regulatory violations.

In the context of data transfer, engineering data presents unique challenges due to its various formats and complexities. Traditional methods of data exchange, such as email or FTP, may prove inadequate for transmitting large and complex datasets or preserving data integrity. Moreover, the need for real-time collaboration among distributed teams necessitates efficient and secure mechanisms for synchronous data exchange.

Furthermore, data autonomy emerges as a critical consideration in engineering data management. Engineers and organizations must retain control over their data assets to safeguard confidential information, comply with regulatory requirements, and maintain operational autonomy. Yet, in centralized data systems, relinquishing control to third-party providers may compromise that exact

data autonomy, exposing organizations to risks of vendor lock-in, data sovereignty violations, and dependency on external entities for data governance.

1.1 Goal

The goal of this paper is to elucidate the design, implementation, and validation of the SBFDBS within the specialized context of engineering data. Through a synthesis of theoretical foundations and practical applications, this study endeavors to demonstrate the efficacy and viability of the SBFDBS as a potent instrument for safeguarding data sovereignty amidst the challenges posed by the contemporary data landscape.

By delving into the intricacies of engineering data management and leveraging the capabilities of the SBFDBS, this research seeks not only to address existing shortcomings, but also to pave the way for in data governance within the engineering domain. In doing so, it endeavors to empower the stakeholders with the tools and frameworks necessary to navigate the complexities of data exchange while upholding the principles of simplicity, autonomy, and sovereignty.

1.2 Research questions and Methodology

To fulfill the objectives of this work, the research questions in Table 1.1 have been formulated.

ID	Research Question
1	How can data sovereignty be maintained while utilizing external data ecosystems?
2	How can data autonomy be maintained while utilizing external data ecosystems?
3	How can software systems for data exchange in external data ecosystems be designed to be interoperable and scalable?

Table 1.1: Research Questions

To initiate the research process, a selection of keywords and phrases were created to encapsulate the essence of the study's focus on data sovereignty, data autonomy, and the design of software systems for secure data exchange within external data ecosystems. These keywords are carefully chosen to ensure inclusivity and relevance to the research questions outlined in the thesis.

The matrix depicted in Table 1.2 lists relevant keywords used to form search terms, serving as the foundation for initiating a snowball search.

Data Sovereignty	Data Autonomy	Data
Data Ecosystems	Decentralized Data Management	Federated Database Systems
Interoperability	System Design	Software Architecture
Validation	Prototyping	

Table 1.2: Keyword Matrix

These search terms are crafted to encompass a broad spectrum of literature relevant to the research questions, facilitating comprehensive exploration and analysis of the subject.

Following the definition of the search Terms, an exhaustive search of academic databases, specifically IEEExplore, SpringerLink and Web of Science, is conducted to gather a diverse range of scholarly works. This initial search serves as the foundation for the snowball methodology, which involves iteratively expanding the search by examining the reference lists of retrieved papers and identifying additional sources through citation analysis.

Table 1.3 presents exemplary search terms employed, alongside the respective scientific databases utilized, the yielded results, and the number of identified relevant papers.

Search Term	Database	Results	Relevant Papers
sovereignty AND autonomy AND data	IEEExplore	21	3
sovereignty AND data AND federated	IEEExplore	18	8
sovereignty AND autonomy AND data AND federated (age > 12 months)	Springer Link	9	8
sovereignty AND autonomy AND interoperability	Web of Science	2	1
system design AND (validation OR prototyping) AND software architecture AND federation AND interoperability	IEEExplore	3	1

Table 1.3: Search Terms examples

It is noteworthy that this list represents only a segment of a larger search endeavor aimed at gathering papers for subsequent application of the snowball methodology.

As the snowball methodology progresses, the focus shifts towards identifying seminal papers and key contributors within the field. Through careful examination of citation networks and bibliographic references, papers that have had a significant impact on the discourse surrounding data sovereignty, data autonomy, and interoperable software systems are identified and included in the review process.

1.3 Structure

The second chapter offers a brief overview of the current state of the art in data exchange, data sovereignty, software design and federated data systems. It sheds light on the diverse application domains of federated data systems in detail. The foundational principles and technical architecture of such systems are analyzed to impart a comprehensive understanding of their functionality.

Chapter three utilizes Requirements Engineering to conceptually design the system. This involves conducting a system requirements analysis that enumerates and explains all essential aspects of the system. Using this requirement analysis, the system is then conceptually designed and represented through models.

Chapter four focuses on the validation of the prototype. It assesses which requirements the prototype fulfills, which have been partially met, and which have been entirely removed. Additionally, it deliberates on the ′threats to validity′ and extensively addresses the scenarios in which these pose actual risks to the validity of the developed system.

The fifth chapter serves as the conclusion of the work. It summarizes all the insights gained throughout the study while simultaneously providing a glimpse into the future. Further research and expansion potentials are discussed and delineated.

2 State of the Art

The exchange of data in external ecosystems presents numerous challenges, particularly in ensuring security, privacy, and sovereignty. As the digital landscape continues to evolve [Eri23, TE23], the need for secure data exchange becomes increasingly critical. This is particularly evident in the context of wireless sensor networks for healthcare applications, where the secure and reliable exchange of health data is paramount [ALK10].

Furthermore, the establishment of data spaces and the evolution of multi-tenancy architecture play pivotal roles in shaping the design and operability of software systems for secure data exchange in external ecosystems [Ott22].

Additionally, the concept of data sovereignty has gained prominence, especially in the context of Indigenous research, where the governance of data on Indigenous populations is a topic of prioritized importance [Coc23]. The interplay of security, privacy, and trust in sociotechnical systems is crucial for identifying external threats and vulnerabilities, contributing to the overall security and trustworthiness of data ecosystems.

Moreover, the security and privacy of data in the medical Internet of Things (IoT) have been the subject of in-depth study, emphasizing the need for robust security measures in data exchange systems [SCL+18]. As the digital landscape continues to expand, the preservation of data sovereignty and the secure exchange of sensitive data remain at the forefront of research and development efforts [HBTD21].

Therefore, the design and implementation of software systems for secure data exchange in external ecosystems must address these multifaceted challenges to ensure operability, scalability, and the preservation of data sovereignty.

2.1 Secure Engineering Data Exchanges

In the pursuit of safeguarding the secure exchange of sensitive engineering data within external data ecosystems, it is imperative to consider various facets of

data exchange and security protocols. The assurance of data exchange security can be achieved through methods such as physical isolation [LCW+19], cryptographic protocols [Can01], and ensuring authenticity, integrity, availability, and non-repudiation of communication services at the process level [CLL+21]. Additionally, leveraging trustworthy blockchain interfaces can furnish a secure interface between blockchains and external platforms, thereby fortifying the reliability of data exchange [ABRS20].

The secure transmission of data within the realm of engineering data and federated database systems is paramount for upholding the confidentiality, integrity, and availability of sensitive information. Federated database systems amalgamate multiple autonomous and heterogeneous databases into a unified virtual system, furnishing users with transparent access while upholding data consistency and security [SL90a]. These systems offer mechanisms for data integration, resolving disparities between different schemes to formulate a federated schema, and enabling seamless access for users by orchestrating queries across multiple databases [AG96].

The incorporation of multilevel security policies has been investigated to augment the security of closely interconnected federated database systems [OS02]. Furthermore, the adoption of integrity lock architectures has been posited to support distributed authorizations and sustain data integrity within federated database federations [LWZZ]. These strategies endeavor to ensure that data is accessed and transferred securely throughout the federated environment.

Authentication in federated database systems poses challenges owing to the autonomous nature of the components, which may not consistently recognize the identity of federation users [YWJ02]. To mitigate this, subject switching algorithms have been devised to proficiently manage access control in federated databases, thereby guaranteeing that only authorized users can access sensitive information [YWJ02].

Furthermore, federated database systems afford a transparent interface to end-users and application developers, presenting federated data sources as a unified system and streamlining the data access process [MPL07]. This transparency is pivotal for ensuring a seamless user experience while upholding the security and integrity of the data traversing across federated databases.

2.2 Attaining Data Sovereignty

To achieve data sovereignty in external data ecosystems, it is crucial to consider various aspects such as international cooperation on standards, secure data

exchange, and the governance of data. International cooperation on standards and secure data exchange across borders are essential to guarantee sovereignty in an interconnected digital economy [KW22]. Additionally, the governance of data on Indigenous populations is long overdue, emphasizing the need for tribal data sovereignty [CSB+17]. Furthermore, the implementation of usage control architecture options can contribute to data sovereignty in business ecosystems by ensuring that the data stays within institutional borders [ZMJ+19], [WNY+21].

The effectiveness of data sovereignty requires the willingness of cooperative members to donate their data altruistically, highlighting the importance of cooperation and willingness in achieving data sovereignty [Cal21]. Trust between participants, data interoperability, and data sovereignty are key requirements that can be met by data spaces, emphasizing the role of trust and interoperability in achieving data sovereignty [Ott22]. Moreover, knowledge-driven data ecosystems equipped with computational methods can exchange and integrate data while preserving personal data privacy, data security, and organizational data sovereignty [GVC+21, SJE+23]. Similarly, a decentralized framework for biostatistics and privacy concerns has been applied to medicine to preserve the privacy of sensitive data and data owners' sovereignty [MFM+20].

The preservation of digital sovereignty is often criticized as a form of protectionism, highlighting the need to address concerns and criticisms in the pursuit of digital sovereignty [CF20]. Cloud federation implies serious security and privacy issues regarding data sovereignty when data is outsourced across different judicial and legal systems, emphasizing the challenges and complexities associated with data sovereignty in cloud environments [ECC16]. The Indian state's project of digital sovereignty must be understood as biopolitical, indicating the multifaceted nature of digital sovereignty and its intersection with political and state interests [Pra21].

In the context of big data ecosystems, secure development, incident response, and privacy-preserving platforms are essential for ensuring the security and privacy of data, contributing to the goal of achieving data sovereignty [MFSFM19, MSFFM20]. Additionally, differential game analysis of enterprises investing in new infrastructure and maintaining social network security under the digital innovation ecosystem highlights the common challenge of ensuring the security of digitalization, emphasizing the need for security in the pursuit of digital sovereignty [CLQJ22]. Furthermore, the interplay of security, privacy, and trust in sociotechnical systems is crucial for identifying external threats and vulnerabilities, contributing to the overall security and trustworthiness of data ecosystems [SAP+19].

Achieving data sovereignty in external data ecosystems requires a multifaceted approach that encompasses international cooperation, governance, trust, pri-

vacy, security, and addressing criticisms and challenges associated with data sovereignty.

2.3 Architecting Data Exchange Software

To design software systems for secure data exchange in external data ecosystems that are operable and scalable, it is essential to consider various aspects such as secure data exchange protocols, multi-tenancy architecture, and scalability in distributed systems.

Secure data exchange protocols play a crucial role in ensuring the confidentiality and integrity of data during exchange. For instance, the OPC (OLE for Process Control) systems provide an interoperable standard method for the secure and reliable exchange of data from industrial devices of multiple vendors to any client software application [GGHT19]. Similarly, a secure Device-to-Device (D2D) protocol specified for facilitating the exchange of health data among citizens and healthcare professionals can be used by software applications, leveraging Bluetooth technologies [KMM+20]. Furthermore, the feasibility of encrypted data exchange between secure software executed in a trusted execution environment (TEE) and the secure logic part of a heterogeneous SoC has been studied, highlighting the importance of secure communication in software systems [BMLB19].

In the context of multi-tenancy architecture, which is increasingly prominent in Software-as-a-Service (SaaS) applications, it is crucial to ensure that software systems are designed to be operable and scalable. Application-level multi-tenancy enables multiple tenants to share common application functionality and resources among each other, emphasizing the need for scalable and manageable software systems [GWLJ13]. Additionally, the use of software improvement techniques has been shown to improve the scalability of software, especially in the context of software design patterns [RFR19]. Moreover, the development of a data security ecosystem composed of country, industry, enterprise, and individual components is crucial for promoting the sound development of the economy and maintaining the order of social and economic development, highlighting the importance of a holistic approach to data security in software systems [MWLJ18].

Scalability in distributed systems is another critical aspect to consider when designing software systems for secure data exchange. Evaluating the scalability of distributed systems is essential to ensure that the software systems can handle increasing workloads and data volumes effectively [JW00]. Furthermore, the performance and scalability of system software on large-scale machines have

often been neglected, emphasizing the need to address scalability challenges in software systems [PFFC]. Additionally, the trade-off between performance, scalability, and cost implications needs to be discussed, particularly when re-architecting software for the cloud, highlighting the importance of considering scalability in cloud-based software systems [XFPM14].

To design software systems for secure data exchange in external data ecosystems that are operable and scalable, it is essential to incorporate secure data exchange protocols, multi-tenancy architecture, and scalability in distributed systems. These considerations are crucial for ensuring the confidentiality, integrity, operability, and scalability of software systems in the context of secure data exchange.

2.4 Architecture of Federated Databases

There are various approaches and implementations to construct the architecture of federated databases systems as shown in the works of Gaedke [MG05], Papadakis [PBM23], Berger [BS08] and Tonne [TSJ+12], which may vary depending on specific application cases and domains. All federated database management systems (DBMS) essentially adhere to the same basic principles. These common principles enable individual entities to autonomously manage their data while still benefiting from the ability to share and utilize it within a controlled framework.

2.4.1 Basic Requirements for Architecture

To better understand the technical architecture of federated DBMS, it is helpful to consider the three basic requirements as outlined by Amit P. Sheth et al. [SL90b]. These requirements—'Distribution,' 'Heterogeneity,' and 'Autonomy'—form the fundamental framework upon which the design and structuring of federated data systems are based. A solid understanding of these requirements and their significance is crucial for adequately grasping the inner mechanisms and the way they are interconnected within this data-driven ecosystem.

2.4.2 Distribution

The requirement of 'Distribution' refers to the partitioning of data across multiple databases. These databases can be stored on a single computer system or on multiple interconnected computer systems, which may be located either in the same place or geographically distributed. Data distribution can occur in vertical and horizontal database partitions in the relational sense. There may also be multiple copies of parts or the entire dataset, which do not necessarily need to be identically structured.

The benefits of data distribution, such as increased availability, reliability, and improved access times, are well-known. In a federated DBMS, data distribution can be strategically employed to harness these advantages. Much of the data distribution in federated DBMS arises from the existence of multiple independent database systems before the establishment of a federated system.

Linking several data systems forms the basis for integrating heterogeneous data from various sources. This enables efficient consolidation and utilization of information present in different databases and formats. Distribution can contribute to enhancing the availability, performance, and scalability of the federated system by distributing the load across various resources and providing redundancy to ensure fault tolerance.

2.4.3 Heterogeneity

The requirement of Heterogeneity pertains to the diversity of differences, particularly in the aspects of various DBMS and in the semantic properties of the data itself. In the context of federated DBMS, heterogeneity refers to the challenge of integrating data from different sources that utilize different data models, structures, constraints, and query languages. These differences can complicate the exchange and processing of data, especially when different organizations or systems utilize different DBMS with varying characteristics.

Heterogeneity at the semantic level arises when there are disagreements regarding the meaning, interpretation, or intended use of data. This can result in seemingly similar data having different meanings, complicating integration and analysis.

In federated DBMS, addressing heterogeneity is crucial, as these systems aim to seamlessly integrate and utilize data from diverse sources. By addressing the challenges of heterogeneity, federated DBMS can facilitate consistent and meaningful analysis across various data sources. This principle underscores the

need for mechanisms and techniques to overcome differences in data models, structures, semantics, and query languages to enable effective communication and collaboration among heterogeneous data sources.

2.4.4 Autonomy

The requirement of Autonomy pertains to the fact that individual organizational units managing different database systems (DBS) are often autonomous. This means that each DBS has separate and independent control and is managed by legally separate entities. Those controlling a DBS are often willing to share data with others as long as they retain control. Therefore, understanding the aspects of component autonomy and how they can be addressed when a DBS participates in a federated DBS is important.

Autonomy can take various forms, such as Design Autonomy, Communication Autonomy, Execution Autonomy, and Association Autonomy. Design Autonomy allows a DBS to choose its own design regarding data management, data model, query language, data conception, and more. Communication Autonomy refers to a DBS's ability to decide whether and how it communicates with other DBS. Execution Autonomy means that a DBS can perform local operations without interference from external operations. Association Autonomy allows a DBS to decide how much functionality and resources it shares with others.

Autonomy is crucial for federated DBMS as it ensures the independence and flexibility of individual DBS. It enables each DBS to make its own decisions regarding design, communication, execution, and participation in federations without impacting the federated DBMS. This supports addressing heterogeneity and strengthens efficient collaboration in a federated DBMS.

However, there may be situations where autonomy needs to be restricted to meet certain requirements. Therefore, it is important to have a balanced understanding of autonomy principles in a federated data system to leverage the benefits of autonomy while ensuring coordination and collaboration.

2.4.5 Basic Technical Architecture

The basic architecture according to Stefan Conrad et al. [CT70] is depicted in figure 2.1.

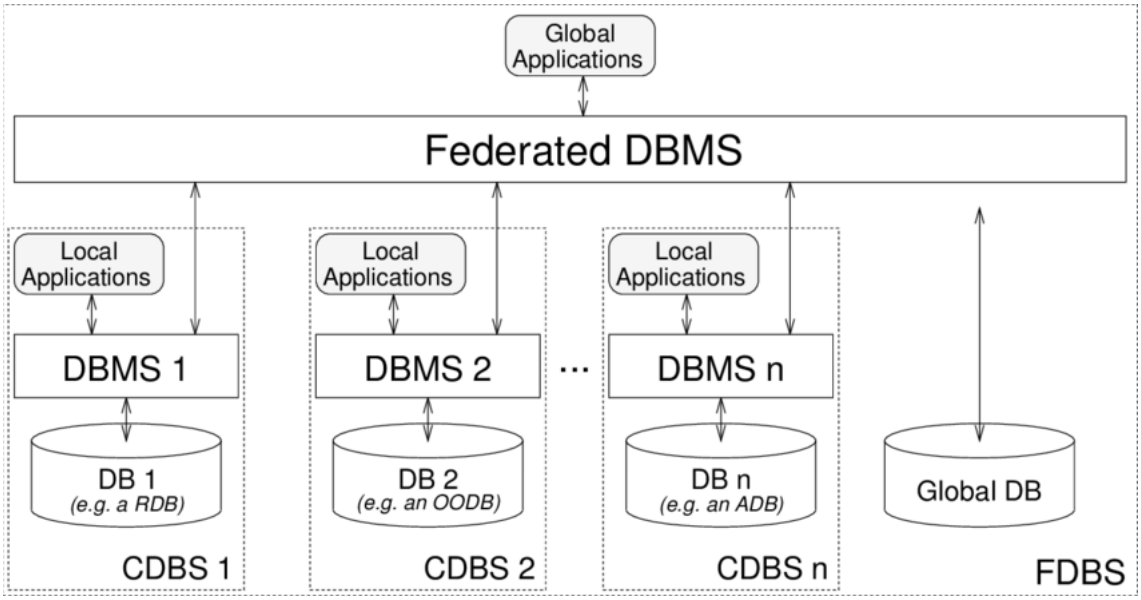

Figure 2.1: Basic architecture of federated data systems [CT70]

To describe the fundamental architecture of federated data systems, they are divided into three layers. The Application Layer, the Federation Layer, and the Autonomous Data System Layer.

The Application Layer forms the top layer of the federated DBMS structure and serves as the direct interface between users or applications and the system. Interaction with the federated data system occurs in this layer, where users or applications send queries to the database. The subsequent layers then process these queries.

The second layer, the Federation Layer, plays a central role as an intermediary between users and distributed data sources. Here, a unified interface for accessing federated data is provided to conceal the complexity of data distribution and heterogeneity from users. An essential component of this layer is the Common Data Model (CDM), which serves as a cross-reference schema and enables the integration of various data source schemas.

The third layer, the Autonomous Data System Layer, represents the level where individual autonomous DBMS exist. This layer consists of various different autonomous DBMS, managed by independent entities and fed with data. Various approaches are used to ensure communication and integration with the Federation Layer, with the CDM being just one of many possible integration concepts [MG05, PBM23, BS08, TSJ+12].

Preserving and ensuring the autonomy of the Common Data Base Systems (CDBS) is the central concern of federated DBMS. This allows the CDBS to continue executing their local operations while participating in the federation. This means that the interfaces of the CDBS are maintained to support existing applications without modifications.

In conclusion, the exploration of federated database systems underscores the critical balance between autonomy, heterogeneity management, and efficient data distribution. As evidenced by the works of Gaedke [MG05], Papadakis [PBM23], Berger [BS08] and Tonne [TSJ+12], the architectural frameworks of these systems provide a structured approach to navigating the complexities of data sharing while maintaining individual data sovereignty.

By understanding the fundamental requirements of distribution, heterogeneity, and autonomy, stakeholders can effectively design and implement federated database systems that promote seamless integration and utilization of diverse data sources. The layered architecture, as delineated by Conrad et al. [CT70], offers a comprehensive framework for orchestrating interactions between users, federated layers, and autonomous data systems.

As technology continues to evolve, the principles shown in this discourse will remain foundational in the development of secure and efficient data ecosystems. By embracing these principles, researchers and practitioners can pave the way for enhanced collaboration, innovation, and data-driven decision-making in diverse domains and applications.

3 System Design

The system design chapter, following the structured approach of ARC42, stands as the foundation for comprehensively outlining the architecture and complexities of the "Secure Bilateral Federated Database System" (SBFDBS). It delves into various aspects of design, covering structural elements, functionalities, and the guiding blueprint for SBFDBS development and operation.

This chapter systematically explores the system's architecture, components, and interactions, providing a thorough understanding of the design choices aimed at preserving data autonomy, sovereignty, and security while facilitating bilateral data transfers within the engineering domain.

By using parts of the ARC42 template, the chapter addresses architectural decisions, explaining the rationale behind each design choice and its alignment with defined requirements and constraints. It also presents architectural views, illustrating the system's structural organization, behavior, and interactions from different perspectives.

3.1 Introduction and Quality Goals

To ensure the secure bilateral transfer of data while maintaining data autonomy and sovereignty, it is essential to establish quality goals that address the concerns of the major stakeholders. For this purpose, the following goals are defined in order of importance.

3.1.1 Interoperability

The architecture must focus on interoperability to ensure seamless bilateral data transfer between different systems and platforms.

3.1.2 Data Sovereignty

It is crucial to prioritize data sovereignty, ensuring that the architecture empowers users to maintain control and ownership of their data throughout the bilateral transfer process.

3.1.3 Security

The architecture must prioritize robust security measures to protect the data during bilateral transfer. This includes encryption techniques to safeguard the data from unauthorized access and cyber-attacks.

3.1.4 Scalabbility

Scalability is essential to accommodate the growing volume of data exchange. The architecture should be designed to handle high transaction volumes and provide reliable security mechanisms.

3.1.5 Trust

To ensure the success and integrity of a federated database system, it is crucial to establish trust both in the system itself and in its users. Trust is essential in this context due to the autonomous and heterogeneous nature of the databases involved. The need for trust in federated database systems is emphasized by the challenges in authentication, where autonomously operated components may not know the identity of federation users [YWJ02].

3.2 User Stories

The following user stories offer an overview of the diverse landscapes where the SBFDBS can be applied. Each use case unveils a unique scenario crafted to highlight the SBFDBS adaptability and efficacy across various sectors. These narratives elucidate the system's functionalities, showcasing its ability to facilitate secure data transfers while upholding data autonomy and sovereignty in distinct operational contexts.

3.2.1 User Stories

i. As a manufacturing company, I seek a secure data exchange system to share proprietary design and production data with partners while safeguarding our data autonomy and intellectual property rights.

ii. As a research scientist, I want to securely exchange experimental data with collaborators while retaining control over the access and usage of my data.

iii. As a healthcare provider, I need to securely share patient records with authorized entities while ensuring that patient data autonomy and privacy are maintained.

iv. As a financial institution, I require a secure platform to exchange sensitive financial data with regulatory bodies while retaining control over data access and compliance.

v. As a government agency, I aim to exchange classified information securely with authorized agencies while maintaining strict control over data access and security.

3.3 Requirements

This chapter defines the critical requirements for the construction of a system facilitating bilateral data exchange while safeguarding individual data autonomy and sovereignty. The primary objective is the establishment of a framework that ensures secure data transfers, endows users with data control, mitigates vulnerabilities, and upholds the principles of data autonomy and sovereignty.

3.3.1 Functional Requirements

The functional requirements of the system serve as a blueprint outlining its operational capacities. These requirements encapsulate fundamental functionalities essential for secure bilateral data transfer while preserving users′ data autonomy and sovereignty. The definitions of these requirements can be found in Table 3.1.

ID	Summary	Requirement
F 1.1	Data request	The system must implement functionality to request the currently available data from all connected autonomous databases and present it to the user.
F 1.2	Data acquisition	The system must provide a mechanism for users to download one or more available data/files from one or multiple connected autonomous databases.
F 2.1	Response Time	All user interactions must be responded to within 1 second or less, ensuring a responsive user experience.
F 2.2	Data Transfer Speed	The system must be capable of transferring 1 GB of data within 5 minutes, including the entire process from initial request to data transfer completion, while maintaining data integrity.
F 2.3	Real-time Updates	The system shall provide real-time updates to users when new data becomes available in the connected autonomous databases, ensuring timely access to the latest information.
F 3.1	Integration of new Systems	The system should support dynamic integration of new autonomous databases at runtime, with newly integrated databases becoming available for data retrieval within 1 second of integration.
F 3.2	Graceful Disconnection Handling	The system must gracefully handle abrupt disconnections of existing logged-in autonomous databases at runtime, ensuring system stability without requiring restarts or failure.
F 3.3	Autonomous Database Management	The system must effectively manage abrupt modifications to autonomous databases, including additions, modifications, or removals of data/files, while maintaining data consistency and availability.
F 4	User Authentication	The system must provide secure user authentication mechanisms, allowing users to log in using Single Sign-On (SSO) methods such as SSID, ensuring data security and user privacy.

Table 3.1: Functional Requirements

3.3.2 Non-Functional Requirements

Parallel to the functional requirements, the non-functional requirements embody qualitative attributes and constraints fundamental to the system's performance and operation. Key among these requirements are stringent adherence to security protocols, robust authentication mechanisms, and compliance with data sovereignty regulations. The detailed definition of these non-functional requirements can be found in Table 3.2.

ID	Summary	Requirement
NF 1	Security	The system shall provide options for secure transmission of data, utilizing protocols such as HTTPS.
NF 2	Data Integrity Verification	The system must incorporate mechanisms to verify the integrity of transmitted data, ensuring its preservation throughout the bilateral transfer process.
NF 3	Logging	The system should include logging functionality to document all data transfer activities, fostering comprehensive records for accountability and traceability purposes.
NF 4	Interoperability	The system must be architected to support interoperability with diverse data systems and storage technologies, facilitating seamless bilateral data exchange across various platforms.
NF 5	Scalability	The system shall be engineered to accommodate escalating volumes of engineering data and user traffic without significant performance degradation over time, with a measurable metric for performance degradation set at less than 10% increase in response time under load.
NF 6	Performance	The system must be capable of sustaining responsiveness when serving a minimum of 5 concurrent users and interacting with at least 3 autonomous databases, with a measurable metric for response time set at less than 500 milliseconds.

Table 3.2: Non-functional requirements

3.4 Stakeholders

The Stakeholders for the system encompass a diverse range of individuals and entities with vested interests in the system's development, implementation, and

outcomes. The identification of stakeholders is crucial for understanding their needs, concerns, and expectations, and for ensuring that the system aligns with their requirements. The stakeholders which are relevant for the System are defined in Table 3.3.

Stakeholders	Description and Interests
System Users	Their concerns revolve around the security and privacy of their data, the ease of use of the system, and the assurance of data autonomy and sovereignty.
Data Protection Authorities	Regulatory bodies and data protection authorities are essential stakeholders due to their role in ensuring compliance with data protection laws and regulations.
Business and Legal Entities	Organizations and legal entities that engage in bilateral data transfer, such as businesses, research institutions, and legal firms, are stakeholders with a vested interest in ensuring that the system facilitates secure and autonomous data transfer in compliance with legal and contractual obligations.
Community and Societal Groups	Stakeholders from community and societal groups may have an interest in ensuring that the system's design aligns with broader societal values, ethical considerations, and the promotion of data sovereignty and autonomy.
Industry Standards Organizations	Entities involved in setting industry standards for data security and autonomy are stakeholders with an interest in ensuring that the system aligns with established best practices and standards.
Research and Academic Institutions	Academic and research institutions have a stake in ensuring that the system aligns with the latest advancements in data security, privacy, and autonomy. Their involvement can contribute to the system's alignment with cutting-edge research and innovation.
Endorsers and Advocacy Groups	Organizations and groups that advocate for data privacy, autonomy, and sovereignty may have a stake in endorsing or influencing the system's design to ensure that it aligns with their advocacy goals.

Table 3.3: Stakeholders

3.5 Constraints

This chapter delves into exploring these constraints, which encompass a range of requirements that restrict the freedom of the decision-making and system design processes. These limitations impact the architectural landscape, dictating design choices and guiding the overall developmental trajectory of software systems.

This system, developed within the context of a Master's thesis, operates under significant constraints, primarily related to financial resources and available workforce. The design and validation phases are not funded, necessitating reliance on open-source solutions or self-developed software to create the prototype. Moreover, there is a stringent time limit of 3 months for the design, prototyping, and validation stages. Due to the budget constraints, acquiring specialized engineering software or security tools necessary for developing a robust system might be challenging, potentially limiting the range of available features or security measures.

Additionally, as this project is the work of a single individual, the scope must remain limited to enable comprehensive validation of the design through the prototype.

Lastly, ensuring secure bilateral transfer of engineering data while maintaining data autonomy can be inherently complex. Developing custom protocols or ensuring compatibility with existing protocols might be time-consuming and resource-intensive. Because time is of the essence, the system is going to use currently proven secure transmission protocols like TLS and SSL.

3.6 Context and Scope

This chapter delineates the scope and context within which the system design is situated, elucidating the challenges encountered, and the strategies employed to navigate them.

The design of the SBFDBS emerges against the backdrop of burgeoning concerns surrounding data security, sovereignty, and sharing within engineering domains. With the exponential growth of digital data and the proliferation of collaborative engineering initiatives, the need for robust mechanisms to facilitate secure data exchanges among stakeholders becomes increasingly pronounced. Traditional approaches to data sharing often entail centralized repositories, which raise concerns regarding data ownership, control, and vulnerability to breaches.

In contrast, federated database systems offer a decentralized paradigm that affords greater autonomy to data custodians while facilitating seamless data access and collaboration. Positioned at the intersection of data security, sovereignty, and federated systems, the SBFDBS embodies a holistic approach to addressing these multifaceted challenges.

The scope of the SBFDBS system design encompasses a multifaceted exploration of its architectural components, functionalities, and validation protocols. Key aspects of the system design include:

- **Architectural Configuration:** An in-depth analysis of the architectural elements comprising the SBFDBS, including data storage mechanisms, access control protocols, and communication interfaces.

- **Functional Specifications:** A comprehensive delineation of the functional requirements and capabilities of the SBFDBS, encompassing user authentication and interoperability with existing data systems.

- **Validation Methodologies:** Rigorous validation processes aimed at scrutinizing the efficacy, robustness, and security of the SBFDBS across diverse operational scenarios. Validation efforts encompass both qualitative and quantitative assessments, leveraging a spectrum of testing environments and methodologies.

3.6.1 Challenges and Considerations

The development and validation of the SBFDBS are beset by a myriad of challenges stemming from resource constraints, temporal limitations, and reliance on open-source tools. These challenges include:

- **Limited Testing Environments:** Resource constraints hinder the diversification of testing environments, potentially compromising the comprehensiveness of validation outcomes.

- **Constraints on Sample Size:** Time and financial limitations impose constraints on sample size for validation endeavors, necessitating a cautious approach to data collection and analysis.

- **Challenges in Conducting Longitudinal Studies:** Temporal constraints necessitate strategic decisions regarding the inclusion/exclusion of system components, guided by considerations of necessity and prior validation efforts.

- **Limited Access to Specialized Validation Tools:** Financial constraints impede access to specialized validation tools, necessitating reliance on open-source and freely available software with potential limitations in functionality and robustness.

- **Time Constraints and Balancing Priorities:** The finite availability of time necessitates pragmatic prioritization of validation activities amidst competing commitments.

- **Reliance on Open-Source and Free Tools:** While cost-effective, reliance on open-source tools may introduce limitations or biases into validation efforts, underscoring the importance of cautious evaluation and validation procedures.

Addressing these challenges requires a blend of methodological rigor, strategic planning, and adaptive resilience, ensuring the integrity and reliability of the SBFDBS design and validation processes. By navigating these challenges adeptly, the SBFDBS endeavors to emerge as a solution in the realm of secure and bilateral engineering data transfers, embodying principles of data autonomy, sovereignty, and federated collaboration.

3.6.2 Business context

The System design hinges on understanding communication partners, inputs, outputs, and interfaces. This chapter delves into the business context, detailing entities which interact with the system and entities which set regulations and restrictions governing these interactions.

In the Figure 3.1 the business context of the SBFDBS is described.

Within the visual representation, the Federated Database occupies a central position, serving as the nexus between System users and data providers. These users encompass various entities, ranging from individual system foreign users to participating business companies and research institutions. The Federated Database also interfaces with autonomous databases, each governed by distinct entities such as businesses, research and academic institutions, or open-source communities.

After initial authentication, system-users initiate data queries to the SBFDBS, which aggregates all valid queried data from the autonomous Database systems. The responsibility of maintaining and expanding the SBFDBS typically falls upon operation-engineers affiliated with the entities providing autonomous databases or volunteer developers from open-source communities.

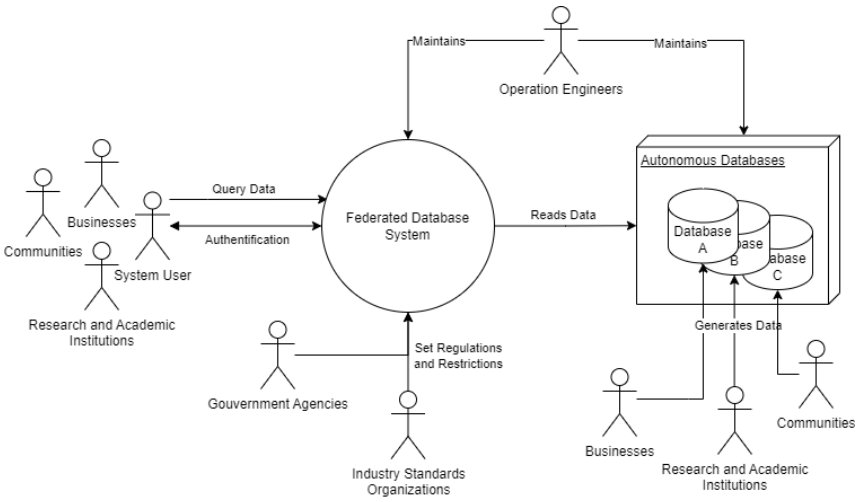

Figure 3.1: Business Context

Additionally, Governmental Agencies and industry standards organizations impose specific regulations and constraints that the SBFDBS must embody. These regulations are contingent upon the field in which the SBFDBS is deployed, necessitating compliance with varying regulatory frameworks and standards.

3.6.3 Technical Context

The technical context of the SBFDBS defines the critical interfaces that bind the system to its environment, encompassing the channels and transmission media pivotal in data exchange. This understanding extends beyond mere connections, mapping domain-specific inputs and outputs to these channels, illuminating how data flows through these interfaces.

Stakeholders vested in infrastructure and hardware design heavily rely on these technical interfaces to inform their architectural decisions. The infrastructure's robustness and the hardware's compatibility hinge upon a clear comprehension of these channels and their alignment with the system's inputs and outputs.

The graphical representation shown in figure 3.2 displays the technical context and serves as a foundational guide, enabling stakeholders to discern the connections and understand how the Secure Bilateral Federated Database System interfaces with its environment, ensuring secure and bilateral data transfers while preserving data autonomy and sovereignty.

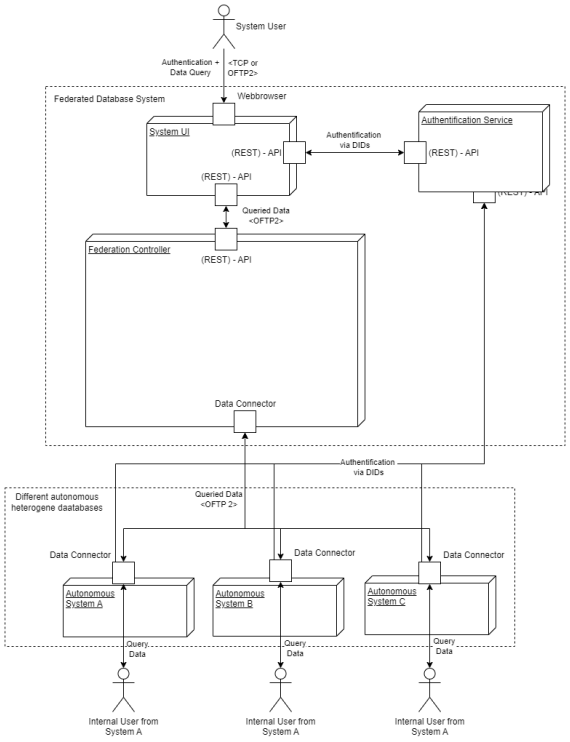

Figure 3.2: Technical Context

The Secure Bilateral Federated Database System (SBFDBS) comprises four primary components: the User-Interface, the Authentication Service, the Federation Controller and the Federation Connectors.

The User-Interface serves as the system through which users engage with the SBFDBS. It can be constructed as a web application to facilitate widespread access for a large user base. The User Interface is accessible in two distinct locations within the SBFDBS. It is present at the upper level through an independent web server that external system users utilize to connect to the SBFDBS. Additionally, it is integrated into the Federation Connector at the lower level. This interface is available for system users associated with a participating organization.

Upon utilizing the Authentication Service, system users authenticate themselves via the User-Interface. This authentication process leverages Self-Sovereign Identity (SSI) [MGGM18, PR21b] and Decentralized Identifiers (DID) [RSL+20], typically stored in local or cloud-based wallets. While the Authentication Service may operate within the SBFDBS, external authentication services can also manage this authentication procedure. Crucially, adherence to the SSI methodology guarantees user autonomy throughout the authentication phase.

Post-authentication, system users can initiate data queries through the User-Interface. These queries are transmitted to the Federation Controller, responsible for mediating all interactions between system users and autonomous databases. The Federation Controller dissects the query and gathers accessible data from autonomous databases using the Connector. Additionally, the Federation Controller maintains a registry of actively logged-in Federation Connectors, facilitating the automatic removal of inaccessible Federation Connectors from the system.

The Federation Connector(s) serves as a conduit between the Federation Controller and autonomous data systems. Deployed within these autonomous data systems, it interfaces between the Federation Controller and the respective autonomous database. This component standardizes heterogeneous local data into a uniform format based on a predefined common data model. The base variant of a Federation Connector, developed by SBFDBS operation-engineers, is provided to autonomous data systems. Subsequently, the operation-engineers of each individual autonomous data systems integrate their unique translation and business logic into the Connector before local deployment within their database systems.

This foundational technical architecture enables the integration of numerous autonomous databases into the system. Importantly, these autonomous systems operate independently, unaware of other systems linked to the SBFDBS. Additionally, system users interact with a unified entity, perceiving the SBFDBS as a singular, cohesive database. Notably, users remain unaware of the specific autonomous database systems connected to the SBFDBS. The utilization of SSI methodologies guarantees autonomous authentication services, obviating the reliance on external institutions.

3.7 Solution Strategies

This section provides a concise overview of solution strategies, addressing key quality goals and referencing detailed explanations in subsequent sections. These strategies align with quality goals such as security, interoperability, data sovereignty, scalability, and trust.

Each strategy is rooted in the problem statement and quality goals, ensuring a robust architecture for secure and autonomous data transfer. Detailed insights into technological intricacies are available in the following sections, offering a comprehensive understanding of the architecture's design and validation.

3.7.1 Interoperability

To ensure seamless interoperability, we've incorporated a pivotal component: the Federation Connector. This system acts as a bridge, enabling the integration of any external system into the SBFDBS. Operating as a Spring application, the connector is deployed locally within each participant's environment. Subsequently, participants tailor the connector to align with their specific system architecture.

The core function of the connector lies in its ability to translate the data structure of a participant's system into a standardized format using a common data model. This model allows database owners to delineate subsets of local data, harmonizing them into a unified virtual database through a global schema. The flexibility of this model lies in its adaptability to and by individual database owners, ensuring seamless integration into the SBFDBS architecture.

The selection of a connector for the SBFDBS is highly fitting for several reasons. Primarily, its adaptability enables diverse systems to seamlessly integrate into the SBFDBS framework, fostering a unified environment conducive to bilateral data exchange. Moreover, the utilization of a standardized data model, ensures a coherent and harmonized structure for disparate data systems, enabling efficient processing within the federation controller.

The connector's role in facilitating interoperability not only aligns with the system's architecture but also ensures a streamlined and standardized approach to incorporate various systems into the SBFDBS ecosystem. Its adaptability and standardized data translation processes solidify its significance as a crucial component within the SBFDBS architecture.

3.7.2 Scalability

In the technical domain, the system demonstrates inherent scalability, primarily facilitated by the ease of integrating new autonomous systems. The linchpin enabling this seamless expansion of data capacity lies within the federation connector—a pivotal component enabling effortless extension of the system's data volume.

While scaling the federation controller poses challenges, it is noteworthy that vertical scaling remains an option. Augmenting the computing power of the federation controller directly contributes to expediting query processing and moderation. However, ensuring horizontal scalability necessitates additional measures within the system's framework.

The accomplishment of this objective necessitates a redefinition of the federation controller's role. By confining its functionalities to mediating interactions between data systems and users, as well as serving as the access point for system login and logout procedures, multiple instances of the federation controller could be instantiated and deployed according to demand. The distribution of traffic load could subsequently be automatically balanced among those instances possessing residual computational resources. To sustain the ability to monitor all presently active federation connectors, the implementation and deployment of a new component are imperative.

This newly introduced component, denoted as "the connection controller," is specifically designated to oversee, monitor, and conduct health checks on all systems currently logged into the federation. This repository, whether it assumes the form of a Database or a dedicated server, governs and facilitates connections to and from the associated databases in response to requests from federation controllers.

The implementation of the connection controller allows the capacity to accommodate multiple federation controllers, thereby fostering vertical scalability without compromising the operational integrity of the system.

Lastly, leveraging a web server as the user interface offers inherent scalability, particularly through technologies like Kubernetes. Kubernetes, an open-source container orchestration platform, boasts diverse autoscaling mechanisms — Horizontal Pod Autoscaler (HPA), Vertical Pod Autoscaler, and Cluster Autoscaler, as detailed by Nguyen [NYK+20]. The utilization of Kubernetes empowers both horizontal and vertical scaling, ensuring adaptability in resource management based on workload demands, as highlighted by Vu [VTK22]. Multiple instances of the same website can operate within a Kubernetes Cluster, wherein automated load balancing and availability mechanisms are inherent functionalities.

3.7.3 Security

To ensure comprehensive security both within and outside the system, the TLS protocol is employed for data transport within the SBFDBS. This protocol facilitates secure data transmission through encryption processes, ensuring data confidentiality and integrity. User authentication is ensured through the integration of the SSI methodology, where users employ their DIDs for authentication, securely stored either locally or in cloud-based secure wallets.

The adoption of the TLS protocol, SSI methodology, and DIDs within the SBFDBS is in accordance with its security requirements. The TLS protocol's capability to establish secure connections and facilitate encrypted data transmission is

pivotal for safeguarding the confidentiality and integrity of sensitive data within the federated database system. Additionally, the SSI methodology, utilizing DIDs for user authentication, aligns with the necessity for a robust and decentralized identity management system, ensuring that only authenticated and authorized users access the SBFDBS [SNE20].

DIDs afford users control over their identities, promoting autonomy and sovereignty within the system, consistent with the core principles of the SBFDBS. Furthermore, the secure storage options for DIDs, whether locally or in cloud-based secure wallets, offer flexibility and convenience to users while upholding the security of their authentication credentials [SNE20].

3.7.4 Trust

The system places a strong emphasis on fostering trust among participants by prioritizing authentication methods that uphold user autonomy and security. In this context, the adoption of the SSI methods for authenticating system users offers significant advantages. SSI, as implemented within the system, eliminates the need for external authentication partners, thereby enabling autonomous and secure user authentication procedures.

Furthermore, the use of DIDs in the SSI process contributes to the autonomy and security of user authentication. DIDs can be autonomously stored, eliminating the need to transmit authentication data to external parties. Participants or system users have the option to store their DIDs in a cloud wallet provided by a third-party service, although this is entirely voluntary. Alternatively, DIDs can be stored locally, prioritizing security over convenience.

Moreover, the system's design reinforces trust by ensuring that no participant or system user has access to detailed information about the participating systems. From the perspective of a system user, interactions with the system appear as if they are accessing a single, monolithic database. Similarly, participating systems are isolated from direct connections to other systems, mitigating potential vulnerabilities. Only the federation controller and federation connector have access to the connection data of the autonomous systems, and they are bound by strict confidentiality measures. Under no circumstances do these systems disclose connection data to any other participant or system user. Consequently, the only connection that any participant or system user has within the system is to the federation controller.

This architectural framework establishes a high level of trust among participants, as sensitive data is never exposed to other participants or system users.

The robust privacy and security measures implemented within the system serve to enhance trust and confidence in the integrity of the platform.

The implementation of SSI and DIDs aligns with the principles of user-centric identity management, empowering individuals with control over their own digital identities. This approach enhances user autonomy and contributes to the overall security and trustworthiness of the system. By prioritizing user privacy and security, the system demonstrates a commitment to fostering a trustworthy and reliable environment for all participants.

3.8 Building Block View

The Building Block View serves as a crucial element in comprehending the SBFDBS architecture. This view delineates the system's static decomposition into distinct building blocks, providing a hierarchical representation across three essential levels.

At the foundational level, the SBFDBS architecture is solely depicted through black boxes. These abstract representations offer a high-level overview, highlighting the structural components without delving into details.

Moving to the second level, a more detailed perspective emerges. Here, the composition of these black boxes is elaborated upon, unveiling the internal constituents and the construction of these fundamental building blocks. This intermediate level bridges the gap between abstraction and detailed system understanding.

Finally, the third level ventures into a meticulous component-centric view. Every component within the SBFDBS system is scrutinized, unraveling the intricate web of interconnections and dependencies. This exhaustive breakdown elucidates the functional elements, their interrelationships, and the granular parts comprising the entire system.

Through this multi-level approach, the Building Block View navigates the structural landscape of the SBFDBS, offering progressively detailed insights into its architecture at various levels of abstraction. In this Level, the comprehensive framework of the entire SBFDBS is unveiled, depicted in Figure 3.3 through distinct black boxes. This diagram offers a panoramic view of the various systems constituting the SBFDBS and elucidates their interconnections and interactions.

The top-level buildup of the SBFDBS is composed out of four subsystems.

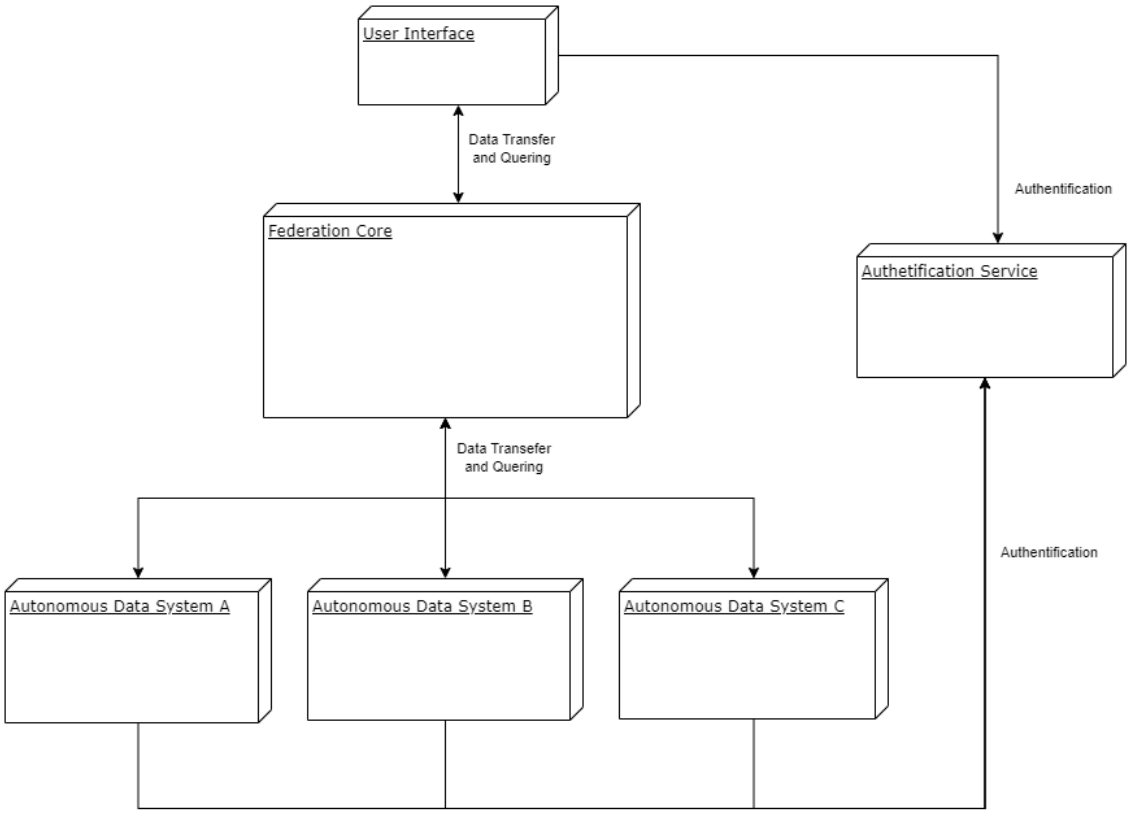

Figure 3.3: Level 1 Building Block View

Federation Core

The Federation Core serves as the central mediator within the SBFDBS. It receives, manages, and processes queries originating from the User Interface. Its primary functions include establishing connections with the connector deployed inside the autonomous Data Systems, aggregating queried data, and facilitating transmission back to the initial requester. Moreover, the Federation Core maintains an updated record of connected Systems, routinely verifying their connectivity.

User Interface

The User Interface functions as the primary interaction point for System Users. It enables users to construct queries through a web-based frontend. Furthermore, the User Interface facilitates user authentication via their DIDs to access the SBFDBS, utilizing the Authentication Service.

Autonomous Data Systems

These systems host the deployed federation connectors, providing essential data to the SBFDBS. The Autonomous Data Systems supply the SBFDBS with data and authenticate themselves through the Authentication Systems.

Authentication Service

The Authentication Service operates as an impartial verifier of the DIDs. It can operate locally within the same network as the Federation Core, or be deployed as a cloud-based service solution. When employing this service in the cloud, ensuring the trustworthiness of the service provider is critical.

3.8.1 Level 2 - Detailed Blackbox view

In this Level, a comprehensive elucidation of the constituent systems forming the SBFDBS is provided, delving into their intricate subsystems encapsulated within each primary system. Figure 3.4 provides a visual representation, offering a detailed overview of the functionalities inherent in each primary system and a comprehensive understanding of the precise components employed to facilitate these functionalities. This detailed exploration aids in unraveling the intricate network of operations within the SBFDBS, shedding light on the nuanced interplay between subsystems and their collective contribution to the system's overall functionality and coherence.

The Federation Core comprises two primary subsystems, one of which supports multiple instances, enabling vertical scaling within the Federation Core:

Federation Controller

The Federation Controller serves as the central mediator within and beyond the Federation Core. It acts as an intermediary for the Federation Connector and the User Interface, facilitating data queries, system logins, and seamless data transfer from autonomous data systems to the initial requester. Its responsibilities encompass forwarding new connections, aggregating data, compressing information, and efficiently transferring data from autonomous databases to the requester.

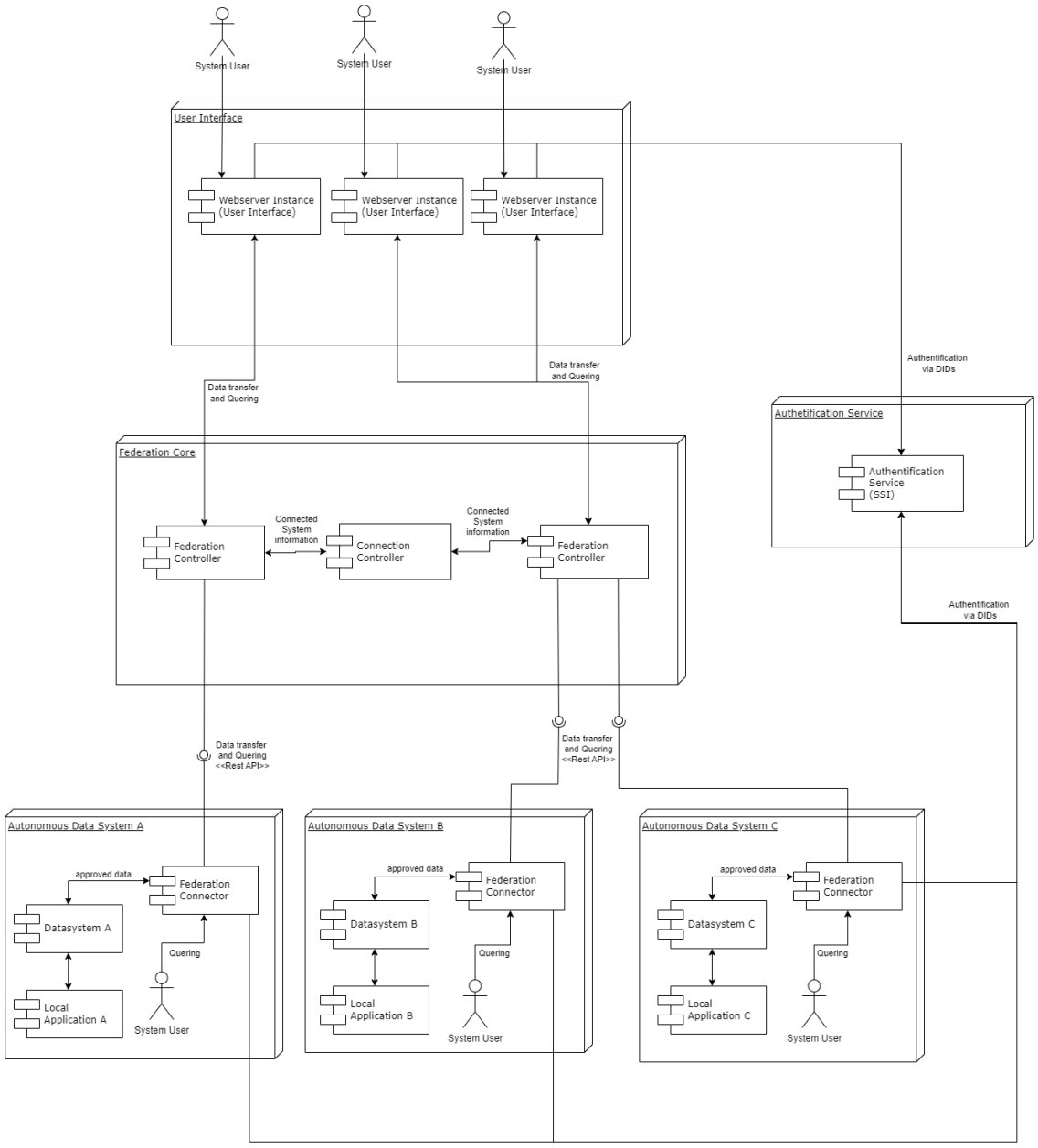

Figure 3.4: Level 2 Building Block View

Connection Controller

The Connection Controller functions as a lightweight system, monitoring all current connected Federation Connectors within the SBFDBS. It receives updates on new connections and gracefully handles disconnections by tracking each connection's status. Furthermore, it provides comprehensive information on connected Federation Connectors to other Federation Controllers upon request. It

also conducts health checks at intervals to ensure the continuity and accessibility of all connected systems, promptly eliminating inactive or non-responsive connections.

Federation Connector

Embedded within the Autonomous Data System, the Federation Connector serves as the linchpin linking the autonomous database to the Federation System. It facilitates the transmission of local data from the Autonomous System to the entire SBFDBS. All communication between the SBFDBS and the Autonomous Data System is channeled through this Connector. Additionally and optionally, it offers a user interface within the participants' internal enterprise network if wanted by the implementing participant.

Webserver

The User Interface primarily comprises multiple instances of the Webserver. These servers operate in a clustered setup, ensuring automatic scalability to meet varying demands and automatic restarts in case of critical system failures. Facilitating interaction with the Web-Based-Frontend, these servers primarily enable System Users to query data from the SBFBDS. Additionally, they serve as the interface for System Users to authenticate themselves through their respective DIDs and the Authentication Service.

Authentication Service

The Authentication Service functions as a lightweight system tasked with validating a System User's DID. This service can operate either within the local confines of the Federation Core or be managed within a cloud-based infrastructure for broader accessibility. This section marks the introduction to Level 3, offering an intricate Building Block view of the SBFDBS. It delineates the subsystems and their constituent components within this intricate system.

Displayed in Figure 3.5, this level expounds upon the individual software components nestled within the expansive SBFDBS framework. By delving into the precise functions of each component, this section provides a nuanced understanding of their interconnections and interactions. Moreover, it sheds light on the intricate network of ports and APIs employed for seamless communication between these components across the network.

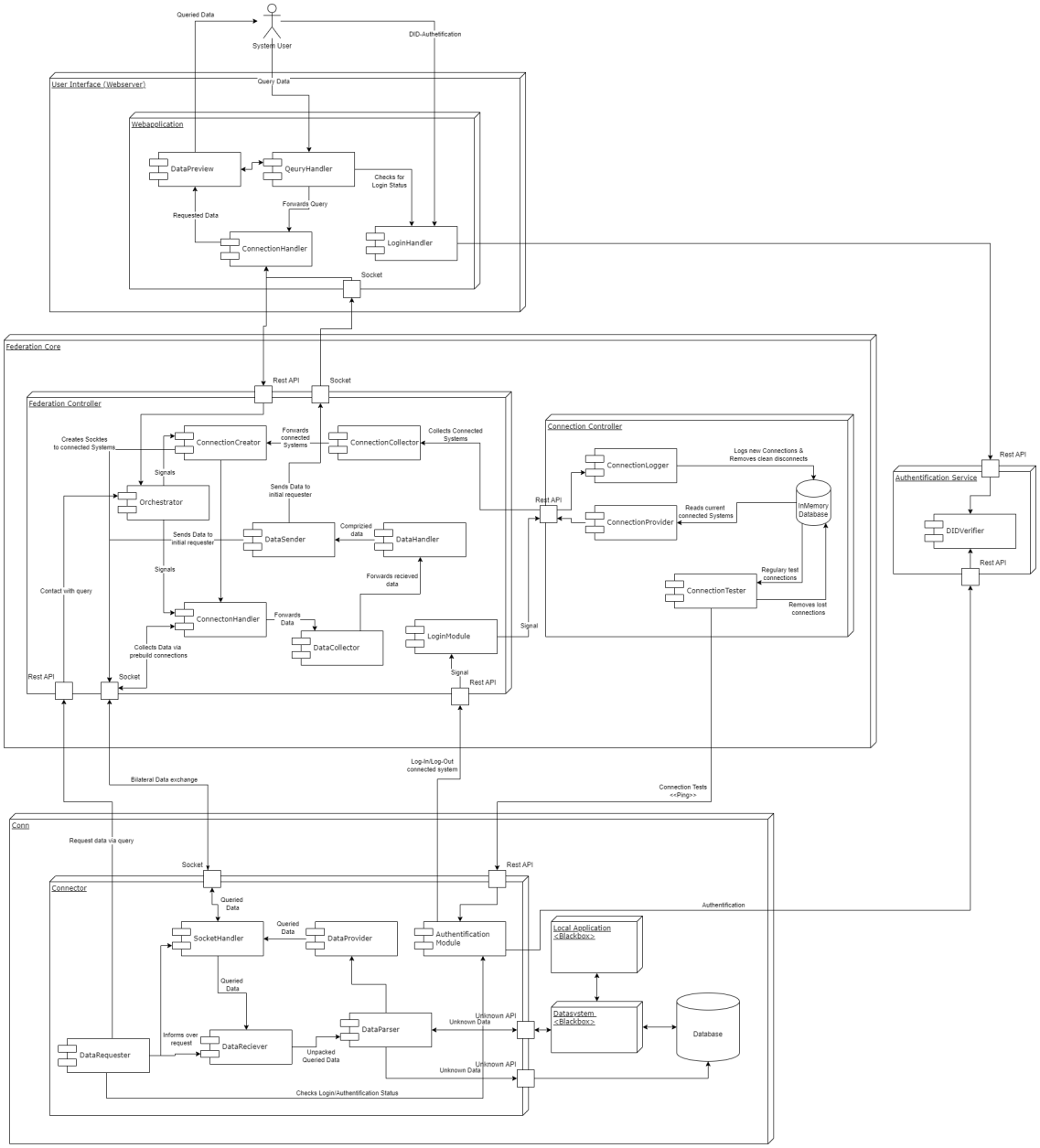

Figure 3.5: Level 3 Building Block View

3.9 SBFDBS Components explanation

A highly modular approach for the SBFDBS was chosen. The goal was that each component has exactly one job to complete, to make developing and maintaining as easy as possible.

To understand the importance of building modular software and the benefits of

having each component perform only one job, it is essential to consider the principles of software development, particularly the SOLID principles, and their relationship to modularity. Modularity in software design refers to the practice of breaking down a system into smaller, self-contained modules, each responsible for a specific functionality. This approach simplifies development and maintenance by isolating changes and reducing dependencies between components.

Modularity is important in software engineering for several reasons. Firstly, it enhances maintainability, as highlighted by [GSA15], who emphasize that modularization leads to an increase in the level of quality attributes such as maintainability, portability, reusability, interoperability, and flexibility. This aligns with the SOLID principle of "Single Responsibility," which advocates for each class or module having only one reason to change, thereby simplifying maintenance.

Furthermore, modularity contributes to flexibility, as discussed by [Par02], who emphasizes that modularization improves the flexibility and comprehensibility of a system while allowing the shortening of its development time. This aligns with the Open/Closed Principle in the SOLID principles, which states that software entities should be open for extension but closed for modification, promoting flexibility without the need for extensive modification of existing code.

In the following Sections, each of the Components of the Federation Core and their functions are explained in short.

3.9.1 Federation Core Components Explained

This section elucidates the functionalities and roles of critical components operating within the Federation Core.

Orchestrator

The Orchestrator, positioned as the nucleus within the Federation Controller, assumes a pivotal role in orchestrating the entire operational flow. As its name implies, this component adeptly manages initial interactions from System Users, meticulously oversees the intricate process of connection establishment, handles data collection, applies compression techniques, and facilitates seamless data transfer. Remarkably, the Orchestrator's scope is purposefully limited to orchestrating; it acts as the signaling hub and manager of other components, ensuring a coherent and synchronized operation throughout the system's functioning.

Connection Creator

Tasked with the crucial responsibility of establishing essential connections necessary to fulfill queries originating from the Orchestrator, the Connection Creator is meticulously dedicated to crafting and initializing sockets exclusively for efficient data transfer. Once these connections are successfully established, they are seamlessly forwarded to the Connection Handler, which assumes further responsibilities for advanced data management and oversight.

Connection Collector

Operating as a vital data conduit, the Connection Collector meticulously gathers essential information regarding all presently connected Autonomous Data Systems, deriving these crucial insights directly from the Connection Controller. Subsequently, this valuable information is relayed and seamlessly integrated into the operational workflow by the Connection Creator, streamlining the subsequent data transmission processes.

Connection Handler

Exercising comprehensive control over all active connections, the Connection Handler diligently monitors and manages these channels, overseeing them to preempt any potential interruptions in the ongoing file transfers. Simultaneously, it maintains a keen vigilance over clean connection closures in the event of errors, while also serving as the primary provider of open connections to other components upon request.

Data Collector

Harnessing the connections masterfully managed by the Connection Handler, the Data Collector engages in a piece-by-piece retrieval of data fragments from external federation connectors. This intricate process culminates in the systematic forwarding of fully transcribed data fragments, segmented to the Data Handler for subsequent processing and unification.

Data Handler

Upon the reception of requested data from the Data Collector, the Data Handler orchestrates the compression and amalgamation of all incoming data streams into a singular, streamlined file format. This compression ensures enhanced efficiency in data transmission back to the original requester, culminating in a comprehensive file that is then seamlessly relayed to the Data Sender for final transmission.

Data Sender

Assuming the critical role of the last mile in data transmission, the Data Sender oversees the seamless transmission of the processed and compiled data back to the original requester, completing the final stage of data transfer within the Federation Core.

Login Module

Functioning as an enabler for the external Federation Connector, the Login Module diligently provides an essential API that facilitates seamless logins and logouts within the Federation Core, ensuring enhanced control and management capabilities.

3.9.2 Connection Handler Components Explanation

This section elucidates the functionalities and roles of critical components operating within the Connection Handler.

Connection Logger

The Connection Logger assumes a pivotal role within the Connection Handler, responsible for logging newly connected systems into the in-memory database housed within its domain. Moreover, it manages the removal process of cleanly logged-out systems from the internal database, ensuring a current and accurate record of system connections.

Connection Provider

Operating as an essential interface for Federation Controllers, the Connection Provider offers a robust REST API. This API serves as a gateway for Federation Controllers to access an up-to-date list of all currently connected federation controllers logged into the SBFDBS. Drawing information directly from the internal database, it furnishes a comprehensive overview of active connections within the system.

3.9.3 Federation Connector Components Explanation

This section provides comprehensive insights into the pivotal roles and functionalities of critical components residing within the domain of the Connection Controller. These components play instrumental roles in facilitating seamless communication and access to crucial connection-related information within the SBFDBS.

Connection Tester

The Connection Tester serves as a system maintenance component, conducting regular checks on existing logged-in Federation Connectors to assess their network availability. Should it detect any unreachable but logged-in Federation Connector, this component initiates its removal from the internal database.

Data Requester

Operating as the strategic orchestrator of data requests within the SBFDBS, the Data Requester navigates connection attempts via a robust REST API, transmitting intricate details regarding the queried data. Subsequently, it primes the Data Receiver to expect socket connection requests from the Federation Core, facilitating a seamless communication pipeline. However, its operational status is contingent upon successful authentication, a validation process verified by interfacing with the authentication module.

Data Receiver

Positioned as the designated recipient of requested data streams facilitated by the Socket Handler, the Data Receiver skillfully navigates the reception of incoming data. Beyond mere reception, it processes this data by applying decompression algorithms and robust integrity checks, ensuring data reliability and consistency.

Socket Handler

Functioning as the vigilant guardian of socket connections initiated during data transmissions, the Socket Handler manages these connections, extending seamless access to the Data Provider and Data Receiver upon request, facilitating an optimized flow of data transfer operations.

Data Provider

Acting as the crucial intermediary between the Federation Core and the requested data, the Data Provider processes the sought-after data. Drawing from the Data Parser, this component compresses the data, ensuring an optimized format for efficient data transmission.

Data Parser

Acting as the pivotal interface connecting the Federation Connector with the Autonomous Data System, the Data Parser proficiently extracts data from the autonomous system. Each participant's operational engineers or developers are required to integrate their requisite business logic into this data parser, facilitating the retrieval of data from their autonomous data system and subsequent provision to the Data Provider, thereby enabling its dissemination throughout the entirety of the SBFDBS.

3.9.4 Authentication Service Components Explained

This section elucidates the functionalities and roles of critical components operating within the Authentication Service.

DID Verifier

Exercising critical oversight over the authentication process, the Authentication Module facilitates the Federation Connector's seamless authentication within the authentication system. Upon successful authentication, it promptly triggers the Login Module within the Federation Core to initiate the Federation Connector's login to the SBFDBS via a REST API.

3.9.5 User Interface Components Explained

This section elucidates the functionalities and roles of critical components operating within the User Interface.

Login Handler

Similar in function to its counterpart within the Federation Controller, this component assumes responsibility for authenticating system users via their DIDs. Leveraging the authentication service, it verifies user credentials, ensuring only permitted access to the system.

Connection Handler

Analogous to its counterpart in the Federation Controller, this component oversees the establishment and management of socket connections, adeptly handling data transmission to and from the Federation Core. It operates by processing system user-defined queries and initiating their transmission to the Federation Core via a REST API. Upon successful data transmission, it unpacks received data for further processing. Lastly, it cleanly closes the socket connection after a successful or failed transmission.

Data Preview

The Data Preview component offers system users the vital capability to preview data before initiating any download processes. It serves as a window into the queried data, displaying essential information such as file names, sizes, and other relevant details. After confirming the preview, it informs the Query Handler about the final query, streamlining subsequent processes.

Query Handler

Operating as the interface between system users and the data querying process, the Query Handler allows the users to construct and formulate queries effectively. Upon assembling complete queries, this component efficiently transmits them to the Connection Handler, initiating the data retrieval process.

3.10 Runtime View

Transitioning from the detailed breakdown of the system's building blocks to the exploration of its runtime dynamics, the focus shifts towards illustrating the practical operation and interaction of the SBFDBS within real-world scenarios. This transition underscores the significance of understanding how the individual components come together to fulfill the system's objectives during execution.

In this section, I delve into the Runtime View of the SBFDBS within the context of the Arc42 template, aiming to provide a clear depiction of the system's behaviors and interactions during operation. Through the presentation of key use cases in the form of sequence diagrams, I elucidate the operational intricacies of the SBFDBS, shedding light on its core functionalities and the manner in which they unfold in real-world scenarios.

By focusing on the three most utilized cases, this section aims to offer a comprehensive insight into the runtime behavior of the SBFDBS, allowing stakeholders to grasp the operational efficacy and intercommunication among its components. Through the visual aid of sequence diagrams, I aim to facilitate a more profound understanding of how the system orchestrates its various processes and interacts with external entities to achieve its objectives.

Figure 3.6 provides a detailed portrayal of the sequential steps involved when a user interacts with the frontend to initiate data retrieval from the autonomous databases. Initially, the user interfaces with the frontend, initiating a data request process. This request is then transmitted to the federation controller, which serves as the central hub for managing data requests within the system. Subsequently, the federation controller leverages its connectivity with all currently active federation connectors, effectively distributing the data retrieval requests across the network. The federation connectors, acting as intermediaries, collect, handle, and facilitate the transfer of the requested data back to the federation controller. Within the federation controller, an orchestrated process ensues, wherein all requested data is aggregated and consolidated. Upon successful compilation of all requested data, the federation controller proceeds

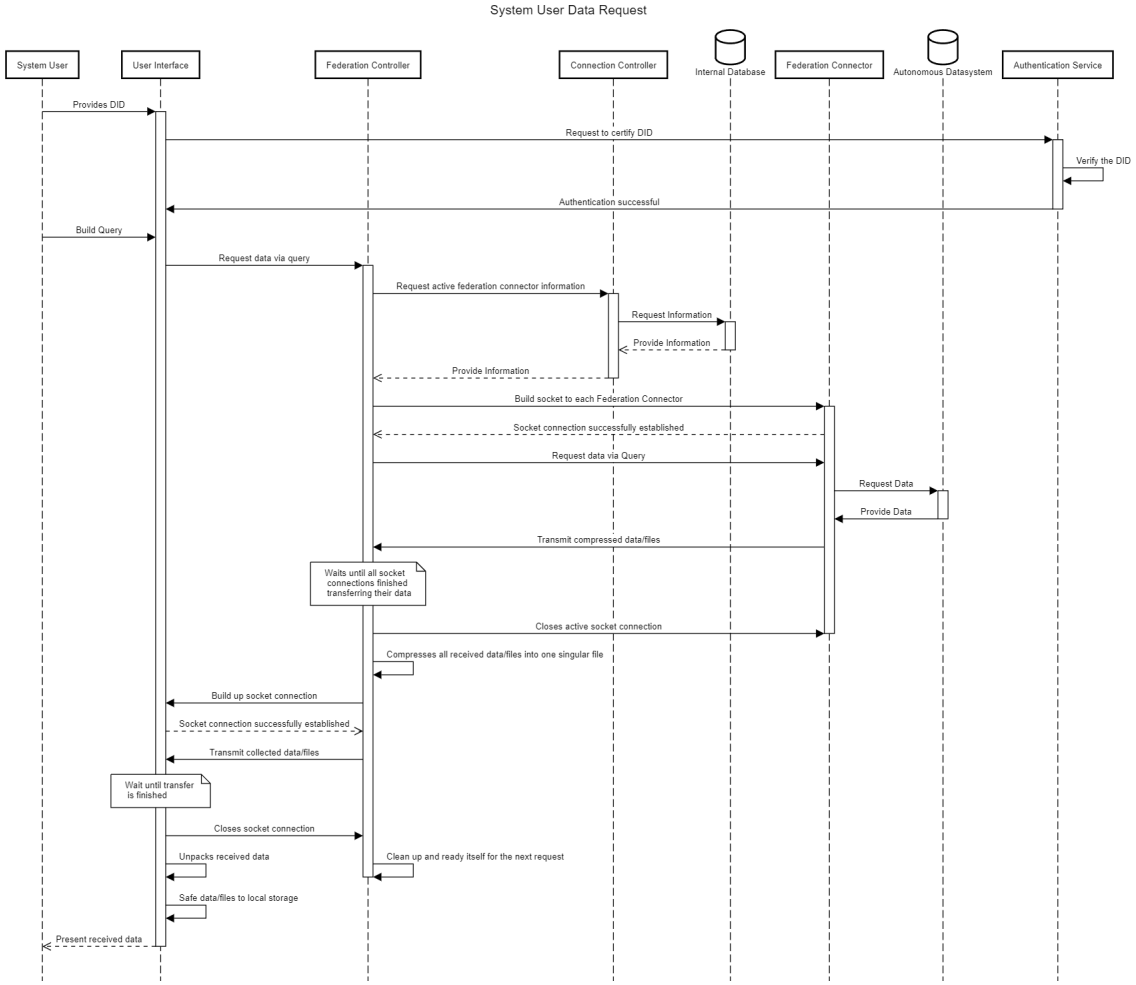

Figure 3.6: Default System User Request

to perform further processing tasks to refine and prepare the data for transmission back to the user interface. This comprehensive process ensures seamless data retrieval and transfer, maintaining the integrity and efficiency of the system's operations.

In Figure 3.7, a detailed depiction of the federation connector's login procedure unfolds. Upon initiation the federation connector undergoes an essential boot-up sequence to validate its operational status thoroughly. Following this initial boot-up phase, the federation connector autonomously proceeds to authenticate its login credentials with the federation controller. Upon receipt of the login request, the federation controller initiates communication with the Authentication Service to assess the integrity and validity of the provided login credentials. Upon receiving confirmation of successful verification, indicating the authentication of the credentials, the federation controller then delegates the verified authentication process to the connection controller. The connection controller

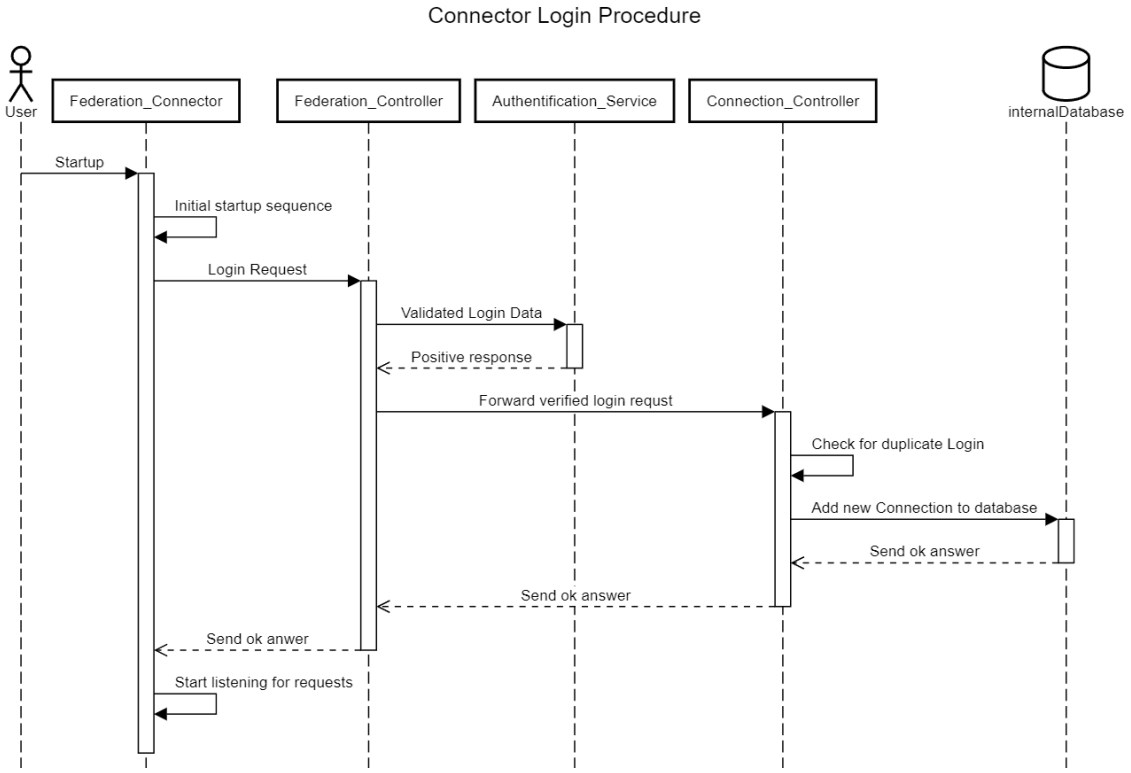

Figure 3.7: Federation Connector Login Procedure

examines the request, scanning for any instances of duplicate entries. When the connection controller identifies no duplicate entries, it promptly notifies the federation controller of the affirmative login status. The federation controller then communicates this positive response to the federation connector. Subsequently, empowered by the confirmed login status, the federation connector transitions into an active listening state, primed to receive and process incoming requests. Conversely, in the event of a negative login response, signifying a discrepancy or irregularity in the authentication process, the federation connector gracefully initiates a shutdown procedure. It notifies the user regarding the termination, providing the reasons for the shutdown.

Figure 3.8 illustrates the automated disconnection process of a federation connector within the SBFDBS, emphasizing the system's commitment to maintaining optimal connectivity by ensuring the presence of only healthy connections. This pivotal process is orchestrated by the connection controller, which systematically conducts regular health checks on all currently active federation connectors.

Initially, the connection controller retrieves a comprehensive list of all connected federation connectors from its internal database. Subsequently, it initiates a health check procedure by dispatching individual health check requests to each

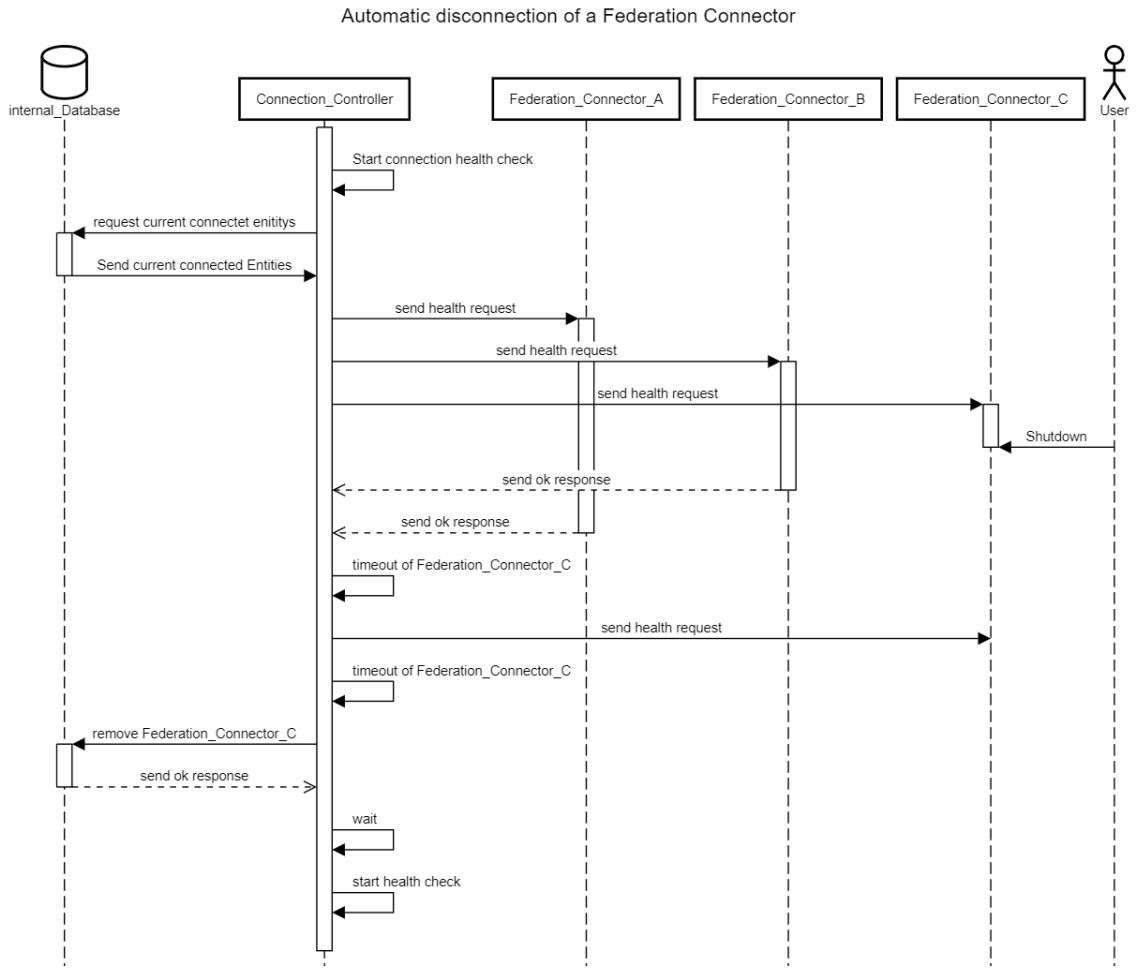

Figure 3.8: Automatic disconnect of a federation connector

connector, monitoring their responses. These responses are categorized as positive, negative, or timed out, with a predefined timeout threshold of one second enforced to swiftly identify and address any non-responsive connections.

In the event of a timeout occurrence, indicating a potential operational issue, the connection controller makes a concerted effort to re-establish communication with the affected connector. If the connection persists in timing out, the connection controller promptly removes it from the internal database, ensuring the integrity of the system's connectivity. Additionally, any federation connector that responds negatively to any health check, irrespective of the reason, is promptly delisted from the database to preemptively mitigate any potential disruptions.

This meticulous process is repeated at regular intervals of three seconds, underscoring the system's dedication to maintaining a roster of exclusively healthy connections within the SBFDBS. By systematically executing these health checks

and swiftly addressing any anomalies, the system optimizes its operational efficiency while safeguarding against potential connectivity issues.

3.11 Technical Dept

While the SBFDBS demonstrates notable advancements in the realm of secure and bilateral data transfers, there exist certain technical debt considerations inherent within its architecture. These potential shortcomings warrant meticulous examination to ensure the system's robustness and reliability in real-world applications.

The most prevalent issue identified is the existence of a single point of failure, namely, the federation connector within the SBFDBS. The federation connector acts as the intermediary, and any malfunction or abrupt cessation of communication by this component between system users and the autonomous databases renders the entire system inoperable. While employing multiple instances could potentially mitigate this concern by redirecting users to a functioning federation controller, such actions may result in the loss of data transfer progress.

Furthermore, the single point of failure poses significant challenges in terms of operational reliability and security. Operation engineers bear the responsibility of maintaining and securing the federation controllers. However, in the event of malicious intent by an operation engineer, there exists the risk of distributing harmful software to unsuspecting users. This compromises the integrity of the data received from federation connectors, which users expect to originate from secure and verified sources.

Moreover, the presence of a single point of failure raises concerns regarding potential discriminatory practices. Operation engineers hold the authority to selectively ban specific autonomous databases from participating in the SBFDBS for various reasons, whether political, personal, or otherwise. This underscores the importance of rigorous oversight of all operation engineers and any modifications made to the federation core's codebase.

Another technical debt inherent within the architecture of the SBFDBS is the potential issue of an overloaded federation controller. As these controllers are deployed within participant's networks, the SBFDBS and its operational engineers lack authority over the type of system or machine on which the federation connector is deployed. If a federation connector is deployed on a sluggish machine or experiences an exceptionally high volume of data requests, it can significantly impact the response time of each individual request involving data from that

federation connector, thereby reducing the overall speed of data transmission within the SBFDBS or, in severe cases, impeding it entirely.

Given the autonomous and self-sovereign nature of the participating data systems, addressing this issue requires a collaborative approach. The primary recourse for mitigating such performance issues involves the operational engineers of the federation core initiating communication with the developers of the problematic autonomous databases. By providing guidance and recommendations aimed at optimizing the performance of these systems, operational engineers can potentially alleviate the strain on federation connectors and improve overall system efficiency.

However, in instances where performance issues persist despite continued engagement with autonomous database developers, more drastic measures may be necessary. One such measure, albeit less preferable, involves excluding the problematic autonomous databases from the SBFDBS network. This action would entail the loss of access to data shared by the excluded databases within the SBFDBS, potentially disrupting data availability and collaboration among participants.

In conclusion, while the SBFDBS represents a significant milestone in the pursuit of secure and bilateral data transfers in engineering domains, acknowledging and addressing the technical debt inherent within its architecture is paramount. By proactively identifying and mitigating potential shortcomings, stakeholders can ensure the system's continued efficacy, resilience, and relevance in an ever-evolving technological landscape.

4 Validation

4.1 Introduction

The validation chapter of this paper serves as a critical examination of the proposed SBFDBS system design, within the context of its practical implementation. This chapter aims to empirically assess the efficacy and functionality of the SBFDBS prototype, developed in accordance with the principles outlined in the Arc42 Template. By subjecting the prototype to validation procedures, I seek to ascertain the extent to which the system design aligns with theoretical frameworks and meets the intended objectives of securely facilitating bilateral transfers of engineering data while upholding user data autonomy and sovereignty.

In the following chapter, I provide an overview of the validation process, detailing the methodology employed, key metrics for evaluation, and the overarching objectives guiding the validation efforts. Additionally, I contextualize the validation within the broader landscape of existing theoretical literature, highlighting areas of convergence and departure between the proposed system design and established frameworks. Through this validation endeavor, I aim to validate the SBFDBS prototype and contribute empirical insights that advance the understanding of secure data transfer systems and inform future iterations of the SBFDBS design.

4.2 Goals of the Validation

The validation process of the SBFDBS prototype is designed to scrutinize and substantiate the efficacy of the system design, ensuring its seamless operation in accordance with the designated objectives. The overarching goals of the validation endeavor are multifaceted, encompassing critical aspects of functionality, interoperability, extensibility, and adherence to data autonomy and sovereignty principles.

The primary objective of the validation is to demonstrate that the SBFDBS prototype operates as intended, facilitating the transfer of diverse data types across

the network. Emphasis is placed on verifying the system's ability to preserve data autonomy and sovereignty throughout the transfer process to each participant, thereby safeguarding users' control over their data assets.

Central to the validation process is the assurance of seamless interoperability, ensuring the SBFDBS prototype's capability to facilitate data transfers across heterogeneous systems without compromising data integrity or security. The validation efforts aim to validate the system's ability to accommodate various data formats and protocols.

Another key focus of the validation is to validate the ease with which new autonomous data systems can be integrated into the SBFDBS ecosystem, even at runtime. The validation process seeks to demonstrate the system's flexibility in accommodating dynamic changes and additions, without necessitating extensive reconfiguration or disruption to existing operations.

Furthermore, the validation endeavors to showcase the extensibility of the SBFDBS prototype by incorporating legal contracts onto files, requiring users' acceptance before accessing associated data. This functionality serves to underscore the system's adaptability to evolving regulatory frameworks and user requirements, enhancing trust and compliance within the system.

In addition to validation, the prototype serves as a demonstrative tool, illustrating the core functionalities of the SBFDBS in a tangible and accessible manner. By providing a vertical slice of the software design, the prototype offers stakeholders a comprehensive understanding of the system's capabilities and potential applications.

It's important to mention that while the prototype validation aims to back up the effectiveness of the SBFDBS design, certain aspects had to be kept out of scope due to time limitations. Specifically, the focus was not placed on validating aspects that are already well-established and solved, such as user authentication using DiDs, implementing HTTPS for secure communication between webservers and the user interface, or logging transferred data within the system. These requirements have proven solutions that are relatively straightforward to incorporate into the SBFDBS with more time and resources on hand.

4.3 Metrics

In the evaluation of the SBFDBS prototype, it is imperative to define and analyze key metrics across various dimensions of system performance and functionality. This chapter outlines the metrics employed to assess the efficacy of the SBFDBS

design in facilitating secure and bilateral transfers of engineering data while preserving data autonomy and sovereignty.

4.3.1 Data Transfer Performance Metrics

Data transfer performance metrics gauge the efficiency and effectiveness of the SBFDBS prototype in facilitating the seamless exchange of engineering data among participants. Key metrics in this category include:

Throughput

Throughput quantifies the volume of data transferred across the network within a specified time frame, typically denoted in bytes per second or records per unit time. A higher throughput signifies an enhanced capacity for data transmission and processing within the SBFDBS ecosystem. This metric is determined by transmitting sizable files through the system and assessing the speed at which they traverse from endpoint to endpoint.

Latency

Latency denotes the duration between the initiation of a data transfer request and its completion, encompassing the time delay incurred during data transmission. Lower latency values indicate swifter data transfer and diminished waiting periods for users accessing or sharing engineering data. In the context of my study, latency measurement involves capturing the total duration for data collection, processing, and transmission across the network.

Reliability

Reliability evaluates the consistency and dependability of data transfers within the SBFDBS prototype, considering factors such as error rates, packet loss, and system downtime. This metric aims to ensure uninterrupted data exchange and minimal disruption to user operations. Reliability assessment entails subjecting data to extensive transmission through the system and subsequently verifying its integrity upon completion to ascertain the system's robustness and reliability.

4.3.2 Data Autonomy and Sovereignty Metrics

It is challenging to quantify data autonomy using precise numerical metrics. Therefore, the metrics employed aim to ensure that participants of the SBDFBS consistently retain autonomy and control over their data and data systems. These metrics encompass:

Participants Control

Participants control assesses the extent of autonomy users maintain in managing and governing their data within the SBFDBS ecosystem. Users should have the capability to modify or delete the data they are sharing with the SBFDBS at any given time. Furthermore, these modifications must be promptly reflected across the entire system, ensuring instantaneous updates and preserving user autonomy.

Legal Control

Legal control emphasizes the system's obligation to afford each participant the option to accompany their shared data with requisite legal documents. These documents must be acknowledged and agreed upon by all interested parties before any data transmission occurs. This mechanism safeguards participants by ensuring that the data they share is utilized only within the agreed-upon and accepted context, thereby preserving their autonomy and legal rights over their shared resources.

4.3.3 Extensibility Metrics

Extensibility metrics assess the scalability and adaptability of the SBFDBS prototype to accommodate evolving user requirements and technological advancements. Key metrics in this category include:

Compatibility

Compatibility measures the ability of the SBFDBS prototype to interact and integrate with diverse data formats. It assesses the ease of data interchange between the SBFDBS ecosystem and external systems, ensuring interoperability and compatibility across disparate technologies.

Ease of Integration

Ease of integration measures the effort required to incorporate new autonomous data systems and functionalities into the SBFDBS ecosystem. It assesses the flexibility and modularity of the system architecture, enabling seamless integration and extension with minimal disruption to existing operations.

Scalability

Scalability evaluates the ability of the SBFDBS prototype to scale up or down in response to changes in data volume, user load, or system requirements. It measures factors such as performance degradation, resource utilization, and system responsiveness under varying workload conditions, ensuring uninterrupted service delivery and optimal resource allocation.

By defining and analyzing these metrics, the evaluation process aims to provide insights into the effectiveness, efficiency, and robustness of the SBFDBS prototype in fulfilling its objectives of secure and bilateral data transfers while preserving data autonomy and sovereignty.

4.4 Data Acquisition Methodology

The validation of the defined metrics for the SBFDBS necessitates a comprehensive data acquisition methodology, encompassing both automated processes and manual testing of individual use cases. Automated processes involve systematically executing predefined tasks and measuring the resulting outcomes, while manual testing involves hands-on manipulation of system components to assess specific metrics, particularly those challenging to quantify.

To exemplify, in assessing reliability, an automated process repeatedly downloads files through the SBFDBS prototype, meticulously monitoring each transfer for errors or incompleteness. The overall reliability of the system is then determined by analyzing the frequency of corrupted or incomplete transfers relative to the total number of transfers conducted during the testing period.

Additionally, in evaluating data autonomy, manual testing is employed to simulate user interactions and gauge the system's responsiveness to user-driven modifications. This entails manually altering attached systems and modifying shared data within a participant's repository, followed by observation and documentation of the system's response to these changes. Through manual testing,

subtle nuances in user autonomy can be discerned, providing valuable insights into the system's adherence to data autonomy principles.

In another instance, to measure participants' control, a systematic approach involves users initiating data modifications within the SBFDBS ecosystem and assessing the promptness and accuracy of system-wide updates in reflecting these changes. This automated process ensures that users retain full autonomy over their data, with modifications being promptly and accurately propagated throughout the system.

By employing a combination of automated processes and manual testing methodologies, the data acquisition phase endeavors to comprehensively validate the defined metrics for the SBFDBS prototype, ensuring robustness, reliability, and adherence to data autonomy principles.

4.5 Prototype

The forthcoming chapter provides an in-depth exploration of the prototype for the SBFDBS. This prototype stands as a pivotal embodiment of the theoretical constructs previously discussed, translating abstract concepts into a concrete system. Developed through a process of iterative refinement, the prototype encapsulates fundamental principles of data autonomy and sovereignty critical for facilitating trustworthy engineering data exchange. Within these pages, I delve into the intricacies of the SBFDBS prototype, offering comprehensive insights into its architecture, functionalities, and validation approaches.

It is imperative to underscore that the prototype outlined in this chapter serves as just that: a prototype. It represents a simplified rendition of the overarching system design, intended primarily as a platform for validating the metrics delineated earlier. Unlike a minimal viable product (MVP), which aims for a comprehensive yet basic functionality, this prototype is best conceptualized as a vertical slice, encompassing critical components essential for proofing the system's design concept. While it may lack the full breadth and depth of features expected in a production-ready solution, it nonetheless provides a tangible framework for assessing the viability and efficacy of key system functionalities. This distinction is pivotal, as it helps manage expectations regarding the prototype's scope and capabilities. Rather than aiming for completeness, the focus is squarely on ensuring that essential components are present and functional, thereby enabling robust validation of the underlying design principles. This pragmatic approach allows for targeted evaluation and iteration, fostering a more agile and effective development process.

4.5.1 Technologies

The technologies employed in the development of the prototype encompass a strategic selection of widely adopted tools and frameworks within the contemporary industry landscape. These choices were made with a keen eye towards leveraging robust, industry-standard solutions while also ensuring cost-effectiveness.

For the backend components, specifically the federation connector and federation controller, a Spring application was utilized. Spring offers a comprehensive ecosystem for building enterprise-grade applications, and its support for basic web functionality and websockets proved instrumental in facilitating real-time communication. Initial communication between components is managed through HTTP(s) endpoints via REST requests, while data transfer is facilitated through Java Sockets, dynamically created as needed and active only during the transmission process. This combination of technologies ensures efficient and secure communication between backend modules.

On the frontend, a Node.js application with Angular was adopted to deliver a user-friendly interface. Node.js provides a scalable and efficient environment for running JavaScript code on the server side, while Angular offers a robust framework for building dynamic web applications. The frontend interface serves as a platform for users to interact with the system, displaying available files/contracts and providing intuitive buttons for downloading them. Similar to the backend, communication with the frontend is achieved through HTTP(s) resources via REST calls, ensuring seamless interaction between frontend and backend components.

Additionally, each of these applications is packaged within Docker containers, further enhancing their portability and facilitating ease of testing. Dockerization enables the rapid deployment of isolated environments, making it simple to simulate new participants within the SBFDBS. This approach streamlines the testing process and promotes scalability, allowing for the efficient evaluation and validation of the system's functionality across diverse scenarios and environments.

4.5.2 Versions

The following frameworks and versions delineate the technologies employed in developing the prototype. The versions are provided for traceability purposes, ensuring transparency and accountability throughout the development process.

Backend (Federation Controller/Federation Connector)

- Gradle 8.5
- Kotlin 1.9.20
- Groovy 3.0.17
- Ant 1.10.13
- Java Virtual Machine (JVM) 17.0.10

User Interface

- Node Package Manager (NPM) 10.2.4
- Angular 17.1.0
- Node 20.11.0
- @angular-devkit/architect 0.1701.0
- @angular-devkit/build-angular 17.1.0
- @angular-devkit/core 17.1.0
- @angular-devkit/schematics 17.1.0
- @schematics/angular 17.1.0
- RxJS 7.8.1
- TypeScript 5.3.3
- Zone.js 0.14.3

Deployment

- Docker 25.0.3

4.5.3 Hardware specifications

To ensure comprehensive traceability and to due to the high impact of the Federation Controller's processing power on the performance of the overall SBFDBS, this section presents the technical specifications of the machines utilized in the validation process. These machines consist of three distinct setups: a Desktop Windows PC which was primarily used for development, testing, and validation as well as two additional machines — a Windows Laptop and a Raspberry Pi 4 with Ubuntu — as benchmarks for system compatibility.

Windows Desktop PC Specifications

The primary machine utilized for development, testing, and validation is equipped with the following hardware configuration:

- CPU: 13th Gen Intel(R) Core(TM) i7-13700KF @ 3.4 GHz
- RAM: 64GB 4000MHz DDR5 SDRAM
- Storage: WD_BLACK SN770 4000MB/s Read 2000MB/s Write
- Operating System: Microsoft Windows 11 Pro
- OS-Version: 10.0.22631 Build 22631
- Network: Realtek Gaming 2.5GbE Family Controller

The hardware configurations of the two machines utilized in the compatibility tests of the validation process are as follows:

Windows Laptop PC Specifications

- CPU: Intel(R) Core(TM) i5-10500H @ 2.5GHz
- RAM: 16GB 1600MHz DDR4 SDRAM
- Storage: Phison Electronics PS5012 3200MB/s Read 1000MB/s Write
- Operating System: Microsoft Windows 11 Pro
- OS-Version: 10.0.22631 Build 22631

- Network: Intel Wi-Fi 6 AX200 160MHz

Raspberry Pi 4 Specifications

- CPU: Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz
- RAM: 8GB LPDDR4-3200 SDRAM
- Storage: KIOXIA EXCERIA microSD 100MB/s Read 100MB/s Write
- Operating System: Linux Ubuntu
- OS-Version: 22.04.4 LTS
- Network: 5.0 GHz IEEE 802.11ac wireless, BLE Gigabit Ethernet

After careful deliberation, these hardware specifications were deemed essential for replication or comparison of the validation process. Other hardware specifications, such as the graphics card, were found to either insignificantly impact measured metrics or have no bearing on the overall results.

4.5.4 Architecture

Within this chapter, a comprehensive exploration of the prototype's architecture is conducted. As previously emphasized, the prototype deliberately focuses on implementing core functionalities essential for gathering requisite data to validate the predefined metrics. This data serves as empirical evidence to ascertain the efficacy of the SBFDBS in fulfilling its intended purpose. Consequently, the architecture of the prototype mirrors that of the SBFDBS, albeit with reduced complexity.

The image depicted in Figure 4.1 shows the architectural blueprint of the prototype, providing a visual representation of its structural elements and interconnections. Through elucidating the architectural intricacies, this chapter aims to elucidate the underlying framework upon which the prototype is built, facilitating a more in-depth understanding of its design rationale and operational dynamics.

Upon examination, it becomes evident that certain components were deliberately omitted from the prototype. Notably, the Authentication System and the Connection Controller are entirely absent. This deliberate omission stems from the realization that these components are unnecessary for validating the core

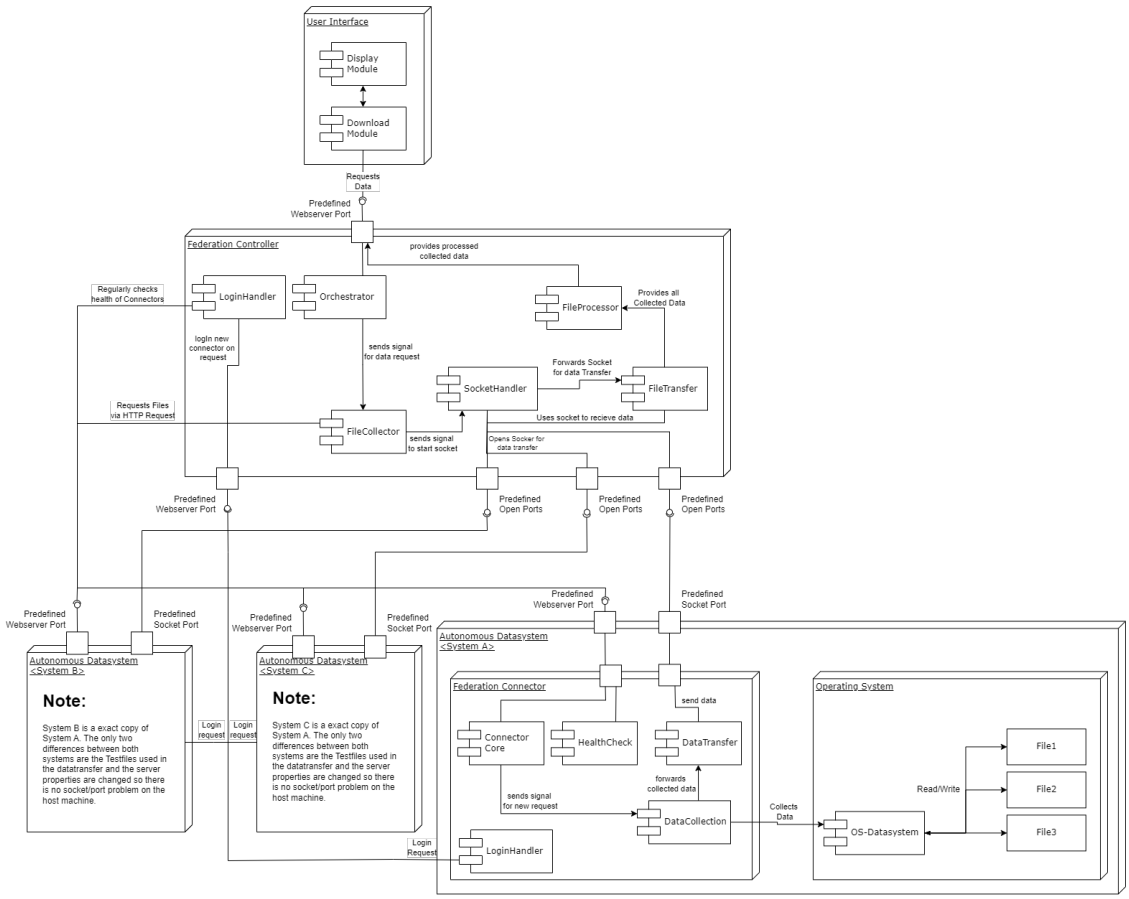

Figure 4.1: Prototype Architecture

principles of the SBFDBS. Similarly, the optional frontend within the Federation Connector of autonomous data systems A, B, and C is omitted, as it is deemed superfluous for validation purposes.

However, the three main components of the SBFDBS—namely, the User Interface, the Federation Controller, and the Federation Connector—are implemented and fully operational within the prototype.

User Interface Component Functionality

The User Interface serves as a foundational application, offering sufficient interactivity to display all available data, review associated contracts for each file, and download files selectively or collectively. This minimalist yet functional interface is powered by an Angular application that operates independently and can be deployed separately from other components. Notably, multiple instances

of the frontend can be instantiated and deployed as needed, ensuring scalability and flexibility in user interactions.

Federation Controller Functionality

In the prototype, the Federation Controller assumes the pivotal role of mediating between the user and the autonomous data systems. It manages all current connections to Federation Connectors and efficiently processes requests for data information and transmission originating from the User Interface.

Additionally, the Federation Controller performs regular health checks on all connected Federation Connectors by issuing HTTP requests to each connector. This proactive approach ensures that only healthy and reachable connectors are utilized for data transmission.

Upon receiving a request to transmit data, the Federation Connector checks all available sources to ascertain the presence of the requested data. If a Federation Connector lacks the requested data, it is disregarded in subsequent procedures. Conversely, if a Federation Connector possesses the requested data, a new socket is dynamically created per required connector.

Subsequently, upon socket creation and establishment, a signal is dispatched to the respective Federation Connector to initiate the transmission of the requested data via the newly opened socket. Upon completion of data transmission, the socket is promptly terminated to optimize resource utilization.

Upon successful retrieval of all requested data, the Federation Controller undertakes data processing, including compression for enhanced transmission efficiency. Once compression is completed, the Federation Connector promptly transmits the requested data back to the frontend to fulfill the initial user request.

Post-transmission, all temporary data and files generated throughout the request are promptly purged, ensuring efficient resource management and maintaining system cleanliness. This systematic approach ensures seamless and efficient data transmission while prioritizing system integrity and resource optimization.

Federation Connector Functionality

The Federation Connector serves as the primary data source within the SBFBDS, facilitating seamless communication between the autonomous data systems and the Federation Controller. Upon initialization, the Federation Connector automatically logs into the Federation Controller and establishes a persistent connection, thereby enabling seamless data transmission.

Subsequently, the Federation Connector provides an HTTP Request Interface for health checks from the Federation Controller, ensuring continuous monitoring and maintenance of system health.

Upon receiving a data request from the Federation Controller, the Federation Connector leverages the individual implemented data gathering function to collect all requested data from the autonomous data system. In the prototype, the data gathering function is realized through the utilization of specified folders on the Federation Connector's operating system (OS). To mimic the decentralized nature of data sources within the SBFBDS, each Autonomous Data system is allocated a unique folder within the Federation Connector's OS-file system. These designated folders serve as repositories for the respective Autonomous Data system's data, encapsulating the varied data sources and ensuring an accurate representation of the system's operational framework.

Following data collection, the Federation Connector compresses all requested data into a single file, optimizing data transmission efficiency. Subsequently, the Federation Connector awaits a signal from the Federation Controller indicating readiness to receive data over a specific socket/port.

Upon completion of data transmission, all temporary data and files generated throughout the request are promptly purged, promoting efficient resource management and system cleanliness.

Furthermore, to maintain system integrity, a mechanism is employed to monitor the status of the Federation Connector. In the event of a shutdown or abrupt cessation of operation, a hook is activated to log the affected Federation Connector out of the system, ensuring that only operational connectors are actively logged in. Even in the event of a malfunctioning hook, the regular health checks initiated by the Federation Controller serve as a safeguard. These health checks swiftly detect any unreachable Federation Connector and consequently remove it from the list of active systems. This proactive approach from all components enhances system reliability and mitigates potential disruptions due to faulty connectors.

4.6 Validation Results

Having elucidated the prototype along with its constituent components and their respective functionalities, the subsequent chapter entails the execution of measurements to gather data for validating the predefined metrics.

4.6.1 Data Acquisition Scenarios

In the quest for data acquisition, a series of scenarios has been meticulously crafted to replicate the predominant use cases of the SBFDBS.

Within these scenarios, the network architecture remains consistent across all tests, comprising one User Interface, one Federation Connector, and three Federation Connectors. Each of these entities operates within its own Docker container, with all containers sharing a common network. Furthermore, the distribution of data across all Federation Connectors is evenly dispersed to guarantee active utilization of all connectors throughout the testing phase.

These predefined use cases are delineated as follows:

Scenario 1: Comprehensive Data Retrieval

In this scenario, a user initiates a request to procure all available data within the SBFDBS. The data comprises various file formats such as .png for images, .mp3 for audio files, .pdf for documents, .xml for spreadsheets, among others. Notably, the accumulated file sizes accessed by the user range from 1 MB to 1 GB. Specifically, the sizes investigated are 5 MB, 10 MB, 100 MB, 500 MB, and 1 GB, respectively.

Scenario 2: Selective File Retrieval

In this scenario, a user specifies a random assortment of files for retrieval from the SBFDBS. The data comprises various file formats such as .png for images, .mp3 for audio files, .pdf for documents, .xml for spreadsheets, and others. The cumulative file sizes accessed by the user in this scenario are 1 MB, 10 MB, 100 MB and 500 MB respectively.

Scenario 3: Contract-Specific Retrieval

In this scenario, a user requests the retrieval of a randomly selected contract associated with a file stored within the SBFDBS. Contracts are available in either .mkd or .pdf file formats. Notably, only one contract is downloaded at a time.

Scenario 4: Random Modifications to Federation Connectors

During the operational phase of the entire SBFDBS system, arbitrary alterations are introduced to a designated Federation Connector. These random modifications encompass the following actions:

- Deletion of a shared file.
- Modification or updating of a shared file.
- Restarting a random Federation Connector.
- Shutdown of a random Federation Connector.
- Initiation and subsequent logging into the Federation Controller by the Federation Connector.

Each of these scenarios undergoes an iterative execution, repeated up to 100 times to accumulate data. This approach not only facilitates the thorough validation of the system but also enables the assessment of its stability over prolonged durations.

These repetitive iterations serve as a robust mechanism for identifying any potential anomalies, inconsistencies, or performance bottlenecks that may arise during the testing phase. Moreover, they provide a solid foundation for the subsequent validation process, wherein the collected data will be analyzed and interpreted to assess the system's adherence to predefined metrics and performance benchmarks.

4.6.2 Data Transfer Performance

This section presents the results obtained from the analysis of the data gathered from the scenarios described earlier.

Throughput

Upon extensive testing, it was observed that the throughput is predominantly contingent upon the available network capacity. The testing infrastructure utilized comprised systems interconnected via wired connections within a 10 Gbit network. Additionally, the system's performance was evaluated on a Raspberry Pi 4 over a 2.5 Gbit 6E W-LAN, with no anomalies detected in this setup.

While further examination in more limited or unstable network environments is deemed necessary to validate the system's resilience under suboptimal network throughput conditions, it's worth noting that such scenarios are considered edge cases and may not be imperative for system validation.

Latency

The latency of the system exhibits considerable variation depending on the total file size requested by the user interface. Before delving into the precise timing between the initial request and the receipt of the final file on the user interface, it is noteworthy that all HTTP requests executed between each component within the aforementioned test network were nearly instantaneous, devoid of any noticeable delays.

The measured delay within the system corresponds to the data collection, preparation, and transmission processes. Notably, the preparation process, which involves compressing all data twice throughout the request, consumes a significant amount of time, particularly contingent upon the size of the data.

Table 4.1 outlines the latency between the initial request and the final transmission of the data to the frontend. The provided measurements represent the average of 100 iterations of the specified scenario with the specified data size.

All measured latencies fall within an acceptable variance relative to the corresponding file size. While further testing with larger file sizes is recommended to assess system performance comprehensively, it is advisable to limit these tests to the most common use cases, which should not exceed 500 MB in total per transfer.

Furthermore, it is pertinent to highlight that users are promptly informed via a popup notification while the file download process is in progress. This proactive measure ensures a seamless and satisfactory user experience, enhancing overall system usability and user satisfaction.

Transfer Type	File Size	Total Request Time
Scenario 1 (All Files)	1 MB	0.23 Seconds
Scenario 1 (All Files)	10 MB	0.85 Seconds
Scenario 1 (All Files)	100 MB	6.53 Seconds
Scenario 1 (All Files)	500 MB	32.44 Seconds
Scenario 1 (All Files)	1000 MB	63.18 Seconds
Scenario 2 (Specific Files)	1 MB	0.216 Seconds
Scenario 2 (Specific Files)	10 MB	0.774 Seconds
Scenario 2 (Specific Files)	100 MB	6.511 Seconds
Scenario 2 (Specific Files)	500 MB	30.037 Seconds
Scenario 3 (Contract)	N/A	0.160 Seconds

Table 4.1: Comparison of Total Request Time for Different Scenarios

Reliability

Throughout all the aforementioned tests, a quality assurance process was conducted to ensure the integrity of the downloaded files. Specifically, 10 random samples were selected from the 100 files produced in each downloading process per scenario. These random samples underwent thorough inspection to verify their data integrity, ensuring that the transmitted data was not corrupted or altered in any manner.

Remarkably, the findings indicate that none of the transmitted data exhibited any signs of corruption or alteration. Consequently, all transmitted data can be confidently utilized as intended without any concerns regarding data integrity.

While it is acknowledged that conducting checks on every single transmitted file would provide greater assurance of flawless transmissions, I maintain that a 10% random sample check suffices for validation purposes. This approach strikes a balance between thoroughness and efficiency, enabling us to confidently validate the integrity of the data transmissions.

4.6.3 Data Autonomy and Sovereignty

Participants Control

To validate this requirement, a hands-on approach was employed. At random intervals and random points during the runtime of the SBFDBS, deliberate alterations or deletions were made to data within one or more Federation Connectors. These modifications were meticulously recorded by the system and promptly implemented.

For instance, if a file was deleted, it was immediately omitted from display in new requests originating from the frontend and rendered inaccessible by the Federation Controller. Similarly, in the event of a file modification, the outdated file was swiftly replaced with the updated version.

Significantly, all tests were conducted while the SBFDBS was operational, and all changes were seamlessly propagated throughout the system without the need for any service restarts. This dynamic demonstration underscores the system's adeptness in upholding participants' control over their data in real-time. It ensures prompt responsiveness to user-driven modifications without disrupting system operations, thereby affirming the system's robustness in preserving data integrity and autonomy.

The sole issue encountered pertained to the modification of data while an active data request involving the altered file was underway. In such instances, a crucial decision had to be made regarding whether the request should proceed with the outdated file or fail altogether. After careful deliberation, it was deemed that data alteration during an active download represents an edge case scenario. Consequently, the most prudent course of action entails failing the request and prompting the user to initiate a new request. This approach ensures consistency and integrity in data transmission processes, mitigating potential discrepancies that may arise from concurrent modifications. While this decision necessitates user intervention to restart the request, it prioritizes system reliability and data accuracy, aligning with the overarching goal of maintaining stringent control over data integrity and autonomy within the SBFDBS ecosystem.

Legal Control

To further ensure legal control over the shared data, an additional system was implemented and thoroughly tested for functionality. This system empowers each participant to define legal documents and attach them to one or more files shared within the system. Subsequently, users are required to explicitly accept

the associated legal document before being granted access to download the respective file through the system.

The functionality of this system was tested using automated end-to-end tests. These tests repetitively simulated the downloading of random a contract, agreement to the associated contract, and subsequent downloading of the corresponding file from the system.

Moreover, to assess the system's resilience and responsiveness to dynamic changes, the contract definitions were altered while the SBFDBS was operational. As previously highlighted in the Participant Control Chapter, these modifications were seamlessly propagated throughout the entire system instantaneously. This was achieved without requiring any service restarts or causing any downtimes, affirming the system's ability to adapt to evolving conditions in real-time.

This robust mechanism guarantees that participants retain full legal autonomy over all data shared, enabling them to define unique conditions for each file independently. Thus, individual participants have the autonomy to establish tailored contracts for each file, reflecting their specific preferences and legal requirements.

By incorporating this system, the SBFDBS reinforces legal compliance and accountability, fostering trust and transparency among participants. It safeguards participants' legal rights and promotes responsible data sharing practices, thereby enhancing the overall integrity and reliability of the system.

4.6.4 Compatibility

To validate this requirement, a diverse selection of files was meticulously chosen for use in all testing scenarios. Different file types and sizes were deliberately included to ensure that the system can seamlessly transfer any file format without encountering corruption or alteration.

As previously stated in the Reliability Chapter, all files transferred through the system retained their original integrity and could be utilized in their intended manner without any compromise.

Furthermore, it is important to emphasize that participants are not required to change or prepare their data files in any manner. The SBFDBS is designed to accommodate shared files without necessitating any alterations or modifications. Participants need only provide the means of accessing the files for the SBFDBS to

function effectively, thereby streamlining the data-sharing process and minimizing user intervention. This aspect underscores the system's compatibility with diverse data formats and its user-friendly approach to data integration.

Ease of Integration

To ensure ease of integration, the SBFDBS was implemented and successfully deployed on various machines, including a Windows Desktop PC, a Windows Laptop PC and an Ubuntu Raspberry Pi 4 Machine. The system ran seamlessly on each of these platforms, demonstrating its versatility and compatibility across different environments.

It is imperative to note that the available resources on the machine directly influence the performance of the system. As data is transmitted through the system, it undergoes compression to optimize transmission efficiency. This compression process requires substantial CPU resources to execute effectively. Therefore, the availability of adequate CPU resources directly correlates with the system's ability to perform data compression efficiently.

Furthermore, to facilitate widespread adoption and future implementation, all source code repositories were made public and downloadable. This ensures accessibility to all potential implementers, allowing them to explore and customize the system according to their specific requirements. By providing open access to the source code, I aim to encourage collaboration and foster innovation within the community.

In addition to source code availability, Docker images were provided for a default use case of the Federation Connector. These Docker images simplify integration for participants by offering a standardized setup with default data provisioning settings. Users can opt to utilize the default data provisioning method, which involves specifying a folder in the operating system for data storage. Alternatively, participants have access to public repositories of the Federation Connector, enabling them to implement customized data provisioning functions tailored to their specific data systems.

Looking ahead, future efforts may involve creating and publicly releasing additional Docker images tailored for popular databases such as Azure Vaults, Amazon Web Store Buckets, MongoDB and PostgreSQL. These images would provide default implementations of data provisioning functions for different database systems, further enhancing ease of integration for participants who may lack the expertise or resources to develop their own provisioning solutions. By expanding the range of available Docker images, I aim to democratize access to the SBFDBS and streamline integration processes for a diverse range of participants.

Scalability

As previously noted, the available resources on the host machine directly influence the system's performance, thereby allowing for vertical scaling. Increasing resources, particularly CPU power, notably enhances the overall runtime of requests within the system.

Given that the prototype focuses on essential components for running the SBFDBS in a small-scale application, it primarily supports vertical scaling. However, the system design inherently incorporates various methods to facilitate vertical scaling, particularly through a microservices approach.

In the system design, all connected federation connectors are managed by a specialized component known as the connection controller. This controller enables the deployment of multiple instances of the federation controller, thereby enabling horizontal scaling by distributing the workload across multiple federation controllers dynamically.

Nevertheless, it's crucial to acknowledge that the scalability of the system is still constrained by the individual autonomous data systems. If a particular system experiences high demand and cannot effectively handle the incoming load, it can impact the performance of the entire SBFDBS, even with multiple federation controllers.

This limitation stems from the system's commitment to preserving the autonomy of each individual participant. While this constraint may pose challenges in certain scenarios, it is rarely encountered in typical use cases. As such, while the limitation persists, it seldom impedes the overall functionality and effectiveness of the SBFDBS in most common scenarios.

4.7 Threads to Validity

In the validation of the SBFDBS prototype, I scrutinize a spectrum of potential threats to validity, encompassing methodological limitations, environmental factors, and inherent biases. These considerations are paramount in safeguarding the accuracy and credibility of my validation efforts, thereby reinforcing the reliability and integrity of the outcomes.

4.7.1 Challenges Arising from Limited Testing Environments

Resource constraints pose significant challenges in diversifying testing environments for evaluating the SBFDBS prototype. The constrained availability of testing resources may inadequately simulate real-world scenarios where the system operates across heterogeneous networks and hardware configurations. Consequently, there exists a risk that validation results may fail to comprehensively capture the system's performance across various configurations and user contexts, potentially introducing uncertainties into my assessments.

4.7.2 Constraints on Sample Size for Testing

The constraints imposed by time and financial limitations necessitate a cautious approach to sample size in validation endeavors. While I express confidence in the adequacy of the collected data to substantiate the successful validation of the SBFDBS, the complexity and variability of potential data systems where the SBFDBS could be deployed demand a more extensive dataset for conclusive claims. Therefore, efforts to augment the sample size remain imperative to bolster the robustness of my validation outcomes.

4.7.3 Challenges in Conducting Longitudinal Studies

The temporal constraints inherent in my research and testing endeavors impose notable challenges, particularly in terms of sample size and the breadth of the feature set within the prototype. The imperative to validate the SBFDBS comprehensively necessitates the implementation and thorough testing of all system components. In managing these constraints, deliberate decisions were made to exclude specific components from the prototype, guided by the assessment of their necessity for validation purposes. This exclusionary process was informed by several factors, including the widespread acceptance and successful implementation of certain core concepts within the field, as well as the existence of prior validation efforts conducted by other researchers.

For instance, one such component omitted from the prototype was the authentication system, which had already undergone rigorous validation processes in previous studies referenced in the literature, including works by Preukschat [PR21a], Weingaertner [WC21] and Schaffner [Sch20]. Given the established validity and widespread adoption of these authentication mechanisms, their inclusion in the prototype was deemed redundant for the purposes of my validation efforts.

4.7.4 Limited Access to Specialized Validation Tools

The financial constraints that impede access to specialized validation tools warrant careful consideration. The predominantly self-implemented or freely available software tools utilized in my validation processes may lack the sophistication and capabilities of their commercial counterparts. Consequently, the reliance on such tools may compromise the comprehensiveness of my validation procedures, particularly in areas such as load testing, security assessments, and compatibility checks. Addressing this limitation requires a judicious balance between resource constraints and the imperative to conduct thorough and rigorous validation assessments.

4.7.5 Time Constraints and Balancing Priorities

The exigencies of balancing validation activities with academic responsibilities, part-time employment, and other commitments necessitate a pragmatic approach to managing time constraints. The finite availability of time and resources to dedicate to design, implementation, and validation endeavors underscores the importance of strategic prioritization and efficient allocation of efforts. However, the inherent trade-offs associated with time constraints may potentially compromise the thoroughness and depth of my validation process, highlighting the need for conscientious planning and resource management to mitigate these risks.

4.7.6 Reliance on Open-Source and Free Tools

Acknowledging the financial constraints that shape my validation efforts, I deliberate the implications of relying heavily on open-source software, free tools or self-developed software. While these resources offer cost-effective solutions, they may lack the robustness, support, or advanced features associated with commercial alternatives. Consequently, the utilization of open-source and free tools may introduce limitations or biases into my validation efforts, underscoring the importance of cautious evaluation and validation of results obtained through such means. Efforts to mitigate these limitations may involve supplementing open-source tools with proprietary solutions where feasible, or augmenting validation procedures to account for potential biases introduced by reliance on open-source resources.

4.8 Validation Conclusion

In conclusion, the validation process of the SBFDBS has provided invaluable insights into its performance, reliability, and adherence to predefined metrics. Through crafted data acquisition scenarios and extensive testing, I have garnered substantial evidence supporting the system's efficacy in facilitating secure and bilateral transfers of engineering data while safeguarding users' data autonomy and sovereignty.

The validation efforts encompassed diverse scenarios, ranging from comprehensive data retrieval to contract-specific retrieval, and included assessments of system throughput, latency, reliability, data autonomy, sovereignty, compatibility, ease of integration, and scalability. Notably, the system exhibited commendable performance across various metrics, demonstrating its ability to accommodate different file formats, sizes, and user preferences.

Key findings indicate that the SBFDBS operates within acceptable latency thresholds, with negligible delays observed in HTTP requests and reasonable transfer times for files of varying sizes. Moreover, reliability tests underscored the system's robustness in maintaining data integrity, as evidenced by the absence of corrupted or altered data in extensive sampling checks.

Furthermore, the system's emphasis on participants' control over their data was reaffirmed through dynamic demonstrations of real-time data modifications and legal control mechanisms. These features enhance user trust and transparency and ensure compliance with legal requirements, fostering responsible data-sharing practices.

Regarding compatibility and ease of integration, the SBFDBS proved versatile and user-friendly, running seamlessly across different platforms and offering open access to source code and Docker images. These initiatives aim to democratize access to the system and encourage collaboration within the community.

While the prototype primarily supports vertical scaling, provisions for horizontal scaling through microservices' architecture signify a pathway for future enhancements. However, it's imperative to acknowledge that scalability remains subject to individual autonomous data systems' capabilities and constraints.

In summary, the validation process has successfully validated the SBFDBS's efficacy in securely facilitating bilateral data transfers while upholding users' data autonomy and sovereignty. The system's robust performance, reliability, and compatibility underscore its potential to address diverse engineering data-sharing needs while fostering trust, transparency, and collaboration among participants.

Furthermore, it's imperative to acknowledge and address potential threats to the validity of my validation process. Methodological limitations, environmental factors, and inherent biases were carefully considered throughout my efforts to ensure the accuracy and credibility of my outcomes. These considerations reinforce the reliability and integrity of my validation efforts, underscoring the confidence in the validity of my findings.

5 Conclusion

In the contemporary digital landscape, characterized by the exponential growth in data generation and collection, the imperative of data security and ownership assumes paramount significance. In response to this imperative, methodologies are warranted to uphold data integrity while facilitating secure data sharing. This paper embarked on addressing this challenge by conceptualizing, researching, and developing a system capable of securely and bilaterally exchanging data without compromising users′ data autonomy and sovereignty, with a specific focus on engineering data exchange.

Throughout this investigation, the paper has delved into the intricate challenges posed by the burgeoning landscape of digital data exchange, particularly within engineering domains. As I traversed this terrain, the overarching inquiries guiding the exploration revolved around the delicate balance between data sovereignty, autonomy, and the need for interoperability and scalability within large-scale data ecosystems.

Leveraging insights from prior research and employing the Arc42 template, I devised a system that enables precisely such functionality. Termed the "Secure Bilateral Federated Database System" (SBFDBS), this system was meticulously crafted and subjected to prototyping resembling a vertical slice of its design, facilitating thorough testing to validate its robust performance, reliability, and compatibility.

The architecture of SBFDBS, depicted in Figure 3.5, predominantly comprises five main components:

- The "User Interface" serving as the interface between end-users and federation connectors
- The "Federation Connectors" acting as mediators between user interfaces and autonomous databases
- The "Federation Controllers" deployed within autonomous databases to facilitate interaction with federation connectors

- The "Connection controller" responsible for managing logged-in federation connectors and disseminating relevant information to federation controllers upon request

- And an "Authentication System" tasked with generating, distributing, and validating credentials like Digital Identifiers (DIDs) for authentication purposes between federation connectors and controllers

Through rigorous prototyping and validation exercises, I have encountered numerous challenges and complexities. Yet, at each juncture, the SBFDBS has proven its mettle, offering a scalable and interoperable solution. Each component of the SBFDBS, from the user interface to the authentication system, plays a pivotal role in orchestrating this delicate dance between security, sovereignty, and autonomy.

Looking ahead, the landscape of digital data exchange continues to evolve, presenting new challenges and opportunities on the horizon. The journey of the SBFDBS does not culminate here; rather, it serves as a stepping stone towards future innovations and advancements in the realm of secure and autonomous data exchange.

5.1 Future Work

The SBFDBS represents an advancement in the realm of secure and autonomous data exchange. As I look towards the future, there are several avenues for further exploration and enhancement that could build upon the foundation laid by the SBFDBS.

5.1.1 Real-World Implementation or Case Study

Exploring the practical implementation of the SBFDBS, the system emerges as a prominent avenue for future endeavors. Within this realm, a particularly compelling prospect lies in its application within specific contexts, such as automotive engineering.

In the automotive sector, where the seamless exchange of critical engineering data among manufacturers, suppliers, and regulatory bodies is paramount, the potential impact of the SBFDBS is profound. Imagining a scenario where a leading automotive manufacturer collaborates to integrate the SBFDBS into their existing data infrastructure ignites excitement. This integration could herald a new

era of secure and bilateral data exchange throughout the supply chain, fundamentally transforming data management practices within the industry.

This implementation or case study of the SBFDBS in such a context would involve several key steps:

- Requirement Analysis: Conducting a comprehensive analysis of the data exchange requirements within the automotive supply chain, including the types of data being exchanged, the frequency of exchange, and the security and privacy requirements.

- System Integration: Integrating the SBFDBS into the existing data infrastructure of the automotive manufacturer, including connectivity with internal databases, ERP systems, and third-party suppliers.

- Pilot Deployment: Deploying the SBFDBS in a pilot environment to assess its performance, reliability, and usability in real-world scenarios. This could involve conducting extensive testing and validation exercises to ensure that the system meets the requirements of all stakeholders.

- User Training and Adoption: Providing training and support to users across the automotive supply chain to facilitate the adoption of the SBFDBS. This could include developing user-friendly interfaces, documentation, and support materials to assist users in navigating the system effectively.

- Evaluation and Feedback: Continuously monitoring and evaluating the performance of the SBFDBS in the field, soliciting feedback from users, and iteratively refining the system based on user experiences and evolving requirements.

By undertaking a real-world implementation of the SBFDBS inside a specified context, researchers and practitioners can validate the efficacy of the system and identify opportunities for further refinement and enhancement. This case study could serve as a blueprint for future deployments of the SBFDBS in other industry sectors, demonstrating its versatility and applicability across diverse domains.

5.1.2 Scalability Optimization

Another area ripe for future work is the optimization of scalability within the SBFDBS framework. While the system has demonstrated robust performance in handling data exchange within engineering domains, scaling it to accommodate

larger datasets and diverse use cases presents an exciting challenge. Future research could focus on refining the architecture and algorithms of the SBFDBS to enhance its scalability without compromising security or autonomy.

5.1.3 Enhanced Interoperability

Interoperability is crucial for ensuring seamless data exchange across heterogeneous systems and platforms. Future iterations of the SBFDBS could explore ways to enhance interoperability with other data exchange protocols and standards, allowing for greater integration with existing infrastructures and ecosystems. This could involve the development of standardized interfaces and protocols for interoperability, as well as compatibility testing with other federated database systems.

5.1.4 Advanced Security Mechanisms

While the SBFDBS prioritizes data security and ownership, there is always room for improvement in the realm of cybersecurity. Future research could focus on integrating advanced security mechanisms, such as homomorphic encryption, secure multi-party computation, or blockchain technology, to further fortify the SBFDBS against potential threats and vulnerabilities. Additionally, exploring novel authentication and access control mechanisms could enhance the system's resilience to unauthorized access and data breaches.

5.1.5 Integration with Emerging Technologies

The SBFDBS is well-positioned to leverage emerging technologies such as edge computing, Internet of Things (IoT), and artificial intelligence (AI) to further enhance its capabilities. Future research could explore ways to integrate these technologies into the SBFDBS ecosystem, enabling new use cases and applications in domains such as predictive maintenance, real-time analytics, and autonomous systems. This could involve the development of specialized modules and connectors for seamless integration with edge devices, IoT platforms, and AI frameworks.

In conclusion, the journey of the SBFDBS does not end with its initial implementation and validation. Instead, it serves as a springboard for future innovations and advancements in the realm of secure and autonomous data exchange. By

exploring the aforementioned avenues for future work, researchers, and practitioners can further refine and extend the capabilities of the SBFDBS, paving the way for a future where data sovereignty, autonomy, and security are not just aspirations but fundamental principles of digital exchange.

Bibliography

[ABRS20] H. Al-Breiki, M. Rehman, and D. Svetinovic. Trustworthy blockchain oracles: review, comparison, and open research challenges. *Ieee Access*, 8:85675–85685, 2020.

[AG96] V. Anand and H. Gunadhi. Data warehouse architecture for dss applciations. *Australasian Journal of Information Systems*, 4, 1996.

[ALK10] M. Ameen, J. Liu, and K. Kwak. Security and privacy issues in wireless sensor networks for healthcare applications. *Journal of Medical Systems*, 36:93–101, 2010.

[BMLB19] E. Benhani, C. Mancillas-López, and L. Bossuet. Secure internal communication of a trustzone-enabled heterogeneous soc lightweight encryption. 2019.

[BS08] Stefan Berger and Michael Schrefl. From federated databases to a federated data warehouse system. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, pages 394–394, 2008.

[Cal21] I. Calzada. Data co-operatives through data sovereignty. *Smart Cities*, 4:1158–1172, 2021.

[Can01] R. Canetti. Universally composable security: a new paradigm for cryptographic protocols. 2001.

[CF20] E. Celeste and F. Fabbrini. Competing jurisdictions: data privacy across the borders. pages 43–58, 2020.

[CLL+21] Y. Chen, J. Li, Q. Lu, H. Lin, Y. Xia, and F. Li. Cyber security for multi-station integrated smart energy stations: architecture and solutions. *Energies*, 14:4287, 2021.

[CLQJ22] Z. Cui, X. Liu, H. Qin, and H. Ji. Differential game analysis of enterprises investing in new infrastructure and maintaining social network security under the digital innovation ecosystem. *Ieee Access*, 10:69577–69590, 2022.

[Coc23] C. Cocq. Data colonialism and data sovereignty in indigenous spaces. *Aoir Selected Papers of Internet Research*, 2023.

[CSB+17] S. Carroll, J. Schultz, E. Briggs, P. Riggs, and N. Palmanteer-Holder. Data as a strategic resource: self-determination, governance, and the data challenge for indigenous nations in the united states. *International Indigenous Policy Journal*, 8, 2017.

[CT70] Stefan Conrad and Can Türker. Active integrity maintenance in federated database systems. 02 1970.

[ECC16] C. Esposito, A. Castiglione, and K. Choo. Encryption-based solution for data sovereignty in federated clouds. *Ieee Cloud Computing*, 3:12–17, 2016.

[Eri23] Ericsson. Annual mobile data traffic worldwide from 2012 to 2029 (in exabytes per month) [graph]. Statista, November 1 2023. Retrieved April 12, 2024.

[GGHT19] J. Gutierrez-Guerrero and J. Holgado-Terriza. Automatic configuration of opc ua for industrial internet of things environments. *Electronics*, 8:600, 2019.

[GSA15] M. Ghasemi, S. Sharafi, and A. Arman. Towards an analytical approach to measure modularity in software architecture design. *Journal of Software*, 10:465–479, 2015.

[GVC+21] S. Geisler, M. Vidal, C. Cappiello, B. Lóscio, A. Gal, M. Jarke, M. Lenzerini, P. Missier, B. Otto, E. Paja, B. Pernici, and J. Rehof. Knowledge-driven data ecosystems toward data transparency. *Journal of Data and Information Quality*, 14:1–12, 2021.

[GWLJ13] F. Gey, S. Walraven, D. Landuyt, and W. Joosen. Building a customizable business-process-as-a-service application with current state-of-practice. pages 113–127, 2013.

[HBTD21] P. Hummel, M. Braun, M. Tretter, and P. Dabrock. Data sovereignty: a review. *Big Data Society*, 8:205395172098201, 2021.

[JW00] P. Jogalekar and M. Woodside. Evaluating the scalability of distributed systems. *Ieee Transactions on Parallel and Distributed Systems*, 11:589–603, 2000.

[KMM+20] A. Kiourtis, A. Mavrogiorgou, S. Menesidou, P. Gouvas, and D. Kyriazis. A secure protocol for managing and sharing personal healthcare data. 2020.

[KW22] H. Kagermann and W. Wahlster. Ten years of industrie 4.0. *Sci*, 4:26, 2022.

[LCW+19] D. Liu, X. Chen, S. Wan, J. Tang, and Y. Cheng. Turing machine-based cross-network isolation and data exchange theory model. *Ieee Access*, 7:125732–125746, 2019.

[LWZZ] W. Li, L. Wang, B. Zhu, and L. Zhang. An integrity lock architecture for supporting distributed authorizations in database federations. pages 189–203.

[MFM+20] P. Mangold, A. Filiot, M. Moussa, V. Sobanski, G. Ficheur, P. Andrey, and A. Lamer. A decentralized framework for biostatistics and privacy concerns. 2020.

[MFSFM19] J. Moreno, E. Fernandez, M. Serrano, and E. Fernandez-Medina. Secure development of big data ecosystems. *Ieee Access*, 7:96604–96619, 2019.

[MG05] Martin Nussbaumer Martin Gaedke, Johannes Meinecke. i2map: an approach to model the landscape of federated systems. In *IEEE International Conference on Web Services (ICWS'05)*, page 798, 2005.

[MGGM18] Alexander Mühle, Andreas Grüner, Tatiana Gayvoronskaya, and Christoph Meinel. A survey on essential components of a self-sovereign identity. *Computer Science Review*, 30:80–86, 2018.

[MPL07] J. Muilu, L. Peltonen, and J. Litton. The federated database – a basis for biobank-based post-genome studies, integrating phenome and genome data from 600 000 twin pairs in europe. *European Journal of Human Genetics*, 15:718–723, 2007.

[MSFFM20] J. Moreno, M. Serrano, E. Fernandez, and E. Fernández-Medina. Improving incident response in big data ecosystems by using blockchain technologies. *Applied Sciences*, 10:724, 2020.

[MWLJ18] R. Ma, Q. Wen, H. Lin, and Y. Jiang. Research on data security ecosystem construction based on big data background. 2018.

[NYK+20] T. Nguyen, Y. Yeom, T. Kim, D. Park, and S. Kim. Horizontal pod autoscaling in kubernetes for elastic container orchestration. *Sensors*, 20:4621, 2020.

[OS02] M. Oliva and F. Saltor. Integrating multilevel security policies in multilevel federated database systems. pages 135–147, 2002.

[Ott22] B. Otto. The evolution of data spaces. pages 3–15, 2022.

[Par02] D. Parnas. On the criteria to be used in decomposing systems into modules. pages 411–427, 2002.

[PBM23] Nikolaos Papadakis, Georgios Bouloukakis, and Kostas Magoutis. Comdex: A context-aware federated platform for iot-enhanced communities. In *Proceedings of the 17th ACM International Conference on Distributed and Event-Based Systems*, DEBS '23, page 37–48, New York, NY, USA, 2023. Association for Computing Machinery.

[PFFC] F. Petrini, J. Fernández, E. Frachtenberg, and S. Coll. Scalable collective communication on the asci q machine.

[PR21a] Alex Preukschat and Drummond Reed. *Self-sovereign identity*. Manning Publications, 2021.

[PR21b] Alex Preukschat and Drummond Reed. *Self-Sovereign Identity: Decentralized digital identity and verifiable credentials*. Manning, 2021. Paperback.

[Pra21] R. Prasad. Bodies and data: the digital sovereignty of the indian state. *Aoir Selected Papers of Internet Research*, 2021.

[Pro23] Proofpoint. Share of organizations worldwide that have experienced a loss of sensitive information as of february 2023, by country [graph]. Statista, May 15 2023. Retrieved April 13, 2024.

[RFR19] M. Rana, U. Farooq, and W. Rahman. Scalability enhancement for cloud-based applications using software oriented methods. *International Journal of Engineering and Advanced Technology*, 8:4208–4213, 2019.

[RSL+20] Drummond Reed, Manu Sporny, Dave Longley, Christopher Allen, Ryan Grant, Markus Sabadello, and Jonathan Holt. Decentralized identifiers (dids) v1. 0. *Draft Community Group Report*, 2020.

[SAP+19] M. Salnitri, K. Angelopoulos, M. Pavlidis, V. Diamantopoulou, H. Mouratidis, and P. Giorgini. Modelling the interplay of security, privacy and trust in sociotechnical systems: a computer-aided design approach. *Software Systems Modeling*, 19:467–491, 2019.

[Sch20] Martin Schaffner. Analysis and evaluation of blockchain-based self-sovereign identity systems. *Technical University of Munich, Munich, Germany*, 2020.

[SCL+18] W. Sun, Z. Cai, Y. Li, F. Liu, S. Fang, and G. Wang. Security and privacy in the medical internet of things: a review. *Security and Communication Networks*, 2018:1–9, 2018.

[SJE+23] A. Sakor, S. Jozashoori, N. Emetis, A. Rivas, K. Bougiatiotis, F. Aisopos, E. Iglesias, P. Rohde, T. Padiya, A. Krithara, G. Paliouras, and M. Vidal. Knowledge4covid-19: a semantic-based approach for constructing a covid-19 related knowledge graph from various sources and analyzing treatments' toxicities. *Journal of Web Semantics*, 75:100760, 2023.

[SL90a] A. Sheth and J. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *Acm Computing Surveys*, 22:183–236, 1990.

[SL90b] Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236, sep 1990.

[SNE20] A. Satybaldy, M. Nowostawski, and J. Ellingsen. Self-sovereign identity systems. pages 447–461, 2020.

[TE23] Transforma Insights and Exploding Topics. Number of internet of things (iot) connected devices worldwide from 2019 to 2023, with forecasts from 2022 to 2030 (in billions) [graph]. Statista, July 1 2023. Retrieved April 12, 2024.

[TSJ+12] Danah Tonne, Rainer Stotzka, Thomas Jejkal, Volker Hartmann, Halil Pasic, Andrea Rapp, Philipp Vanscheidt, Bernhard Neumair, Achim Streit, Ariel Garcia, Daniel Kurzawe, Tibor K´lm´n, Jedrzej Rybicki, and Beatriz Sanchez Bribian. A federated data zone for the arts and humanities. In *2012 20th Euromicro International Conference on Parallel, Distributed and Network-based Processing*, pages 198–205, 2012.

[VTK22] D. Vu, M. Tran, and Y. Kim. Predictive hybrid autoscaling for containerized applications. *Ieee Access*, 10:109768–109778, 2022.

[WC21] Tim Weingaertner and Oskar Camenzind. Identity of things: Applying concepts from self sovereign identity to iot devices. *The Journal of The British Blockchain Association*, 2021.

[WNY+21] S. Welten, L. Neumann, Y. Yediel, L. Santos, S. Decker, and O. Beyan. Dams: a distributed analytics metadata schema. *Data Intelligence*, 3:528–547, 2021.

[XFPM14] H. Xiong, F. Fowley, C. Pahl, and N. Moran. Scalable architectures for platform-as-a-service clouds: performance and cost analysis. pages 226–233, 2014.

[YWJ02] J. Yang, D. Wijesekera, and S. Jajodia. Subject switching algorithms for access control in federated databases. pages 61–74, 2002.

[ZMJ+19] J. Zrenner, F. Möller, C. Jung, A. Eitel, and B. Otto. Usage control architecture options for data sovereignty in business ecosystems. *Journal of Enterprise Information Management*, 32:477–495, 2019.

Attachments

1 Code Repositories

The source code associated with this paper is openly accessible for a period of one year following the submission of this thesis, up until April 30, 2025. It is hosted in three distinct public GitHub repositories, each governed by the Apache 2.0 License.

The user interface prototype is available at the following GitHub repository:

https://github.com/CodeOwlCove/Thesis_User_Interface

The federation controller prototype can be accessed through the following GitHub repository:

https://github.com/CodeOwlCove/Thesis_Federation_Controller

Additionally, the federation connector prototype is available at:

https://github.com/CodeOwlCove/Thesis_Federaion_Connector

It should be noted that the Connection Controller and Authentication Service were excluded from the prototyping and validation phase and therefore remain to be prototyped.