

MSc Computer Science

Final Project

**Identification and Implementation of Suitable Decentralized Identifier Methods
for Self-Sovereign Identity Wallets in a Data Space**

Author:

Amadeus Darren Leander

Chair and Supervisor:

dr. João Luiz Rebelo Moreira (UT)

Committee:

dr. Faizan Ahmed (UT)

dr. Maarten Everts (UT)

External Supervisors:

Willem Datema (TNO)

Simon Dalmolen (TNO)

University of Twente – Faculty of Electrical Engineering, Mathematics & Computer Science

TNO – Data Ecosystems Department

**UNIVERSITY
OF TWENTE.**

TNO innovation
for life

Abstract

The Dataspace Protocol was recently released to the public by the International Data Spaces Association as a standardized technical specification to tackle rising interoperability issues among participants within the same and across different data spaces, complementing the functional components of data spaces outlined in the Reference Architecture Model (IDS-RAM). Despite this, this protocol only provides guidelines solely for the data exchange process through the IDS Connector component and does not touch upon other essential components described in the IDS-RAM. Ongoing efforts currently exist to define a similar protocol for the Identity Provider component and to introduce a decentralized identity management system within a data space. This overall thesis study aims to contribute to these existing processes by exploring the suitability of Decentralized Identifiers (DIDs) in a Self-Sovereign Identity (SSI) wallet for the domain of data spaces. The study begins with performing a collection of commonly supported DID methods from different IDS Connector implementations and independent SSI wallets by surveying technical documentations and version control systems. Afterwards, an evaluation rubric following W3C recommendations is constructed by selecting criteria relevant to the context of data spaces identity management. With these prepared artefacts, each of the gathered DID methods are evaluated individually using the generated evaluation rubric, of which the results are compared against each other to decide their applicability in the domain of this study. This comparison process draws out one candidate DID method deemed most suitable for the data spaces context. Finally, a validation is conducted by integrating the selected DID method to an existing data space SSI wallet, and different scenarios encompassing interoperability, security and usability are conducted to measure its actual fit and appropriateness.

Keywords: decentralized identifiers, self-sovereign identity, data spaces, decentralized identity management, identity provider

Table of Contents

Abstract	2
Table of Contents	3
List of Figures	6
List of Tables	8
1. Introduction	10
1.1. Background.....	10
1.2. Problem Statement.....	12
1.3. Research Objective	12
1.4. Research Questions.....	13
1.5. Research Approach	14
1.6. Report Structure.....	16
2. Exploratory Study	17
2.1. Fundamental Terminologies	17
2.1.1. Data sharing.....	17
2.1.2. Data sovereignty.....	17
2.1.3. Trustworthiness	17
2.1.4. Interoperability	17
2.1.5. Data space	17
2.1.6. JavaScript Object Notation (JSON)	18
2.2. International Data Spaces (IDS)	19
2.3. IDS Reference Architecture Model 4 (IDS-RAM 4).....	19
2.3.1. IDS Connector.....	20
2.3.2. Identity Provider.....	20
2.4. Dataspace Protocol 2024-1	20
2.4.1. Control Plane.....	21

2.4.2.	Data Plane	21
2.5.	Self-Sovereign Identity (SSI) wallet.....	22
2.6.	Verifiable Credentials (VCs)	22
2.6.1.	Verifiable Presentations (VPs)	22
2.7.	Decentralized Identifiers (DIDs)	22
2.7.1.	DID Document	22
2.7.2.	did:web	23
2.8.	Workflow of DIDs, VCs and VPs.....	24
2.9.	Previous Work.....	24
2.9.1.	Categorization of DID Methods	24
2.9.2.	Underlying Technologies for DIDs	25
2.9.3.	Evaluation of Specific DID Methods	25
3.	DID Methods Selection	27
3.1.	Knowledge Sources for Collection.....	27
3.2.	Collection Results.....	30
3.2.1.	In other IDS Connectors.....	30
3.2.2.	In independent SSI wallets	33
3.3.	Selection Results.....	38
4.	DID Methods Evaluation.....	40
4.1.	Rubric References.....	40
4.2.	Rubric Creation.....	43
4.3.	Evaluation Results	51
4.3.1.	Benchmark DID Method.....	51
4.3.2.	Selected DID Methods	51
4.4.	Evaluation Discussions.....	58
4.5.	Evaluation Conclusions	61

5. Verification & Validation	63
5.1. Implementation Design.....	63
5.1.1. Current System & Challenges	63
5.1.2. Modified System	68
5.2. Verification	74
5.2.1. Interoperability	74
5.2.2. Security.....	80
5.3. Validation.....	85
5.3.1. Usability	85
6. Final Remarks	91
6.1. Conclusions	91
6.2. Limitations.....	94
6.3. Future Work	94
References	96
Appendix A: Timeline of Research Topics & Final Project	100
Appendix B: DID Method Evaluation Rubric & Scoring Sheet	101
Appendix C: DID Method Evaluation Results using Evaluation Rubric	109

List of Figures

Figure 1: Design Science Methodology engineering cycle (Wieringa, 2014)	14
Figure 2: Flowchart of Research Outline based on Design Science Methodology.....	16
Figure 3: Architecture of a data space (International Data Spaces Association, n.d.b)	18
Figure 4: Example of JSON	18
Figure 5: Example of JSON-LD.....	19
Figure 6: Example of JSONL.....	19
Figure 7: Interactions of components in the IDS-RAM 4 System Layer (Otto et al., 2022) ...	20
Figure 8: Interactions of components in the Dataspace Protocol (Steinbuss, 2024).....	21
Figure 9: Example of a DID Document	23
Figure 10: Screenshot of IDSA Data Connector Report detailed connector overview	27
Figure 11: Screenshot of IDSA Data Connector Report overview list.....	28
Figure 12: Screenshot of OWF Digital Wallet Overview	29
Figure 13: Class Diagram of Current TSG SSI wallet	64
Figure 14: Sequence Diagram of Current DID Creation.....	65
Figure 15: Sequence Diagram of Current DID Resolution	66
Figure 16: Class Diagram of Modified TSG SSI wallet with Strategy pattern.....	69
Figure 17: Sequence Diagram of Modified DID Creation.....	70
Figure 18: Sequence Diagram of Modified DID Resolution	72
Figure 19: Result of DID Creation with did:web.....	76
Figure 20: Result of DID Creation with did:tdw	77
Figure 21: Result of DID Creation with an unsupported DID method	77
Figure 22: Result of DID Resolution with did:web	78
Figure 23: Result of DID Resolution with did:tdw	79
Figure 24: Result of DID Resolution with an unsupported DID method	79
Figure 25: Initial state of resolved DID Document before did:web attack	81

Figure 26: State of resolved DID Document after did:web attack.....	82
Figure 27: Initial state of resolved DID Document before did:tdw attack.....	83
Figure 28: State of resolved DID Document after did:tdw attack.....	83
Figure 29: Credentials issued by Data Space Wallet to Data Space Participants.....	86
Figure 30: Default Catalog Request page on Control Plane	87
Figure 31: Result of did:web to did:web Catalog Request.....	87
Figure 32: Result of did:tdw to did:tdw Catalog Request.....	88
Figure 33: Result of did:web to did:tdw Catalog Request	89
Figure 34: Result of did:tdw to did:web Catalog Request	89
Figure 35: GANTT chart of Research Topics & Final Project.....	100

List of Tables

Table 1: IDS Connectors labelled as SSI-implementing based on the IDSA Data Connector Report (Giussani & Steinbuss, 2024) (as of 28 March 2024)	31
Table 2: Open-source IDS Connector implementations, based on the IDSA Data Connector Report, with additional descriptions regarding SSI (as of 28 March 2024).....	32
Table 3: Independent SSI wallets, curated from the Digital Wallet Overview, and their supported DID methods (as of 14 March 2024)	37
Table 4: Summary of findings of current DID method implementations.....	39
Table 5: Criteria categories of W3C DID Method Rubric v1.0 (Andrieu et al., 2021).....	41
Table 6: Extraction of data space identity provider characteristics from core design choices of DSSC building blocks	42
Table 7: Selection process of W3C DID Method Rubric v1.0 criteria based on data space identity provider characteristics extracted from DSSC	50
Table 8: Summary of DID Method Rubric Scoring Results	58
Table 9: Categorization of Evaluated DID Methods based on Similar Characteristics	61
Table 10: Interoperability Verification Scenario – DID Creation Interoperability.....	75
Table 11: Interoperability Verification Scenario – DID Resolution Interoperability	75
Table 12: Security Verification Scenario – Web Infrastructure Infiltration.....	80
Table 13: Usability Validation Scenario – Catalog Request.....	86
Table 14: Resulting DID Methods Evaluation Rubric for this Thesis Study	107
Table 15: Resulting Scoring Sheet for DID Methods Evaluation	108
Table 16: Resulting Evaluation Scoring Sheet of did:webs	111
Table 17: Resulting Evaluation Scoring Sheet of did:tdw	114
Table 18: Resulting Evaluation Scoring Sheet of did:key.....	116
Table 19: Resulting Evaluation Scoring Sheet of did:jwk	118
Table 20: Resulting Evaluation Scoring Sheet of did:peer.....	121
Table 21: Resulting Evaluation Scoring Sheet of did:pkh	123
Table 22: Resulting Evaluation Scoring Sheet of did:ion	125

Table 23: Resulting Evaluation Scoring Sheet of did:ebis 128

Table 24: Resulting Evaluation Scoring Sheet of did:sov 131

Table 25: Resulting Evaluation Scoring Sheet of did:ethr 133

Table 26: Resulting Evaluation Scoring Sheet of did:cheqd 136

Table 27: Resulting Evaluation Scoring Sheet of did:web 138

1. Introduction

1.1. Background

The importance of data and the immense value that it holds is no longer a question in this modern digital era (Robotics & Automation News, 2021). Worldwide operations in collecting massive amounts of diverse data in all forms triggers the attention towards opportunities of data sharing across different organizations with the intent of collaboration, believed to have the value potential to instigate many future possibilities allowing for further technological, business, and societal advancements (European Parliament, 2023). However, proper data sharing comes with many hindering obstacles, such as accessibility of organizations to discover and connect with each other, interoperability of different shared data formats, and security of the confidential and private nature of data (Curry et al., 2022).

Recent studies regarding the emerging concept of data spaces is now a central focus of many global organizations interested in data sharing. Data spaces is an abstract solution that provides a secure room for a diverse group of data consumers and data providers to connect with one another and enable consumers to gain access to the data owned by data providers. All of this occurs whilst complete control over the data is still maintained by the data providers. There is no central data pooling involved by the data space as all data remains in full administration of each data provider. Data spaces aim to address the challenges in data sharing by balancing between the pressing needs of data exchange and data sovereignty (Otto, 2022a).

The International Data Spaces (IDS), maintained by a society under the name of the International Data Spaces Association (IDSA), is one example of an initiative that attempts to concretize the theoretical concept of data spaces. The IDSA defines the IDS Reference Architecture Model (IDS-RAM) specification, with its latest release the IDS-RAM 4, as a global standard for building and participating in data spaces, providing functional and design guidelines on the components needed for data sharing and identity verification, while ensuring data sovereignty and privacy. This specification defines a component called the IDS Connector as the primary trusted software interface that is used to initiate and perform data sharing interactions among participants (Otto et al., 2022).

Despite the growing adoption of the IDS-RAM standards by diverse organizations and consortiums, the specification itself is loosely defined and does not address specific technical details, resulting in different implementation approaches across various data spaces and organizations (International Data Spaces Association, n.d.a). This implementation discrepancy potentially leads to interoperability issues among different IDS Connectors, regardless of an intra- or inter-data space setting. Without the presence of a technical standardization, organizations must continuously adapt their IDS Connector architecture and protocol implementations according to each data space that they wish to participate in (International Data Spaces Association, 2023b).

As an attempted solution to handle this surging issue of technical interoperability, the IDSA, together with its interested parties, are currently pushing forward experiments and research on an architectural shift, fabricating the Dataspace Protocol (International Data Spaces Association, 2023c). Recently, the first stable version, Dataspace Protocol 2024-1, has been released to the public by the IDSA. It is initially intended to act as the core reference for the technical implementation of each data space component with the aim to ensure seamless interoperability among different participants (International Data Spaces Association, 2024a).

The introduction of the Dataspace Protocol to the data space community has solidified the knowledge and understanding of both prospective and current data space participants on how to build a technical implementation of an IDS Connector. However, on its own, it is insufficient to fulfil all the demands for technical specifications of every data space component. The only focus of the Dataspace Protocol is on setting out the specific technicalities for the IDS Connector's Control Plane; the coordinating layer of the IDS Connector based on this new specification, which encompasses the communication of catalogues, contracts, and data for data exchange procedures between participants. Though crucial, data exchange only achieves one of the many aspects of data spaces. Specifications for other elements such as identity management, participant metadata storage, auditability trails and vocabulary hubs remain untouched and undiscussed. Likewise, similar protocols defining the technical standards of these remaining IDS components are equally important and must be introduced for the global growth and outreach of data space adoption.

The Eclipse Dataspace Working Group¹, in which the IDSA is also a part of, is currently one of the frontrunners in studies and contributions regarding the creation of these protocols. One of the maturing protocol drafts in serious development is the Eclipse Dataspace Decentralized Claims Protocol². This protocol aims to build technical guidelines regarding the identity management of a data space. Identity management, in charge of enabling a data space participant to prove its identity, authenticity and authorization to ensure trusted and secure data sharing, is emphasized in both the IDS-RAM 4 (Otto et al., 2022) and the Dataspace Protocol 2024-1 (International Data Spaces Association, 2024b), described as a crucial component of the name Identity Provider.

The existing IDS-RAM 4 specification describes abstract functional guidelines for the Identity Provider implementation to introduce an identity management system where a single principal component is in control of the identity of all participants within the data space (Otto et al., 2022). Despite this blueprint, centralized identity management is recently believed to have the potential of being an additional barrier for security and interoperability in data spaces due to the need for reintegration with different centralized Identity Providers every time an IDS Connector wishes to participate in a new data space.

Instead, ongoing studies are moving away from the usage of centralized identity management and nearing towards an approach of decentralized identity management, including what is currently being performed in the Eclipse Dataspace Decentralized Claims Protocol. Decentralized identity management brings forward the idea that each IDS Connector is responsible of its own identity, by carrying with itself its own personal standardized Identity Provider to every data space that it participates in, enabling to perform two essential tasks: firstly, to communicate with the data space's own Identity Provider for requesting admittance into the data space, and secondly after approval of data space participation, to communicate with other data space participants as a prerequisite before conducting the data sharing processes.

Aligned with the European Commission's proposed revision for the electronic Identification, Authentication and Trust Service 2.0³ (eIDAS 2.0) regulation, which aims for identity management interoperability leading to long term collaboration and partnership, experiments

¹ <https://www.eclipse.org/org/workinggroups/dataspace-charter.php>

² <https://projects.eclipse.org/proposals/eclipse-dataspace-decentralized-claims-protocol>

³ <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52021PC0281>

to integrate decentralized identity management in data spaces have been implemented by utilizing Self-Sovereign Identity (SSI) wallets (van den Wall Bake et al., 2024). In short, an SSI wallet is a digital container which allows entities to store, manage and share their own digital identity information without relying on any other third-party system (Kubach et al., 2020). Representation of this digital identity information is typically incorporated by using World Wide Web Consortium (W3C) recommendations of Decentralized Identifiers (DIDs), Verifiable Credentials (VCs) and Verifiable Presentations (VPs). In brief, DIDs are globally unique digital identifiers that can prove authentication over itself based on a trusted underlying technology of choice (Sporny et al., 2022b), VCs are trustworthy and tamper-evident digital claims made about an entity by an issuing authority, and VPs are a subset of one or more VCs that can be used for the purpose of sharing to other entities for verification (Sporny et al., 2022a).

1.2. Problem Statement

Although the Dataspace Protocol provides further clarity on the processes for data sharing in data spaces, the lack of specifications for the technical implementations of other IDS components provides vast opportunities for greater contribution of developing standardized protocols. Being the important ingredient that identity management is in data spaces, a technical protocol, like what the Dataspace Protocol is to the IDS Connector, is most desired to concretize the technicalities of decentralized Identity Providers implemented through SSI wallets. Despite SSI wallets and its inherent technologies, itself not being completely new topics in the field of information technology, the context of its utilization within a data space domain is fresh and requires further refinement and assessment.

1.3. Research Objective

The Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek (TNO) is an independent not-for-profit research organisation that is enthusiastic to participate in and develop a variety of state-of-the-art global innovations and technical advancements; in which data spaces and the IDS are no exceptions. Stemming from this interest, TNO as a member of the IDSA actively brings substantial contributions to the IDS-RAM and the Dataspace Protocol across current and previous release versions (TNO, n.d.).

The current initiative to define a standard technical protocol for data space identity management is a research topic of interest for TNO. Together with the IDSA, TNO wishes to contribute to the Eclipse Dataspace Decentralized Claims Protocol by conducting further studies on standardizing SSI wallet technologies for data spaces, specifically about the suitability of underlying DID methods support in SSI wallets.

At the present time, a working in-house IDS Connector, the TNO Security Gateway⁴ (TSG), built with compliance to the IDS-RAM and Dataspace Protocol standards is already developed by TNO. This IDS Connector is composed of the Control Plane and Data Plane components, as outlined by the Dataspace Protocol. Along with these components is a new SSI wallet implementation integrated under the TSG. In this TSG SSI wallet implementation, a prospective data space participant can store a DID in their SSI wallet and use this stored DID to register itself to an existing data space by including it in a participation request sent to the data space's

⁴ <https://gitlab.com/tno-tsg/dataspace-protocol/tno-security-gateway>

SSI wallet. The data space's SSI wallet then issues a VC, following the OpenID for Verifiable Credential Issuance⁵ protocol, to the participant to signify that it has been approved to participate in the data space. Once the participant is in possession of this VC, it can be used to generate and share VPs, through the OpenID for Verifiable Presentations⁶ protocol, to prove identity for data exchange with other participants in the data space.

Having said that, this SSI wallet implementation comes with its own imperfections. Currently, the implementation only supports one DID method, the did:web method, which was chosen due to reasons of ease and familiarity of integration. It is unclear and uncertain whether the did:web method is the best fit for data spaces, especially surrounding concerns related to security, reliability, and flexibility. Therefore, an attempt to extend the SSI wallet to accommodate other suitable DID methods is not only desirable, but essential for its long-term effectiveness and contribution to the data space identity management technical protocols.

The main objective of this thesis study is to discover which from the wide selection of DID methods is most suitable for usage in this domain of SSI wallets in data spaces, by evaluating their characteristics such as underlying technology, security risk, performance, and other related factors, against the specific core principles of data spaces identity management. Based on the results of an evaluative analysis, an implementation of the most suitable DID method will be developed on the TSG SSI wallet, and a validation against the existing did:web will be carried out to confirm the implementation results.

1.4. Research Questions

Based on the background, problem statement and research objectives as described above, the following main research question and supporting sub-research questions are defined for this thesis study.

MRQ: How to identify suitable Decentralized Identifier (DID) methods for implementation in Self-Sovereign Identity (SSI) wallets in the context of a data space?

SRQ 1: What are the common DID methods used in SSI wallets, based on recent European and global trends?

The first sub-research question aims to explore popular DID methods used in similar systems to gather foundational knowledge about the current landscape of DID methods before selecting those to consider for recommendation.

SRQ 2: What standardized rubric can be used to compare the selected DID methods in the context of usage in a data space?

The second sub-research question aims to identify existing recognized DID method rubrics or models, and develop a framework that will be used in this study to evaluate the suitability of different DID methods.

⁵ https://openid.net/specs/openid-4-verifiable-credential-issuance-1_0.html

⁶ https://openid.net/specs/openid-4-verifiable-presentations-1_0.html

SRQ 3: How to compare the selected DID methods based on the derived evaluation rubric?

The third sub-research question aims to focus on comparing the selected DID methods by using the created rubric, outlining each of their fundamentals, benefits, and drawbacks.

SRQ 4: Based on the evaluation results, which DID method is the most suitable for implementation in an SSI wallet within a data space domain?

The fourth sub-research question aims to interpret the results of the evaluation process from SQR3 to produce a final selection for a candidate of the most suitable DID method.

SRQ 5: How can support for the new DID method be introduced into an existing data space SSI wallet?

The fifth sub-research question aims to examine the practical aspects of integrating the selected DID method from SRQ4 into TNO’s data space SSI wallet, covering the software design and technical implementation.

SRQ 6: How can the new SSI wallet implementation be verified and validated in the context of usage in a data space?

The sixth sub-research question aims to verify validate whether the new DID method and its implementation in TNO’s data space SSI wallet meets the intended purposes and expectations.

1.5. Research Approach

This thesis study's approach is in accordance with the Design Science Methodology (DSM) for Information Systems and Software Engineering by Wieringa (2014). “Design Science” is defined as the design and investigation of certain artifacts in specific contexts to reach desired effects. In this thesis study, the “artifact” is defined as the implementation of DID methods on SSI wallets, and the “context” is defined as IDS-based data spaces. The DSM introduces an engineering cycle that is composed of five subsequent parts: Problem Investigation, Treatment Design, Treatment Validation, Treatment Implementation, and Implementation Evaluation, as illustrated in Figure 1.

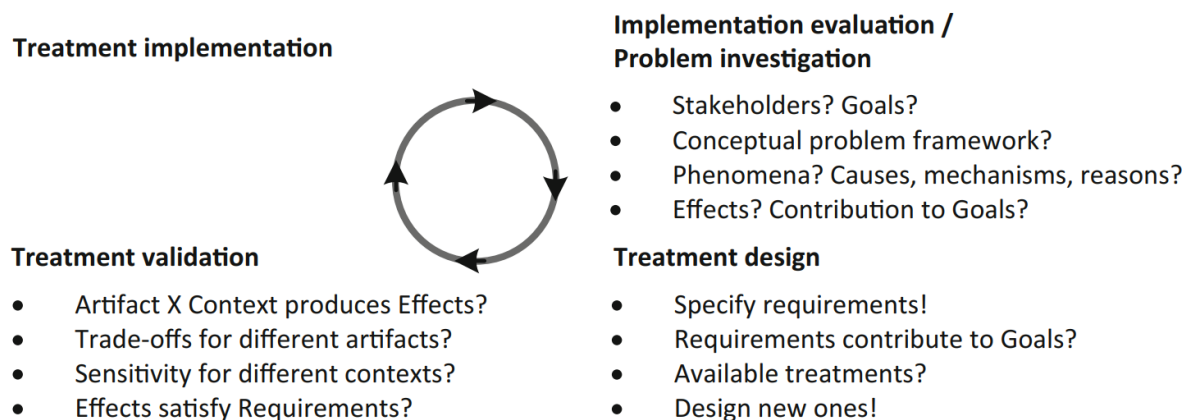


Figure 1: Design Science Methodology engineering cycle (Wieringa, 2014)

The first three parts of the cycle are considered as the design cycle and will become the principal guidance for this thesis study. The final two parts of the cycle complete the engineering cycle, where the results of the design cycle are transferred, used, and validated in the real world. Due to time constraints, the scope of this thesis study only encompasses the design cycle aspect of DRM, as follows:

1) Problem Investigation

This initial step of the DRM focuses on discovering the context and current situation, defining what improvements are needed, and related work that has been conducted. With respect to this thesis study, the first step encompasses performing an exploratory study on the related fundamental terminologies, technologies and work to build the foundational understanding of the study, and then conducting data collection of commonly implemented DID methods by performing a survey on other IDS Connectors and independent SSI wallets, and finally formulating an appropriate evaluation rubric for comparing DID methods usage in data spaces.

2) Treatment Design

In the second step, the produced DID method evaluation rubric is used to perform the evaluation on the collected DID methods, both artifacts from the previous step, aiming to single out the candidate for most suitable DID method to be implemented. The software design for the SSI wallet to support this new DID method alongside the existing implemented DID method is developed. Implementation of this new software design for the SSI wallet is also conducted in this step.

3) Treatment Validation

The last step of the design cycle aims to validate the DID method evaluation and implementation results. Performance of the SSI wallet using the newly implemented DID method will be compared against performance of the preceding DID method.

Figure 2 below shows the exact actions performed in each part of the cycle in the form of a flowchart for further clarity of the outline of this research.

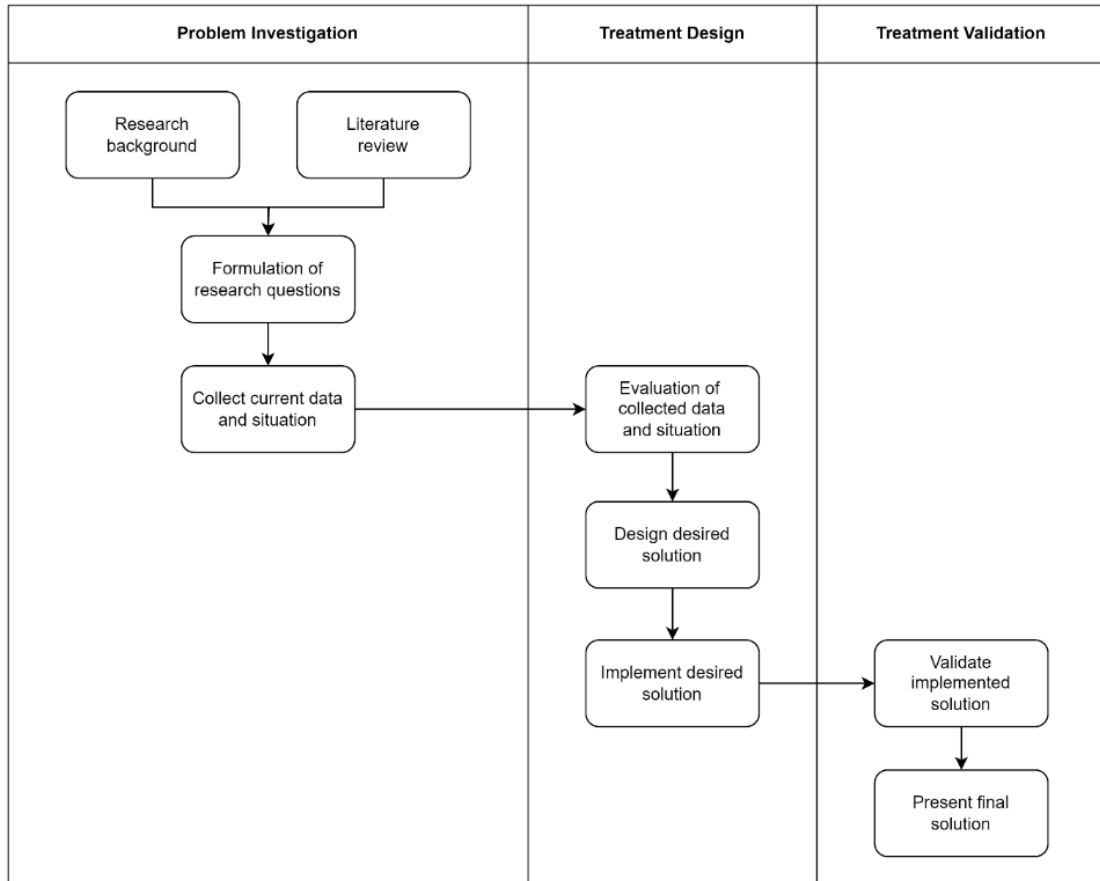


Figure 2: Flowchart of Research Outline based on Design Science Methodology

Although not covered in this thesis, the possibility of extending this research project to the fourth and fifth parts of the cycle is not completely ruled out. As the main aim of this thesis is to contribute to the real technical standardizations of data space Identity Providers, organizations such as TNO or others participating in the IDSA could take the results of this thesis into consideration and attempt to introduce it to their real data space scenarios.

1.6. Report Structure

There are six chapters in this thesis study report, organized as follows: Chapter 1 introduces the background, motivation, and research questions of the study. Chapter 2 presents an exploratory study encompassing terminologies and technologies that will be used to accomplish the research project and previous work relating to the study's topic. Chapter 3 describes the DID methods selection, which includes a collection process from the selected knowledge sources, as well as discussions regarding results of the data gathering process and a final selection of DID methods for evaluation. Chapter 4 demonstrates the DID methods evaluation process, beginning from comparisons of the core material references to establish the rubric, the formulation of the evaluation rubric, and ending with the actual evaluation process on the selected DID methods. Chapter 5 addresses a verification and validation on the results of the DID methods evaluation process, by creating a design implementation of the candidate DID method on an existing SSI wallet and performing several scenarios that covers various aspects of identity management in data spaces. Chapter 6 summarizes and concludes the entire thesis study, also briefly emphasizing current limitations and future work that could be conducted regarding this topic.

2. Exploratory Study

2.1. Fundamental Terminologies

2.1.1. Data sharing

Data sharing is the collaborative exchange and usage of data between several parties towards a common vision in mind. It is not just a mere one-off trade of data for internal usage within each party; data sharing aims to achieve a shared goal through the usage of data (Otto, 2022b).

2.1.2. Data sovereignty

Data sovereignty is the ability for an entity to be self-determined and in complete control of the data that they own, in regards of who can access it, how it can be accessed, and at what costs (Pettenpohl et al., 2022).

2.1.3. Trustworthiness

Trustworthiness is the degree of trust of a trustor in a trustee, which can vary based on one context to another. Trustworthiness can be decomposed into three other qualities: reliability, truthful information communication and transparency. Indicators of trustworthiness differ in many forms depending on its application, such as information about the implementation of privacy and security measures within a system, or data certificates and provenance information for systems handling large amounts of data (Amaral et al., 2020).

2.1.4. Interoperability

As defined by the European Interoperability Framework, interoperability is the capability of different organizations, with the same beneficial goals in mind, to interact with one other and involve the sharing of knowledge and information through exchanging data between their ICT systems. Four layers of interoperability are distinguished: 1) legal: ensures that organizations operating under distinct legal frameworks and policies can work together, by clarifying that legislations should not block each other; 2) organizational: ensures that business processes, responsibilities and expectations among different organizations are commonly agreed and aligned towards mutual goals; 3) semantic: ensures that the meanings (semantics) and formats (syntaxes) of the data and information are well-preserved throughout exchanges between organizations; and 4) technical: ensures that the infrastructures and applications of different organizations can properly link their systems and services together, for example through interface specifications and secure communication protocols (European Commission, 2017).

2.1.5. Data space

A data space is a distributed data integration concept that enables its participants to perform secure, trustworthy and interoperable data sharing without the involvement of a central data store, contrary to traditional central data integration concepts. In data spaces, data is instead stored and managed on each individual source, and the exchange is performed directly between the two participants (data provider and data consumer) upon request. There is also no requirement for the data on each source to follow a common pre-defined schema (Otto, 2022b). The data space itself is not involved in handling any data during the exchange process, granting complete data sovereignty to the data owners. The data space merely manages a catalog of data resources from each provider, including the regulations by which the data can be used (Otto, 2022a).

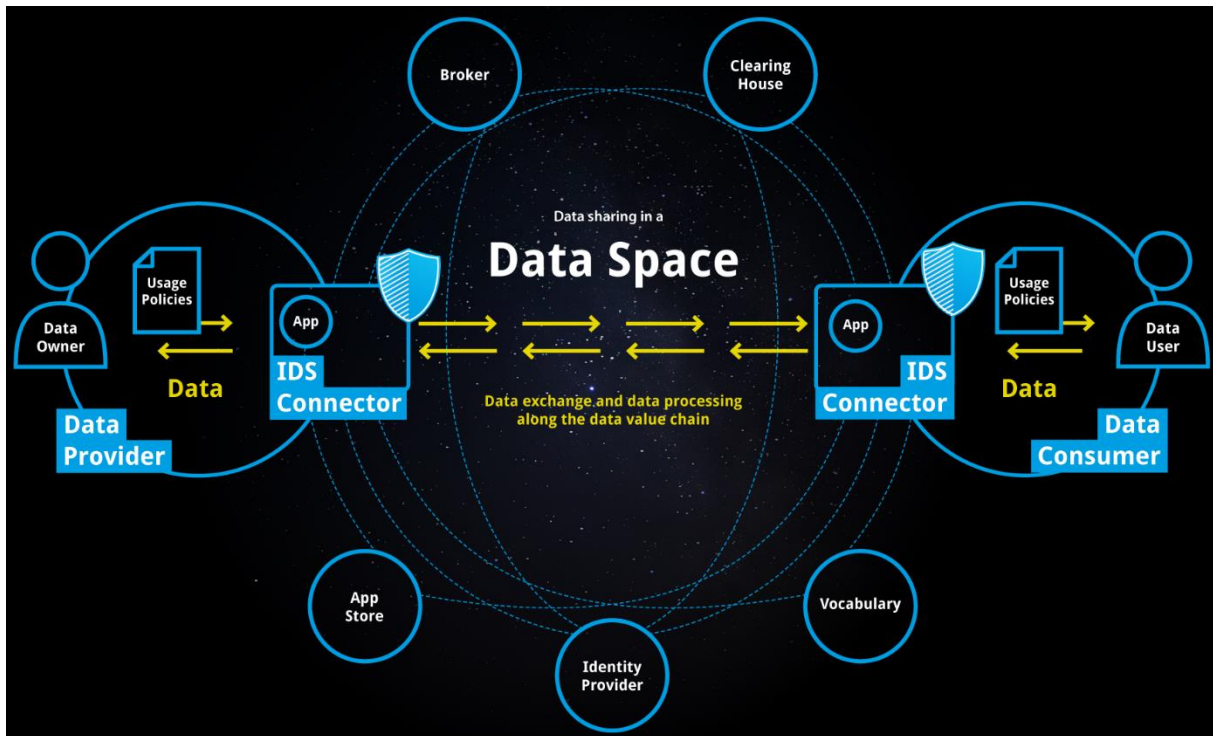


Figure 3: Architecture of a data space (International Data Spaces Association, n.d.b)

2.1.6. JavaScript Object Notation (JSON)

JavaScript Object Notation, or JSON for short, is a text-based syntax designed to enable standardized and structured data exchange between various programming languages. It utilizes a format that is composed of braces, brackets, colons, and commas, which are intended for different purposes. As inferred from its name, JSON was inspired by JavaScript's object literals, however, the internal data structures of the JavaScript programming language are not imposed onto other programming languages that consume the format. JSON itself only defines a technical and structured format for the data exchange process and does not encompass the shared understanding regarding the meaning, semantics, and syntaxes of the data, which should be refined further through other means between the data producer and consumer (ECMA International, 2017).

```
{
  "squadName": "Super hero squad",
  "homeTown": "Metro City",
  "formed": 2016,
  "secretBase": "Super tower",
  "active": true,
  "members": [
    {
      "name": "Molecule Man",
      "age": 29,
      "secretIdentity": "Dan Jukes",
      "powers": ["Radiation resistance", "Turning tiny"]
    }
  ]
}
```

Figure 4: Example of JSON

2.1.6.1. JSON-Linked Data (JSON-LD)

JSON-Linked Data, or commonly abbreviated as JSON-LD, is an extension of the JSON standard, providing it capabilities to serialize Linked Data whilst still building off the structure defined by JSON, aiming for easy integration with minimal changes. Linked Data is a concept of creating a network of standard-abiding machine-readable data across different web-based environments, by allowing an entity to begin at one instance of Linked Data and follow embedded links to other instances of Linked Data hosted on different web locations. An additional `@context` property is included in the JSON-LD document to denote where to find the interpretation rules of the different properties contained in the document. Moreover, JSON-LD is a concrete syntax of the Resource Description Framework (RDF), meaning that a JSON-LD document fulfills both as an RDF and a JSON document, also representing an instance of an RDF data model (Sporny et al., 2020).

```
{
  "@context": "https://json-ld.org/contexts/person.jsonld",
  "name": "Manu Sporny",
  "homepage": "http://manu.sporny.org/",
  "image": "http://manu.sporny.org/images/manu.png"
}
```

Figure 5: Example of JSON-LD

2.1.6.2. JSON Lines (JSONL)

JSON Lines, shortened as JSONL, is a text format that is essentially a file containing multiple newline-delimited valid JSON values. It allows multiple structured JSON values to be stored in a single location and processed as one record at a time. JSONL is commonly used for storing log messages or processing large messages across different parties (JSON Lines, n.d.).

```
{"name": "Gilbert", "wins": [["straight", "7♣"], ["one pair", "10♥"]]}
{"name": "Alexa", "wins": [["two pair", "4♠"], ["two pair", "9♠"]]}
{"name": "May", "wins": []}
{"name": "Deloise", "wins": [["three of a kind", "5♣"]]}
```

Figure 6: Example of JSONL

2.2. International Data Spaces (IDS)

The International Data Spaces, governed by the International Data Spaces Association (IDSA), is a real-life technology-agnostic initiative of a data space implementation, designed to be interoperable with other data spaces. The standards to build or participate in an IDS-compliant data space are defined in the IDS Reference Architecture Model (International Data Spaces Association, 2023a).

2.3. IDS Reference Architecture Model 4 (IDS-RAM 4)

The IDS Reference Architecture Model 4 is defined by IDSA as currently the most up-to-date standardized blueprint for trusted and sovereign data exchange within a virtual data space ecosystem. It outlines the architecture, components, and protocols necessary to build and participate in a working data space. However, the IDS-RAM 4 does not explain in detail how each component and their protocols are to be developed. The IDS-RAM is composed of five distinct layers: Business Layer, Functional Layer, Information Layer, Process Layer, and

System Layer. Figure 7 shows an architectural overview of the System Layer (Otto et al., 2022). Two components related to this study are discussed in the following subsections.

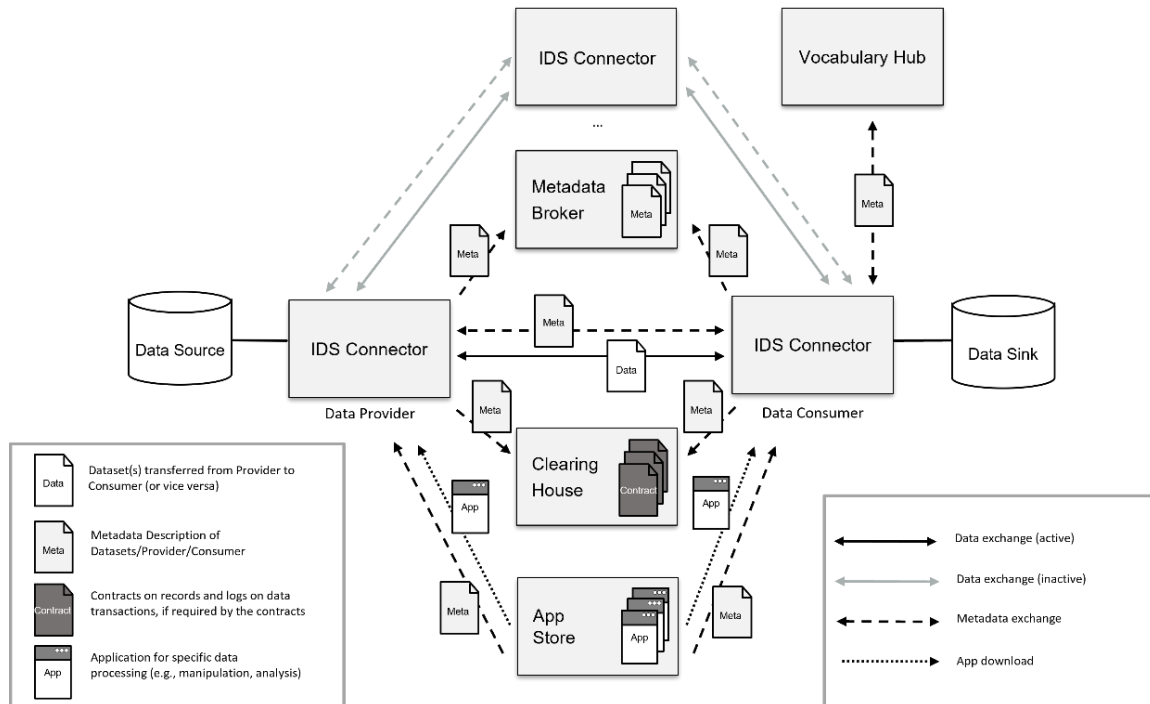


Figure 7: Interactions of components in the IDS-RAM 4 System Layer (Otto et al., 2022)

2.3.1. IDS Connector

An IDS Connector is the main component that facilitates the interactions for the data sharing process. It acts as a trusted gateway to connect one organization and its data to other organizations and their data via standardized data exchange endpoints, allowing interoperability with organizations of diverse types (Otto et al., 2022).

2.3.2. Identity Provider

An Identity Provider is the main driver for Identity and Access Management (IAM) in a data space ecosystem to ensure aspects of identification, authentication, and authorization. The Identity Provider is composed of three components: Certificate Authorities (CAs) for issuing and managing identity claims, Dynamic Attribute Provisioning Service (DAPS) to provide tokens for connector interactions, and Participant Information Services (ParIS) to provide business related information of the data space participants (International Data Spaces Association, 2022).

2.4. Dataspace Protocol 2024-1

The Dataspace Protocol 2024-1 is a technical specification of how the IDS System Layer can be realized. It aims to facilitate knowledge about the technicalities to achieve technical interoperability of data sharing among different entities that abide by usage control policies. The specifications define the schema and protocol formats for data space participants to publish a data catalog, negotiate contract agreements, and engage in the actual data transfer process (International Data Spaces Association, 2024b). As presented on Figure 8, two layers, the Control Plane and Data Plane make up the architecture of the Dataspace Protocol, of which descriptions are provided in the following subsections. The approach of this architecture is

rooted in the common best practices of distributed systems, such as those seen in software-defined networking (Kreutz et al., 2015) and microservices service mesh (Li et al., 2019), which aims to separate the management and configuration tasks (Control Plane) from the data processing and operational functions (Data Plane) in the hopes to enhance scalability, security, and flexibility.

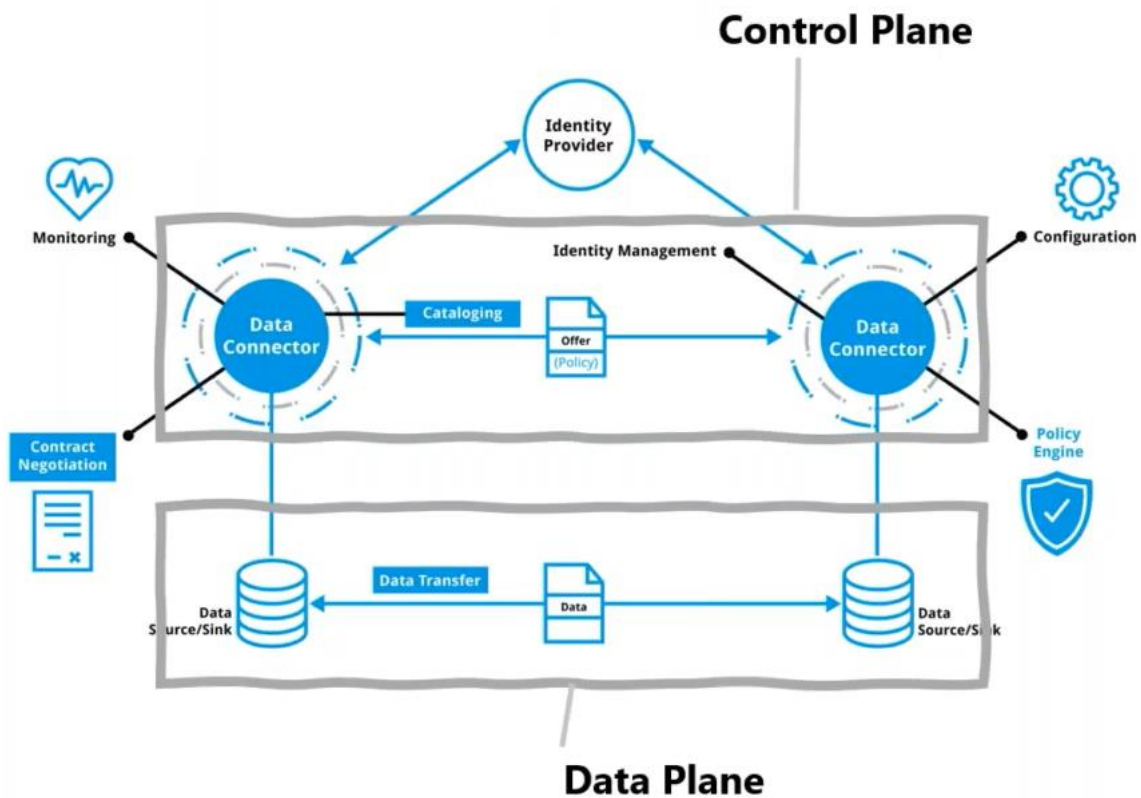


Figure 8: Interactions of components in the Dataspace Protocol (Steinbuss, 2024)

2.4.1. Control Plane

A Control Plane is the main coordinating layer within the technical specifications of the IDS Connector. It is composed of three core functionalities: Catalog Request, Contract Negotiation, and Transfer Process. The Catalog Request shows what kind of data is offered by the data space participant and how it is offered to other participants, following the formats of the Data Catalog Vocabulary⁷ (DCAT) standards. The Contract Negotiation allows negotiation of usage control policies among data space participants before engaging in data exchange, abiding by the formats in the Open Digital Rights Language⁸ (ODRL) standards. The Transfer Process initiates and manages the practical data exchange. Specific protocols for each of the three functionalities are defined in the Dataspace Protocol (International Data Spaces Association, 2024b).

2.4.2. Data Plane

Unlike the Control Plane which only manages indicators of the initiation and completion of the data exchange interaction, the Data Plane is the component of the IDS Connector that performs the actual data exchange process. This process could be implemented in the form of different wire protocols such as HTTP, Kafka, etc. as its specification is not defined in the Dataspace Protocol, allowing for reuse of previous data exchange implementations. Data exchange on the

⁷ <https://www.w3.org/TR/vocab-dcat-3/>

⁸ <https://www.w3.org/TR/odrl-model/>

Data Plane is only triggered and performed once all the processes on the Control Plane are completed (International Data Spaces Association, 2024b).

2.5. Self-Sovereign Identity (SSI) wallet

A Self-Sovereign Identity wallet is the concept of a digital wallet that allows holders to independently maintain and control their digital identities, for the main purpose of proving their identity to other digital entities or services to establish trusted digital relationships. This decentralized identity model is an alternative to other traditional approaches, which are commonly either creating a new account at each service provider or relying on a single account with a large identity provider. SSI wallets aim to solve the issue of these dependencies on centralized or federated identity model, as it gives complete sovereignty to holders regarding which trusted credentials they want to share to verifiers. The concept of trusted credentials and identities in SSI wallets relies on Verifiable Credentials and Decentralized Identifiers (Preukschat and Reed, 2021).

2.6. Verifiable Credentials (VCs)

Verifiable Credentials are a set of one or more digital claims made about an entity that are trustworthy, tamper-evident, and cryptographically verifiable. These credentials can contain various information relating to the entity, the issuer, or other specific details. An entity who asserts a claim about another entity and generates a verifiable credential for it is called an Issuer. An entity who possesses one or more verifiable credentials and is able to generate verifiable presentations from them is called a Holder. An entity who processes verifiable credentials and verifiable presentations is called a Verifier (Sporny et al., 2022a).

2.6.1. Verifiable Presentations (VPs)

Verifiable Presentations can be generated by extracting information from one or more VCs so that only a subset is shared to requesting parties to confirm that the entity is in ownership of verifiable credentials that fulfill the requested characteristics (Sporny et al., 2022a).

2.7. Decentralized Identifiers (DIDs)

Decentralized Identifiers are a new type of globally unique digital identifiers that can be self-generated by entities via a trusted underlying technology of choice, referred to as a verifiable data registry (VDR). The specifications outlining guidelines for a new DID infrastructure is called a “DID method”. Each entity can have multiple DIDs for different identities and usages. This is the opposite of existing globally unique identifiers; which are commonly issued, assigned, and revoked by centralized or federated external authorities, leaving no control for the entity itself. The format of a DID must abide by `did:<method-name>:<method-specific-identifier>` (Sporny et al., 2022b). There are currently nearly 200 DID methods recognized by W3C, with various types of underlying verifiable data registries such as distributed ledger technology (DLT), cryptographic keys, web domains, etc. (Steele & Sporny, 2023).

2.7.1. DID Document

DID Documents are JSON-LD files that contain information associated with a DID identifier, allowing an entity claiming to be the owner of a certain DID to prove control over it. In the

initial DID creation process, a DID Document is stored on the VDR based on the DID identifier. These DID Documents can then be resolved by retrieval from the VDR through referring to the DID identifier itself. DID Documents can typically contain cryptographic key materials of different types (the verificationMethod array of objects property), the usages of the defined key materials (properties such as authentication, assertionMethod, etc. as defined in the W3C DID specification), and the trusted services associated with the DID (the service array of objects property) (Sporny et al., 2022b).

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/jws-2020/v1"
  ],
  "id": "did:example:123",
  "verificationMethod": [
    {
      "id": "did:example:123#key-0",
      "type": "JsonWebKey2020",
      "controller": "did:example:123",
      "publicKeyJwk": {
        "kty": "EC",
        "crv": "P-256",
        "x": "Er6KSSnAjI700bRWh1aMgqyIOQYrDJTE94ej5hybQ2M",
        "y": "pPVzCOTJwgikPjuUE6UebfZySqEJ0ZtsWFpj7YSPGEk"
      }
    }
  ],
  "authentication": [
    "did:example:123#key-0",
  ],
  "assertionMethod": [
    "did:example:123#key-0",
  ],
  "service": [
    {
      "id": "did:example:123#linked-domain",
      "type": "LinkedDomains",
      "serviceEndpoint": "https://bar.example.com"
    }
  ]
}
```

Figure 9: Example of a DID Document

2.7.2. did:web

The did:web is a DID method specification that depends on the existing reputation of web domains, utilizing web URLs as the method specific identifiers. Examples of DIDs using the did:web method specification include did:web:w3c-ccg.github.io and did:web:example.com. The DID Document JSON-LD file is stored under the /.well-known path of the web domain (Gribneau et al., 2023).

2.8. Workflow of DIDs, VCs and VPs

According to Brunner et al. (2021), the flow of integration between DIDs, VCs and VPs is as follows. For this explanation, the roles that will be used are Issuer (of a VC), Holder (of a VC) and Verifier (of a claim).

1) Issuance

An Issuer must firstly create a DID for themselves based on their DID method of choice to be used as a form of their identity. The Issuer then requests for the Holder's DID. If the Holder does not have a DID yet, they must also create it based on their DID method of choice, which need not to be the same as the Issuer. Once the Issuer has collected the Holder's DID, the Issuer creates a VC that contains at least the DIDs of both the Issuer and Holder, and digitally signs the VC with their DID. This VC is then sent to the Holder.

2) Sharing

When the Holder is already in possession of the VC created by the Issuer, the Holder is free to select a subset of the claims in the VC to share it with any Verifier. The Holder performs this by creating a VP containing the selected claims that they are willing to share. This VP is then shared with a Verifier. There are no set standards for the communication method between the Holder and Verifier.

3) Verification

Unlike the Issuer and Holder, a Verifier is not required to hold a DID. The VP that a Verifier receives from a Holder can contain one or more claims about the Holder. To verify each claim, the Verifier must retrieve the DID Documents of both the Issuer and Holder (resolvable from the DIDs stored in the VP), and then uses the cryptographic keys stored in the DID Documents to verify that the signatures of the claim are indeed signed by the Issuer and Holder. Moreover, the Verifier might need to check if the VC is still valid and has not been revoked. The Verifier then can check if the claim is valid.

2.9. Previous Work

Both the DID v1.0⁹ and VC Data Model v1.1¹⁰ specifications have just recently received recommendation status by W3C in the year 2022, meaning that the maturity of this field is still young. Understandably, though it exists, the amount of academic and scientific publications regarding these concepts are still shallow. Albeit most studies still aim to introduce the basic concepts and workflows such as in the subchapters above, there also exists some substantial research regarding the comparison and evaluation of different DID methods; these studies are discussed in this subchapter.

2.9.1. Categorization of DID Methods

Bistarelli et al. (2023) conducted a study to categorize all W3C-compliant DID methods (as of May 2023) based on the characteristic of their underlying technology, which are defined as public ledger, private ledger, permissioned ledger, permissionless ledger, non-ledger, and ledger-agnostic. Their results show that most DID methods currently adopt non-public blockchains, and out of those that use public blockchains, Ethereum is the most popular. The

⁹ <https://www.w3.org/TR/did-core/>

¹⁰ <https://www.w3.org/TR/vc-data-model/>

study also highlights which DID method specifications have included concrete implementations and remain as drafts as of 2022.

On a more comprehensive categorization process, the study by Hoops et al. (2022) develops a new taxonomy to classify DID methods. The taxonomy disperses DID methods into several categories, that the authors call “DID method instance group”, based on characteristics related to use case (general/specific), registry technology (self-contained/DLT/web service/distributed hash table), deployment requirement (none/independent/coordinated), operation support (CR/CRUD), explicit cost (free/write fee/recurring costs/write fee & recurring cost), identifier format (human-readable/not), and DID Document capabilities (minimal/basic/keys/services/arbitrary). The goal of the creation of this taxonomy is to become a first point of reference for practitioners to understand the differences between DID methods and continue to build upon that knowledge to make an informed decision regarding which DID method to implement for a specific use case.

Although these studies do attempt to make a formal categorization of different DID methods based on their behaviors and characteristics, they are performed at a generic level and do not particularly encompass any single scope. Since the specific use case of using SSI wallets in data spaces is still a rapidly growing concept, it remains unknown up to what level of generality in respect to other scopes of SSI wallets can it be matched to. Hence, further studies should be conducted that precisely target comparative analysis of DID methods in the context of usage in data spaces.

2.9.2. Underlying Technologies for DIDs

Coming from a different approach, the study by Alizadeh et al. (2022) attempts to directly compare different underlying verifiable data registries that are commonly used as underlying technologies for DIDs against each other, without working with the specific DID methods. The experiments were conducted to measure the read time performance of an Ethereum-based blockchain and Kademia-based distributed hash table as verifiable data registries, in which the authors made the verdict that a distributed hash table is more recommended for usage as a DID verifiable data registry based on their results.

Despite the study resulting in ample analysis between the performance of the two verifiable data registries, the study does not present a recommendation on which exact DID methods to use. From an implementor’s standpoint, it does not provide much clarity because the same verifiable data registry could be used by multiple distinct DID methods, each with their own additional unique characteristics depending on the implementation of the DID method maintainer.

2.9.3. Evaluation of Specific DID Methods

In another study, Fdhila et al. (2021) performs an evaluation on six DID methods, basing the analysis on a combination of the W3C DID Method Rubric and SSI principles. The six DID methods (did:btc, did:v1, did:ethr, did:sov, did:web, did:peer) were selected based on the authors’ expertise in the field of SSI, specifically aiming to cover the different architectural designs of blockchain-based, non-blockchain-based, public permissioned, public permissionless, and pairwise. The evaluation criteria are spread into four categories: rulemaking (degree of decentralization), operation (execution of CRUD), security (potential attacks on integrity, correctness, and privacy), and implementation (actual implementation and utilization challenges). The study's results concluded that there is no single winner among the evaluated

DID methods, as each brings their own advantages and disadvantages. The choice of implementation heavily depends on the context of the use case.

This category of related work addresses the missing piece of the previous subsections in the way that it takes a specific subset of DID methods into focus for evaluation. However, the drawback of this study is that it does not explicitly disclose the concrete thoughts and procedures leading up to this DID methods selection's decision. Since this thesis study aims to play a significant contribution for standardizing the technical specifications of Identity Providers, there is a need to have a grounded foundation in the choice of DID methods to be evaluated.

3. DID Methods Selection

3.1. Knowledge Sources for Collection

The initial step of this thesis study is to perform a systematic collection of DID methods, and then make an informed selection producing a subset of DID methods for evaluation. This should be carried out to preserve neutrality and prevent the bias of selecting a set of DID methods based on the author’s preferences. To carry this out, two distinct categories of knowledge sources are elected as references for DID methods usage.

The first category is other usages of DIDs within the context of data spaces. As previously established in prior chapters, TNO is not the sole organization that researches on and contributes to the IDS; there also exists other organizations within the IDSA who perform related work as well. It is justified that these other organizations also have varying implementations of IDS Connectors being developed and integrated to several data spaces in the field. The monthly release of the Data Connector Report¹¹ by the IDSA regularly documents the current available implementations of data connectors that are built based off the IDS-RAM specification. The document lists numerous IDS Connectors (see Figure 11) and breaks them down across distinctive characteristics (see Figure 10), so that potential users can understand each of them in more detail. Thus, the Data Connector Report will be the main source of knowledge for reference against other DID implementations in the data space context.

2.1.23 TNO Security Gateway (TSG)	
Connector Overview	
Name	TNO Security Gateway (TSG)
Maintainer	TNO
Logo of the connector or company logo	TNO
Peculiarity of the connector	Multi-purpose connector
More Information	GitLab repository ⁷⁴
Connector Details	
Type	A generic open-source solution
Maturity	TRL 8
Portability	Agnostic
License	Open-source
IDS Certification	Yes (Connector Certification - Concept Review) ⁷⁵
Adoption examples	SCSN ⁷⁶ - actively used in production. Different European and Dutch projects.
Deployment options	Cloud
Service level	Self-service
Access & Usage Control	
Access control	Yes
Type of access control	» OAuth (Open authorization, standard/framework for REST/APIs) » API key (Manage access through a unique code for programming interface)

Figure 10: Screenshot of IDSA Data Connector Report detailed connector overview

¹¹ <https://internationaldataspaces.org/data-connector-report/>














Connectors with up-to-date information							
Section	Name of connector	Maintainer	Open source	IDS certified	Identity management	Deployment options	Service level
2.1.1	ALSOV Connector					<ul style="list-style-type: none"> Edge On-premises Cloud 	Platform-aaS
2.1.2	Boot-X Connector						Connector-aaS
2.1.3	ECI IDS Connector powered by TNO				X.509	Cloud	<ul style="list-style-type: none"> Connector-aaS Self-service
2.1.4	Eclipse Dataspace Components (EDC)		✓			Not specified	Service level is best effort of the open-source-community
2.1.5	EdgeDS Connector		✓			<ul style="list-style-type: none"> Edge On-premises Cloud IoT/CPS/OT 	Connector-aaS
2.1.6	EGI Datahub Connector		✓		X.509	<ul style="list-style-type: none"> On premises Cloud 	Platform-aaS
2.1.7	EDNA-X EDC Connector		✓		did:web	<ul style="list-style-type: none"> On-premises Cloud 	<ul style="list-style-type: none"> Connector-aaS Self-service
2.1.8	FIWARE Data Space Connector		✓		X.509	<ul style="list-style-type: none"> On-premises Cloud 	<ul style="list-style-type: none"> Connector-aaS Self-service
2.1.9	GATE Dataspace Connector		✓		X.509	On-premises	Platform-aaS
2.1.10	GDSO Connector - Tyre Information Service		Partially		AWS Cognito	<ul style="list-style-type: none"> Edge On-premises Cloud IoT/CPS/OT 	Platform-aaS
2.1.11	HEALTH-X dataLOFT EDC				SSI	<ul style="list-style-type: none"> On-premises Cloud 	Self-service
2.1.12	IIOC (Intel IONOS Orbiter Connector)		✓		SSI	<ul style="list-style-type: none"> Cloud IoT/CPS/OT 	
2.1.13	Kharon IDS Connector				Kharon	<ul style="list-style-type: none"> Edge Cloud 	<ul style="list-style-type: none"> Connector-aaS Platform-aaS
2.1.14	Mitsubishi Dataspace Connector		Partially			IoT/CPS/OT	Self-service
2.1.15	OneNet Connector		✓		X.509	On-premises	Connector-aaS

Figure 11: Screenshot of IDSA Data Connector Report overview list

The second category is the usage of DID in independent SSI wallets. The concept of SSI was not especially constructed for the single purpose of data spaces. The main driver for the conception of SSI wallets was for decentralized identity management. This means that SSI wallets can, and do, exist as standalone entities within the field of cyber security. Hence, this study will also extract information regarding DID methods that are supported by current available SSI wallets in the market. The Digital Wallet Overview¹², provided and maintained by the Digital Wallet and Agent Overviews Special Interest Group¹³ of the OpenWallet Foundation (OWF)¹⁴, provides a detailed list of 33 SSI wallets showing their characteristics, thus will be used as a reference (see Figure 12). The OWF is an organization that aims to enable collaboration on the development of open-source digital wallet technologies and govern their best practices (Graham & Goldscheider, 2023). The information provided on the Digital Wallet Overview is based on a survey, containing questions regarding core technical implementations, that was sent out to many well-known generic SSI wallet vendors. Its inception was initially

¹² <https://openwallet-foundation.github.io/digital-wallet-and-agent-overviews-sig/>

¹³ <https://github.com/openwallet-foundation/digital-wallet-and-agent-overviews-sig>

¹⁴ <https://openwallet.foundation/>

started by TNO under the TNO SSI Lab¹⁵, however, it has since been accepted by and handed over to the OWF in the hopes to allow for greater contribution (de Kok & van Leuken, 2023).

Wallet Overview											
General							Credential Format	Encoding Scheme	Signature Algorithm	Holder Identifier	Issuer Identifier
Wallet	Deployment	Organisational Wallet	Open Source	Links to app	Support	API					
AceID Wallet	-	-	-	---	-	-	-	-	-	-	-
Apple Wallet	-	-	-	---	-	-	-	-	-	-	-
Atala PRISM	-	-	No	---	-	-	JWT-VC, soon:AnonCreds	JSON, JSON-LD	ECDSA, soon: CL	did:prism	did:prism
Authenticator	-	-	-	---	-	-	-	-	-	-	-
BC Wallet	-	-	-	---	-	-	-	-	-	-	-
Blockcerts Wallet	-	-	Yes	---	-	-	-	-	-	-	-
Blockpass	-	-	No	---	-	-	-	-	-	-	-
CardShare Wallet	-	-	No	---	-	-	AnonCred, JSON-LD	JSON	BBB+	did	did:sov
Data Wallet	-	Yes, but branded as Enterprise Wallet Platform by Grant.io	No	▶ Ⓞ Ⓜ	e-mail	-	AnonCreds, JWT-VC, ICAO DTC, X.509	JSON	CL, ES256, ECDSA	did:ebio, did:key, link:secrets, X.509	did:ebio, did:key, did:sov, X.509
Datakeeper	-	-	No	---	-	-	LDP-VC	JSON-LD	ECDSA	did:eth	did:eth
Digital ID Wallet	-	-	-	---	-	-	-	-	-	-	-
status Wallet	-	-	A new version of the wallet - called S...	▶ Ⓞ -	-	-	AnonCred	JSON	CL	Link secret	Link secret
Gataca	-	-	No	---	-	-	LDP-VC, JWT-VC	JSON-LD	EdDSA, RSA, secg256k1, P-256, CatEa	did:key	did:gatac
Iss_ID: irol	-	-	No	---	-	-	JWT, JWT-VC, LDP-VC	-	BBB+ ECDSA	did:eth, did:ion, did:web, did:jwk, did:key	did:web, did:key, did:jwk
helix id	-	-	partly open sourced	---	-	-	JWT, JWT-VC	compact and JSON serialization, JSON, JSON-LD	BBB+ ECDSA	did:key, ENS, did:eth	did:key, ENS, di...

Figure 12: Screenshot of OWF Digital Wallet Overview

From these two sources, it is expected that a large pool of DID methods will be collected. Although it is a great possibility that not every single one of the DID methods gathered will be used for the evaluation process, these two sources of knowledge will become the foundation in leading to the focused subset of DID methods to be evaluated.

¹⁵ <https://github.com/tno-ssi-lab/wallet-overview>

3.2. Collection Results

3.2.1. In other IDS Connectors

Table 1 below documents all current IDS Connector implementations that integrate SSI as a means of decentralized Identity Provider, according to the overview provided in the March 2024 release of the Data Connector Report (Giussani & Steinbuss, 2024).

The results show that roughly only around one third (10 out of the 28) of the IDS Connector implementations are currently reported as utilizing SSI, in which only two of them (FIWARE Data Space Connector & Prometheus-X Dataspace Connector) are truly open-source, meaning that the other 8 implementations are inaccessible by the public. Two IDS Connector implementations (Amadeus EONA-X EDC Connector & truzzt IIOC IoT Connect) were also flagged as open-source by the Data Connector Report, but neither their version control repository nor technical documentation were discovered. However, out of these 10 SSI-implementing IDS Connectors, the Data Connector Report does specify 6 connectors that support did:web for identity management. Based on these findings, the overall results are 1 connector implementation supporting did:ebsi, 6 connector implementations supporting did:web, and 3 remain unknown due to the incapability of auditing their version control repository or technical documentation. It can be concluded that the results of this initial data gathering are not exactly diverse, as the amount of openly available information is limited.

Despite this, for cases in the real world, there may be IDS Connector maintainers who are experimenting on the usage of SSI for their Identity Provider, but still remain undocumented by the IDSA, similar to what is currently happening at TNO. Due to this hypothesis, further review was performed on IDS Connector implementations that are listed as open-source on the Data Connector Report. The review was carried out by accessing the version control repository and technical documentation of each open-source IDS Connector, and performing analysis on the current work. Out of the 16 connectors indicated as open-source in the Data Connector Report, findings from a review of 11 of them are shown in Table 2 below. The five excluded connector implementations consists of the TSG, and those four already discussed in the paragraph above, such as FIWARE Data Space Connector, Prometheus-X Dataspace Connector, Amadeus EONA-X EDC Connector, and truzzt IIOC IoT Connect. An additional IDS Connector implementation, the Tractus-X EDC was also discovered during the review process and included in the findings.

The results of the analysis in Table 2 show that only two other IDS Connector implementation are currently truly working with SSI technologies, which are the Eclipse Dataspace Components – Framework and the Tractus-X EDC. The Mitsubishi Electric Dataspace Connector also enables the usage of SSIs, however its architecture is largely built comparably based off the Eclipse Dataspace Components – Framework, hence they can be regarded as identical. The Tractus-X EDC was also initially based on the Eclipse Dataspace Components – Framework, however substantial architectural and design modifications have influenced it to become its own unique implementation.

Name of connector	Source code available?	Supported DID methods?	Notes
FIWARE Data Space Connector	YES	did:ebasi	https://github.com/FIWARE/data-space-connector/tree/main/doc/flows/service-interaction-m2m
Prometheus-X Dataspace Connector	YES	did:web	According to source code, usage of SSI is still in progress. https://github.com/Prometheus-X-association/dataspace-connector/tree/main
Huawei Boot-X Connector	NO	did:web	Currently closed-source but planning to be open-source in the future.
Amadeus EONA-X EDC Connector	NO	did:web	Open-source according to Data Connector Report, but version control repository not found.
sovity CaaS (Connector-as-a-Service)	NO	did:web	Closed-source.
Tech2B SCSN Connector	NO	did:web	Closed-source.
Telekom DIH Connector	NO	did:web	Closed-source.
Fraunhofer HEALTH-X dataLOFT EDC	NO	UNKNOWN	Currently closed-source but planning to be open-source in the future.
truzzt IIoC IoT Connect (Intel IONOS Orbiter Connector)	NO	UNKNOWN	Open-source according to Data Connector Report, but empty version control repository. https://gitlab.truzzt.com/iioC/iioC-core
Tekniker IDS Connector	NO	UNKNOWN	Currently closed-source but planning to be open-source in the future.

Table 1: IDS Connectors labelled as SSI-implementing based on the IDSA Data Connector Report (Giussani & Steinbuss, 2024) (as of 28 March 2024)

Name of connector	Supported DID methods?	Notes
Eclipse Dataspace Components – Framework	did:web did:ion	did:ion only mentioned in documentation but unimplemented. https://github.com/eclipse-edc/Connector/tree/main/extensions/common/iam/decentralized-identity https://github.com/eclipse-edc/Publications/blob/main/Identity%20Management/DID_EDC.md
Mitsubishi Electric Dataspace Connector	did:web did:ion	Forked from Eclipse Dataspace Components. https://github.com/huebl/DataSpaceConnector/tree/main/extensions/common/iam/decentralized-identity
Tractus-X EDC	did:web	Not on the IDSA Data Connector Report, but discovered later on during this review process. https://github.com/eclipse-tractusx/tractusx-edc/tree/main/edc-extensions/ssi/ssi-miw-credential-client
Intracom Telecom EdgeDS Connector	NOT SUPPORTED	https://github.com/jkalogero/EdgeDS
EGI DataHub Connector	NOT SUPPORTED	https://egitlab.iti.es/euhubs4data/egi-datahub-connector https://docs.egi.eu/users/aai/check-in/
GATE Dataspace Connector	NOT SUPPORTED	https://github.com/gate-institute/DataspaceConnector
GDSO Connector – Tyre Information Service	NOT SUPPORTED	https://gdso-org.github.io/tech-doc/docs/getting-started/authentication
OneNet Connector	NOT SUPPORTED	https://github.com/european-dynamics-rnd/OneNet
Fraunhofer Silicon Economy EDC	NOT SUPPORTED	https://git.openlogisticsfoundation.org/silicon-economy/base/ids/silicon-economy-edc
sovity Open-Source EDC Connector	NOT SUPPORTED	https://github.com/sovity/edc-extensions https://github.com/sovity/edc-ui
Engineering TRUE Connector	NOT SUPPORTED	https://github.com/Engineering-Research-and-Development/true-connector/blob/main/doc/advancedConfiguration/identityproviders.md
Fraunhofer Trusted Connector	NOT SUPPORTED	https://github.com/Fraunhofer-AISEC/trusted-connector https://industrial-data-space.github.io/trusted-connector-documentation/docs/overview/

Table 2: Open-source IDS Connector implementations, based on the IDSA Data Connector Report, with additional descriptions regarding SSI (as of 28 March 2024)

3.2.2. In independent SSI wallets

Table 3 presents the findings of DID implementations in existing self-standing SSI wallets. Although the Digital Wallet Overview listed an abundance of SSI wallets that the OWF intended to extract information from, the list is not exhaustive as many wallet vendors provided incomplete information or did not respond at all. Hence, additional research was conducted to retrieve this missing information. Moreover, to ensure that the information used for this study is up to date, the DID methods support that were already listed on the Digital Wallet Overview were also confirmed to the latest progressions of each SSI wallet.

Both of these additional processes done by searching for and reviewing the technical documentation and version control repositories of the SSI wallet implementations. This additional information is also indicated in Table 3. The results of this analysis shows a considerable amount of variation of DID method support, although several common DID methods repeatedly appear among different SSI wallet implementations.

Out of the total 32 SSI wallets that were reviewed, the technical documentation or version control repository of 18 of them were publicly discoverable. 4 of them were confirmed to not support DIDs. 12 of them showed either in the technical documentation or version control repository that DID methods were supported, in which 11 out of the 12 stated either addition or subtraction of DID methods support compared to the latest version of the Digital Wallet Overview. The other 2 SSI wallets did not explicitly state the support of DIDs, however the DID methods were listed on the Digital Wallet Overview and used as a reference.

Regarding the other 14 SSI wallets in which no technical documentation or version control repository were found, 9 of them have DID methods listed on the Digital Wallet Overview hence the overview is used as the source of truth for these cases. The implementations of the remaining 5 SSI wallets remain unknown due to the absence of information.

For easier visibility and understanding, Table 3 orders the SSI wallets in the clusters of characteristics as stated in the above paragraphs.

Name of SSI wallet	Tech doc / Source code?	DID supported?	Supported DID methods?	Notes
Apple Wallet	https://developer.apple.com/documentation/passkit/apple_pay_and_wallet https://developer.apple.com/documentation/walletpasses	NO	NOT SUPPORTED	
Blockpass IDN	https://docs.blockpass.org/ https://github.com/blockpass-org	NO	NOT SUPPORTED	
Cleverbase Vidua	https://cleverbase.com/en/developer-documentation/	NO	NOT SUPPORTED	
Privacy by Design Foundation Yivi	https://irma.app/docs/what-is-irma/	NO	NOT SUPPORTED	
IOHK Atala PRISM now “Hyperledger Labs Open Enterprise Agent”	https://github.com/hyperledger-labs/open-enterprise-agent https://iohk.io/en/blog/posts/2023/12/04/iog-contributes-atala-prism-to-hyperledger-foundation/	YES	did:prism did:peer	did:prism retrieved from Digital Wallet Overview, did:peer added after reviewing source code documentation.
Government of BC Wallet	https://digital.gov.bc.ca/digital-trust/about/about-bc-wallet/ https://github.com/bcgov/bc-wallet-mobile https://github.com/openwallet-foundation/credo-ts https://github.com/bcgov/issuer-kit	YES	did:web did:key did:jwk did:peer did:sov did:indy did:cheqd	Uses credo-ts for DID methods support.
Hyland Credentials Blockcerts Wallet	https://github.com/blockchain-certificates/cert-issuer?tab=readme-ov-file#working-with-dids	YES	Multiple	Uses the DIF universal resolver.
KayTrust Wallet	https://developer.kaytrust.id/Specs/	YES	did:ev	did:ev not listed on Digital Wallet Overview.
MATTR Wallet	https://learn.mattr.global/docs/vii-platform/dids/structure https://learn.mattr.global/tutorials/dids/overview-create	YES	did:key did:web did:ion	did:ion no longer supported, according to technical documentation.
Metadium MYKEEPiN	https://github.com/coinplug/mykeepin-sdk-py	YES	did:meta	did:meta not listed on Digital Wallet Overview.

Animo Paradym Wallet	https://github.com/animo/paradym-wallet https://docs.paradym.id/learn-about-paradym	YES	did:jwk did:key did:web	
Sphereon Wallet	https://sphereon.com/sphereon-products/sphereon-wallet/ https://github.com/Sphereon-Opensource/mobile-wallet	YES	did:jwk did:key did:ion did:cheqd did:web did:lto did:factom did:ethr did:pkh did:accumulate did:ebsi	did:pkh retrieved from Digital Wallet Overview, not found in technical documentation and source code. did:accumulate and did:ebsi not listed on Digital Wallet Overview. did:ebsi still under development.
Talao	https://talao.io/blog/building-future-proof-digital-identity-wallets-with-talao-technology-stack/ https://talao.io/talao-wallet/	YES	did:ebsi did:ion did:web did:tz did:key did:jwk did:hedera did:pkh did:polygonid	did:jwk, did:hedera, did:pkh and did:polygonid not listed on Digital Wallet Overview.
walt.id	https://github.com/walt-id/waltid-identity/tree/main/waltid-did https://docs.oss.walt.id/issuer/sdks/manage-dids/overview	YES	did:key did:jwk did:web did:cheqd did:ebsi did:iota did:velocity	did:ebsi and did:iota mentioned in source code documentation, but not implemented yet. did:velocity mentioned in Digital Wallet Overview, but not mentioned in technical documentation and source code.
Microsoft Authenticator	https://learn.microsoft.com/en-us/entra/verified-id/decentralized-identifier-overview	YES	did:web	did:web not listed on Digital Wallet Overview.

Microsoft Entra	https://learn.microsoft.com/en-us/entra/verified-id/decentralized-identifier-overview	YES	did:web	did:web not listed on Digital Wallet Overview.
iGrant.io Data Wallet	https://docs.igrant.io/docs/ https://developer.igrant.io/	YES but no DID spec., refer to Digital Wallet Overview	did:ebsi did:key did:sov	
ZADA	https://github.com/lycheeventures/zada-wallet	YES but no DID spec., refer to Digital Wallet Overview	did:sov	
Ego Company CertiShare Wallet	NOT FOUND refer to Digital Wallet Overview	YES	did:sov	
Rabobank Datakeeper	NOT FOUND refer to Digital Wallet Overview	YES	did:ethr	
Gataca	NOT FOUND refer to Digital Wallet Overview	YES	did:key did:gatc	
Gimly Tap ID	NOT FOUND refer to Digital Wallet Overview	YES	did:ethr did:ion did:web did:jwk did:key	
helix id	NOT FOUND refer to Digital Wallet Overview	YES	did:key did:ethr	
Identry	NOT FOUND refer to Digital Wallet Overview	YES	did:sov	
Lissi Wallet	NOT FOUND refer to Digital Wallet Overview	YES	did:sov did:indy	
Spherity Wallet	NOT FOUND refer to Digital Wallet Overview	YES	did:ethr	
Validated ID VIDwallet	NOT FOUND refer to Digital Wallet Overview	YES	did:key did:jwk did:ala did:ebsi did:ethr	

AceBlock AceID Wallet	NOT FOUND refer to Digital Wallet Overview	UNKNOWN	UNKNOWN	
Thales Digital ID Wallet	NOT FOUND refer to Digital Wallet Overview	UNKNOWN	UNKNOWN	
Esatus Wallet	NOT FOUND refer to Digital Wallet Overview	UNKNOWN	UNKNOWN	
IDEMIA Mobile ID	NOT FOUND refer to Digital Wallet Overview	UNKNOWN	UNKNOWN	
idento.one	NOT FOUND refer to Digital Wallet Overview	UNKNOWN	UNKNOWN	

Table 3: Independent SSI wallets, curated from the Digital Wallet Overview, and their supported DID methods (as of 14 March 2024)

3.3. Selection Results

The results of the data collection above are summarized in Table 4. The first column lists all of the DID methods that were supported at least in one instance among the reviewed IDS Connectors and SSI wallets. The second column gives a short description of the underlying technology utilized for the verifiable data registry, categorized as “Non-ledger” or “Ledger”. The last three columns represent a component of the usage frequency of the DID methods, where the first of the three columns tally usage in other IDS Connectors, the second in independent SSI wallets, and the third in total of combined frequency.

Based on these findings, 24 distinct DID methods were collected in total from the two knowledge sources. Table 4 displays the collected DID methods firstly in clusters of underlying technology, and then sorts each of them by usage frequency within each cluster. With respect to the underlying technology, the results show that ledger-based verifiable data registries are overall more popular than its non-ledger-based counterparts. 18 DID methods use some form of DLT (blockchain/Hashgraph/directed acyclic graph), meanwhile only 5 of the DID methods collected are categorized as not dependent on any ledger technologies, and 1 DID method with unknown underlying technology (did:accumulate). No related specifications and documentations were found for the did:accumulate method, hence it will be neglected in this study.

Considering usage frequency, it can be concluded that did:web is the most popular supported DID method with a total of 18 instances, where there is an equal split of 9 usages in both other IDS Connector implementations and independent SSI wallets. The did:web method is also the most popular DID method supported on the SSI wallets of IDS Connectors, given the small sample space of two other DID methods support. On the other hand, did:key is the most popular DID method supported by independent SSI wallets, with all 11 counts attributed to it. Overall, did:web is the most popular supported non-ledger-based DID method, just edging did:key by 1 count. Among the ledger-based DID methods, a tie of 6 usages between did:ion, did:ebis, did:sov, and did:ethr are at the top of the ranks.

In the interest of time, it is determined that only a subset of the overall collected DID methods will be considered for in-depth evaluation in the next step of this thesis study. For diversification purposes, the choice of subset will include the top 5 most frequently supported DID methods based on the two categories of underlying technologies. This leads to 10 DID methods included in the subset, {did:web, did:key, did:jwk, did:peer, did:pkh} for non-ledger-based DID methods, and {did:ion, did:ebis, did:sov, did:ethr, did:cheqd} for ledger-based DID methods.

However, as often previously mentioned, the did:web method is already currently supported by the TSG SSI wallet. In order to maintain the heterogeneity of the subset of DID methods, and also considering the popularity of the usage of did:web, it is desired that a similar but enhanced DID method is evaluated in replacement. Two of such DID methods were discovered via online search. Firstly, the Trust Over IP Foundation (2023) recently published an article announcing the conception of a DID method that they deemed as the successor of did:web, called the did:webs which incorporates self-certifying techniques to the did:web architecture. Secondly, Curran (2024) introduced a new DID method called did:tdw that builds on the did:web method and provides additional functionalities for self-certification and verifiable history. Hence, the focused set of DID methods that will be used for further evaluation are = {did:webs, did:tdw, did:key, did:jwk, did:peer, did:pkh, did:ion, did:ebis, did:sov, did:ethr, did:cheqd}.

DID Method	Underlying Technology	Usage Frequency		
		Other IDS Connectors	Independent SSI wallets	In total
did:web	Non-ledger (Web)	9	9	18
did:key	Non-ledger (Self)	0	11	11
did:jwk	Non-ledger (Self)	0	7	7
did:peer	Non-ledger (Self)	0	2	2
did:pkh	Non-ledger (Self)	0	2	2
did:ion	Ledger (Bitcoin Blockchain)	2	4	6
did:ebsi	Ledger (EBSI Blockchain)	1	5	6
did:sov	Ledger (Sovrin Blockchain)	0	6	6
did:ethr	Ledger (Ethereum Blockchain)	0	6	6
did:cheqd	Ledger (cheqd Cosmos Blockchain)	0	3	3
did:indy	Ledger (Hyperledger Indy Blockchain)	0	2	2
did:prism	Ledger (Cardano Blockchain)	0	1	1
did:gatc	Ledger (Ethereum, Hyperledger Fabric, Hyperledger Besu, Quorum Blockchains)	0	1	1
did:ev	Ledger (Ethereum Blockchain)	0	1	1
did:meta	Ledger (Metadium Blockchain)	0	1	1
did:lto	Ledger (LTO Blockchain)	0	1	1
did:factom	Ledger (Factom Blockchain)	0	1	1
did:tz	Ledger (Tezos Blockchain)	0	1	1
did:hedera	Ledger (Hedera Hashgraph)	0	1	1
did:polygonid	Ledger (Polygon Blockchain)	0	1	1
did:ala	Ledger (Alastria Blockchain)	0	1	1
did:iota	Ledger (IOTA DAG)	0	1	1
did:velocity	Ledger (Velocity Blockchain)	0	1	1
did:accumulate	Unknown	0	1	1

Table 4: Summary of findings of current DID method implementations

4. DID Methods Evaluation

4.1. Rubric References

Once the assortment of DID methods is collected, the next step of this thesis study is to prepare for the evaluation process. To perform a comprehensive analysis of the DID methods, the application of an evaluation rubric is crucial to ensure a structured, consistent and objective comparison. According to Dickinson and Adams (2017), an evaluation rubric is a set of criteria that is used for explicit and standardized evaluations, with the aim to better determine the quality and success of the entity being evaluated. As the field of DIDs itself is still a relatively new concept, presently there exists only few alternatives of evaluation rubrics for DID methods.

The W3C DID Method Rubric v1.0¹⁶ is the most popular, as it is directly endorsed by W3C as a group note. This rubric includes an extensive scoring guide for a range of different aspects of a DID method, mainly surrounding its decentralization. Before using the rubric, W3C recommends that the profuse amount of criteria outlined in the rubric be selected based on the specific use case of the evaluation (Andrieu et al., 2021). This rubric was also used by Fdhila et al. (2021) as a reference when creating their custom evaluation rubric. The criteria categories of the W3C DID Method Rubric are summarized in Table 5. In addition to those presented in Table 5, the W3C DID Method Rubric website also includes some potential additional criteria for the future, albeit currently excluded from the actual rubric itself because they are deemed as still unfitting for the time being.

Criteria Category	Description
Rulemaking	By whom and how were the specifications made? Criteria: Open contribution (participation); Transparency; Breadth of Authority; Public vs private economies; Cost
Design	What specifications are there? Criteria: Permissioned operation; Interoperability; Scope of usage
Operations	How are the specifications executed and ensured? Criteria: Financial accountability; Limited resource resolution; Limited resource registration
Enforcement	How is the violation of specifications identified and addressed? Criteria: Auditability
Alternatives	What available alternative implementations exist? Criteria: Active implementations; Market share; Platform support; Language support; Rogue risk; Forkability
Adoption & Diversity	How widely is the specification used? Criteria: Acceptance; Users; International adoption; In which countries; Language support
Security	What security guarantees are there?

¹⁶ <https://www.w3.org/TR/did-rubric/>

	Criteria: Robust Crypto; Expert Review; Future Proofing; Self Certification; Availability; Evolution; Many Eyes; Diffuse Control; Regulatory Compliance
Privacy	How is privacy assured? Criteria: Per-DID constraints on visibility; Cros-DID Leakage; Incentives for Multicontext DID; Revocation of Consent

Table 5: Criteria categories of W3C DID Method Rubric v1.0 (Andrieu et al., 2021)

The plethora of criteria encompassed in the W3C DID Method Rubric is extensive and may become overwhelming for most potential evaluators. In line with W3C’s recommendation and as a means to simplify and transform the rubric to be more easily understood by newcomers in the field of DIDs, Cunningham et al. (2022) have attempted to push forward an improved and focused rubric criteria that aims to help interested individuals and organizations of all experience levels, backgrounds and roles to quickly get a grasp on which existing DID methods to implement, or at least to exclude from their consideration. This new proposed rubric builds off the existing W3C DID Method Rubric by selecting criteria specifically related to standardization, and adding new related questions that the authors determined as appropriate. It essentially creates distinct subsets of the current rubric for three different roles described in the paper: standards bodies, product owners, and implementors. Despite these efforts, as of the time of writing, this new rubric is still incomplete and remains a document draft from the 11th edition of the Rebooting the Web of Trust event.

The evaluation rubric used for this thesis study will refer to the W3C DID Method Rubric as the core foundation; the criteria descriptions and corresponding scoring guide used in the resulting evaluation rubric will be identical to that from the W3C DID Method Rubric. However, as advised by W3C itself, only a subset of the numerous criteria will be selected for actual usage, in this instance with reference to the exact domain of data spaces. This selection process will follow a similar manner to that performed for the proposed rubric by Cunningham et al. (2022). A tabular checklist will be utilized to show clarity of mapping between evaluation criteria and domain characteristic, where the y-axis represents each criteria outlined in the W3C DID Method Rubric, and the x-axis represents characteristics of identity providers of data spaces.

With the aim to have a strong basis for the choice of characteristics of data space identity providers, the specific characteristics will be based on the information provided by the Data Spaces Support Centre¹⁷ (DSSC). The DSSC is a consortium that aspires to contribute to the establishment of common data spaces with respect to European Union values, by performing exploration on the different needs of many data space initiatives, defining common requirements for them, and organize best practices accelerating the construction of sovereign data spaces. These best practices are presented in the form of various business-, organizational- and technical-related building blocks, in which the two most related to building a data space identity provider fall within the Data Sovereign & Trust pillar, namely the Identity & Attestation Management and Trust Framework building blocks. The Identity & Attestation Management building block refers to the management of identities and attestations within a data space to guarantee reliability and integrity by setting and verifying participant information through the use of recognizes technical standards and frameworks. The Trust Framework building block

¹⁷ <https://dssc.eu/>

outlines how to ensure that a participant in a data space complies with specific rules and a shared set of standards.

Although the DSSC does provide a lengthy description of what each building block should provide, it does not define clear-cut characteristics that can be directly utilized for this thesis study. However, the DSSC does present a set of core design choices in the form of open-ended questions as a driver for consideration when implementing each building block. In this thesis study, the characteristics of the identity provider of data spaces will be derived from the core design choices of the Identity & Attestation Management and Trust Framework building blocks. Table 6 below shows the extraction of these characteristics, showing the core design choices of each building block. The results of this characteristics extraction will then later be used in the criteria selection for creation of the final evaluation rubric.

DSSC Building Block	Core Design Choices	Extracted Characteristics
Identity & Attestation Management	How do you represent information about entities in a data space to make it verifiable and enable trust and interoperability?	Verifiability; Trustworthiness; Interoperability;
	How do you ensure the reliability of information on participants' identities?	Authenticity;
	How do you identify the participants in a data space?	Authenticity;
Trust Framework	How is it verified that an entity is a real and valid legal entity?	Verifiability;
	How can claims/attestations made by participants be validated and verified?	Validity; Verifiability;
	How is policy enforcement ensured for the usage of data?	Compliance;
	How are claims signed, and how can Trust anchors validate and verify these signatures?	Validity; Verifiability;
	What are the technical enablers to implement a secure, robust trust framework, and how is interoperability ensured?	Trustworthiness; Interoperability;

Table 6: Extraction of data space identity provider characteristics from core design choices of DSSC building blocks

4.2. Rubric Creation

The results of the selection process of the W3C DID Method Rubric criteria based on the characteristics of data space identity providers are displayed in Table 7. The format of Table 7 adheres to what was described in Chapter 4.1.

From the eight criteria categories established by the W3C DID Method Rubric, two categories were excluded from the selection process. These were the Alternatives and Adoption & Diversity criteria categories. The reasoning behind this is because it can be argued that a large portion of the criteria making up these two categories surround the usage frequency and popularity of the DID methods, and in this thesis study, research regarding those factors were already conducted in the form of DID methods collection from alternative IDS Connector implementations and independent SSI wallets. Moreover, on the W3C DID Method Rubric itself, the criteria listed under these two categories are still incomplete, evident by the fact that there are no concrete scoring schemes provided yet, as W3C faces difficulty in constructing the scores because the adoption of DID methods in real production environments is still growing. W3C plans to add the standardized scorings for these categories in the near future once there is more widespread adoption of DIDs.

Considering the elimination of the two categories mentioned above, 25 criteria in total across six distinct categories (Rulemaking, Design, Operation, Enforcement, Security, and Privacy) were evaluated and underwent the selection process. In the cells of Table 7, an “X” is marked to indicate that the criteria satisfies the specific characteristic of that column. 21 out of the 25 handled criteria fulfil at least one out of the six data space identity provider characteristics. These 21 criteria will be considered for the resulting DID methods evaluation rubric to be used in the next stages of this thesis study. The complete rubric, and its derived scoring sheet, can be seen on Appendix B.

A difference in the resulting rubric from this study compared to the reference W3C DID Method Rubric lies in the scoring scale that is applied. The W3C DID Method Rubric operates using a scoring scale of A-D, where A being the highest and D being the lowest score. Instead of an alphabetical scale, the new rubric will utilize a scoring scale of 1-4, where 4 is given as the highest score and 1 as the lowest. This adjustment was made so that it will be easier to perform a calculation on the total cumulative scores of each evaluated DID method.

A brief discussion supporting the decisions made on the relationship of each criterion to characteristics are given below:

- 1) Open contribution (participation): This criterion discusses how open the degree of participation towards governance of the DID method specification is for different parties. By allowing more contributors to participate in the specification, the sense of validity and verifiability grows as more parties can cross-check the proposed contributions before it is accepted into the specification. Moreover, as the number of specification maintainers increase, the validation and verification performed leads to increased trustworthiness of DID users. Involvement of a greater number of diverse participants in governing the specification will also open the opportunity of compliance towards more regulations, and usage interoperability as it accommodates all these different parties. On the other hand, allowing the contributions of many parties does not always guarantee authenticity, as the true intentions of the contributors are not always transparent.

- 2) **Transparency:** This criterion considers how visible the specification governance processes are. The more visible the discussions and decisions on governance of the DID method specification, the higher the sense of authenticity will be, as it leaves less room for mischievous intents. Additionally, this practice increases the trustworthiness of the DID method as any interested party can view the complete historical inception of the specification and judge its fairness. Moreover, since all processes related to the specification governance are transparent, it will be possible to easily verify and validate the governance processes as any interested party can assess these factors themselves. On the contrary, the factors of interoperability and compliance are not directly related to the transparency of the rulemaking process, as transparency only touches upon the openness of specification governance process, instead of the actual adjustments to the specification.
- 3) **Breadth of Authority:** This criterion examines how many and which contributors currently actively participate in the governance process of the DID method specification. The interoperability of the DID method will likely increase when there is contribution from a greater breadth of parties. Likewise, compliance to regulations will also improve as each independent party contributing to the specification will consider fulfilling their regulatory requirements. Furthermore, the more parties contributing to the specification, the easier it is to trust it as there will be a more diverse set of parties who give attention to it. With greater participation, more parties will also be able to verify and validate the proposed changes made to the specification, to ensure that the changes are not made arbitrarily. However, authenticity may not be guaranteed, as the motivations and influences driving the different parties may not always be entirely transparent to the public.
- 4) **Public vs private economies:** This criterion attempts to evaluate the openness of the economic interest of the parties that govern the DID method specification. When the economic interests of a DID method are directed towards the greater good of society, it is more likely that the DID method is trustworthy as DID users can believe that the parties who govern the DID method specification only have benevolent intentions. Vice versa, when the economic interests of the governing body of a DID method specification is profit-driven, then it would only be natural that DID owners have some scepticism on adopting the DID method, as DID users may have suspicions on future plans of the body. Moreover, the more open the economic interests of the governing parties, the higher the likelihood for maintaining its authenticity.
- 5) **Cost:** This criterion focuses on assessing how expensive participation (time, money, and effort) is in terms of both governance of the specification and utilization of the DID method itself. In most cases, lower participation costs lead to a higher number of interested parties contributing to the specification and using the DID method. A greater number of participants means that it leads to greater interoperability through the diversity of contributors. The factor of costs can also be a catalyst that could indirectly affect other factors, such as when a DID method specification that is too expensive for participation leads to less contributors, there is greater potential for lower levels of trustworthiness, verifiability, and validity.

- 6) **Permissioned operation:** This criterion discusses whether permissions are required to perform DID operations on the VDR. The more permissions and restrictions there are to perform operations on the VDR, the less interoperable the DID method becomes, as there are less parties who can freely utilize it. Additionally, the level of permissions of a DID method could also impact its trustworthiness. In hindsight, it seems true that increasing permissions on the usage of the VDR of a DID method could strengthen its trustworthiness as only approved parties can interact with it, however counterintuitively, the presence of additional permissions could mean that the governing body of the VDR only allows access to parties that fulfil certain characteristics of their preferences, thus also impacting the validity of the DID method.
- 7) **Interoperability:** This criterion considers whether the DID method places restrictions on the technology that can work with it, such as specific SSI wallet implementations. At first glance, it is simple to understand that the more diverse and flexible technologies that the DID method can work with, the more interoperable it becomes. Restraining the types of technology that a DID method supports could have broader indirect impacts towards factors such as validity, verifiability and trustworthiness, as it may seem that the DID method maintainers are biased towards certain parties.
- 8) **Scope of usage:** This criterion examines how wide the scope of a DID method can be used, taking into account the scopes of universal, contextual, paired and central. The different levels of scopes certainly influence interoperability of the DIDs, as DIDs with a more universal scope can be utilized across different scenarios, whereas a DID with centralized scope could only be valid for one certain domain. Moreover, allowing the universal creation and usage of DIDs could increase its trustworthiness as it is more broadly accessible in contrast with DIDs that have limited scopes. Also as a result of this, universal DIDs allow for greater verifiability and validity. In contrast, authenticity is not a factor as DIDs created and used in paired or centralized environments could also provide the same degree of authenticity as universal or contextual DIDs.
- 9) **Financial accountability:** This criterion attempts to evaluate how transparent the economics of the DID method are. This criterion is similar to the Transparency criteria, however comes from a different point of view, in which the financial operations of the DID method are considered. A DID method that provides a more open view of its financial operations definitely will increase the level of trustworthiness, as its transparency will instil confidence to its users. Furthermore, it ensures a sense of authenticity and validity that the maintainers of the DID method are legitimate.
- 10) **Auditability:** This criterion focuses on assessing how the changes performed on a DID Document of this DID method can be cryptographically proven. DID methods that allow anyone to audit the cryptographic proof of the history of changes of the corresponding DID Document will embed a sense of confidence that the DID and DID Document are trustworthy, as all past operations performed on it are visible and cryptographically verifiable. Moreover, providing the cryptographic proof history to any party allows for anyone to verify that the DID Document has not been tampered with, guaranteeing the authenticity and validity of the information that is contained in the DID Document.
- 11) **Robust Cryptography:** This criterion discusses the level of security that is provided by the required cryptographic algorithms and key types supported by the DID method.

When a DID method supports cryptographic algorithms and key types that requires higher security levels, trustworthiness increases as users have more confidence that the DID method is protected from malicious activities. Additionally, this sense of trust from the increased protection will also ensure that the DID and DID Document are authentic, valid and less likely to be tampered with. Requiring a higher security level also makes the DID method more easily compliant with the requirements of different regulations.

- 12) Expert Review: This criterion considers how popular, approved, and experienced the cryptographic and security measures of the DID method are. Support for well-vetted and robust security measures can lead to more trust and confidence when using the DID method, as there is a proven record of accomplishment that the security measures used are credible. Due to that, there is also more trust that the DID and DID Document are and will remain valid and authentic with the strong protections in place. Supporting popular security measures also often leads to the possibility of interoperability as many other systems utilize the same measures. Additionally, well-known cryptographic and security measures are commonly required in many regulations, hence there is higher chance for the DID method to comply with diverse regulations.
- 13) Future Proofing: This criterion examines how easy it is for the DID method and its VDR to adopt improved security measures in the future. Immediately, the main related factor that comes to mind is interoperability. The ease of upgrading security measures enhances the degree of interoperability and compatibility with other technologies, paving way for the DID method to evolve further. Furthermore, the simplicity of adopting newer and better security mechanisms also significantly expands the DID method's ability to comply and fulfil the heightened demands of a more broad variety of regulations.
- 14) Self Certification: This criterion attempts to evaluate how related the randomness of the DID identifier is with the inception keys stored in the DID Document. When DID identifiers are derived directly from the DID's inception keys, trustworthiness will be increased among DID receivers as they know that the risk of impersonation is lowered. The DID identifier and inception keys stored in the DID Document can also be verified against each other to ensure that the contents are what they claim to be, thus maintaining the validity of the DID. As a result from these factors, the degree of authenticity is strengthened as genuineness is highly upheld.
- 15) Availability: This criterion focuses on assessing how robust and strong the VDR used by the DID method is against malignant risks attempting to suppress information flow. As the DID method's VDR is more immune and reliable, the more trustworthy it becomes because DID owners have more confidence in the DID method being consistently available. Building on top of that, assuring the availability of the VDR also ensures the validity of the DIDs and DID Documents as they remain accessible and persistent. Additionally, when the VDR is susceptible to risks that prevent its information flow to be available, it reduces the ability of parties to verify the DID and its corresponding verification methods in the DID Document.
- 16) Evolution: This criterion discusses whether the correctness of the current state of a DID Document can be checked against its historical proof from the DID method's VDR. When a VDR maintains a publicly accessible, strongly connected and complete state

evolution of a DID Document, users will have more trust and confidence towards the DID, as the current state can be verified against any past changes made regarding it. On the opposite end, when evidence of a proper evolution does not exist, then parties only rely on weak trustworthiness of the DID owner, and it will be difficult to believe in the authenticity and validity of the DID Document.

- 17) Many Eyes: This criterion considers whether the code of the DID method specification itself is public, how many contributors it has, and whether there are mechanisms to publish vulnerability reports. As reiterated multiple times in previous criteria, the presence of many contributors solidifies the trustworthiness and verifiability of the DID method, as it means various parties are interested in maintaining it. Moreover, the presence of security reporting mechanisms also help to improve the DID method's validity, as vulnerabilities are raised in a prompt manner. This criterion is not directly related to authenticity, as the number of active contributions does not entirely guarantee the authenticity, since different contributors may have different intentions that may not align exclusively with the goals of the DID method.
- 18) Regulatory Compliance: This criterion examines whether the DID method utilizes cryptographic mechanisms that abide by legal regulations. Naturally, this criterion is most related to the characteristic of compliance, in a sense that a DID owner should utilize a DID method that fulfils all, or at least most, of the regulatory obligations that the DID owner is interested in, and also in their particular domain of application.
- 19) Per-DID constraints on visibility: This criterion attempts to evaluate how visible or restricted a DID method's VDR is to the public. VDRs that are completely kept in private reduces the risk of any malevolent actions made by irresponsible parties as the audience to which the VDR is visible to is restricted. This will instil greater trustworthiness and ensure the authenticity of the information provided on the VDR. On the other hand, VDRs that are public with no visibility restrictions are also beneficial in terms of allowing for better verifiability of DIDs and DID Documents, as the information regarding DID Documents can be verified more easily.
- 20) Cross-DID Leakage: This criterion focuses on assessing how feasible it is to manage multiple DIDs whilst maintaining privacy and not demonstrating relations leading to the same owner. The more the entropy and the isolation among different DIDs generated from the same DID method by a single DID owner, the greater trust established for the DID method. DID owners then can trust that their DIDs are not traceable to one another, upholding their privacy. The characteristics of authenticity and verifiability may sound related to this criterion, however they focus more on the correctness of the DIDs and DID Documents itself.
- 21) Revocation of Consent: This criterion discusses whether a created DID and DID Document can be revoked and removed. The characteristic that comes directly into mind is trustworthiness. When a DID method provides possibilities for DID revocation, it aids in enhancing the trust of DID users, as they can be ensured that the resolvable DIDs are those that are still active and have valid DID Documents. Moreover, the ability to revoke consent on DIDs also aligns with several regulatory compliances, for example EU GDPR's "right to be forgotten".

Aside from discussing the 21 criteria that were selected to be included in the resulting evaluation rubric, explanations on why the remaining four criteria were excluded are given below:

- 1) Limited resource resolution: This criterion considers how much memory is needed in order to resolve a DID. A strong argument to include this criterion in the final evaluation rubric is the reasoning that lower minimum requirements of processing memory provides more interoperability that allows more devices to be able to independently perform DID resolutions, instead of redirecting the operation to other larger and powerful devices. However, the domain of data spaces itself actively works with enormous amounts of data, thus this already presents evidence that data space software components are generally deployed on larger systems to support the heavy processing load. Hence, this criterion is deemed irrelevant for the context of this study.
- 2) Limited resource registration: This criterion examines how much memory is needed in order to create and register a DID. Similarly to the previous criteria above, although requiring less memory to perform the DID creation operations would drastically increase the DID method's interoperability for devices of different sizes and capabilities, the domain of data spaces itself already requires large and powerful machines in order to perform its data sharing activities, hence this criterion is also excluded from the final evaluation rubric of this study.
- 3) Diffuse Control: This criterion attempts to evaluate whether mechanisms to decentralize the control over a DID to several distinct parties are supported. One could argue that this criterion is important in order to ensure trustworthiness because risk levels will decrease as the responsibility is no longer centralized to a single party when DID control is dispersed. However, the chances of needing diffuse of control in the context of data spaces is low, as a single DID should be owned and maintained by a single party controlling the data space connector, be it an individual or an organization. Therefore, the relevance of this criterion for the context of this study is minimal.
- 4) Incentives for Multicontext DIDs: This criterion focuses on assessing whether the DID method provides incentives for reusing a DID in multiple contexts. Within the domain of data spaces, it can be assumed that for the most part DIDs will be reused across different data spaces. There is not much urgency to create a brand-new DID when joining a data space for the first time, as this only adds to the complicatedness of using SSI in data spaces, where identity management actually only expresses a small component of the whole complex data space ecosystem. Additionally, in some cases, data space participants might want to be identified as the same party when participating across different data spaces to present consistency, and utilizing the same DID will help to achieve that. As a result, this criterion was exempt from the final evaluation rubric for this study.

		Characteristics of Identity Providers for Data Spaces (extracted from DSSC building blocks)					
		Verifiability	Trustworthiness	Interoperability	Authenticity	Validity	Compliance
Criteria from W3C DID Method Rubric v1.0	Rulemaking						
	Open contribution (participation)	X	X	X		X	X
	Transparency	X	X		X	X	
	Breadth of Authority	X	X	X		X	X
	Public vs private economies		X		X		
	Cost			X			
	Design						
	Permissioned operation		X	X		X	
	Interoperability			X			
	Scope of usage	X	X	X		X	
	Operation						
	Financial accountability		X		X	X	
	Limited resource resolution						
	Limited resource registration						
	Enforcement						
	Auditability	X	X		X	X	
	Alternatives						
	Active implementations						
	Market share						
	Platform support						
Language support							

Rogue risk						
Forkability						
Adoption & Diversity						
Acceptance						
Users						
International adoption						
In which countries?						
Language support						
Security						
Robust Cryptography		X		X	X	X
Expert Review		X	X	X	X	X
Future Proofing			X			X
Self Certification	X	X		X	X	
Availability	X	X			X	
Evolution	X	X		X	X	
Many Eyes	X	X			X	
Diffuse Control						
Regulatory Compliance						X
Privacy						
Per-DID constraints on visibility	X	X		X		
Cross-DID Leakage		X				
Incentives for Multicontext DIDs						
Revocation of Consent		X			X	X

Table 7: Selection process of W3C DID Method Rubric v1.0 criteria based on data space identity provider characteristics extracted from DSSC

4.3. Evaluation Results

The results of the evaluation process are described in the subsections below. Each subsection exclusively discusses a single DID method that was evaluated. The rubric evaluation forms used for the evaluation process are not included in this section, but instead situated in Appendix C for readability purposes of this paper. The discussions provided in each subsection are divided into three partitions: Summary, Benefits, and Drawbacks. The Summary portion explains the underlying technology and behavioural characteristics that the DID method brings. The Benefits portion elaborates the beneficial values that the DID method provides. Lastly, the Drawbacks portion highlights the weaknesses contained within each DID method.

4.3.1. Benchmark DID Method

Aside from the 11 DID methods that were selected for evaluation in the previous chapters, the existing supported `did:web` DID method was also evaluated to represent its stance as an anchoring benchmark.

4.3.1.1. `did:web`

Summary: `did:web` is a DID method that relies on the usage of web domains and DNS. A `did:web` is created by registering a web domain and hosting a web server under that domain, then taking this web domain as the DID and deploying the DID Document on the web server. The DID Document is stored by default under the `./well-known` path in the form of a JSON file. The DNS server practically acts as the VDR. Updating a `did:web` DID Document means replacing the current file with a new one, and deactivating a `did:web` DID Document means removing the file completely from the web server.

Benefits: The implementation is very easy and straightforward, as the methods involved are familiar for most developers. The DID Document is easily discoverable and universally resolvable as an interested party will just need to perform a REST API GET request to retrieve it. It can also be considered as free of costs as most organizations looking to implement DID methods would be highly likely to already own a web domain and host a web server.

Drawbacks: As it depends on the web infrastructure, the security and privacy of `did:web` relies entirely on web SSL/TLS certificates, which although generally provides great security, are not completely immune from malicious attacks. Although revoking a DID is possible by removing the DID Document from the web server, it may raise uncertainty on whether the DID has actually been removed, or whether the DID Document is only temporarily unresolvable because the web server is offline. There are no inherent mechanisms to retrieve a verifiable change history of previous versions of the DID Document, unless other technologies such as an online version control system are used, hence all trust must be placed on the current DID Document.

4.3.2. Selected DID Methods

4.3.2.1. `did:webs`

Summary: As the name suggests, `did:webs` is based off similar technology to that of `did:web`; `did:webs` also depends on web domains and DNS as a storage location of the DID Document JSON file. Two striking differences lie in how the identifier is built and DID Document is generated. Aside from web domains, `did:webs` also utilizes the usage of Key Event Receipt Infrastructure (KERI). The role of KERI is to record any changes that happens to the

cryptographic keys used in the DID Document. The DID Document is always regenerated based on the latest logs stored on the KERI file, instead of creating it manually. A did:webs identifier begins with the web domain, and is appended with a KERI autonomic identifier, which is connected to a KERI CESR file where the history of key changes are recorded.

Benefits: The adoption of KERI brings extra self-certification to the DID, as the KERI identifier used as a part of the DID is generated for the specific KERI CESR file used. With the existence of the KERI CESR file, a verifiable history of the cryptographic keys are recorded and accessible via the web, hence it's possible to audit the history of a did:webs DID Document. This also adds additional guarantees to the security of the DID and DID Document, as all updates can only be performed to the KERI CESR file, and the DID Document JSON file is only regenerated after. Deactivation of a did:webs DID is performed by rotating the cryptographic keys stored in the KERI CESR file to null and regenerating the DID Document again. This will prevent confusion during resolving a DID as there is a difference between a revoked DID and the web server being offline, whilst also retaining the DID Document history. Public discoverability and universal usage also remains the same as did:web, since did:webs still relies on the web domain as a VDR. Moreover, no additional costs are involved when using KERI.

Drawbacks: KERI is still considered a new technology, hence it still needs more adoption to be considered as battle-tested. Although the inclusion of the KERI identifier on the did:webs identifier increases the degree of self-certification, the other half of the DID identifier still utilizes the web domain, which is a random value of choice by the DID owner. Fundamentally, did:webs still relies on the web, where there are always possibilities of attacks by malicious actors worldwide.

4.3.2.2. did:tdw

Summary: did:tdw, which stands for Trust DID Web, is an improvement from the did:web specification. Aside from only depending on web and DNS technology like did:web, did:tdw takes it to another level by including additional mechanisms to ensure security and verifiability. Instead of using a normal JSON file containing the DID Document, a did:tdw DID Document is stored on a JSON Lines (JSONL) file, where each entry to the log file is a JSON that describes that state of the DID Document, including its version and create time, kept in chronological order from creation to deactivation. A self-certifying identifier (SCID) is generated from the initial DID Document, where this SCID is included to the did:tdw DID identifier (either as a subdomain or path), along with the web domain name where the JSONL file is stored. The SCID will remain unique forever, even when the JSONL file is transported to other web domains. For each JSONL entry, an entry hash is created. Moreover, a data integrity proof signature is created for each JSONL entry to guarantee the authenticity.

Benefits: did:tdw is easily discoverable and permissionless as it is hosted on the web, hence anyone can publicly access it. As the JSONL file chronologically stores all versions of the DID Document, its history is available to be audited and also verifiable. Moreover, did:tdw does not only support updates, but also allows for deactivation of DIDs so that no further operations can be performed. As the SCID is always globally unique, DID owners are free to move the JSONL file to other web domains, subsequently also changing the web domain part of the DID itself, but still able to guarantee ownership and historical verification by using the same SCID. did:tdw

is by principle free, however the DID owner must be in possession of a web domain that is hosted online.

Drawbacks: Currently, did:tdw is still maintained by a closed group: the Government of British Columbia team. It is open to contributions by the public, but the final decisions are still made by the BCGov team. Since a large part of the did:tdw DID still relies on using the web domain, it is not entirely self-certifiable, despite a part of the DID adopting the SCID. Additionally, entities could infer that different did:tdw DIDs are owned by the same owner due to using the same web domain. Web security must be ensured when using did:tdw, as its dependence on the web may be exploited by malicious actors. The specification states a way for did:tdw users to be backwards compatible with did:web, however substantial migration will be needed when moving entirely to did:tdw as the DID Documents are based on different file formats.

4.3.2.3. did:key

Summary: The did:key DID method makes use of the public key of the static cryptographic key pair to become the DID identifier, and also to generate the DID Document. The did:key identifier is entirely made of the public key itself. did:key is a generative DID method, hence there is no actual VDR in use; the DID Document is generated each time the DID is resolved. The DID Document generation process relies on certain rules that are set on the specification, but mainly centres around including the public key into the DID Document. Due to these situations, it is not possible to update or deactivate did:key DIDs.

Benefits: Given the way its implementation is specified, did:key is arguable the simplest and easiest DID method to be implemented. Its usage of the cryptographic public key as the method specific identifier definitely guarantees self-certification and identifier entropy. The specification also does not define specific restrictions on which cryptographic algorithms that can be used, it only recommends some examples of algorithms, hence it is flexible and dependent on each DID owner. There is zero cost of implementation as generating cryptographic keys are free of charge and does not require any special permissions. Most developers are also familiar with this process and hence the time and effort needed to learn how to utilize did:key is short. Complete privacy can also be achieved as only holders of the DID can resolve it, hiding it from the public.

Drawbacks: Although not specifying the cryptographic algorithm allowed to be used gives flexibility, it means that DID owners can potentially utilize algorithms that are not strong and secure enough. Moreover, solely relying on the cryptographic key pair means that there is no actual VDR and it is not possible to update or deactivate the DID, hence there is also no verifiable proof of history for the DID Document. Updating the DID Document would mean changing the cryptographic keys used in both the DID and the DID Document, thus creating a new DID entirely. In the situation where the cryptographic key pair has been compromised, there is no possibility to recover the DID, as its existence is permanent and cannot be revoked.

4.3.2.4. did:jwk

Summary: did:jwk essentially has the same characteristics as did:key, as they are both generative DID methods with the idea of expanding a generated cryptographic key pair as a DID Document. The main difference is that did:jwk specifies the JSON Web Key (JWK) format

must be used for the cryptographic keys, and disallows other cryptographic key formats. The did:jwk identifier uses an encoding of this JWK value.

Benefits: Generally the same benefits as did:key because they are based on similar technology. The implementation of this DID method is very easy, as DID creation only involves generating a JWK cryptographic key pair, and encoding the public key value to derive the DID identifier, which also guarantees a high degree of self-certification. Moreover, no costs are involved in the DID method operations as there are no fees associated to generate the cryptographic keys and the DID Document. Additionally, despite specifying a fixed cryptographic key format, no specific algorithms are specified as restrictions.

Drawbacks: Similar to its benefits, the drawbacks of did:jwk are generally the same as did:key. The freedom of cryptographic algorithm choice does not securely guarantee that the algorithm used to generate the cryptographic keys is sufficiently strong and secure. It is also impossible to modify or remove the DID, since there is no VDR involved in maintaining a did:jwk DID to verifiably keep track of its history. The only noticeable new drawback is that did:jwk is only maintained by one individual, unlike most other DID methods which are maintained by a community or organization.

4.3.2.5. did:peer

Summary: Unlike other DID methods in this evaluation, which are considered as anywise DIDs (DIDs that can be created and used universally), did:peer specifies a method for creating pairwise/n-wise DIDs. A pairwise DID is a DID that is intended to be known by exactly one other party, hence the name “pair”, while an n-wise DID is intended to be known by exactly “n” number of other parties. The did:peer specification defines five different methods for creating a DID and its corresponding DID Document. Both the DID and DID Document are then shared manually to other parties through a secure protocol. The parties receiving the DID and DID Document are considered as peers, and each peer must also in turn send their DID and DID Document to other peers. The VDR for storing did:peer DIDs and DID Documents is localized and defined individually by each peer, which could make use of technologies such as a database or local cache. Hence, resolving a DID does not involve any other parties.

Benefits: All five did:peer creation mechanisms ensure a high degree of self-certification as the DID is created by performing some processes on the inception key. Moreover, since the DIDs are not shared to the public and stored locally, in principle complete privacy can be maintained between parties of the peer relationship. Additionally, creating a did:peer is free and permissionless, as it does not depend on any specific VDR. The specification also advises against reusing the same DID in multiple peer relationships, as it is best to keep the did:peer DID unique for each relationship to prevent exploitations.

Drawbacks: Not storing the DIDs on a universal VDR could also limit its discoverability and resolvability, as new peers joining the relationship will need more attention and cannot entirely onboard themselves directly. Moreover, security mechanisms are not strongly enforced, as most of it is based on the trustworthiness built upon the relationship of the peers, and the guarantee that they will not misuse the DIDs and DID Documents that they receive. Additionally, by default did:peer DIDs are not updatable, as each peer only receives a copy of the DID Document once, for it to store in their local VDR.

4.3.2.6. did:pkh

Summary: The principle concept of did:pkh is comparable to that of did:key and did:jwk. did:pkh is also a purely generative DID method, meaning that only the DID is created upon initiation, the DID Document is regenerated each time the DID is resolved. The difference lies in the values that the DID relies on for its inception. Instead of using cryptographic keys, did:pkh utilizes existing blockchain addresses, which typically follow the CAIP10 format and are built using public key hashes (hash of cryptographic keys). The CAIP10 format includes a part that defines which blockchain the address belongs to, for example eip155:1 meaning Ethereum mainnet.

Benefits: Similar to the cryptography keys in did:key and did:jwk, the blockchain addresses used as the DID are also used for verification in the DID Document, hence there is high degree of self-certification. Privacy of the DIDs is also possible as they are not stored on a public VDR, they are only shared to those who the DID owner intends it to be received by.

Drawbacks: Being a generative DID means that it is not possible to update and delete did:pkh DIDs, even when it is compromised. As a negative impact to this, it is not possible to audit the evolution of did:pkh DID Documents. Using did:pkh for longer periods of time can have greater risks due to this. A difference from did:key and did:jwk is that did:pkh requires more costs, as the DID owner must be in pre-possession of a blockchain address, meaning a blockchain transaction must be performed, which could incur substantial fees. Additionally, the authenticity of did:pkh solely depends on the blockchain used; a vulnerability in the blockchain means the did:pkh DID may also be impacted.

4.3.2.7. did:ion

Summary: did:ion is a DID method that fundamentally relies on the Bitcoin blockchain. However, it does not directly make operations to the Bitcoin network. Instead, did:ion utilizes a public permissionless network, called the Identity Overlay Network (ION), which is built using the Sidetree protocol and acts as a data link layer atop of the Bitcoin network. ION acts as an intermediary between the DID and the Bitcoin network, where it allows the capability of batching up to 10000 DID operations into a single Bitcoin transaction, or called as an ION transaction in this case. The batching process is performed automatically by the ION nodes. DID owners can choose to maintain their own ION node or use existing ones in the public.

Benefits: Batching of DID operations into a single ION transaction drastically reduces the usage costs, as in theory it cuts the cost of a DID operation to 1/10000 of a Bitcoin transaction. Similar to other blockchain-backed DID methods, it is possible to audit the change history of a DID Document stored on did:ion as each operation performed on the DID Document are written to the public ledger. Privacy and self-certification are also maintained as the did:ion identifier is composed from the cryptographic keys used in the DID Document, instead of using any Blockchain address.

Drawbacks: When a DID owner chooses to deploy and maintain an ION node, they are responsible of paying the transaction fees associated with the batch writes to the Bitcoin network.

4.3.2.8. did:ebssi

Summary: The European Blockchain Services Infrastructure (EBSI) is a blockchain network that is maintained and governed by the European Commission and the European Blockchain Partnership. It was created with the intention to be used for digital identity through leveraging decentralized identifiers and verifiable credentials. EBSI defines two separate DID specifications for did:ebssi; v1 for legal entities (registration of DID Documents on the EBSI ledger), and v2 for natural persons (relying only on cryptographic key pairs). The method specific identifier of did:ebssi v1 is composed of an encoding of the cryptographically-generated EBSI subject identifier, whereas it is the public key of the generated cryptographic key pair also used in the DID Document for did:ebssi v2.

Benefits: The did:ebssi specification for legal entities behaves similarly to other blockchain-based DIDs, in which the audit history of DID Documents is securely stored on the network, deactivation of DIDs is possible, and is universally resolvable through the public EBSI network. For both types of specifications, did:ebssi already sets a strong base of minimum security level, and also naturally complies to many European regulations as it is itself maintained by the EC. The did:ebssi specification for natural persons is easy to implement as it only involves creating cryptographic keys, which is mostly familiar for many developers.

Drawbacks: Despite being created and endorsed directly by the EC, it still does not seem mature enough for wider adoption, as the specification feels vague and leaves many questions unanswered. Community involvement to the specification also remains limited, as the did:ebssi specification repository is currently private, let alone allowing for any opportunities for public contributions. Moreover, EBSI currently curates a list of approved wallets that are usable and already integrated with did:ebssi. In the v1 specification for legal entities, writing a did:ebssi DID Document to the EBSI network requires additional permissions and preliminary registration to the EBSI Authorisation service. On the other hand, the v2 specification for natural persons acts very much like the did:key specification, meaning it does not support updates and deactivation, hence it will be difficult to recover from situations where the cryptographic keys are compromised.

4.3.2.9. did:sov

Summary: did:sov relies on storing the DIDs and DID Documents on the Sovrin network, governed by the Sovrin Foundation. The Sovrin network is a public permissioned blockchain that is designed specifically for the purpose of supporting self-sovereign identity, built atop of Hyperledger Indy. Users write information related to their DID and DID Document to the Sovrin network. There is a cost for each write transaction to the Sovrin network, and since it is a permissioned network there are also subscription fees related to registering as an official transaction endorser to Sovrin. Most of the time, users can delegate their writes to the Sovrin transaction endorsers. The did:sov identifier is in the format of a UUID which is generatable in two possible ways: 1) using standard UUID algorithms, 2) taking the first 16 bytes of the DID's inception keys.

Benefits: did:sov supports common features that are expected from ledger-based DID methods, for example auditability of DID Document evolution via the Sovrin ledger, and revocation of a DID by deactivation to prevent any future changes to the DID Document. Both methods of generating the did:sov identifier ensure randomness, however using the inception keys leads to

higher self-certification. The high degree of randomness also protects user privacy and prevents the users from being traced back from the DID.

Drawbacks: The fees associated with operations on the Sovrin network are not cheap, and can accumulate to much greater costs in the long run. Moreover, since Sovrin is a permissioned network, registration needs to be carried out if a DID owner wants to become a transaction endorser to actually perform writes on the Sovrin network; the onboarding subscription fee for this is costly. Additionally, the method to generate the did:sov identifier by using UUID makes it unrelated to its inception keys, hence no self-certification.

4.3.2.10. did:ethr

Summary: did:ethr is a ledger-based DID method that relies on storing its DID Document on an off-chain registry, following the ERC-1056 smart contract standard, that is related to the public Ethereum blockchain network. A difference from the other evaluated ledger-based DIDs is that did:ethr not only supports the usage of cryptographic public keys as its identifier, but also Ethereum addresses, for example in key pair account or smart contract form. All DID operations (create, read, update, delete) are performed using ERC-1056 smart contract standards and onto an off-chain Ethereum DID registry of choice instead of directly writing operations to the Ethereum blockchain.

Benefits: As a ledger-based DID method, did:ethr fulfils core characteristics such as having a verifiable change history of the DID Document stored on the blockchain network, and also revoking validity of a DID and DID Document. Ethereum has been regarded as a robust network for quite some time now, hence it has a strong secure record of accomplishment of public usage. The Ethereum network provides universal discoverability for did:ethr DIDs due to its public and permissionless nature. Moreover, did:ethr reduces usage costs in comparison to other ledger-based DID methods as it depends initially on an off-chain registry, instead of completely relying directly on the Ethereum blockchain for every operation.

Drawbacks: Using existing Ethereum addresses as the method specific identifier decreases the self-certification of the DID from the inception keys in its DID Document. Although minimized, each update and delete operations are still written to the Ethereum blockchain and incurs costs, which could in the long run add up to substantial fees.

4.3.2.11. did:cheqd

Summary: The did:cheqd DID method is incepted and maintained by the Cheqd company. Cheqd is a technology company that provides a variety of services related to trusted data markets for other companies and organisations, such as privacy-preserving payments infrastructure and self-sovereign identity. The core of Cheqd's services is the public permissionless Cheqd blockchain network, built based on the Cosmos blockchain framework and created with the main purpose for self-sovereign identity. There are two ways to generate the did:cheqd identifier: 1) the preferred method of using any UUID generation algorithm, 2) the Hyperledger Indy-compatible method of using the encoding of the DID's inception keys.

Benefits: Since did:cheqd depends on the Cheqd blockchain network as the VDR, it possesses the common characteristics of other ledger-based DID methods, such as complete deactivation

of a DID to prevent future operations, and also a connected and verifiable chronological overview of a DID Document’s evolution. As it relies on the public permissionless Cheqd network, did:cheqd allows anyone to universally utilize the decentralization benefits that it provides. Both choices of did:cheqd identifier creation methods result in high entropy, thus making it difficult to infer the owners of different DIDs. did:cheqd was created with a good sense of interoperability in mind allowing for the support of other Hyperledger Indy-based DID methods, although not entirely backwards compatible.

Drawbacks: Although Cheqd has opened communication channels through social media for community suggestions, currently only the Cheqd team have made contributions to the did:cheqd specification on GitHub. Like other blockchain-based DID methods, costs are involved when performing DID operations on the Cheqd network, although arguably more reasonable than the networks utilized by other DID methods evaluated in this study. Using the preferred method of generating the did:cheqd identifier using UUID reduces the degree of self-certification, as the identifier is completely random and unrelated to the keys of the DID Document.

4.4. Evaluation Discussions

The results of the cumulative scores from the evaluation of the selected DID methods using the formulated evaluation rubric is ranked and summarized in Table 8 below. The individual score breakdowns of each evaluated DID method can be observed in Appendix C.

Rank	DID Method	Score
0 (benchmark)	did:web	59
1	did:ion	70
2, 3	did:webs	67
	did:tdw	
4, 5	did:ethr	63
	did:peer	
6, 7	did:cheqd	62
	did:sov	
8	did:pkh	61
9, 10	did:key	59
	did:jwk	
11	did:ebsi	53 (natural person) 52 (legal entities)

Table 8: Summary of DID Method Rubric Scoring Results

Based on the results shown in Table 8, the most obvious key takeaway is that most of the ledger-based DID methods are ranked in the upper half, whilst most of the non-ledger based DID methods sit at the bottom half of the spectrum. Out of the 11 evaluated DID methods, all but one resulted in a final score that is equal to or higher than did:web, albeit the range of scores among the evaluated DID methods not being much dispersed considering the possible scores on the scale of 21 to 84.

When solely determining based on the resulting scores, did:ion seems to be the strongest candidate for usage in a data space environment, as it has the highest resulting score. However, upon closer inspection of the qualitative evaluation results, it is deduced that all of the ledger-

based DID methods, which includes did:ion, did:sov, did:ethr, did:cheqd, and did:ebis, have similar characteristics with only slight differences in certain properties. All ledger-based DID methods are based on some type of decentralized network that provides functionality that are most needed to ensure the trustworthiness that the adoption of DIDs in data spaces aim to achieve. Examples of these are strong and robust choices of security mechanisms to provide a secure network catering to the general public, a connected verifiable and traceable proof of DID Document evolution history ensuring the validity of the DID, and the possibility to perform complete deactivation of DIDs to render them invalid from any future usages. In spite of that, these solid benefits come with significant drawbacks. A major disadvantage is that none of the ledger-based DID methods come free of charge. For all ledger-based DID methods evaluated in this study, payment of a certain amount of fee is required for at least one type of DID operation, potentially adding up to a greater sum in the long term. Another substantial drawback is that the knowledge required to work with distributed ledger technologies is still a growing skill that is not so common, therefore a steeper learning curve is expected before someone is able to begin being involved.

The unique properties offered by each of these DID methods is what results in the variance of scores among the ledger-based DID methods. For did:ion, its key beneficial characteristic is that thousands of DID operations can be batched into a single blockchain transaction, which drastically saves costs in scenarios where many DID operations must be executed. For did:sov, although it relies on the Sovrin network which was especially constructed for self-sovereign identity, its permissioned nature prevails as a major drawback, as it leads to additional startup costs and requires onboarding procedures before being allowed to perform writes to the actual network, on top of the existing costs to make the actual writes for the DID operations. For did:ethr, its uniqueness of allowing the usage of Ethereum addresses as the identifier greatly simplifies the implementation process, however also reduces the self-certification of the identifier towards the inception keys in its DID Document. For did:cheqd, the Cheqd network itself is commercialized and maintained by a for-profit technology company providing services to other parties. This means that future decisions of the Cheqd company should be closely monitored when opting for this DID method. The outlier among these ledger-based DID methods is did:ebis. The did:ebis DID method scores significantly lower than all other evaluated DID methods, independent of the underlying technology. This is largely accredited to the immaturity of its specification in these early stages of its establishment. However, as it is allowed for room to grow, did:ebis could definitely develop into one of the strongest candidates for a favourite DID method adoption within the EU in the future, considering the fact that did:ebis is governed directly by the European Commission.

The results of this evaluation process not only expose the similarities and differences among ledger-based DID methods, but also further distinguished the distribution of non-ledger-based DID methods into separate categories (as summarized in Table 9). One of the categories that can be extracted from this evaluation process is key-based DID methods. Key-based DID methods, which include did:key, did:jwk, and did:pkh, are essentially inherently generative DID methods. This means that DID Documents built following the specifications of this type of DID method do not rely on any concrete VDR because the DID Documents are regenerated using the public cryptographic key value used as the DID identifier each time it is resolved. The result of the evaluation shows that the main difference among these three key-based DID methods is the key structure and format used. For did:key, there are no restrictions on which cryptographic keys and formats are allowed. For did:jwk, the format of the cryptographic keys used must be

in the form of JSON Web Key. For did:pkh, public key hashes retrieved from CAIP-10 blockchain addresses are used. Although key-based DID methods are intuitively the easiest to implement as nothing else needs to be prepared aside from establishing the DID identifier itself, the downside is that they do not provide essential characteristics of DIDs, such as robust security, DID revocation, and a trustable chain of historical changes. In the event where malicious actors gain possession of the corresponding private cryptographic keys, it is not possible to recover from such attack, and the DID will forever be compromised. Due to this, it is best to adopt key-based DID methods for short term, temporary usage in environment where all parties are within close contact of each other.

Based on its name, the did:peer DID method gives the impression that it may be the most suitable for usage in a data space domain. DIDs, VCs and VPs in the context of data spaces are only exchanged between parties of an established relationship within a data space, which suits the concept of n-wise DIDs that is supported by did:peer. However, upon further analysis through the evaluation rubric, it is uncovered that did:peer may not be the correct solution for this use case. A substantial reason is due to the fact that there is no universally available VDR involved for resolving did:peer DIDs, but instead it relies on each peer in the relationship to maintain their own local VDR to store the DIDs and DID Documents of every other peer. For data space identity providers and participants, this practice is certainly undesirable as this means that each participant must keep a collection of DIDs and DID Documents of other participants in the same data space. A major inconvenience arises when the number of participants of the data space dynamically change over time. On the contrary, not having the local VDR means that it is not possible to verify the DIDs retrieved from VCs and VPs, hence data space participants are not able to verify and authenticate their transactions. Moreover, did:peer encourages the creation of new DIDs for every new peer relationship. Although it is understandable from a security perspective, it may end up being troublesome for data space participants having to manage many several distinct DIDs when they participate in different data spaces.

A point of intrigue from the results of this evaluation process lies in the category of web-based DID methods. This category contains DID methods which are fundamentally constructed from the idea of web technology. The DID method currently implemented in the TSG SSI wallet, did:web, is the most basic version of this type of DID methods. Other DID methods of this type included in the evaluations are did:webs and did:tdw, of which can be categorized further into improved web-based DID methods. Both of these DID methods share similar characteristics, and resultingly have equal scores. They are interesting alternatives to the stronger group of ledger-based DID methods, as their scores rank the second highest overall, only trailing behind did:ion. Although not dependant on any blockchain networks, both did:webs (through KERI) and did:tdw (through JSONL), provide the features typically expected from decentralized ledger-based DIDs, such as a provable and permanent evolution history of the cryptographic keys used in the DID Document, as well as the possibility to revoke a DID to prevent it from being usable anymore in the future. Due to the supporting KERI and JSONL files still being plainly stored on public web servers, it is evident that the security provided by both of these DID methods are not as powerful as those accommodated by their blockchain-based counterparts. However, considering the other benefits such as zero additional costs and ease of implementation, the level of robustness in their security measures are a fair trade-off. Though their implementation is not as simple as that of key-based DID methods, the implementations

of both did:webs and did:tdw are still easy to perform as it mainly relies on common web technology with additional cryptographic measures.

Characteristic Category	DID Methods
Ledger-based	did:ion did:sov did:ethr did:cheqd did:ebsi
Key-based	did:key did:jwk did:pkh
Peer-based	did:peer
Improved Web-based	did:webs did:tdw

Table 9: Categorization of Evaluated DID Methods based on Similar Characteristics

4.5. Evaluation Conclusions

After a comprehensive evaluation process among the selected DID methods facilitated by the rubric, and conclusions drawn from the discussions above, it can be determined that the improved web-based DID methods of did:webs and did:tdw are potentially the most suitable DID methods for candidate implementations in the context of this study. The data spaces domain requires the adoption of identity management that is decentralized, reliable and secure, of which this category of DID methods fulfil. It is believed that these requirements can be achieved with a combination of the support for web technology and additional cryptographic mechanisms that both did:webs and did:tdw provides. A balance is struck between the factors of the decentralization, robustness, and history-persistence of ledger-based DID methods, and the implementation-ease and costs-free of other non-ledger based DID methods.

However, to create a clearer and more focused stance for a DID method suggestion by removing the ambiguity of recommending several similar DID methods from a category, and considering the constraints situated in terms of time and effort, only one DID method –either did:webs or did:tdw– will become the final recommendation as the outcome of this study. In order to make an informed decision on this choice, the following discussions present a comparative analysis between the two similar DID methods.

A striking key difference between did:webs and did:tdw is the structure of the supplementary mechanisms that is included in addition to the existing web technology used by did:web. As mentioned in previous sections, did:webs utilizes Key Event Receipt Infrastructure (KERI), whereas did:tdw makes use of JSON lines (JSONL) files. KERI is a nascent technological concept which enables recording each changes of controlled cryptographic keys in a sequential and linked manner according to a strict predefined standard format, whose adoption is still growing within the niche focus area of decentralized trust. On the other hand, JSONL is an extension of the popular and renowned JSON data format, which is widely adopted in most organizations across different markets, and is essentially a file containing multiples lines of JSON format data. This aspect infers that the underlying technology used by did:tdw is simpler and easier to implement rather than did:webs, as the technology that did:tdw uses is more

commonly familiar for many technology professionals, thus leading to less steep of a learning curve and requires less time for getting familiar with.

Additionally, another point to consider is the varying degrees of flexibility offered by the two DID methods, in terms of user control over the DID Document contents. In `did:webs`, changes to the DID Document can only be reflected by first modifying the KERI file as its source of truth, and then regenerating the DID Document. Creating updates to the DID Document is restricted as direct modifications to it is not a possibility. In contrast to that, the lines of JSON in the JSONL file utilized by `did:tdw` contains representations of the changes on the DID Document itself in a standardized manner. Hence, users have complete freedom of how they want to modify the DID Document, without having any limitations or boundaries. This shows that `did:tdw` allows direct control of the DID Document itself instead of needing to undergo a generative-system that depends on other factors such as in `did:webs`.

Despite the greater freedom of DID Document modification, `did:tdw` does not provide a possibility to directly retrieve the DID Document from the hosting web server like that of `did:web`. Additional processing must be performed on the JSONL file to resolve the DID identifier into the actual DID Document. On the contrary, the DID Document is served accessibly and plainly as a normal JSON file on the web server when using `did:webs`. The retrieval process of a `did:webs` DID Document is thus simple and straightforward similar to using a plain `did:web`, albeit still providing stronger trust guarantees. Although the `did:tdw` DID resolution process involves additional mechanisms compared to `did:webs`, the needed processing effort and time can be deemed negligible since JSONL files are not expected to be extremely lengthy due to the expected infrequent changes on the contents of DID Documents. The complexity of `did:tdw` DID resolution is also observably much simpler than that of `did:webs` DID creation, given the familiarity of the technology.

In light of these comparisons between `did:tdw` and `did:web`, the outcome of the considerations from this study provides a recommendation to implement `did:tdw` on the TSG SSI wallet due to its simplicity, familiarity and transparency, which outweigh its drawbacks. Regardless of this decision, the implementation of `did:webs` is not completely ruled out for future studies.

5. Verification & Validation

In order to properly confirm that the results of the evaluation process were accurate and can truly be provided as a recommendation for the Eclipse Dataspace Decentralized Claims Protocol, verification and validation processes must be performed. According to IEEE (2017), verification refers to determining whether the results of an activity conform to the initial requirements, and validation refers to determining whether the results of an activity fulfil the actual needs and intended usage. The following subchapters outline the preparations, processes and results of the verification and validation processes.

5.1. Implementation Design

A proof-of-concept implementation will be conducted as a prelude to the verification and validation processes, of which functionalities to support the new selected DID method (did:tdw) is introduced to the existing TSG SSI wallet system alongside the current did:web DID method functionalities. This subchapter describes the current software design of the TSG SSI wallet, the challenges that exist to implement the new incoming DID method, and the resulting modified software design.

5.1.1. Current System & Challenges

The TSG SSI wallet was developed using the NestJS¹⁸ framework in the TypeScript¹⁹ programming language. In this framework, three base components are specified for building web backend applications; 1) Controllers: components in charge of handling requests and responses from and to the client; 2) Providers: any component, such as services, repositories, factories, helpers, etc., which can be injected as a dependency to other components; 3) Modules: components that provide the metadata to organize the structure of the application. The class diagram in Figure 13 below represents the current state of the software design of the TSG SSI wallet, where the classes are labelled with <<controller>>, <<provider>>, and <<module>> to show the type of each component. Irrelevant classes that are unrelated to DID functionalities have been excluded to provide more clarity and focus specifically for this study.

By referring to the class diagram, it can be deduced that the main classes supplying functionalities related to DIDs are the DidService and DidResolverService classes. Several other classes rely on these classes as dependencies to make use of their methods. The DidService class provides methods specifically for did:web DID identifier and DID Document creation, in-memory storage of the DID identifier itself, and also saving and retrieving the DID Document from the database. The DidResolverService class contains a single method which performs resolution from a did:web DID identifier to the corresponding DID Document following the did:web specifications. The DidService class has a dependency to the DidDocumentsRepository, which is an interface to the database table used for performing create, read, update and delete operations on DID Documents to and from the database. The /.well-known path, where the DID Document JSON file is located, is served by the DIDController class. On the other hand, the DIDManagementController class exposes an endpoint to retrieve the DID Document directly from the database under a generic path, irrespective to the supported DID method.

¹⁸ <https://nestjs.com/>

¹⁹ <https://www.typescriptlang.org/>

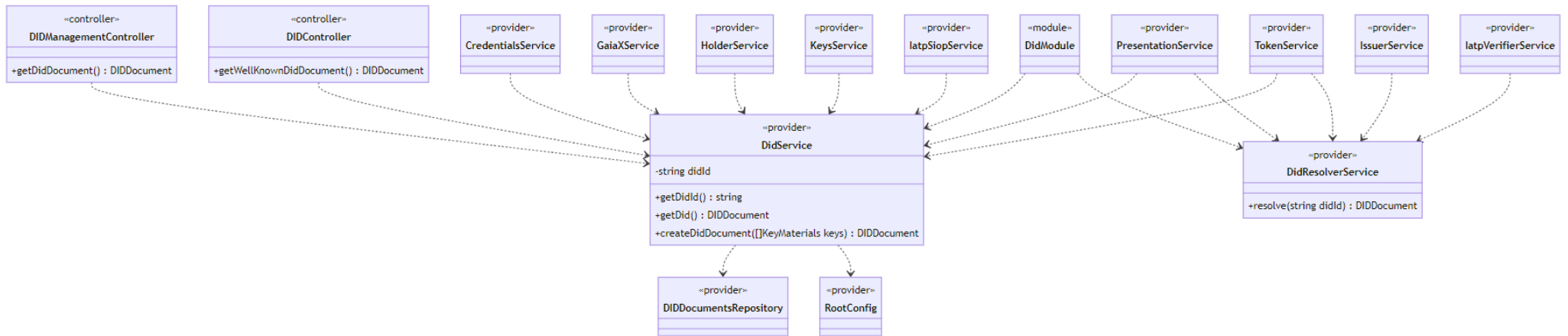


Figure 13: Class Diagram of Current TSG SSI wallet

Given the above class diagram, Figures 14 and 15 below represents the flow of interactions between classes for DID creation and DID resolution respectively, in the form of a sequence diagram.

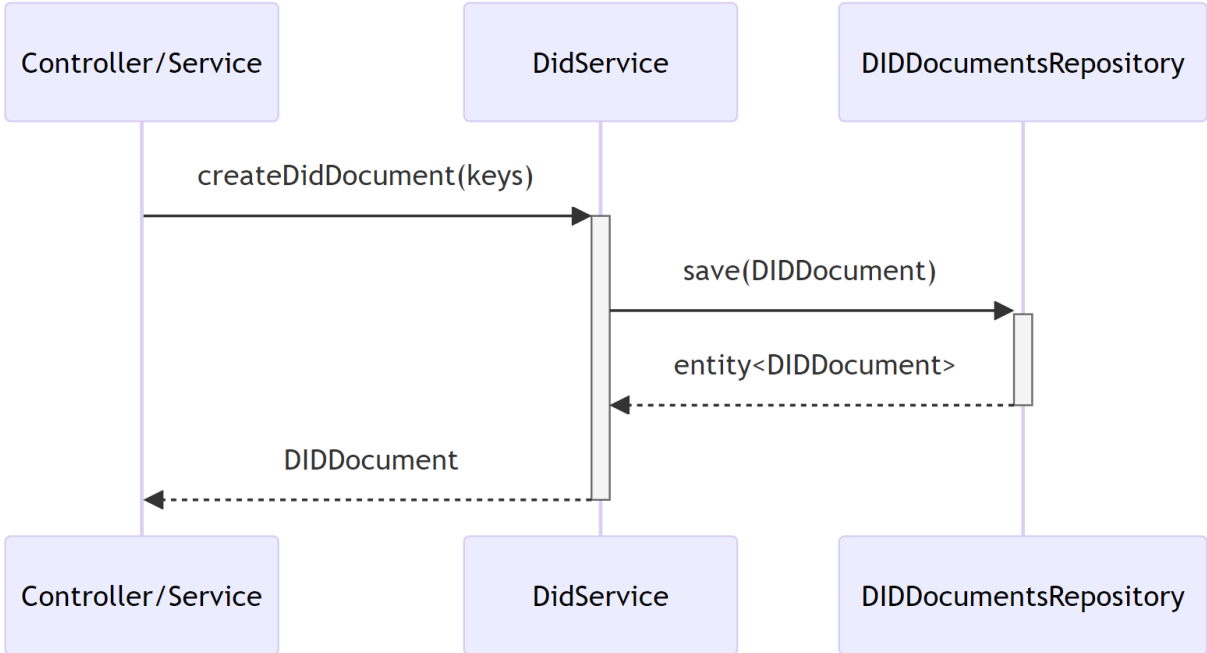


Figure 14: Sequence Diagram of Current DID Creation

The sequence diagram of DID Creation, as shown on Figure 14, illustrates the step-by-step process of how the TSG SSI wallet handles DID creation:

1. A Controller or Service class calls the createDidDocument() method of the DidService class and passes a list of verification keys as the keys parameter.
2. The DidService class then creates the did:web DID Document with the correct DID identifier based on the current web domain and passed verification keys.
3. The DidService class then stores the created DID Document to the database via the DidDocumentsRepository interface.
4. The DidService class then responds to the calling Controller or Service class by returning the created DID Document.

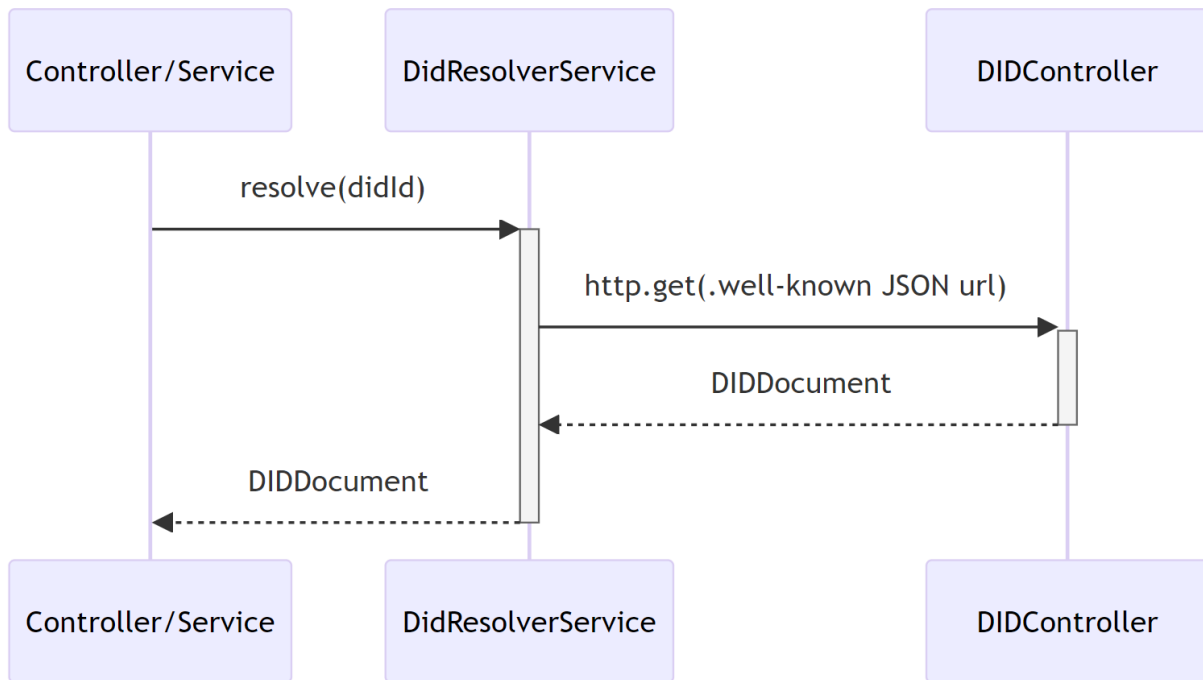


Figure 15: Sequence Diagram of Current DID Resolution

The sequence diagram of DID Resolution in Figure 15 describes the step-by-step process of how the TSG SSI wallet handles DID resolution:

1. A Controller or Service class calls the resolve() method from the DidResolverService class and passes the DID identifier value as a didId string.
2. The DidResolverService class then transforms the didId value to the corresponding URL value.
3. The DidResolverService class then appends the /.well-known/did.json path to the URL and performs an HTTP GET API request to the complete URL.
4. The HTTP GET API request is handled by the DidController class, which performs processing to retrieve the DID Document from the database (not illustrated in the sequence diagram), and returns it as a response to the DidResolverService class.
5. The DidResolverService class then responds to the calling Controller or Service class by returning the retrieved DID Document.

With the current software structure, the TSG SSI wallet performs excellently for the existing use case of supporting a single did:web DID method. Creating an implementation to support the did:tdw (as the DID method to be implemented based on previous evaluation results) is simple and straightforward, because there exists a TypeScript library²⁰ developed by the maintainers of the did:tdw specification. However, the real problem lies in the fact that the software design of the TSG SSI wallet was not created with the consideration of adopting another DID method; the existing software design is built for only one fixed DID method. An easy and sufficient enough solution is merely to just include the bare minimum of new logic, classes and files to support the did:tdw DID method. Despite providing a working solution, this approach does not abide by proper software engineering best practices, and may bring about future obstacles when there are requirements to dynamically support implementations and solutions for other DID methods. To tackle this challenge, a suitable software design pattern must be introduced to the TSG SSI wallet's software design.

To assess the possible design pattern candidates for consideration, an overview of popular design patterns was retrieved from Shvets (2021). Initially, creational design patterns, namely Factory Method and Abstract Factory, were thought to be promising candidates, because they allowed the creation of objects from different subclasses through the interface of a common superclass. However, upon further study, it was discovered that the aim of creational design patterns is to accommodate problems where there is a need to dynamically support the creation of a suit of related objects based on a certain strategy. When looking back at the problem at hand, the resulting objects produced by every DID method is the same, which are the DID identifier and the DID Document. There is no need to create different subclasses to represent the DID Documents of different DID methods as they are all structurally uniform abiding to a common data model, as defined by Sporny et al. (2022b) as maintainers of the W3C recommendation document. Hence, an implementation of a creational design pattern would not be an appropriate solution to this problem.

Instead of creational design patterns, the focus was shifted towards examining behavioural design patterns. This was decided with the reasoning that all DID methods inherently aspire to provide the exact same structural entities (DID identifier and DID Document) only with different associated content values, meaning that the only difference lies in the underlying algorithms, or behaviour, used to create them and produce the different content. According to Shvets (2021), the Strategy pattern is a behavioural design pattern that allows the definition of a group of similar algorithms that are each encapsulated into their own classes, of which the objects instantiated from these classes are interchangeable for each other through a common interface. The Strategy pattern essentially intends to solve problems where a program attempts to achieve something specific in several different ways, while all resulting to the same outcome. Given this account, it is evident that the characteristics of the Strategy pattern and what problem it ought to settle clearly matches greatly to the problem at hand with the current TSG SSI wallet's software design. The following subchapter described how the Strategy pattern is introduced to the TSG SSI wallet.

²⁰ <https://github.com/bcgov/trustdidweb-ts>

5.1.2. Modified System

Figure 16 below illustrates the Strategy pattern being implemented on the TSG SSI wallet's software design. Basing off from Shvets' (2021) explanation of the Strategy design pattern, the current methods related to support for did:web DID functionalities are extracted into its own classes `DidWebStrategy` and `DidWebResolverStrategy` from the original classes of `DidService` and `DidResolverService` (also called context classes). The new classes implement the `DidStrategy` and `DidResolverStrategy` interfaces respectively, which describes method definitions originally implemented in the context classes for DID Document creation and resolution. Additionally, the `DidStrategy` interface also specifies a new method for dynamic DID identifier creation based on the selected strategy. Similarly, new strategy classes called `DidTdwStrategy` and `DidTdwResolverStrategy` are also created to encapsulate the new algorithms for processing did:tdw DIDs, of which also naturally implement the required interfaces. Additionally, the `DidTdwStrategy` class has a dependency on `DidLogsRepository`, which is an interface to the database table that is used to manage DID Logs.

In the context classes, instead of the actual implementation of DID functionalities, references to objects of the `DidStrategy` and `DidResolverStrategy` interface types are contained in the respective context classes, and the original methods only delegate the processing of the method calls by calling the strategy objects instead of performing it on their own. For the `DidService` class, the choice of which DID strategy to utilize depends on a `didMethod` string value in the `RootConfig` class, of which is a dependency of the `DidService` class. Hence, only one instance of DID strategy will be instantiated as the DID strategy is already selected from the startup of the system. For the `DidResolverService` class, the option of DID resolver strategy to utilize depends on the prefix of the `didId` value that is passed as a parameter. This means that the decision of which DID resolver strategy implementation to use is dynamically made at runtime, hence each DID resolver strategy is created and stored in a map data structure.

Regarding the DID-related controllers, upon closer inspection, it is noticed that the endpoint exposed by the `DIDController` class, which is the `/.well-known` path for retrieval of the JSON file containing the DID Document, is only specific for the use case of did:web and unused by did:tdw DIDs. Hence, the controller is renamed to `DIDWebController`, and its dependency on the `DidService` class is replaced directly with the `DidWebStrategy` class. For did:tdw specifically, a new `DIDTdwController` class having a dependency to the `DidTdwStrategy` class is created to expose an endpoint for retrieving the JSONL file containing did:tdw DID Logs. The `DIDManagementController` class remains as it is because it can be used generically for any DID method.

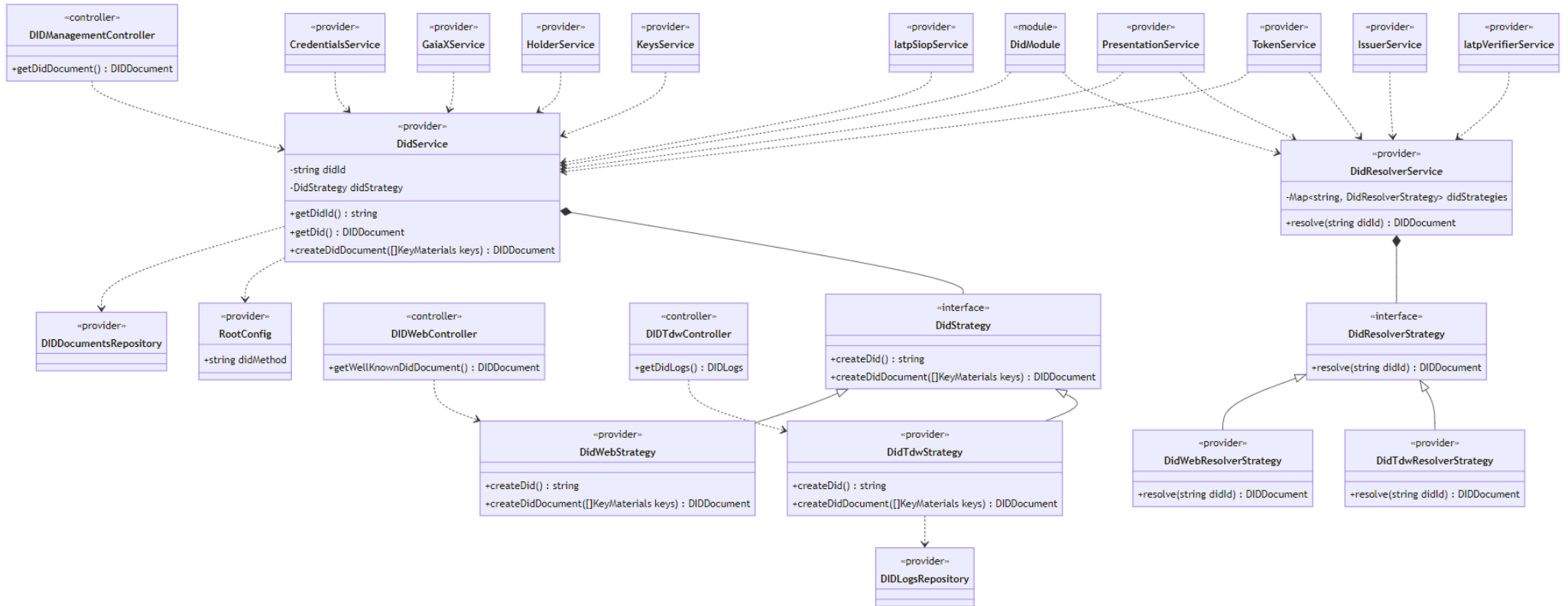


Figure 16: Class Diagram of Modified TSG SSI wallet with Strategy pattern

Alongside the modifications to the software design depicted by the modified class diagram, the flow of method calls sequence for DID creation and DID resolution also changes, as shown on the sequence diagrams in Figures 17 and 18.

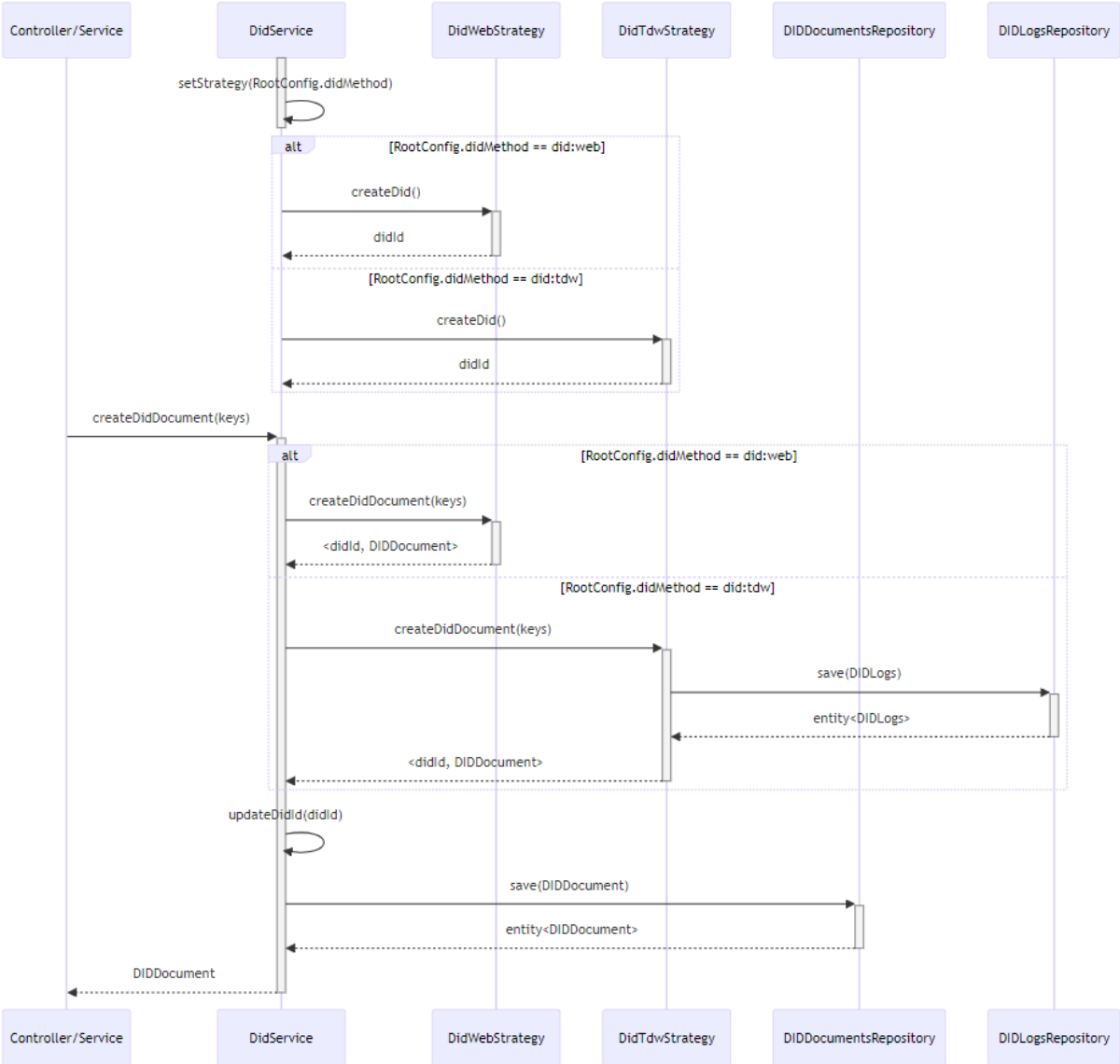


Figure 17: Sequence Diagram of Modified DID Creation

The updated sequence diagram of DID Creation in Figure 17 describes the step-by-step process of how the TSG SSI wallet handles DID creation. Below is the sequence flow for did:web.

1. At application startup, the DidService class reads the didMethod value from the RootConfig class. The DidService class then creates and sets the DID strategy object to be used based on this value (either as DidWebStrategy or DidTdwStrategy classes).
2. After instantiating the DID strategy object, the DidService class makes a call to the createDid() method of the DID strategy object, which returns the DID identifier value according to the DID method specifications.
3. A Controller or Service class calls the createDidDocument() method of the DidService class and passes a list of verification keys as the keys parameter.
4. The DidService class will then forward the call by calling the createDidDocument() method of the DID strategy object, also passing in the list of verification keys.

5. The DID strategy object will create the DID Document according to the corresponding DID method specification.
 - a. For the DidWebStrategy class, the DID Document will be created like in the previous flow.
 - b. For the DidTdwStrategy class, the DID Document is firstly created, and then the DID Logs. The DID Logs is then stored to the database through the DIDLogsRepository interface.
6. The DID strategy object then returns a response to the DidService class containing an updated didId value and the created DID Document. A new didId value is returned because in most cases, the value of the didId depends on the verification keys used, which is only received after the createDidDocument() method is called.
7. The DidService class then updates the didId value that it stores to the new value received from the DID strategy object.
8. Afterwards, the DidService class then stores the created DID Document to the database via the DidDocumentsRepository interface.
9. The DidService class then responds to the calling Controller or Service class by returning the created DID Document.

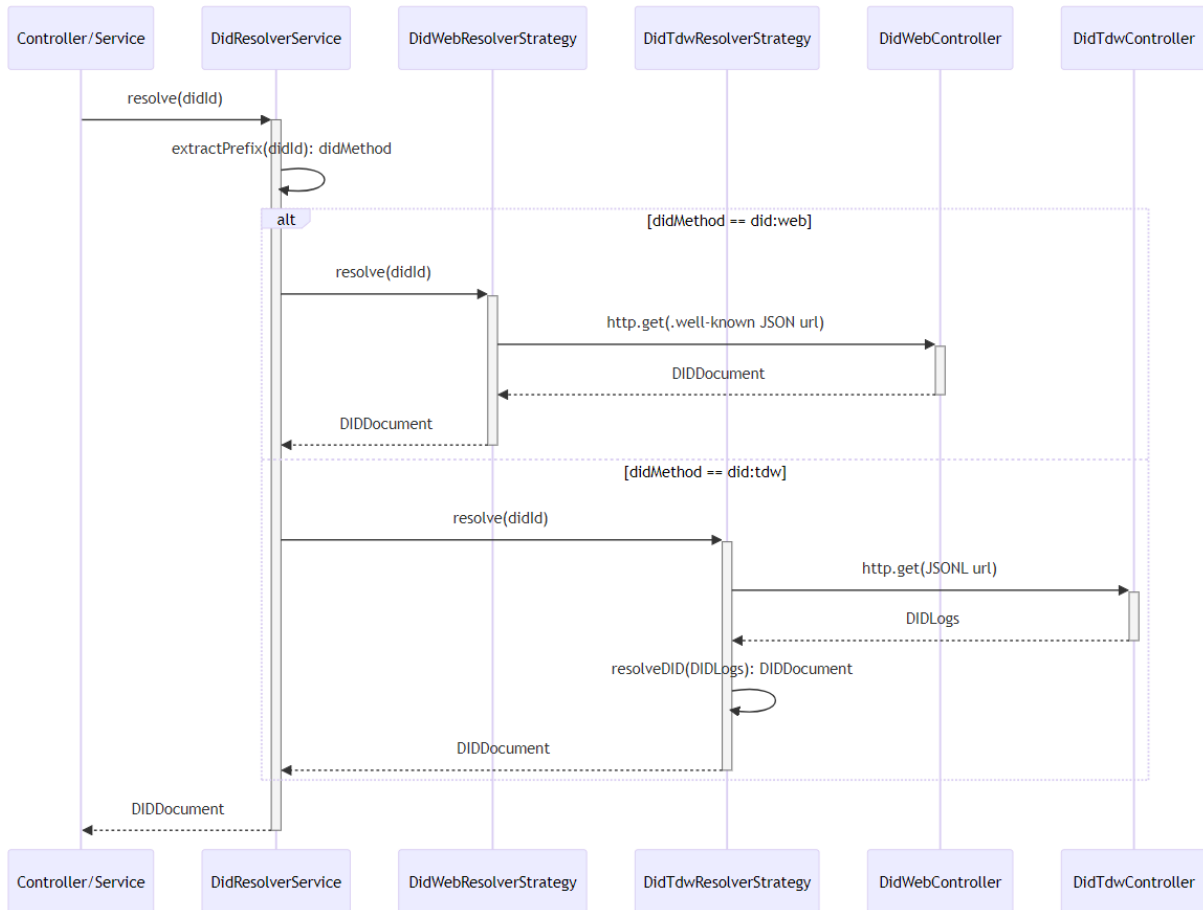


Figure 18: Sequence Diagram of Modified DID Resolution

The updated sequence diagram of DID Resolution, as shown on Figure 18, illustrates the step-by-step process of how the TSG SSI wallet handles DID resolution:

1. A Controller or Service class calls the resolve() method from the DidResolverService class and passes the DID identifier value as a didId string.
2. The DidResolverService class extracts the prefix from the passed didId value to determine the DID method of the DID.
3. Depending on the DID method, the DidResolverService class will perform a resolve() method call to the DID resolver strategy object, also passing the didId value.
 - a. For the DidWebResolverStrategy class,
 - i. The didId value will be transformed to the corresponding URL value, and appended with the /.well-known/did.json path.
 - ii. An HTTP GET API request is performed on the complete URL.
 - iii. This HTTP GET API request is handled by the DidWebController class, which performs processing to retrieve the DID Document from the database (not illustrated in the sequence diagram), and returns it as a response to the DidWebResolverStrategy class.
 - b. For the DidTdwResolverStrategy class,
 - i. The didId value will be transformed to the corresponding URL value, and appended with the /did.jsonl path.
 - ii. An HTTP GET API request is performed on the complete URL.

- iii. This HTTP GET API request is handled by the `DidTdwController` class, which performs processing to retrieve the DID Logs from the database (not illustrated in the sequence diagram), and returns it as a response to the `DidTdwResolverStrategy` class.
 - iv. The `DidTdwResolverStrategy` class then performs processing to resolve the DID Document from the retrieved DID Logs.
 4. The DID strategy objects will then return the retrieved DID Document to the `DidResolverService` class.
 5. The `DidResolverService` class finally responds to the calling Controller or Service class by returning the resolved DID Document.

5.2. Verification

Coming back to the initial stages of the criteria selection process for the DID method evaluation rubric, this study incorporated the DSSC core design choices and extracted basic data space identity management principles from them. As a scheme for verification, the core design choices and extracted principles are revisited to ascertain the newly selected DID method and its implementation design to the existing system. Two categories of verification scenarios will be covered in this study: Interoperability (which encapsulates the Interoperability principle) and Security (which covers the Trustworthiness, Authenticity and Verifiability principles). The scenario categories are discussed further in the following subchapters, along with their results and discussions about their evaluation.

5.2.1. Interoperability

5.2.1.1. Scenario Description

The Interoperability scenario aims to assess the ability of the solution to allow the TSG SSI wallet to maintain specifically its technical interoperability among different supported DID methods. The goal is to achieve complete seamlessness and uniformity in the experience that any user has when using the TSG SSI wallet, irrespective of the supported DID method. The TSG SSI wallet must behave the same from the client's standpoint when performing functionalities related to DIDs, namely DID creation and DID resolution. For the current state of the system, this scenario means triggering the same actions for both did:web and did:tdw, and ensuring that all actions result to the same outcomes in both scenarios. This scenario is further split into two parts: 1) DID creation, and 2) DID resolution.

5.2.1.1.1. DID Creation

The exact steps to conduct the verification scenario on DID creation is described in Table 10 below. An existing GET /management/did endpoint on the TSG SSI wallet is utilized for this scenario, as it performs a direct query to the database to retrieve the created DID Document and conducts no actual DID resolution processing. The process should be performed twice, once with the did:web DID method configuration, and a second time with the did:tdw DID method configuration.

Verification Scenario:	DID Creation Interoperability
Action Steps:	1. Update the RootConfig.didMethod configuration to either did:web or did:tdw, depending on the scenario round.
	2. Start the TSG SSI wallet application. On start up, the appropriate DID identifier and DID Document will be automatically created.
	3. Perform an HTTP GET API request to the /management/did endpoint of the TSG SSI wallet application.
Expected Results:	1. A DID Document which follows the W3C DID specification is received as a response from the request to the /management/did endpoint.
	2. The DID identifier prefix matches with the DID method specified in the configurations before start up, and follows the format defined by the corresponding DID method specification.

Expected Failure:	ERROR1: The RootConfig.didMethod configuration is set to a DID method value other than did:web or did:tdw. There should be an error upon starting the application.
--------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------

Table 10: Interoperability Verification Scenario – DID Creation Interoperability

5.2.1.1.2. DID Resolution

The steps to perform interoperability verification on DID resolution is explained in Table 11 below. The TSG SSI wallet exposes an endpoint to perform DID resolution via HTTP API request, which is the GET /management/did/resolve/{didId} endpoint. The endpoint performs DID resolution for the {didId} path parameter value through using the resolve() method of the DidResolverService class, meaning that it follows the mechanisms outlined in the respective DID method specifications of each supported DID method. This DID resolver endpoint accepts both the did:web and did:tdw DID methods, independent of the DID method supported by the instance of TSG SSI wallet used to resolve the DID identifier. Similar to the DID creation scenario, the process below should also be performed twice, once with a did:web DID identifier, and a second time with a did:tdw DID identifier.

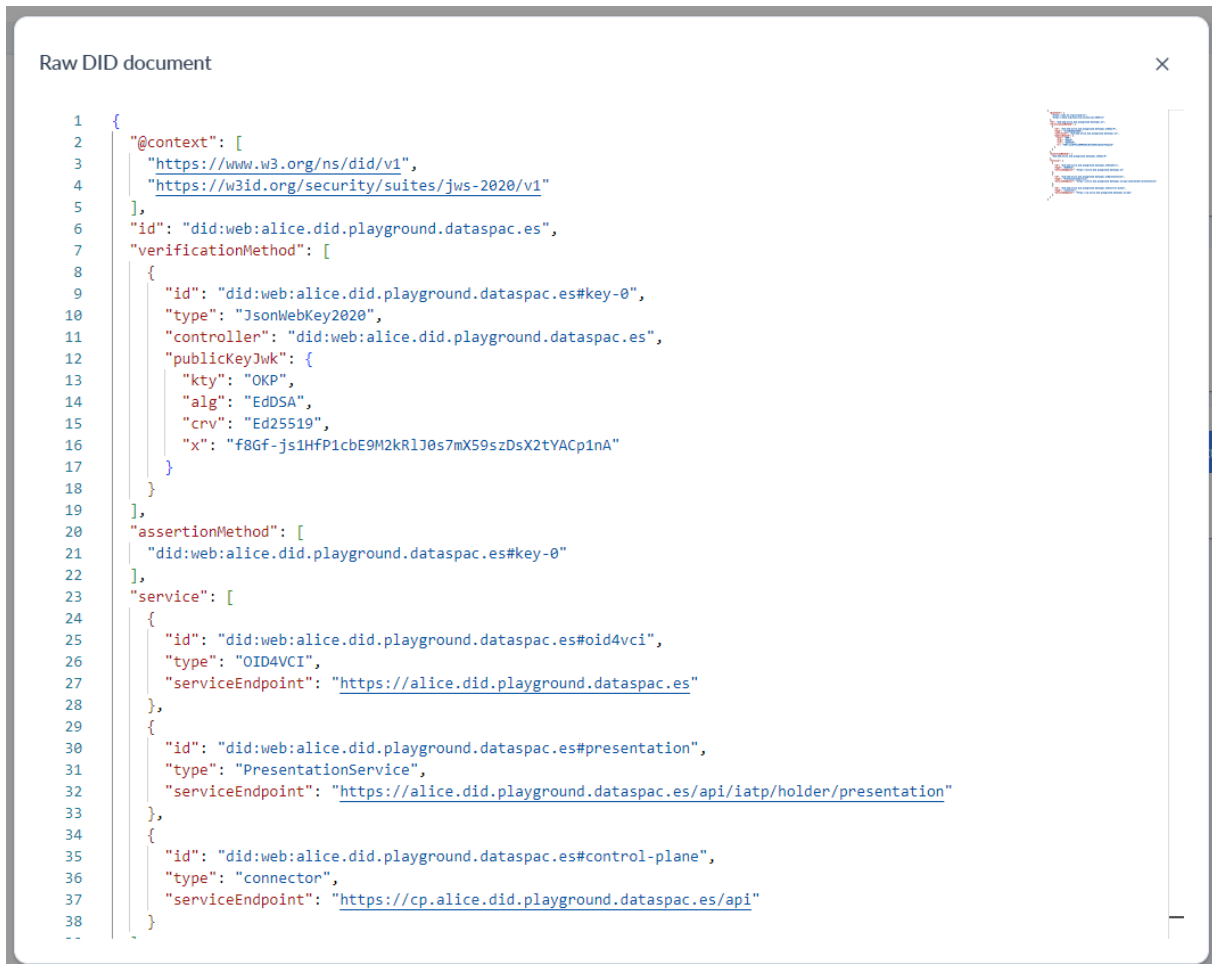
Verification Scenario:	DID Resolution Interoperability
Action Steps:	1. Ensure that the DID identifier used is valid with prefix either did:web or did:tdw, depending on the scenario round.
	2. Ensure that a VDR exists that hosts the DID Document for the DID identifier used.
	3. Perform an HTTP GET API request to the /management/did/resolve/{didId} endpoint, where the {didId} path parameter is replaced by the actual DID identifier.
Expected Results:	1. A DID Document is received as a response from the /management/did/resolve/{didId} endpoint.
	2. The id field in the DID Document is the same as the DID identifier used in the {didId} path parameter.
Expected Failure:	ERROR1: The DID identifier trying to be resolved is neither of DID method did:web nor did:tdw. There should be an error that the DID resolver does not support this DID method.

Table 11: Interoperability Verification Scenario – DID Resolution Interoperability

5.2.1.2. Scenario Results

5.2.1.2.1. DID Creation

The TSG SSI wallet user interface (UI) makes a call to the /management/did endpoint to retrieve the DID Document and display it on the UI dashboard. The screenshot in Figure 19 below shows the resultant DID Document in the TSG SSI wallet UI dashboard when following the DID Creation Interoperability verification scenario with the did:web DID method configuration. Figure 20 displays the screenshot of the same, but with the did:tdw DID method configuration.



```
1 {
2   "@context": [
3     "https://www.w3.org/ns/did/v1",
4     "https://w3id.org/security/suites/jws-2020/v1"
5   ],
6   "id": "did:web:alice.did.playground.dataspac.es",
7   "verificationMethod": [
8     {
9       "id": "did:web:alice.did.playground.dataspac.es#key-0",
10      "type": "JsonWebKey2020",
11      "controller": "did:web:alice.did.playground.dataspac.es",
12      "publicKeyJwk": {
13        "kty": "OKP",
14        "alg": "EdDSA",
15        "crv": "Ed25519",
16        "x": "f8Gf-js1HfP1cbE9M2kRlJ0s7mX59szDsX2tYACp1nA"
17      }
18    }
19  ],
20  "assertionMethod": [
21    "did:web:alice.did.playground.dataspac.es#key-0"
22  ],
23  "service": [
24    {
25      "id": "did:web:alice.did.playground.dataspac.es#oid4vci",
26      "type": "OID4VCI",
27      "serviceEndpoint": "https://alice.did.playground.dataspac.es"
28    },
29    {
30      "id": "did:web:alice.did.playground.dataspac.es#presentation",
31      "type": "PresentationService",
32      "serviceEndpoint": "https://alice.did.playground.dataspac.es/api/iatp/holder/presentation"
33    },
34    {
35      "id": "did:web:alice.did.playground.dataspac.es#control-plane",
36      "type": "connector",
37      "serviceEndpoint": "https://cp.alice.did.playground.dataspac.es/api"
38    }
39  ]
40 }
```

Figure 19: Result of DID Creation with did:web

```

Raw DID document
1  {
2    "@context": [
3      "https://www.w3.org/ns/did/v1",
4      "https://w3id.org/security/suites/jws-2020/v1"
5    ],
6    "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy",
7    "controller": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy",
8    "assertionMethod": [
9      "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#key-0"
10   ],
11   "verificationMethod": [
12     {
13       "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#key-0",
14       "controller": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy",
15       "type": "JsonWebKey2020",
16       "publicKeyJwk": {
17         "kty": "OKP",
18         "alg": "EdDSA",
19         "crv": "Ed25519",
20         "x": "x5SHU3QviViuQicGRUyf6Ni0fxRWRzix__J3s12-rPmE"
21       }
22     }
23   ],
24   "service": [
25     {
26       "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#oid4vci",
27       "type": "OID4VCI",
28       "serviceEndpoint": "https://charlie.did.playground.dataspac.es"
29     },
30     {
31       "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#presentation",
32       "type": "PresentationService",
33       "serviceEndpoint": "https://charlie.did.playground.dataspac.es/api/iatp/holder/presentation"
34     },
35     {
36       "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#control-plane",
37       "type": "connector",
38       "serviceEndpoint": "https://cp.charlie.did.playground.dataspac.es/api"
39     }
40   ]
41 }

```

Figure 20: Result of DID Creation with did:tdw

By comparing the two snippets above, it is evident that configuring different DID methods maintain the outcome of the DID Document creation. The structure of both DID Documents comply to the W3C DID specification, and have the appropriate values for verification methods, assertion methods and services. The DID identifier generated, as seen in the id field, is also true according to the DID method configured, and clearly shows the distinguishing prefixes.

Aside from the successful scenarios above with did:web and did:tdw configurations, the results of the verification scenario to produce a failure output is shown below Figure 21. In this scenario, the DID method configuration is set to did:ion.

```

11:46:03 PM - Starting compilation in watch mode...

11:46:06 PM - Found 0 errors. Watching for file changes.
Configuration is not valid:
- config did.method does not match the following rules:
  - isIn: method must be one of the following values: did:web:, did:tdw:, current config is `did:ion:`
Waiting 30 seconds before exit

```

Figure 21: Result of DID Creation with an unsupported DID method

5.2.1.2.2. DID Resolution

Figures 22 and 23 below presents screenshots of the resolved DID Documents, for did:web and did:tdw respectively, retrieved via performing HTTP GET API requests to the TSG SSI wallet using the web browser to fulfil the DID Resolution Interoperability verification scenario. The same TSG SSI wallet, with the URL <https://did.playground.dataspac.es/>, is utilized for both cases.



```
1 {
2   "@context": [
3     "https://www.w3.org/ns/did/v1",
4     "https://w3id.org/security/suites/jws-2020/v1"
5   ],
6   "id": "did:web:alice.did.playground.dataspac.es",
7   "verificationMethod": [
8     {
9       "id": "did:web:alice.did.playground.dataspac.es#key-0",
10      "type": "JsonWebKey2020",
11      "controller": "did:web:alice.did.playground.dataspac.es",
12      "publicKeyJwk": {
13        "kty": "OKP",
14        "alg": "EdDSA",
15        "crv": "Ed25519",
16        "x": "f8Gf-js1HfP1cbE9M2kRlJ0s7mX59szDsX2tYACp1nA"
17      }
18    }
19  ],
20  "assertionMethod": [
21    "did:web:alice.did.playground.dataspac.es#key-0"
22  ],
23  "service": [
24    {
25      "id": "did:web:alice.did.playground.dataspac.es#oid4vci",
26      "type": "OID4VCI",
27      "serviceEndpoint": "https://alice.did.playground.dataspac.es"
28    },
29    {
30      "id": "did:web:alice.did.playground.dataspac.es#presentation",
31      "type": "PresentationService",
32      "serviceEndpoint": "https://alice.did.playground.dataspac.es/api/iatp/holder/presentation"
33    },
34    {
35      "id": "did:web:alice.did.playground.dataspac.es#control-plane",
36      "type": "connector",
37      "serviceEndpoint": "https://cp.alice.did.playground.dataspac.es/api"
38    }
39  ]
40 }
```

Figure 22: Result of DID Resolution with did:web

```

1 {
2   "@context": [
3     "https://www.w3.org/ns/did/v1",
4     "https://w3id.org/security/suites/jws-2020/v1"
5   ],
6   "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy",
7   "controller": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy",
8   "assertionMethod": [
9     "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#key-0"
10  ],
11  "verificationMethod": [
12    {
13      "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#key-0",
14      "controller": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy",
15      "type": "JsonWebKey2020",
16      "publicKeyJwk": {
17        "kty": "OKP",
18        "alg": "EdDSA",
19        "crv": "Ed25519",
20        "x": "x5HU3QviViuQicGRUyf6NiOfxRWRzix__J3s12-rPmE"
21      }
22    }
23  ],
24  "service": [
25    {
26      "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#oid4vci",
27      "type": "OID4VCI",
28      "serviceEndpoint": "https://charlie.did.playground.dataspac.es"
29    },
30    {
31      "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#presentation",
32      "type": "PresentationService",
33      "serviceEndpoint": "https://charlie.did.playground.dataspac.es/api/iatp/holder/presentation"
34    },
35    {
36      "id": "did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy#control-plane",
37      "type": "connector",
38      "serviceEndpoint": "https://cp.charlie.did.playground.dataspac.es/api"
39    }
40  ]
41 }

```

Figure 23: Result of DID Resolution with did:tdw

Examining the two HTTP GET API responses side-by-side, it is apparent that the TSG SSI wallet has no issues with resolving either a did:web DID or a did:tdw DID without fail. The DID Documents of the two example DID methods used in this scenario were successfully resolved from the given DID identifiers, which are also displayed on the id field of the DID Documents.

Apart from the successful DID resolution scenarios shown for the did:web and did:tdw DID identifiers, Figure 24 displays the output failure when an attempt is made to resolve an unsupported DID method, in this case a did:pkh DID.

```

1 {
2   "name": "AppError",
3   "status": "BAD_REQUEST",
4   "code": 400,
5   "message": "Resolver does not support the did:pkh: method"
6 }

```

Figure 24: Result of DID Resolution with an unsupported DID method

5.2.1.3. Scenario Evaluation

Considering the results of the Interoperability verification scenarios above when supplied with different parameters, it can therefore be concluded that the new software design of including the Strategy pattern correctly fulfils the requirement of supporting different DID methods on the TSG SSI wallet. The solution accurately addresses the challenge of adapting the previously rigid and fixed software design to a more dynamic and flexible iteration, as evident by the results displaying the expected differences between the values of the DID methods and DID Documents upon creation and resolution, whilst maintaining consistency in its core behaviours.

5.2.2. Security

5.2.2.1. Scenario Description

The Security scenario attempts to assess the trustworthiness, authenticity and verifiability of the newly implemented DID method. The aim of this scenario is to exploit the vulnerabilities that exist in the did:web DID method, and investigating if the newly supported DID method brings about a cure to the current security risks that did:web poses.

Due to solely relying on hosting a JSON file on a web server as a means for DID resolution, the existing did:web DID method is substantially liable to susceptibility by unwanted parties. In the event where a malicious actor gains malevolent access to an entity's web infrastructure, they can intentionally infiltrate the web server and replace the hosted DID Document with ease, thus also replacing the verification keys associated with the DID. The malicious actor can then create VCs and VPs with their new set of keys, and parties resolving the DID identifier are unable to detect that the DID Document has been mischievously tampered by an actor that is neither the owner nor the maintainer of the DID.

The explicit steps required to perform the verification scenario on the security vulnerability of did:web is explained in Table 12 below. Since this security verification scenario is conducted within a local environment, complete control over the TSG SSI wallet instance is possessed, hence the scenario can be simulated by directly modifying the DID Document on the TSG SSI wallet server without requiring additional server infiltration mechanisms. This scenario should be performed twice, once with a TSG SSI wallet instance that hosts a did:web DID, and another round with an instance that hosts a did:tdw DID.

Verification Scenario:	Web Infrastructure Infiltration
Action Steps:	1. Ensure that TSG SSI wallet is hosting a DID Document that matches with the DID method to be used for this round.
	2. Access the TSG SSI wallet server and modify the values that are involved in the DID resolution process with new values (DID Document for did:web and DID Logs for did:tdw).
	3. Perform an HTTP GET API request to the /management/did/resolve/{didId} endpoint, where the {didId} path parameter is replaced by the actual DID identifier.
Expected Results: did:web	1. A DID Document is received as a response from the /management/did/resolve/{didId} endpoint.
	2. The id field in the DID Document is the same as the DID identifier used in the {didId} path parameter.
Expected Results: did:tdw	1. The /management/did/resolve/{didId} endpoint responds an error with a message stating that the DID resolution process failed.

Table 12: Security Verification Scenario – Web Infrastructure Infiltration

5.2.2.2. Scenario Results

5.2.2.2.1. did:web Attack

In the beginning, upon starting up the TSG SSI wallet application with the DID method configuration set to did:web, resolving the DID identifier will return the created DID Document as shown in the HTTP GET API result in Figure 25.



```
1 {
2   "@context": [
3     "https://www.w3.org/ns/did/v1",
4     "https://w3id.org/security/suites/jws-2020/v1"
5   ],
6   "id": "did:web:localhost%3A3000",
7   "verificationMethod": [
8     {
9       "id": "did:web:localhost%3A3000#key-0",
10      "type": "JsonWebKey2020",
11      "controller": "did:web:localhost%3A3000",
12      "publicKeyJwk": {
13        "kty": "OKP",
14        "alg": "EdDSA",
15        "crv": "Ed25519",
16        "x": "gndKcZtgUVKWTv2pGj7sBPy33MBdcR0g53EIAjoGI8"
17      }
18    }
19  ],
20  "assertionMethod": [
21    "did:web:localhost%3A3000#key-0"
22  ],
23  "service": [
24    {
25      "id": "did:web:localhost%3A3000#oid4vci",
26      "type": "OID4VCI",
27      "serviceEndpoint": "https://localhost:3000"
28    },
29    {
30      "id": "did:web:localhost%3A3000#presentation",
31      "type": "PresentationService",
32      "serviceEndpoint": "http://localhost:3000/api/iatp/holder/presentation"
33    }
34  ]
35 }
```

Figure 25: Initial state of resolved DID Document before did:web attack

Afterwards, the server where the TSG SSI wallet application is run on is accessed, and the DID Document is modified with new malicious values, specifically the public cryptographic key value (publicKeyJwk property) used as a verificationMethod, and also the endpoint of all service entities (serviceEndpoint property). When reperforming the DID resolution request, the results as illustrated on Figure 26 shows that the DID resolution process still succeeds. This is due to the fact that the resolution process of did:web only involves retrieving the created DID Document from storage, and performing no other additional processes to verify it. Since there are no noticeable differences or warnings, a client resolving this DID would essentially have no sense of awareness that it was modified by an actor other than the DID owner, and continue to trust its authenticity and validity.

```
localhost:3000/api/management/did/resolve/did%3Aweb%3Alocalhost%253A3000
1 {
2   "@context": [
3     "https://www.w3.org/ns/did/v1",
4     "https://w3id.org/security/suites/jws-2020/v1"
5   ],
6   "id": "did:web:localhost%3A3000",
7   "verificationMethod": [
8     {
9       "id": "did:web:localhost%3A3000#key-0",
10      "type": "JsonWebKey2020",
11      "controller": "did:web:localhost%3A3000",
12      "publicKeyJwk": {
13        "kty": "OKP",
14        "alg": "EdDSA",
15        "crv": "Ed25519",
16        "x": "malicious-key-value"
17      }
18    }
19  ],
20  "assertionMethod": [
21    "did:web:localhost%3A3000#key-0"
22  ],
23  "service": [
24    {
25      "id": "did:web:localhost%3A3000#oid4vci",
26      "type": "OID4VCI",
27      "serviceEndpoint": "https://malicious.domain.com"
28    },
29    {
30      "id": "did:web:localhost%3A3000#presentation",
31      "type": "PresentationService",
32      "serviceEndpoint": "http://malicious.domain.com"
33    }
34  ]
35 }
```

Figure 26: State of resolved DID Document after did:web attack

5.2.2.2.2. did:tdw Attack

On application startup of the TSG SSI wallet with the did:tdw DID method configuration, a resolve request to the DID identifier will produce a response containing the DID Document as shown in the web browser screenshot of Figure 27.



```
1 {
2   "@context": [
3     "https://www.w3.org/ns/did/v1",
4     "https://w3id.org/security/suites/jws-2020/v1"
5   ],
6   "id": "did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr",
7   "controller": "did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr",
8   "assertionMethod": [
9     "did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr#key-0"
10  ],
11  "verificationMethod": [
12    {
13      "id": "did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr#key-0",
14      "controller": "did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr",
15      "type": "JsonWebKey2020",
16      "publicKeyJwk": {
17        "kty": "OKP",
18        "alg": "EdDSA",
19        "crv": "Ed25519",
20        "x": "gndKcZtgUVKwTVv2pGj7sBPY33MBdcR0g53EIAjoGI8"
21      }
22    }
23  ],
24  "service": [
25    {
26      "id": "did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr#oid4vci",
27      "type": "OID4VCI",
28      "serviceEndpoint": "https://localhost:3000"
29    },
30    {
31      "id": "did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr#presentation",
32      "type": "PresentationService",
33      "serviceEndpoint": "http://localhost:3000/api/iatp/holder/presentation"
34    }
35  ]
36 }
```

Figure 27: Initial state of resolved DID Document before did:tdw attack

For did:tdw, the DID Logs are modified instead of the direct DID Document, since this is the actual source of truth processed to perform DID resolution. The values updated with the malicious content remains identical to the did:web scenario, which are the publicKeyJwk of verificationMethod and the serviceEndpoint of service. Figure 28 shows the result of reperforming the HTTP GET API request via web browser for resolving the did:tdw DID after performing the above changes. It can be clearly observed that the DID Document is not successfully resolved, and an error is returned in the response.



```
1 {
2   "name": "AppError",
3   "status": "BAD_REQUEST",
4   "code": 400,
5   "message": "Could not load DID document for did:tdw:localhost%3A3000:bb3tywamedxunbj6xxx5sczrbtjr"
6 }
```

Figure 28: State of resolved DID Document after did:tdw attack

This phenomenon can be attributed to the course of actions carried out when a did:tdw DID is resolved. The resolution process of did:tdw involves retrieving the DID Logs JSON Lines values from the hosting web server, and then processing the DID Logs to finally resolve the actual DID Document. Due to did:tdw's nature of storing a chained history of DID Log states and maintaining verifiable signatures built for each DID Log entry value, a change in the

contents of the DID Logs would lead to detection of a discrepancy in the existing and expected values during the resolution process, hence indicating a possible instance of malicious and unwanted tampering, thus an error message is returned. This particular characteristic of did:tdw prevents malevolent actors who gain access to the web infrastructure from freely modifying the contents of the DID Document, unless they are also in possession of the private part of the verification keys (which is a greater problem as they will not only be able to achieve legitimate overwrites of the DID Logs, but also completely pretend to be the rightful owner of the keys and use it for other malicious purposes). Otherwise, the design of did:tdw successfully increases the trustworthiness of DID Documents, as the authenticity and verifiability is guaranteed.

5.2.2.3. Scenario Evaluation

As the above results shown by the two Security scenarios, involving did:web and did:tdw respectively, it is clear that did:tdw provides an additional layer of protection to the TSG SSI wallet. In comparison to using the naïve did:web DID method, using the did:tdw DID method improves the aspect of security, in terms of the trustworthiness of the DID Document, the authenticity of modifications made to it, and also verifiability of a certain DID identifier to be resolved to a DID Document. This is achieved by reducing the risk of compromise by malicious actors who gains access to the web server, as they would not be able to freely alter the DID Document without being in possession of the actual cryptographic keys used to control the DID, given that these keys are stored in a separate isolated secret manager according to best practices. However, in the situation where the cryptographic keys are also compromised, then the malicious actor would be able to gain full control over the DID and DID Document. This is an aspect that is not yet addressed by did:tdw.

5.3. Validation

Completing the coverage of DSSC principle in the verification and validation processes, the validation scenario attempts to validate the Usability of the new solution, which relates to the Validity and Compliance principles. The subchapters below discuss further regarding this Usability scenario, including evidence of the results and a brief evaluation.

5.3.1. Usability

5.3.1.1. Scenario Description

The Usability scenario focuses on confirming the validity and compliance of the new solution on the TSG SSI wallet in the context of a real data space. Where the two verification scenarios only involve assessing the TSG SSI wallet on its own, this validation scenario also considers the inclusion of the TSG Control Plane. The goal of this validation scenario is to observe the actual usage of the did:tdw DID method in the end-to-end flow of data transfer between two participants within a data space environment. This involves not only usage of DIDs, but also the creation and sharing of VCs and VPs.

This Usability scenario is further broken down into four cases: 1) did:web to did:web, 2) did:tdw to did:tdw, 3) did:web to did:tdw, and 4) did:tdw to did:web. Each case is intended to cover the different possible types of interactions between consumer and provider data space participants. In order to fulfil these scenarios, four TSG IDS Connectors and five TSG SSI wallets are created. One TSG SSI wallet represents the data space wallet that issues VCs to data space participants, and the four pairs of TSG IDS Connectors and SSI wallets represent each individual data space participant. The data space wallet and two participant wallets will be run under the did:web configuration, and the other two participant wallets will utilize the did:tdw configuration.

Once the VCs are issues to each data space participant denoting their participation in the data space, a Catalog Request²¹ is performed from the consuming to the providing TSG IDS Connectors. This scenario is performed four times, in accordance to the four cases previously mentioned. The Catalog Request functionality inherently looks up a certain IDS Connector by their Control Plane URL and SSI wallet DID identifier, and then makes a request to view all the datasets that it provides. In this flow, DID resolution is performed to verify both the issuer and holder DIDs of the VC. In real practice, the Catalog Request should return datasets based on what is contained in the TSG Data Plane. However, for this validation simulation, the TSG Data Plane was excluded for efficiency reasons, since there are no DID-related functionalities performed directly on it. Hence, it is expected that for all validation cases of this scenario, the dataset will be empty despite successfully discovering the requested IDS Connector. For this thesis study, this simulation flow is sufficient since the Catalog Request is the only interaction in the data space transaction process that involves the SSI wallets. All other flows after it (Contract Negotiation and Transfer Process) no longer involve the SSI wallet because trust is already gained from the initial stages of IDS Connector discovery in the Catalog Request phase. Table 13 summarizes the exact steps required to perform this Usability validation scenario.

²¹ <https://docs.internationaldataspaces.org/ids-knowledgebase/v/dataspace-protocol/catalog/catalog.protocol>

Validation Scenario:	Catalog Request
Action Steps:	1. Ensure that there are two TSG IDS Connectors (Control Plane + SSI wallet) running with DID methods according to this round of validation.
	2. Ensure that there is one TSG SSI wallet running as the data space wallet, which issues VCs for the other two TSG IDS Connectors to become participants of the data space.
	3. Access the TSG Control Plane UI and perform a Catalog Request to the desired TSG IDS Connector, specifying the TSG Control Plane URL and the TSG SSI wallet DID.
Expected Results:	1. The desired TSG IDS Connector is discovered with the correct DID method.
	2. The Catalog is empty since there is no TSG Data Plane connected to the TSG IDS Connector.

Table 13: Usability Validation Scenario – Catalog Request

5.3.1.2. Scenario Results

Figure 29 illustrates a screenshot from the TSG SSI wallet UI of the data space wallet showing the VCs that are issued by the data space wallet to the data space participants. The VPs are generated automatically once the VCs are issued to the data space participant wallets, however are not shown on the UI. From this snippet, it can be observed that the data space wallet’s (Issuer) DID is `did:web:did.playground.dataspac.es`. The DIDs of the two `did:web` TSG IDS Connectors are `did:web:alice.did.playground.dataspac.es` and `did:web:bob.did.playground.dataspac.es`, and the DIDs of the two `did:tdw` TSG IDS Connectors are `did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy` and `did:tdw:dan.did.playground.dataspac.es:bm5xyowovfo2a4k4oybrjoutdqwn`.

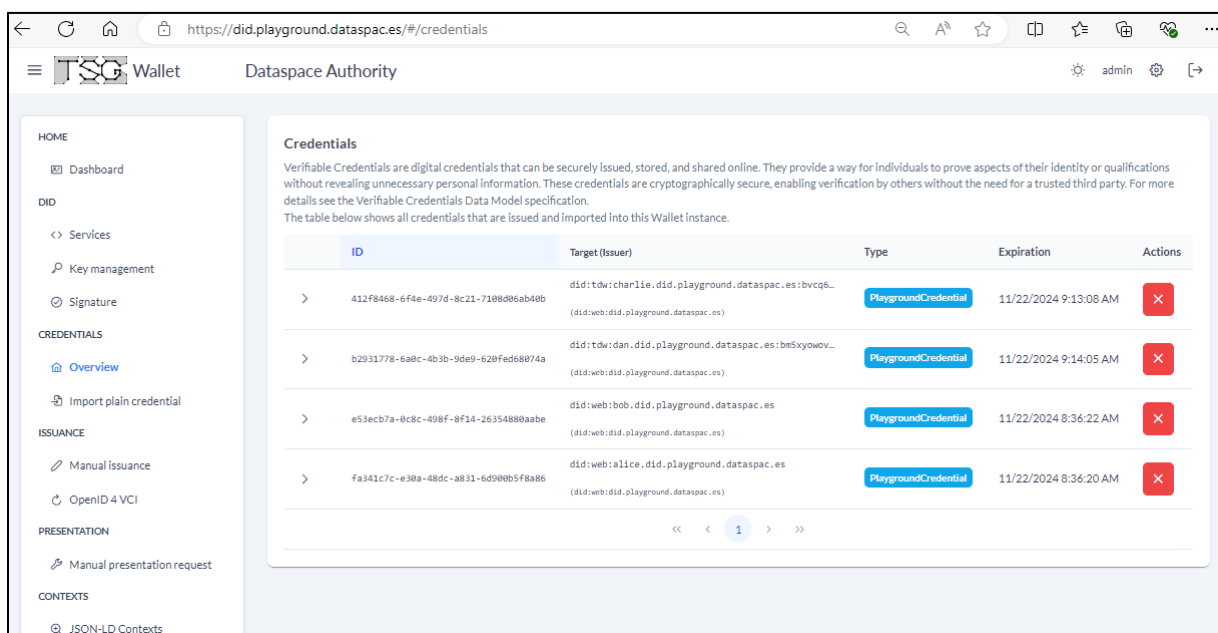


Figure 29: Credentials issued by Data Space Wallet to Data Space Participants

The image on Figure 30 displays the default Catalog Request page on the TSG Control Panel UI when it is initially opened. Requesting a Catalog from another IDS Connector involves specifying the URL of the Control Plane and the DID identifier of the SSI wallet.

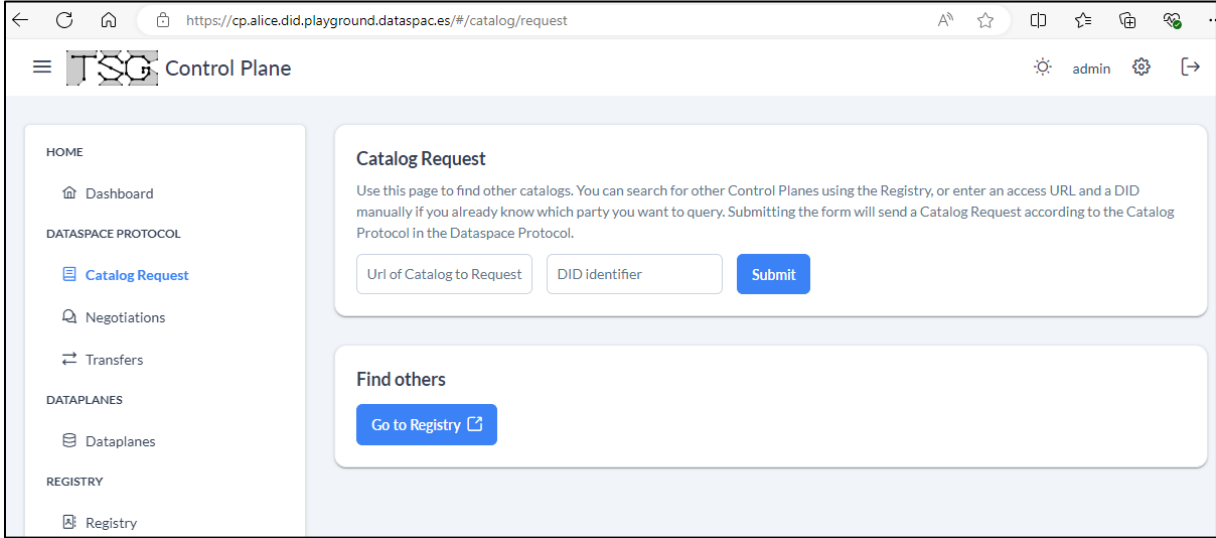


Figure 30: Default Catalog Request page on Control Plane

5.3.1.2.1. did:web to did:web

The did:web to did:web Catalog Request scenario will involve the TSG IDS Connectors with DIDs did:web:alice.did.playground.dataspac.es and did:web:bob.did.playground.dataspac.es, where the former will be making the request for catalogs of the later. The result is shown in Figure 31.

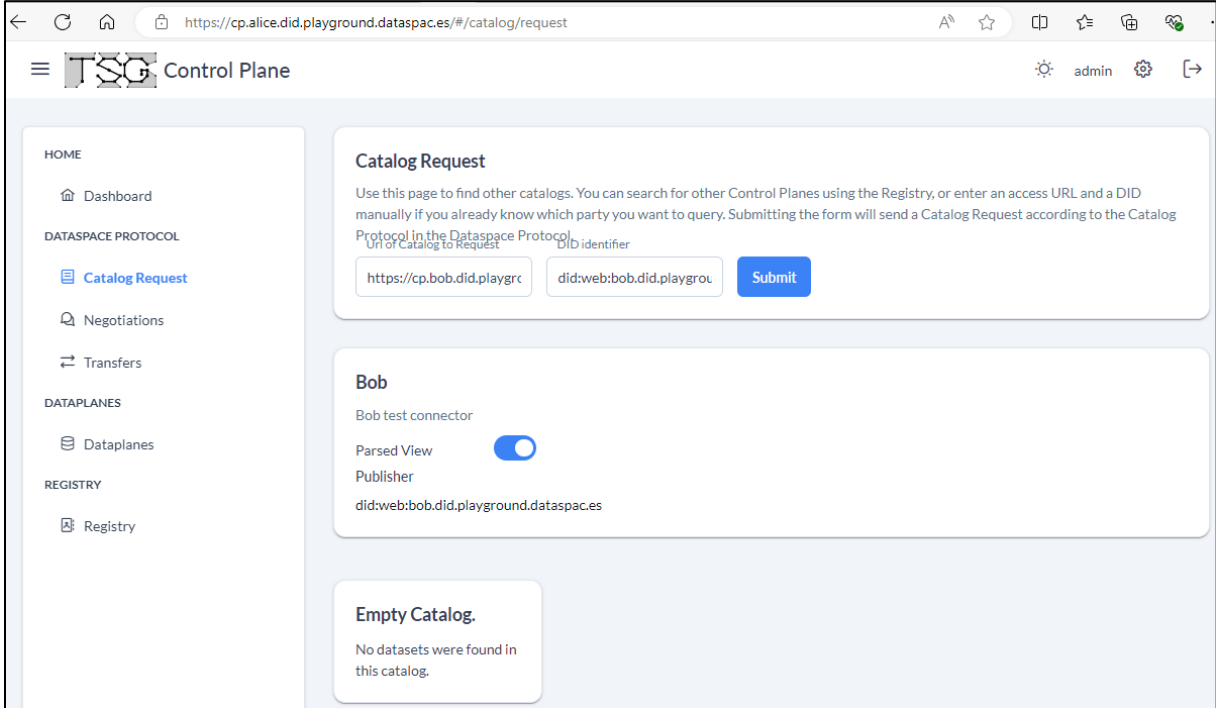


Figure 31: Result of did:web to did:web Catalog Request

5.3.1.2.2. did:tdw to did:tdw

For the did:tdw to did:tdw Catalog Request scenario, the TSG IDS Connectors `did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy` and `did:tdw:dan.did.playground.dataspac.es:bm5xyowovfo2a4k4oybrjoutdqwn` are used. Figure 32 displays the results of this scenario.

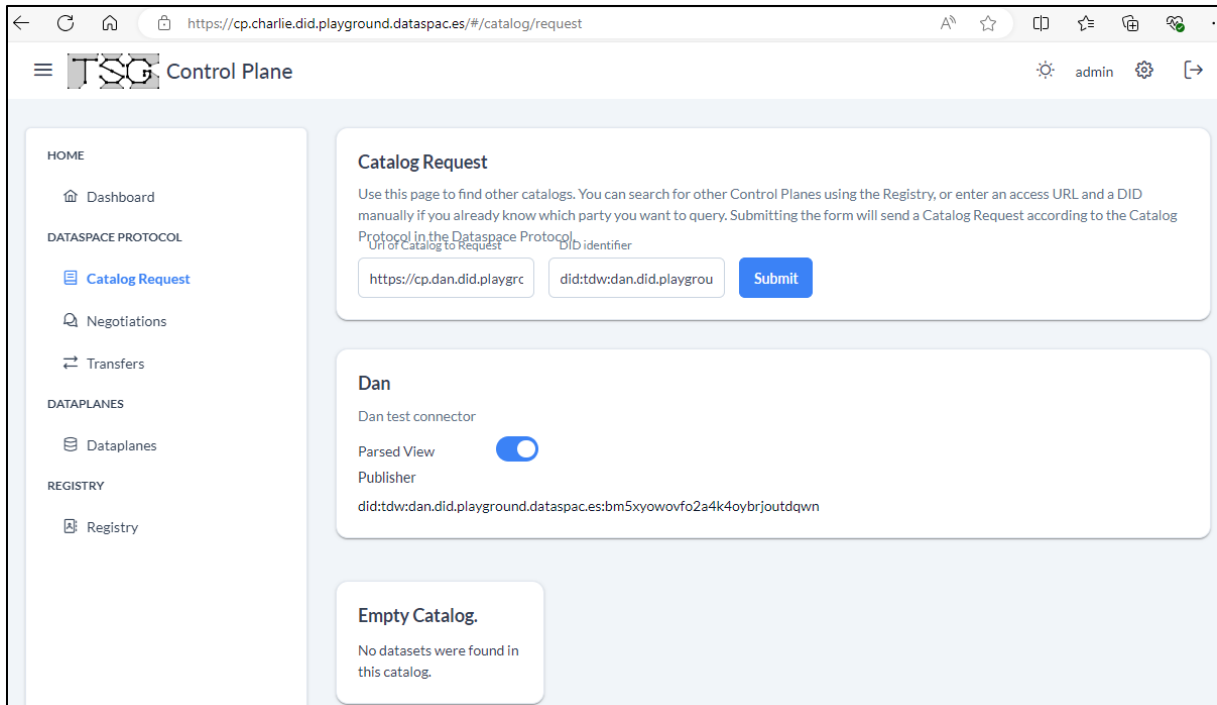


Figure 32: Result of did:tdw to did:tdw Catalog Request

5.3.1.2.3. did:web to did:tdw

The did:web to did:tdw Catalog Request scenario is performed by involving the `did:web:bob.did.playground.dataspac.es` and `did:tdw:charlie.did.playground.dataspac.es:bvcq6fyzf2572c4tjmf7bwlqgyyy` IDS Connectors. The result of this Catalog Request scenario is shown in Figure 33.

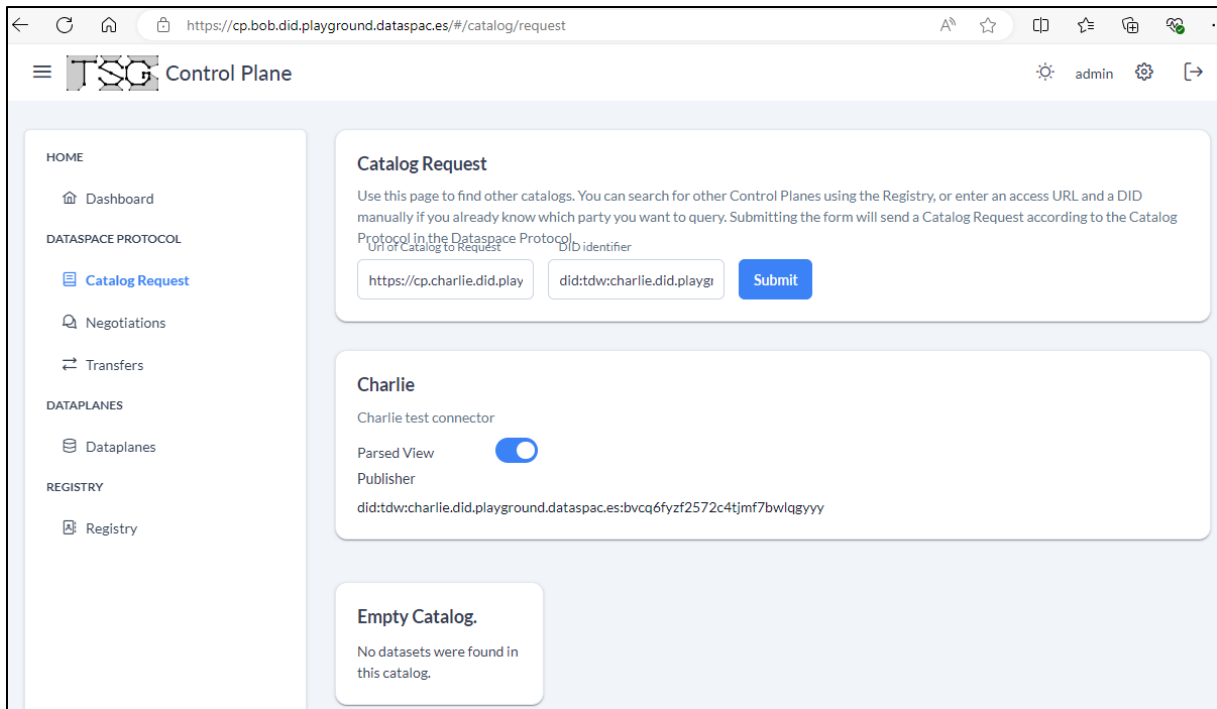


Figure 33: Result of did:web to did:tdw Catalog Request

5.3.1.2.4. did:tdw to did:web

Figure 34 illustrates the Catalog Request for the did:tdw to did:web scenario, which is conducted by using the did:tdw:dan.did.playground.dataspac.es:bm5xyowovfo2a4k4oybrjoutdqwn and the did:web:alice.did.playground.dataspac.es TSG IDS Connectors.

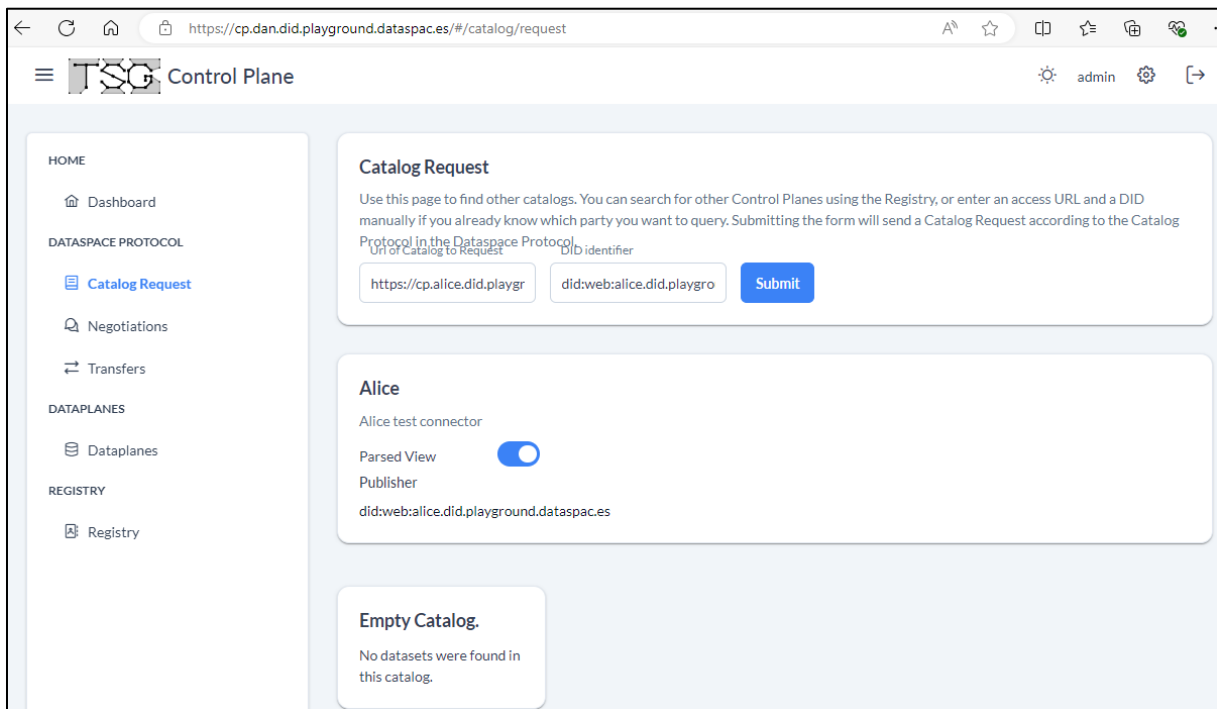


Figure 34: Result of did:tdw to did:web Catalog Request

5.3.1.3. Scenario Evaluation

With these validation results of the Usability scenarios, it is evident that the new did:tdw DID method introduced to the data spaces context is able to function properly and seamlessly, within interactions among the same and different DID methods. The new DID method does not fail to comply with the existing standards and specifications that are defined for data space interactions, and additionally the VCs and VPs involved in the Catalog Request flow remain valid when utilizing the new did:tdw DID method.

6. Final Remarks

6.1. Conclusions

This thesis aims to contribute to ongoing efforts to standardize a protocol for the technical specifications of data space identity management. With the decentralized-nature of data spaces, this study contributes to the possibilities of integrating Self-Sovereign Identity (SSI) wallets as a form of decentralized identity provider, specifically exploring the suitability of different Decentralized Identifier (DID) methods used in SSI wallets for data spaces. The research questions in Chapter 1 lay the foundation of what knowledge is to be gained through this study, with the main research question being “How to identify suitable Decentralized Identifier (DID) methods for implementation in Self-Sovereign Identity (SSI) wallets in the context of a data space?”. Discussions regarding the answers to each of the six sub-research questions are given below:

SRQ 1: What are the common DID methods used in SSI wallets, based on recent European and global trends?

The exhaustive list of supported DID methods, gathered from selected knowledge sources including other IDS connectors and independent SSI wallets, is available to audit in Tables 1, 2, and 3 in Chapter 3.2. Table 4 in Chapter 3.3 summarizes these findings to provide a more concise and readable overview. Results show that did:web is the most frequently supported among all other DID methods independent of the underlying technology. A four-way tie between did:ion, did:ebis, did:sov, and did:ethr top the most frequent supported ledger-based DID methods. Based on this gathering process, a total of 11 final DID methods contained in the following subset: {did:webs, did:tdw, did:key, did:jwk, did:peer, did:pkh, did:ion, did:ebis, did:sov, did:ethr, did:cheqd} were chosen to be analysed for further evaluation in the next steps of the study. The main factors considered behind making this decision were popularity and underlying technology.

SRQ 2: What standardized rubric can be used to compare the selected DID methods in the context of usage in a data space?

As decentralized identity management is a growing field, the presence of standardized rubrics for evaluating DID characteristics are still limited. The most prominent, extensive and reliable rubric, the W3C DID Method Rubric v1.0, is by no surprise governed by W3C themselves as the maintainer of DID technology. There currently exists attempts by other parties to compose a more focused subset of the rubric based on specific contexts of usage, however there are no substantial results yet. Building off from this situation, the evaluation rubric incorporated for this thesis study was a reduced set of the W3C DID Method Rubric, where the included criteria were selected based on the basic characteristics of identity providers in a data space, with reference to the Data Spaces Support Centre. The methodology of selection process can be seen in Table 7 in Chapter 4.2. The complete derived rubric is shown in Table 14 and the scoring sheet in Table 15, both in Appendix B.

SRQ 3: How to compare the selected DID methods based on the derived evaluation rubric?

The complete resultant scoring sheets of each selected DID method and the currently supported did:web method used in the evaluation process with reference to the derived evaluation rubric is included in Appendix C. More succinct and coherent summaries encompassing the

underlying technology, advantages, and disadvantages of each evaluated DID method are discussed in Chapter 4.3. Analysis based on total accumulated score rankings in Table 8 and characteristics groupings are discussed in Chapter 4.4. The results of this study indicated that different DID methods with distinct underlying technologies can be classified into groups based on the similar characteristics that they offer. The characteristic groups observed from the list of evaluated DID methods in this study include ledger-based, key-based, peer-based and web-based. The different characteristics of the underlying technologies also dictate the behaviors of the DID methods, which determine their benefits and drawbacks. Ledger-based DID methods provide the most decentralization and verifiability due to their reliance on secure distributed ledger technologies, however can lead to bloating fees and extended initial learning times. Key-based DID methods are by far the easiest to implement in terms of technicality due to their generative nature, but provide the least level of security measures. Peer-based DID methods are equipped with high standards of trust and authenticity due to the requirement to preserve private peer relationships among DID users, yet poses a challenge in decentralization as implementation heavily relies on maintaining local VDRs. Improved web-based DID methods offer ease of implementation by using web technology, however have different degrees of trustworthiness and verifiability across different DID methods.

SRQ 4: Based on the evaluation results, which DID method is the most suitable for implementation in an SSI wallet within a data space domain?

Building off from the results of discussions in SRQ3 based on Chapter 4.4, conclusions of the selected most suitable DID method are drawn in Chapter 4.5. When only considering from the standpoint of the total cumulative scores, it is evident that ledger-based DID methods edge out non-ledger-based DID methods, where did:ion is at the top of the table. Apart from did:ebis, all other ledger-based DID methods outperform all non-ledger-based DID methods, aside from improved web-based DID methods like did:webs and did:tdw. However, the study concludes that the suitability of ledger-based DID methods was not the best fit for the context of data spaces due to its more difficult and costly implementations despite it offering the most security overall. Instead, improved web-based DID methods like did:webs and did:tdw were selected for their balance of adequate security guarantees and simpler implementation procedures. The trade-offs are deemed acceptable, considering that identity management is only one aspect of the whole data space transaction process. In the end, the did:tdw DID method was the sole recommendation for usage in SSI wallets of a data space, due to reasons of using more familiar technology and flexibility of control over the actual DID Documents owned.

SRQ 5: How can support for the new DID method be introduced into an existing data space SSI wallet?

Chapter 5.1.1 includes a class diagram of the existing TSG SSI wallet software design, and sequence diagrams illustrating the flow of interactions between class objects. Despite its suitability for the current use case of supporting did:web, the present software design of the TSG SSI wallet is a fixed solution and was not built with the consideration of the possibility of adopting another DID method. Instead of going through the direct and naïve route of simply adding new files, classes, and logic beside the existing ones on the TSG SSI wallet platform, a design pattern was introduced as a best practice for better software design and engineering. After considering the suitability several design patterns, the Strategy pattern was chosen as the most compatible with the challenges in hand. The new class diagram portraying the Strategy

pattern implemented on the TSG SSI wallet software design, and the corresponding modifications to the sequence diagrams, can be seen in Chapter 5.1.2. Development on the TSG SSI wallet application was performed based on the designs represented by the class diagram and sequence diagrams.

SRQ 6: How can the new SSI wallet implementation be verified and validated in the context of usage in a data space?

With the aim to appropriately verify and validate the results of the evaluation and implementation, concepts adopted in the early stages of this thesis study were revisited. The DSSC core design choices and the principles extracted from them, as used to create the evaluation rubric explained in Chapter 4.2, were reconsidered to build the related verification and validation scenarios. Two distinct verification scenarios (Interoperability, and Security) and one validation scenario (Usability) were generated for the new TSG SSI wallet implementation as described in Chapters 5.2 and 5.3. The Interoperability verification scenario aimed to verify that the TSG SSI wallet is able to interchangeably support both the existing did:web and new did:tdw DID methods without any significant discrepancies, aside from those defined in each of their specifications. The Security verification scenario aimed to evaluate the benefits that did:tdw brought to treat the security vulnerabilities that exists in did:web. The Usability validation scenario aimed to demonstrate that the did:tdw DID method is usable in the TSG IDS Connector throughout the entire data space transaction workflow, without any divergences from using did:web. The results and evaluation of these validation process are also presented in Chapters 5.2 and 5.3.

Given the answers to each of the above six research questions, and in line with the original objective of this thesis study, a strong recommendation can be given to the Eclipse Dataspace Working Group regarding the adoption of the did:tdw DID method by the Eclipse Dataspace Decentralized Claims Protocol. Though the existing recommendation of the did:web DID method also fulfills the bare minimum for usability, the results of this study demonstrates that the suitability of did:tdw within the context of data spaces brings more beneficial characteristics, as supported by both theoretical comparisons and practical proofs of implementation. In regard to DIDs and SSIs in general terms, the growing and promising field of decentralized identity management, driven by SSI and coupled with the diverse range of available DID methods, hold many benefits for adoption across different sectors, of which the suitability of the DID methods should be evaluated based on the specific context of interest, as such performed in this thesis study.

6.2. Limitations

One obvious limitation to this thesis study is the incorporation of an existing evaluation rubric instead of creating a new one from the ground up. The reasoning behind this choice was due to the considerable amount of time and effort that would be required to generate a new evaluation rubric, of which arguably could become an independent research study on its own. Involving an existing evaluation rubric to this study reduces the workload, as the evaluation rubric is only one supporting part of the overall study. Despite this, creating a custom rubric for the exact context of this study could generate more accurate evaluations rather than using a generic solution.

Another limitation lies in the implementation of the validation design. A library created by maintainers of the did:tdw specification providing its related features was utilized to support it in the TSG SSI wallet. Despite it being developed by the same group of people writing the specification, it was noticed that there were some inconsistencies between the actual library implementation and the specifications of did:tdw and W3C DID. For the continuation and success of this study, a clone of the library was created and hosted on TNO's public GitLab repository²², and modified to not only amend the defective feature, but also enable proper integration with the TSG SSI wallet.

6.3. Future Work

Based on the findings and discussions of this thesis study, several ideas to expand the work further could be explored upon in the future. Firstly, reiterating what was stated as a limitation, it could be intriguing to conduct further studies on establishing a context-specific DID method evaluation rubric exclusively for the domain of data spaces. New criteria that are derived from essential factors of data spaces could be incorporated to the new evaluation rubric. Moreover, instead of solely reusing the criteria from an existing evaluation rubric as it is, future work can be performed to assess how these criteria can be refined and expanded for the specific domain. This approach could also be applied when performing DID method evaluation for other contexts.

Another point of interest could be to perform interviews and surveys on maintainers of the other IDS Connectors and SSI wallets regarding the reasoning behind their decision to support the DID methods that they chose. In this thesis study, only a survey on information already available online (source code/technical documentation/reports) was conducted to identify the supported DID methods, without delving deeper into the factors that led them to make those decisions. Studying and analyzing these factors further could uncover more notions and rationales to why some DID methods are more favorable than others, which can be tailored further to the context of data spaces.

Towards the later stages of this thesis study, the maintainers of the did:tdw DID method released a newer version of the specification to the public which included substantial updates to the DID method's inner workings. This latest version of the specification encompassed valuable features such as enhanced security guarantees using an optional witnesses-like mechanism, a more space-efficient data representation for the JSON Lines file, and stricter rules on the did:tdw DID identifier format. Though these new features remain unimplemented up to the conclusion of

²² <https://gitlab.com/tno-tsg/dataspace-protocol/utils/trustdidweb-ts>

this thesis, further work could be conducted to analyze the benefits and methods of incorporating them to the existing TSG SSI wallet software design.

Addressing future work on a more general note, further studies could be conducted to ideate solutions for issues surrounding other levels of interoperability, namely semantic interoperability. This thesis, alongside the standards that it spotlights, purposefully only encompasses the aspect of technical interoperability, and neglects any of the other levels. Semantic interoperability among data space participants, both within the same and across different data spaces, remains an open issue that leads to many discussions within the data space world, such as standardizations of ontologies and vocabularies for data sharing across sectors.

References

- Alizadeh, M., Andersson, K., & Schelen, O. (2022). Performance Analysis of Verifiable Data Registry Solutions for Decentralized Identifiers. 2022 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), Gold Coast, Australia, 2022, pp. 1-8, doi: 10.1109/CSDE56538.2022.10089278.
- Amaral, G., Guizzardi, R., Guizzardi, G., & Mylopoulos, J. (2020). Ontology-Based Modeling and Analysis of Trustworthiness Requirements: Preliminary Results. In: Dobbie, G., Frank, U., Kappel, G., Liddle, S.W., Mayr, H.C. (eds) Conceptual Modeling. ER 2020. Lecture Notes in Computer Science, vol 12400, pp. 342-352. Springer, Cham. https://doi.org/10.1007/978-3-030-62522-1_25.
- Andrieu, J., Hardman, D., Appelcline, S., Guy, A., Lohkamp, J., Reed, D., Sabadello, M., & Terbu, O. (2021). DID Method Rubric v1.0. World Wide Web Consortium (W3C) Group Note. Retrieved 12 March 2024 from <https://www.w3.org/TR/did-rubric/>.
- Bistarelli, S., Micheli, F., & Santini, F. (2023). A Survey on Decentralized Identifier Methods for Self Sovereign Identity. Proceedings of the Italian Conference on Cyber Security (ITASEC 2023), Bari, Italy, May 2-5, 2023. <https://ceur-ws.org/Vol-3488/paper05.pdf>.
- Brunner, C., Gallersdorfer, U., Knirsch, F., Engel, D., & Matthes, F. (2021). DID and VC: Untangling Decentralized Identifiers and Verifiable Credentials for the Web of Trust. In Proceedings of the 2020 3rd International Conference on Blockchain Technology and Applications (ICBTA '20). Association for Computing Machinery, New York, NY, USA, 61–66. <https://doi.org/10.1145/3446983.3446992>.
- Cunningham, C., Franco, D. C., & Grant, R. (2022). Is This DID Method Ready to be Endorsed? Useful Rubric Criteria. Rebooting the Web of Trust 11 Draft Document. Retrieved 14 March 2024 from <https://github.com/WebOfTrustInfo/rwot11-the-hague/blob/master/draft-documents/is-my-did-method-ready-to-endorse.md>.
- Curran, S. (2024). NEW!! The Last DID Method = Trust DID Web did:tdw. In: Saul, H., Windley, E. (eds) Internet Identity Workshop 38: Book of Proceedings, Mountain View, CA, USA, April 16-18, 2024. Retrieved 26 April 2024 from https://github.com/windley/IIW_homepage/blob/6b0308f951ce7bcd14efea3c53e553317d78d483/assets/proceedings/IIW38_Book_of_Proceedings.pdf?raw=true.
- Curry, E., Scerri, S., & Tuikka, T. (2022). Data Spaces: Design, Deployment, and Future Directions. In: Curry, E., Scerri, S., Tuikka, T. (eds) Data Spaces. Springer, Cham. https://doi.org/10.1007/978-3-030-98636-0_1
- de Kok, W., & van Leuken, M. (2023). Digital identity wallets: a technical overview. TNO ICT, Strategy & Policy. Retrieved from <https://publications.tno.nl/publication/34641566/4ZIT6d/kok-2023-digital.pdf>.
- Dickinson, P., & Adams, J. (2017). Value in evaluation – The use of rubrics. Evaluation and Program Planning, Volume 65, pp. 113-116. <https://doi.org/10.1016/j.evalprogplan.2017.07.005>.

ECMA International. (2017). ECMA-404: The JSON Data Interchange Standard. 2nd Edition, December 2017. Retrieved 15 September 2024 from <https://ecma-international.org/publications-and-standards/standards/ecma-404/>.

European Commission. (2017). New European Interoperability Framework: Promoting seamless services and data flows for European public administrators. European Union: Luxembourg. Retrieved 9 September 2024 from https://ec.europa.eu/isa2/sites/default/files/eif_brochure_final.pdf.

European Parliament. (2023). Boosting data sharing in the EU: what are the benefits?. European Parliament. Retrieved 26 February 2024 from <https://www.europarl.europa.eu/topics/en/article/20220331STO26411/boosting-data-sharing-in-the-eu-what-are-the-benefits>.

Fdhila, W., Stifter, N., Kostal, K., Saglam, C., & Sabadello, M. (2021). Methods for Decentralized Identities: Evaluation and Insights. In: Gonzalez Enriquez, J., Debois, S., Fettke, P., Plebani, P., van de Weerd, I., Weber, I. (eds) Business Process Management: Blockchain and Robotic Process Automation Forum. BPM 2021. Lecture Notes in Business Information Processing, vol 428. Springer, Cham. https://doi.org/10.1007/978-3-030-85867-4_9.

Giussani G., & Steinbuss S. (2024). Data Connector Report. International Data Spaces Association, (13), March 2024. <https://doi.org/10.5281/zenodo.10869525>.

Graham, G., & Goldscheider, D. (2023). Why the World Needs an Open Source Digital Wallet Right Now. OpenWallet Foundation in partnership with The Linux Foundation Research. Retrieved from <https://project.linuxfoundation.org/hubfs/LF%20Research/OpenWallet%20Open%20Digital%20Wallet%20-%20Report.pdf>.

Gribneau, C., Prorock, M., Steele, O., Terbu, O., Xu, M., & Zagidulin, D. (2023). did:web Method Specification. World Wide Web Credentials Community Group (W3C-CCG). Retrieved 3 March 2024 from <https://w3c-ccg.github.io/did-method-web/>.

Hoops, F., Muehle, A., Matthes, F., & Meinel, C. (2023). A Taxonomy of Decentralized Identifier Methods for Practitioners. 2023 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS), Athens, Greece, 2023 pp. 57-65, doi: 10.1109/DAPPS57946.2023.00017.

IEEE. (2017). IEEE 1012-2016: IEEE Standard for System, Software, and Hardware Verification and Validation. Retrieved 19 September 2024 from <https://standards.ieee.org/ieee/1012/5609/>.

International Data Spaces Association. (2023a). How to Build Data Spaces? - 1. Gather knowledge. Retrieved 26 February 2024 from <https://docs.internationaldataspaces.org/ids-knowledgebase/v/how-to-build-data-spaces/>.

International Data Spaces Association. (2023b). Dataspace Protocol – ensuring data space interoperability. Retrieved 5 March 2024 from <https://internationaldataspaces.org/dataspace-protocol-ensuring-data-space-interoperability/>.

International Data Spaces Association. (2023c). Your guide to data spaces: From the IDS-RAM to the Dataspace Protocol. Retrieved 5 March 2024 from

<https://internationaldataspaces.org/your-guide-to-data-spaces-from-the-ids-ram-to-the-dataspace-protocol/>.

International Data Spaces Association. (2024a). IDSA releases stable version of the Dataspace Protocol. Retrieved 5 March 2024 from <https://internationaldataspaces.org/idsa-releases-stable-version-of-the-dataspace-protocol/>.

International Data Spaces Association. (2024b). Dataspace Protocol 2024-1. Retrieved 5 March 2024 from <https://docs.internationaldataspaces.org/ids-knowledgebase/v/dataspace-protocol>.

International Data Spaces Association. (n.d.a). Dataspace Protocol Overview - Advancing interoperability: the Dataspace Protocol. Retrieved 5 March 2024 from <https://internationaldataspaces.org/offers/dataspace-protocol-overview/>.

International Data Spaces Association. (n.d.b). Data Spaces: Where the future of data happens. Retrieved 19 September 2024 from <https://internationaldataspaces.org/why/data-spaces/>.

JSON Lines. (n.d.). JSON Lines: Documentation for the JSON Lines text file format. Retrieved 18 September 2024 from <https://jsonlines.org/>.

Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-Defined Networking: A Comprehensive Survey. In Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, Jan. 2015, doi: 10.1109/JPROC.2014.2371999.

Kubach, M., Schunck, C. H., Sellung, R., & Rossnagel, H. (2020). Self-sovereign and Decentralized identity as the future of identity management?. Open Identity Summit 2020, Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020, 35.

Li, W., Lemieux, Y., Gao, J., Zhao, Z., & Han, Y. (2019). Service Mesh: Challenges, State of the Art, and Future Research Opportunities. 2019 IEEE International Conference on Service-Oriented System Engineering (SOSE), San Francisco, CA, USA, 2019, pp. 122-1225, doi: 10.1109/SOSE.2019.00026.

Nagel, L., & Lycklama, D. (2021). Design Principles for Data Spaces. OPEN DEI Position Paper. Version 1.0. Belin. <https://h2020-demeter.eu/wp-content/uploads/2021/05/Position-paper-design-principles-for-data-spaces.pdf>.

Otto, B. (2022a). A federated infrastructure for European data spaces. Communications of the ACM. 65. pp. 44-45. 10.1145/3512341.

Otto, B. (2022b). The Evolution of Data Spaces. In: Otto, B., ten Hompel, M., Wrobel, S. (eds) Designing Data Spaces. Springer, Cham. https://doi.org/10.1007/978-3-030-93975-5_1.

Otto, B., Steinbuss, S., Teuscher, A., & Bader, S. (2022). International Data Spaces – Reference Architecture Model 4 (IDS-RAM 4). Retrieved 26 February 2024 from <https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/>.

Pettenpohl, H., Spiekermann, M., & Both, J. R. (2022). International Data Spaces in a Nutshell. In: Otto, B., ten Hompel, M., Wrobel, S. (eds) Designing Data Spaces. Springer, Cham. https://doi.org/10.1007/978-3-030-93975-5_3

Preukschat, A., & Reed, D. (2021). Self-Sovereign Identity. Manning Publications, 9781638351023.

Robotics & Automation News. (2021). How has Data Become the World's Most Valuable Commodity?. Robotics & Automation News. Retrieved 20 February 2024 from <https://roboticsandautomationnews.com/2021/07/22/how-has-data-become-the-worlds-most-valuable-commodity/44267/>.

Shvets, A. (2021). Dive into Design Patterns. Refactoring.Guru.

Sporny, M., Longley, D. & Chadwick, D. (2022a). Verifiable Credentials Data Model v1.1. World Wide Web Consortium (W3C) Recommendation. Retrieved 27 February 2024 from <https://www.w3.org/TR/vc-data-model/>.

Sporny, M., Longley, D., Sabadello, M., Reed, D., Steele, O., & Allen, C. (2022b). Decentralized Identifiers (DIDs) v1.0. World Wide Web Consortium (W3C) Recommendation. Retrieved 26 February 2024 from <https://www.w3.org/TR/did-core/>.

Sporny, M., Longley, D., Kellogg, G., Lanthaler, M., Champin, P., & Lindström, N. (2020). JSON-LD 1.1. World Wide Web Consortium (W3C) Recommendation. Retrieved 15 September 2024 from <https://www.w3.org/TR/json-ld/>.

Steele, O., & Sporny, M. (2023). DID Specification Registries. World Wide Web Consortium (W3C) Group Note. Retrieved 26 February 2024 from <https://www.w3.org/TR/did-spec-registries/>.

Steinbuss, S. (2023). IDSA Tech Talk | Unveiling the Dataspace Protocol. YouTube, uploaded by International Data Spaces Association, 30 November 2024, <https://www.youtube.com/watch?v=zZHqH5zN1Ac>.

TNO. (n.d.). Data sharing: the key to further digitalization. Retrieved 28 February 2024 from <https://www.tno.nl/en/digital/digital-innovations/data-sharing/data-sharing-digitalisation/>.

Trust Over IP Foundation. (2023). Announcing Public Review of the did:webs Method Specification. Retrieved 20 March 2024 from <https://trustoverip.org/news/2023/12/15/announcing-public-review-of-the-didwebs-method-specification/>.

van den Wall Bake, A., Jansen, V., Kluiters, L., & Travinka, Y. (2024). Impact of eIDAS revision and EU Digital Identity landscape on data spaces development. Whitepaper by Centre of Excellence for Data Sharing and Cloud. <https://coe-dsc.nl/wp-content/uploads/2024/01/CoE-DSC-eIDAS-impact-on-data-spaces.pdf>

Wieringa, R. J. (2014). Design Science Methodology for Information Systems and Software Engineering. Springer. <https://doi.org/10.1007/978-3-662-43839-8>.

Yasuda, K., Lodderstedt, T., Chadwick, D., Nakamura, K., & Vercammen, J. (2022). OpenID for Verifiable Credentials: A Shift in the Trust Model Brought by Verifiable Credentials. OpenID Foundation. Retrieved 17 March 2024 from https://openid.net/wordpress-content/uploads/2022/06/OIDF-Whitepaper_OpenID-for-Verifiable-Credentials-V2_2022-06-23.pdf.

Appendix A: Timeline of Research Topics & Final Project

The GANTT chart in Figure 35 displays the overall timeline plan of the entire thesis. The Practical-related and Paper-related cell rows highlighted in darker blue show how the work related to the practical implementation and thesis paper writing are split respectively. The cell row highlighted in yellow and green show the separation between the work that will be carried out in the Research Topics and Final Project. Despite the formation of this plan, there are possibilities that the actual timeline will deviate from it, depending on the real progress of research and implementation.

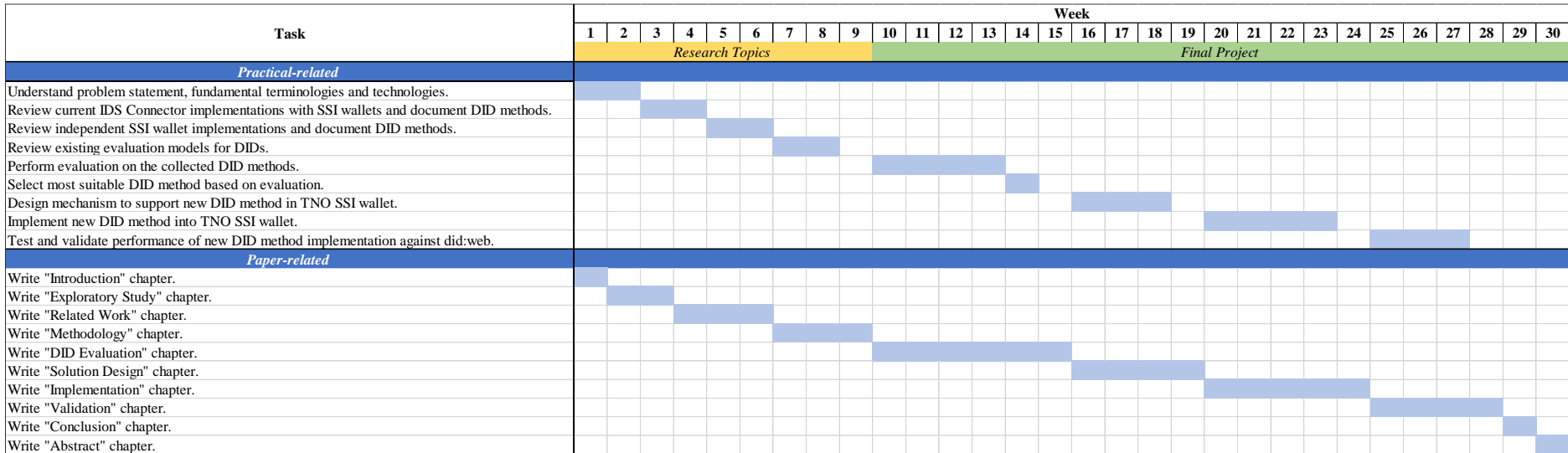


Figure 35: GANTT chart of Research Topics & Final Project

Appendix B: DID Method Evaluation Rubric & Scoring Sheet

Criteria	Score			
	4	3	2	1
Rulemaking				
<p>Open contribution (participation)</p> <p>How open is participation in governance decisions?</p>	Anyone can participate in an open, fair process where all participants have equal opportunity to be heard and influence decisions.	Anyone can comment and contribute to open debate, but decisions are ultimately made by a closed group.	Debate is restricted to a selected but known group.	Debate is conducted in secret by an unknown group.
<p>Transparency</p> <p>How visible are rulemaking processes?</p>	Agendas and participation details for all meetings are publicly announced, the meetings are broadcast in real-time to any listeners, and all minutes and recordings are captured in real-time and publicly reviewable in perpetuity.	Minutes of meetings are reviewable by the public, including all votes and who cast them, but real-time observation may be limited.	All current rules are publicly available.	Rules may be changed without public notice.
<p>Breadth of Authority</p> <p>How many independent parties actively participate in the governance authority?</p>	Rules are decided by an open set of multiple parties.	Rules are decided by a closed set of multiple parties.	Rules are decided by a single known entity.	Rules are decided by an unknown party.

<p>Public vs private economies</p> <p>How privatized is the economic interest of the governing authority?</p>	<p>The governing authority is established for the public good without rents or remuneration.</p>	<p>The governing authority is established for the common good of a limited set of parties.</p>	<p>The governing authority is established to enhance profits for a limited set of parties.</p>	<p>The governing authority is established to extract rents.</p>
<p>Cost</p> <p>How expensive is it to participate in governance (in time, money, or effort)?</p>	<p>Free to all.</p>	<p>Inexpensive, but accessible.</p>	<p>Modest cost for interested parties.</p>	<p>Expensive and restricted/Not possible to participate.</p>
Design				
<p>Permissioned operation</p> <p>To use the DID Method, do you need permission?</p>	<p>Anyone can participate fully (full read/write and participation in consensus).</p>	<p>Anyone can read/write, but consensus mechanism is permissioned.</p>	<p>Anyone can read, but writing and consensus is permissioned.</p>	<p>All participation is permissioned.</p>
<p>Interoperability</p> <p>Does the DID Method restrict access or functionality to particular wallet implementations per the specification?</p>	<p>Any wallet can work with any resolver on any registry.</p>	<p>Any wallet can work with multiple resolvers and multiple registries.</p>	<p>Some implementations of some wallets can work with some resolvers.</p>	<p>There is a single combined suite of resolver, registry, and wallet.</p>

<p>Scope of usage</p> <p>How widely can DIDs of this Method be used?</p>	<p>Universal: DIDs can only be created and used universally, between any number of parties.</p>	<p>Contextual: DIDs can be created and used contextually, between any set of collaborating parties.</p>	<p>Paired: DID can be created and used pairwise, between any two parties.</p>	<p>Central: DIDs can only be created and used with a single, centralized party.</p>
Operation				
<p>Financial accountability</p> <p>How transparent and fair are the economics of the Method?</p>	<p>All operational finances are transparent and accounted for.</p>	<p>Compensation for primary operators is transparent.</p>	<p>Some financial flows are visible.</p>	<p>Operation is privatized with no visibility.</p>
Enforcement				
<p>Auditability</p> <p>Who can retrieve cryptographic proof of the history of changes to a given DID Document?</p>	<p>Anyone.</p>	<p>Only a select group, including parties not involved in a given DID transaction.</p>	<p>Only parties to the transaction.</p>	<p>Not available.</p>
Security				
<p>Robust Crypto</p> <p>What is the lowest security level provided by the combination of algorithms and key types that the method requires its</p>	<p>No combination of required features produces a profile with less than 256 bits of security.</p>	<p>Between 128 and 256 bits. (Conventional wisdom — NIST recommendation — says that this level of security is adequate until the next revolutionary breakthrough.)</p>	<p>Less than 128 bits. (NIST recommends replacing this security by 2030.)</p>	<p>Less than 112 bits. (Security is obsolete today.)</p>

implementations to support?				
<p>Expert Review</p> <p>Does the system use cryptographic and security primitives that are well vetted by technical experts, and battle hardened in the school of experience?</p>	<p>Experts generally consider the system very secure, and this opinion is reinforced by a track record of secure production use.</p>	<p>The theoretical security of the system looks excellent, and no known attacks or substantive criticisms are unaddressed. However, limited review or limited experience informs the opinion.</p>	<p>Credible reports of vulnerabilities or design shortcomings have not been addressed.</p>	<p>The system actively uses mechanisms that are officially deprecated.</p>
<p>Future Proofing</p> <p>How friendly is the system to adopting post-quantum crypto, larger hashes, or other measures that upgrade its security?</p>	<p>Any user of the system can easily upgrade their crypto at any time.</p>	<p>No code changes are needed, but the whole system needs to be reconfigured to allow new crypto.</p>	<p>Code changes must be implemented before new crypto is possible.</p>	<p>Code changes must be implemented, and migration of all existing data must be performed, before new crypto is possible.</p>
<p>Self Certification</p> <p>To what extent is the entropy used to create an identifier demonstrably connected to the party that created its inception key?</p>	<p>Identifier entropy is directly derived from inception keys. Initial pre-rotation preserves this quality.</p>	<p>Identifiers are assigned by a VDR rather than chosen by users. The VDR uses a cryptographically robust RNG. This creates a chain of custody that's as strong as the VDR's integrity.</p>	<p>Identifiers are related to inception keys with some caveats, or they are assigned by a VDR using an algorithm that has vulnerabilities.</p>	<p>Identifiers are arbitrary. They have no chain of custody before they are registered on a VDR. DID squatting is possible.</p>

<p>Availability</p> <p>How robust are protections against attempts to suppress information flow, whether legal (cease and desist) or technical (denial of service)?</p>	<p>The VDR is practically immune from this risk.</p>	<p>The VDR has reasonable protections in place. However, motivated and well resourced attackers could temporarily disrupt access in a targeted context.</p>	<p>Attackers could permanently disrupt access in a targeted context.</p>	<p>—</p>
<p>Evolution</p> <p>Is the current state of a DID document provably correct from a history that's visible to anyone who can resolve the DID?</p>	<p>Every evolution of state is recorded, accessible, and linked appropriately to its predecessor. Arbitrary versions can be queried and proved correct, and they have a reasonably useful timestamp.</p>	<p>Adequate evidence of proper evolution exists, and a forensic analysis could prove correctness. However, it's not exposed for consumption of ordinary resolvers, it lacks supporting metadata, or it's exposed in a very suboptimal way.</p>	<p>Limited evidence of proper evolution exists.</p>	<p>No evidence of proper evolution exists; the users have to trust the system's assertion that current state resulted from something appropriate.</p>
<p>Many Eyes</p> <p>Is the code of the method published, does it have many contributors, and does it have a published vulnerability reporting (responsible disclosure) mechanism?</p>	<p>The code is public. It has hundreds of contributors. CVEs or similar reports have been published and handled appropriately.</p>	<p>The code is public, but the list of contributors is small. No vulnerability reporting mechanism has been announced, or it's been announced but has no demonstrable track record.</p>	<p>The code is partly private.</p>	<p>The code is entirely private.</p>

<p>Regulatory Compliance</p> <p>Does the system use cryptographic mechanisms that satisfy legal requirements in relevant jurisdictions (e.g., FIPS-certified algorithms, requirements for encryption back doors, etc.)?</p>	<p>The method uses only algorithms that are officially endorsed by whatever regulatory bodies are important to the reviewer. For example, a reviewer in the USA might assign this score if cryptography is FIPS-approved.</p>	<p>The method uses algorithms that are ignored rather than endorsed by the relevant authority -- or uses a combination of algorithms with the practical outcome that interop is unlikely using only approved settings.</p>	<p>The method uses cryptography that is not aligned with requirements of the relevant authority.</p>	<p>—</p>
Privacy				
<p>Per-DID constraints on visibility</p> <p>What provisions are made for restricting visibility of DIDs to audiences other than the general public?</p>	<p>Fully private is possible.</p>	<p>VDR is visible to "all", but "all" is a restricted audience with enforced terms of service or other disincentives to abuse.</p>	<p>VDR is fully public with no constraints.</p>	<p>—</p>
<p>Cross-DID Leakage</p> <p>How possible is it to control multiple DIDs, without having an observer of one DID be able to deduce that another DID has the same controller?</p>	<p>Each DID is entirely isolated; there is no practical way to infer relationships of common control between them.</p>	<p>It may be possible to infer relationships of common control based on timing and the IP address of clients, but not based on data persisted in the VDR itself. Inference is expensive or impractical.</p>	<p>It is possible to infer relationships based on data persisted in the VDR itself, and the inference is easy.</p>	<p>—</p>

<p>Revocation of Consent</p> <p>How does one revoke consent for the storage of a DID?</p>	<p>The VDR fully supports removal of data that no longer has consent. Once something is removed, nothing internal to the system allows it to be recovered, and the system no longer qualifies as a data controller.</p>	<p>The DID method only supports a current view, and the state of an identifier in its VDR can be updated to a deleted or removed state. Thus, the identifier ceases to resolve. However, a permanent record of the previous state of the identifier remains, and could be used to reconstruct historical data. The VDR thus remains a data controller for deleted identifiers, at least in concept.</p>	<p>The DID method supports both current and previous views of an identifier. While deletes are possible, the old state of an identifier is a fully supported feature, so adding the identifier to the VDR is irrevocable.</p>	<p>–</p>
--------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------

Table 14: Resulting DID Methods Evaluation Rubric for this Thesis Study

DID Method		
Specification URL		
Current status of specification		
DID Specification Registries		
Underlying technology		
Criteria	Evaluation	Score
Open contribution (participation)		
Transparency		
Breadth of Authority		
Public vs private economies		
Cost		
Permissioned operation		
Interoperability		
Scope of usage		
Financial accountability		
Auditability		
Robust Crypto		
Expert Review		
Future Proofing		
Self Certification		
Availability		
Evolution		
Many Eyes		
Regulatory Compliance		
Per-DID constraints on visibility		
Cross-DID Leakage		
Revocation of Consent		
Total Score		

Table 15: Resulting Scoring Sheet for DID Methods Evaluation

Appendix C: DID Method Evaluation Results using Evaluation Rubric

C.1. did:webs

DID Method	did:webs	
Specification URL	https://trustoverip.github.io/tswg-did-method-webs-specification/	
Current status of specification	Draft (Implementors Draft v0.9.15)	
DID Specification Registries?	Yes	
Underlying technology	Publishes a DID Document under an existing web domain that is discoverable by DNS, additionally uses Key Event Receipt Infrastructure (KERI) to provide a secure chain of cryptographic key events.	
Criteria	Evaluation	Score
Open contribution (participation)	The specification for did:webs is maintained by the Trust Over IP Foundation, which consists of many steering members and contributors (both organizational and individual) worldwide. Anyone can participate in contributing to the specifications as it is published on a public GitHub repository, however final decisions are still governed by the Trust Over IP Foundation.	3
Transparency	As of writing, there are no indications of any regular meetings to discuss proposed changes to the specification. Contributions are proposed via GitHub open issues and pull requests, where the development can be audited.	3
Breadth of Authority	In the end, approval of contributions are made by members of the Trust Over IP Foundation, which consists of a diverse set of parties.	3
Public vs private economies	The Trust Over IP Foundation, under the Joint Development Foundation, is a not-for-profit community that aims to contribute to the community to build frameworks for digital trust on the internet.	4
Cost	Participating to the specifications via the public GitHub does not require any fees, but takes time and effort. Creating the DID can be considered cost-free as the only additional expenses is to maintain a web domain and web server, however most organizations implementing DID methods would already have this in place. Additionally, the cost of implementing KERI is free.	4
Permissioned operation	There are no permissions that restrict the usage of did:webs. As long as a web domain is owned, a did:webs DID can be created by hosting the DID and KERI related files. Any party can then also retrieve the DID Document without any restrictions.	4
Interoperability	No specific dependencies restrict did:webs from being interoperable, as it is based on the web. Moreover, did:webs is interoperable with did:web through performing certain conversion methods.	4

Scope of usage	As it is hosted on a web domain, the usage of did:webs is universal. Moreover, KERI itself also has means to restrict communications to specifically for pairwise.	4
Financial accountability	The financials of parties are not shared with each other for did:webs.	1
Auditability	The public keri.cesr file stores all change history of cryptography used in the DID Document, and it is viewable by anyone accessing the path.	4
Robust Crypto	When relying solely on SSL/TLS certificates, there security level ranges from 112 to 256 bits, however since did:webs depends on KERI for additional security, KERI currently supports cryptographic algorithms where the minimum security level is 128 bits.	3
Expert Review	The security of did:webs relies on KERI, which incorporates well-vetted cryptographic algorithms. However, KERI itself is still new and there is limited experience in its usage.	3
Future Proofing	As KERI currently only supports 3 cryptographic algorithms, code changes must be performed. However, the KERI protocol is built so that other cryptographic algorithms can be easily integrated.	2
Self Certification	The method specific identifier is composed of two parts, firstly an arbitrary web domain name of choice by the DID owner, and the second part a KERI autonomic identifier that is generated using cryptography on the KERI key event logs.	2
Availability	Generally, security by web certificates are regarded as sufficient. did:webs takes this to another level by incorporating KERI, reducing malicious risks, yet not entirely preventing them.	3
Evolution	All past states of the cryptographic keys used in the DID Document (did.json) are stored and linked (both forwards and backwards) in the KERI file.	4
Many Eyes	The code is publicly available on GitHub, however the number of contributors is still not large. Vulnerability reporting is performed through raising issues on GitHub.	3
Regulatory Compliance	Evidence exists that did:webs attempts to comply with EU regulations, for example by using KERI which abides by the EU GDPR.	4
Per-DID constraints on visibility	Since did:webs relies on discoverability via the web, it is public with no restrictions on visibility. However, similarly to did:web, its discoverability can be limited by using additional web mechanisms such as whitelisting or blacklisting IP addresses.	3
Cross-DID Leakage	Although the KERI portion of did:webs is derived from random inception keys, the web portion still relies on a web domain which is identifiable. If a DID owner has two DIDs that are both stored under the same web domain, there is a possibility that someone can identify that both DIDs belong to the same owner.	3
Revocation of Consent	It is not advised to remove the DID and KERI files from the web server as it will raise confusion on whether the DID has	3

	been revoked or whether the web server is currently offline. Instead, deactivating the DID is possible by rotating the KERI keys to null and regenerating a new DID Document.	
Total Score		67

Table 16: Resulting Evaluation Scoring Sheet of did:webs

C.2. did:tdw

DID Method	did:tdw (Trust DID Web)	
Specification URL	https://bcgov.github.io/trustdidweb/	
Current status of specification	Draft	
DID Specification Registries?	No	
Underlying technology	Publishes a JSON Lines (JSONL) file, containing a verifiable history of DID Logs equipped with a self-certifying identifier (SCID), entry hash and data integrity mechanisms, under an existing web domain that is discoverable by DNS.	
Criteria	Evaluation	Score
Open contribution (participation)	The did:tdw specification is owned by the Government of British Columbia (BCGov) team. The source code is publicly available on GitHub, where anyone can open issues and create pull requests to contribute to the specification. Final decisions on contribution acceptances are governed by the BCGov team.	3
Transparency	There seems to be no live meetings for discussions about did:tdw. All discussions are carried out via GitHub, and the working discussions can be observed from GitHub issues and pull requests.	3
Breadth of Authority	Despite being open to contributions by any party, the final decisions regarding rule making for the specification is carried out by members of the BCGov team.	2
Public vs private economies	Although supported under a country's government body, the goal of the BCGov team is to enable open-source solutions to developers from any sector, without any financial incentives.	4
Cost	Participating in governance of the did:tdw is free, but it costs the contributors time and effort. Similarly to other web-based DID methods, there are fees for maintaining a web domain and server, however can be neglected because most organizations will already have it setup. The additional mechanisms and standards that did:tdw uses, such as JSONL, SCID and data integrity proofs, do not involve any fees.	4
Permissioned operation	Anyone can fully participate in all operations of a did:tdw DID. There are no specific permissions put in place, as long as a domain name is already owned. The additional security mechanisms also do not restrict certain criteria for them to be used.	4
Interoperability	As it is hosted on the web, there are no specific technology restrictions for creating and using did:tdw. All technologies are able to work with the mechanisms performed for did:tdw. A corresponding did:web DID can also be created alongside a did:tdw DID, allowing for backwards compatibility, though the did:web will have none of the security mechanisms guaranteed by the did:tdw specification.	4

Scope of usage	As did:tdw relies on a web domain, it can practically be created and used universally by anyone worldwide, as long as there is an internet connection.	4
Financial accountability	By default, there is no visibility regarding the financial accountability of DID owners and receivers.	1
Auditability	did:tdw retains a DID log with cryptographic proof in the form of a JSONL file which can be accessed under the correct path based on the DID.	4
Robust Crypto	Since did:tdw is based on web technology, its main security guarantee relies on the web domain's SSL/TLS certificate, whose bits of security range between 112 to 256. However, the additional mechanisms carried out to improve security, namely SCID, entry hash, and data integrity, are required to use cryptographic algorithms with at least 128 bits of security.	3
Expert Review	did:tdw defines security-ensuring processes that are widely acknowledged in web technologies, meaning that these mechanisms are generally regarded as secure and well-vetted by experts.	4
Future Proofing	Only one suite of cryptographic algorithms are currently supported by did:tdw, hence code changes will be needed. However, the specification owners do advise a mechanism for using pre-rotation keys to avoid post-quantum attacks.	2
Self Certification	The did:tdw method specific identifier is built using two values, one part from a random web domain of choice by the DID owner, and the other part from an SCID generated from the initial DID Document.	2
Availability	did:tdw for the most part relies on security from the web, which is generally strong but could be exposed by malicious actors with attacks such as MitM. Regarding protections for the DID Document, the security mechanisms in place, such as SCID, entry hash, and data integrity, provide more security than just plainly relying on the DNS and the web.	3
Evolution	The current DID Document is extracted from the did.jsonl JSONL file, which actually keeps a record of each version of the changes to the DID Document in the form of DID logs.	4
Many Eyes	The code for did:tdw is available publicly on GitHub, however there are still only a handful of contributors as it is still a growing specification. Creating GitHub issues can be considered as vulnerability reporting.	3
Regulatory Compliance	Though not explicitly stated, there is no reason for did:tdw to not comply with regulations as it depends on web technology.	4
Per-DID constraints on visibility	By default, the visibility of did:tdw DIDs are public as they are hosted on the web. However, actions can be taken to restrict access, such as whitelisting or blacklisting certain IP addresses.	3
Cross-DID Leakage	Although the SCID is added as a part of the did:tdw DID, a portion of the method specific identifier still utilizes web domains. Different did:tdw DIDs stored under different	3

	paths of the same web domain could potentially be identified as coming from the same owner.	
Revocation of Consent	The did:tdw specification explains the possibility of deactivating a DID by adding a “deactivated: true” field in the last entry of the DID logs of the JSONL file. The DID owner may also remove the cryptography keys from the DID Document to show that it is no longer supposed to be used to perform other DID operations, however this is optional.	3
Total Score		67

Table 17: Resulting Evaluation Scoring Sheet of did:tdw

C.3. did:key

DID Method	did:key	
Specification URL	https://w3c-ccg.github.io/did-method-key/	
Current status of specification	Draft (Unofficial Draft v0.7 – 02 Sep 2022)	
DID Specification Registries?	Yes	
Underlying technology	Utilizes static cryptographic keys and expands them into DID Documents without an actual VDR.	
Criteria	Evaluation	Score
Open contribution (participation)	did:key is maintained by the W3C Credentials Community Group. Although there is possibility for anyone to contribute to the specifications through the process of opening issues and pull requests on GitHub or initiating discussions on their mailing list, all decisions are still ultimately made by members of the W3C CCG.	3
Transparency	All communications regarding the specifications are carried out via open issues and pull requests, and are openly accessible via GitHub. However, there is limited possibility to audit it in actual real-time.	3
Breadth of Authority	Although there currently exists 7 contributors to the specification, 6 of them are part of the same organization (Digital Bazaar). Hence, it can be considered as being ruled by one single known entity despite the specification being under control of the W3C CCG.	2
Public vs private economies	Despite being governed under the W3C CCG, the did:key specification is still majority maintained by a single organization, hence it can be considered that the specification was initially built to support their organization. However, it does not seem that the organization attempts to extract any profits from it.	3
Cost	It is cost-free to participate to the specifications on GitHub, however it will take time and effort. There are zero costs involved to create and maintain the DID as it relies on static cryptographic keys.	4
Permissioned operation	Anyone is able to generate and utilize cryptographic key pairs without any restrictions.	4
Interoperability	Static cryptographic keys can work with any technology, hence there are no restrictions in interoperability.	4
Scope of usage	Any party can create cryptographic keys and use it as their DIDs. Moreover, did:key does not rely on a VDR but instead performs a generative process each time it is resolved. This means that once a DID is shared, the DID Document can be resolved (generated) universally.	4
Financial accountability	The financial accountability of operators of the DID are not transparent and unknown since there is no VDR of some sort.	1
Auditability	Auditability is not possible because the DID Document of did:key cannot be updated. Since did:key is a generative DID method, changing the keys shown in the DID	1

	Document means changing the key used in the DID itself, thus creating a new DID altogether.	
Robust Crypto	The security level of did:key relies on the algorithm used to create the cryptographic key pair. Although there is no strict mandatory requirement, the DID specification lists several recommended algorithms which range in security level from 112 to 256 bits.	2
Expert Review	The variety of cryptographic methods recommended by did:key are generally recognized as secure and well-implemented in production. However, since the choice of cryptographic algorithm is in the hands of the DID owner, and they could decide to use a cryptographic algorithm that is not popular yet.	3
Future Proofing	Since did:key only relies on the DID to independently generate the DID Document, any alternative cryptography can easily be adopted.	4
Self Certification	A did:key identifier is directly created by using the cryptographic public key, hence the DID is clearly connected with the inception keys.	4
Availability	There is technically no VDR for did:key as it is a generative DID method. Despite this, it is not immune to risks from security compromise. Moreover, since did:key does not support update and revoke, once the key used in the DID is compromised, the DID will be permanently unrecoverable.	2
Evolution	The DID Document is not updatable in did:key, hence it is not possible to view any evolution history.	1
Many Eyes	The did:key specification code is public, but has only few contributors. Vulnerabilities are tracked as open issues on GitHub.	3
Regulatory Compliance	did:key does not specifically adhere to any authority as it is meant to be used for local and temporary environments.	2
Per-DID constraints on visibility	Since did:key relies on cryptographic key pairs, it does not rely to any public VDR and is hence completely private besides to those the DID is shared to.	4
Cross-DID Leakage	Each generated cryptographic key pairs are isolated from one another and cannot be inferred as they have high entropy.	4
Revocation of Consent	Since did:key is purely generative, it is not possible to deactivate the DID and DID Document. Once created, the DID is permanent.	1
Total Score		59

Table 18: Resulting Evaluation Scoring Sheet of did:key

C.4. did:jwk

DID Method	did:jwk	
Specification URL	https://github.com/quartzjer/did-jwk/blob/main/spec.md	
Current status of specification	Release (Version 1.0 – 15 Feb 2023)	
DID Specification Registries?	Yes	
Underlying technology	Deterministic transformation of JSON Web Key (JWK) to DID Document.	
Criteria	Evaluation	Score
Open contribution (participation)	The did:jwk specification is owned and maintained not by an organization, but instead by a single person. Suggestions and contributions are openly available on GitHub via issues and pull requests, however the maintainer of the specification has the final say.	3
Transparency	It is possible to observe all discussions through issues and pull requests opened on GitHub, however it is not performed in real-time as live updates are not shared.	3
Breadth of Authority	Though open-source, the rules are ultimately decided by a single known entity (the only maintainer and code owner).	2
Public vs private economies	The did:jwk specification is created and maintained by an individual personnel, hence albeit currently seeming to be built for the public good, there is more possibility that deviation from this goal can occur in the future.	3
Cost	It is cost-free to participate to the specifications on GitHub, however it will take time and effort. There are zero transaction costs to create and maintain the DID since it is based on local JWKs.	4
Permissioned operation	All operations are permissionless as anyone is able to generate and use JWKs without any restrictions.	4
Interoperability	There are no restrictions in the technology that is able to work with JWKs.	4
Scope of usage	did:jwk DIDs can be created and used for universal purposes, as long as it is shared to the designated entities.	4
Financial accountability	The financial accountability of operators of the DID are not transparent and unknown as there is no real VDR.	1
Auditability	Updates to a DID Document is not supported in did:jwk, hence it is not possible to even retrieve any change history, let alone its cryptographic proof.	1
Robust Crypto	JWK is only a structural definition of representing a cryptographic key, it does not specify the cryptographic algorithms that can be used. Hence, did:jwk also does not specify it. This decision is determined by the DID owner. The specification does, however, provide examples with cryptographic algorithms that provide 128 bits of security.	2
Expert Review	Both did:jwk and JWK do not enforce particular cryptographic algorithms that must be used, hence it is quite difficult to make a decision on this. In the example provided on the did:jwk specification, they use algorithms that are well-vetted and considered as secure.	3

Future Proofing	Alternative cryptography can easily be integrated flexibly with did:jwk in the future since the choice is made by the DID owner.	4
Self Certification	The actual did:jwk identifier is directly created by using an encoding of the JWK, hence they are closely tied.	4
Availability	did:jwk essentially does not have a VDR, as it is a generative DID method that generates the DID Document based on the DID each time it is resolved. This disallows the possibility of updating the DID Document, which will change the DID itself. Hence, permanent negative effects could occur if the JWK is compromised.	2
Evolution	It is not possible to make updates for did:jwk without changing the DID value, hence it is not possible to update it, let alone view the change history.	1
Many Eyes	The code is public on GitHub, however there are currently only 4 contributors with the last one being nearly 1 year ago.	3
Regulatory Compliance	There are no specific regulations that did:jwk tries to adhere to as it is intended for local temporary short-term usage.	2
Per-DID constraints on visibility	The process of generating and using a did:jwk DID can be completely private since it does not rely on any public VDR. The DID is only visible to parties that it is shared to.	4
Cross-DID Leakage	Each DID is created using an encoding of its own JWK, hence it is not possible to infer relations with other DIDs.	4
Revocation of Consent	Since did:jwk is a generative DID method by nature, revocation or deletion cannot be performed once the DID has been shared to other parties.	1
Total Score		59

Table 19: Resulting Evaluation Scoring Sheet of did:jwk

C.5. did:peer

DID Method	did:peer	
Specification URL	https://identity.foundation/peer-did-method-spec/index.html	
Current status of specification	Draft (v1.0)	
DID Specification Registries?	Yes	
Underlying technology	Only resolvable by parties in an established relationship, and each party contributes their own n-wise DID to maintain the relationship. A protocol is used for communicating the DID and DID Document to peers, and each peer stores this data into a local cache or database as the VDR.	
Criteria	Evaluation	Score
Open contribution (participation)	The did:peer specification is governed by the Decentralized Identifier Foundation which encompasses a varied range of contributors worldwide. Possibility for contribution is open to anyone by creating issues and pull requests as the code is hosted on public GitHub, however approvals for contributions are still managed by members of DIF.	3
Transparency	There are no live agendas regarding discussions about did:peer, however all past and current discussions in GitHub issues and pull requests can be accessed by the public.	3
Breadth of Authority	DIF remains as the maintainer of the did:peer specification, hence new contributions still have to be approved by members of the organization. There are currently a substantial set of contributors coming from diverse backgrounds.	3
Public vs private economies	Aside from DIF being a non-profit community striving to establish an open decentralized identity ecosystem, the did:peer method itself is encapsulated in a way that it relies solely on itself and peers in the relationship.	4
Cost	It is cost-free to participate to the specifications on GitHub, however it will take time and effort. There are zero transaction costs to create and maintain a did:peer.	4
Permissioned operation	Creating a did:peer does not require any permissions aside from agreements among peers. Reading a did:peer DID and DID Document can only be performed by peers who have received them from the owner.	4
Interoperability	Any technology of choice is able to implement did:peer. There are no specific restrictions. Each peer can define their own method of storage for the DIDs and DID Documents that they receive.	4
Scope of usage	did:peer only allows a created DID to be shared by a certain number of parties via an established relationship. It should not be used outside of this relationship.	2
Financial accountability	Since maintaining the DID itself does not involve any costs, the financial accountability of peers are also not discussed.	1
Auditability	Updates are not supported in the did:peer specification, hence a change history does not exist. Once a DID owner	1

	shares a copy of the DID and DID Document, each peer only receives and stores this copy locally using the medium of their choice (for example: database, local cache) which acts as the VDR.	
Robust Crypto	did:peer specifies several different ways to generate a DID (method 0, 1, 2, 3, 4). However, none of them require a specific algorithm to generate the cryptographic keys. There are some DID generation methods that requires the usage of SHA256 to use a hashed value as the DID. Moreover, sharing of the DID Document to other peers must be performed using a protocol that is encrypted, following the DIDComm protocol.	2
Expert Review	It is difficult to judge the security of the cryptographic algorithms as none are explicitly specified, meaning users make the choice on their own. Regarding general security primitives, did:peer relies on the fact that DIDs are only shared among peers in a relationship, and not outside. Hence, the DID Document can only be retrieved by these selected group of entities. This may sound risky, but in principle it seems sufficient as long as all peers are trusted.	3
Future Proofing	Currently, did:peer specifies certain cryptographic methods to follow. However, they are defined in a standardized way so that future updates of cryptographic methods do not need much changes on the user's end, allowing for backwards compatibility.	4
Self Certification	The inception keys are generated by a secure random algorithm, and the DID is built by processing the public part of this key pair with other mechanisms, hence there is high self-certification. Changing the public key on the DID Document will result in mismatch between DID and DID Document.	4
Availability	did:peer relies on trusted relationships among DID owners and receivers, which is believed to be secure in this scenario. However, there is always a risk of man-in-the-middle attack when establishing the connection between peers.	3
Evolution	It is not possible to check if the current state of a DID Document is correct based on its history because did:peer does not support updates, hence there is no history. Only a single copy of the initial DID Document is stored in each peer's local VDR of choice.	1
Many Eyes	The code is publicly available on GitHub with 25 contributors so far, and vulnerability reports can be made using GitHub issues.	3
Regulatory Compliance	There is no incentive for did:peer to comply to any regulations as the DIDs are not shared to the public. As long as the parties involved in the peer relationship agree, then internal regulations have been complied to.	2
Per-DID constraints on visibility	Since the DID is only shared to peers of a relationship, it is completely possible to keep it private from the public. An	4

	agreement between peers to not expose the DID to others parties should be honoured by each peer.	
Cross-DID Leakage	It is advised that a new DID is created and not reused for each different peer relationship. Each DID is created with high randomness, hence it is difficult to deduce DIDs from the same owner.	4
Revocation of Consent	Complete revocation of consent for a DID can be achieved, by communicating the deletion of the DID to all those that hold it. Otherwise, if a DID holder deletes its local copy, then it just removes itself from the peer relationship.	4
Total Score		63

Table 20: Resulting Evaluation Scoring Sheet of did:peer

C.6. did:pkh

DID Method	did:pkh	
Specification URL	https://github.com/w3c-ccg/did-pkh/blob/main/did-pkh-method-draft.md	
Current status of specification	Draft	
DID Specification Registries?	Yes	
Underlying technology	Generates DIDs and DID Documents using existing blockchain addresses, that typically conforms to the CAIP-10 specification based on public key hash (pkh).	
Criteria	Evaluation	Score
Open contribution (participation)	The did:pkh method is governed by the W3C Credentials Community Group. There are a handful of active contributors. Contributions are made on did:pkh's public GitHub repository, and final decisions are made by members of the W3C CCG.	3
Transparency	Previously, regular meetings were held with the agenda to discuss GitHub pull request and issues created for the DID method. Meeting schedules would be broadcasted via the W3C CCG mailing list, and meeting minutes are also stored on GitHub. Currently, these regular meetings are on an indefinite hiatus. As of now, the discussions are performed asynchronously via GitHub.	3
Breadth of Authority	Although potential contributions are accepted from any parties, the group that mainly decides the rules for did:pkh come from W3C CCG.	3
Public vs private economies	With the did:pkh specification being maintained by the W3C CCG, the specification itself was created with the intention of the public good. The economic interests of the blockchains however, are varied depending on the selection of the DID owner.	3
Cost	Participating in the DID method specification is free, however it takes time and effort to contribute to the GitHub. Creating a did:pkh DID itself is free, but it requires the possession of blockchain addresses, which typically have their own related fees based on the respective blockchains.	2
Permissioned operation	Anyone with an approved blockchain address is allowed to fully participate in the DID method as there are no restrictions. However, permissions vary for writing to each supported blockchain network; these are not associated with did:pkh permissions.	3
Interoperability	There are no specific technologies that are specifically tied to the usage of did:pkh.	4
Scope of usage	Universal usage of did:pkh DIDs are possible as long as they are shared to the intended parties. Anyone in possession of the DIDs can essentially generate the DID Document.	4
Financial accountability	Since did:pkh relies on blockchain addresses, the financial operations can be retrieved from the blockchain networks if needed.	4

Auditability	It is not possible to audit the change history of a did:pkh DID as updates are not possible due to its generative nature.	1
Robust Crypto	The did:pkh specification itself does not define any mandatory cryptographic algorithms that must be used. It states that the cryptographic algorithm depends on the blockchain of choice. In general, it can be regarded that blockchains require a cryptographic algorithm with more than 128 bits of security.	3
Expert Review	The public regards the cryptography and security of blockchains are generally regarded as very secure and robust, evident by their popularity.	4
Future Proofing	Since the choice of cryptographic algorithm depends on the blockchain of choice used to create the DID, code changes may be required on the blockchain side.	2
Self Certification	The degree of self-certification of did:pkh is very high since the DID is created from existing blockchain addresses, which are also used as the verifying key in the DID Document.	4
Availability	The availability of did:pkh depends on the blockchain that it relies on, which are for the most part robust, but are not immune from risk of attacks.	3
Evolution	It is not possible to audit the evolution of a DID Document as updating DID Documents is not supported in did:pkh.	1
Many Eyes	The code is published on a public GitHub, however there is still only a small list of contributors within the W3C CCG.	3
Regulatory Compliance	The regulatory compliance depends on which blockchain is used and also what regulations the user is interested in.	3
Per-DID constraints on visibility	Fully constraining the visibility of a did:pkh DID is possible as it is a generative DID, hence it is not stored on a public VDR. Only people receiving the DID can resolve it.	4
Cross-DID Leakage	The dependence of did:pkh towards blockchain addresses prevent the possibility of complete anonymity, as the possibility of tracing different transactions to the same owner is not completely dismissed from all of the blockchains that did:pkh supports.	3
Revocation of Consent	Deletion is not possible using did:pkh since it is a generative DID, thus anyone who has a hold of the DID can always resolve (and generate) the DID Document. Consequentially, it is best to use did:pkh only in local environments for shorter time periods.	1
Total Score		61

Table 21: Resulting Evaluation Scoring Sheet of did:pkh

C.7. did:ion

DID Method	did:ion	
Specification URL	https://github.com/decentralized-identity/ion	
Current status of specification	Release (v1.0.4 – 9 Jun 2022)	
DID Specification Registries?	Yes	
Underlying technology	Registering DID Documents on a public permissionless network called the Identity Overlay Network (ION), which is based on the purely deterministic Sidetree protocol, running atop Bitcoin blockchain as a data link layer and depending solely on Bitcoin's timechain.	
Criteria	Evaluation	Score
Open contribution (participation)	The did:ion specification is maintained by the Decentralized Identity Foundation, which is composed of a community of international contributing organizations. Anyone can submit proposal for changes via GitHub issues, and they will be assessed by members of DIF.	3
Transparency	All decisions made to the specifications are open and based on discussions conducted during development and working group meetings, in which all community members are able to participate. It is required to become a member of DIF to participate in these meetings.	4
Breadth of Authority	DIF is purposely comprised of an open and diverse collection of international organizations, where each contributor has equal say in the decisions made.	3
Public vs private economies	Aside from the fact that did:ion is governed by the non-profit DIF, both ION and Bitcoin were created for complete public usage and not involving the financial gains of any parties.	4
Cost	It is cost-free to participate to the specifications on GitHub, however it will take time and effort to attend the development meetings and working group discussions. There is a cost to perform Bitcoin transactions, albeit it is considered very small as one transaction by ION can compose of up to 10000 DID operations. There is a minimum transaction fee for each operation, which is set at 0.001 of the normalized Bitcoin transaction fee of the past 100 blocks. Moreover, making an ION transaction requires depositing some amount of money for a certain predefined time duration in case more transactions are performed in the near future, with the aim to prevent spam.	2
Permissioned operation	ION, being based off Bitcoin, is permissionless by nature. Additionally, anyone can install and run an ION node on their own to be a part of the growing ION network.	4
Interoperability	Anyone can participate with did:ion via any technology. There are no restrictions on the software to be used.	4
Scope of usage	Since ION uses Bitcoin, all DIDs are recorded on a public ledger and accessible by anyone, hence its usage is universal.	4
Financial accountability	The Bitcoin blockchain records all financial transactions that occur in each operation.	4

Auditability	All operations conducted on ION for did:ion are recorded on the Bitcoin blockchain, hence the history of a did:ion DID Document can be audited.	4
Robust Crypto	did:ion allows the support of any JWK representation in their DID Documents. Although JWK only defines a format and not a specific cryptographic algorithm, generally this equals to a security level of at least 128 bits.	3
Expert Review	The security of ION relies heavily on Bitcoin. The Bitcoin blockchain is arguably regarded as one of the most secure blockchain currently available in the cryptocurrency domain.	4
Future Proofing	Upgrading the cryptography is possible as long as it still follows the JWK representations, however additional security upgrades may also be involved in the ION code, or even possibly on the Bitcoin code.	2
Self Certification	A did:ion identifier is created by deriving values from its generated cryptographic keys used in the DID Document, hence it is definitely self-certified.	4
Availability	The specification guarantees ION's availability as the number of ION nodes grow. The more the ION nodes, the more available and redundant ION data is. Each ION node has a copy of all transaction data. However, ION is not completely free from any vulnerability risks.	3
Evolution	All history of updates are stored on the Bitcoin blockchain, which is a distributed ledger, hence the evolution of a DID Document can be verified from it.	3
Many Eyes	The code is publicly hosted on GitHub and has around 30 contributors. Vulnerability issues are also addressed on GitHub.	3
Regulatory Compliance	Given the large influence and scale of Bitcoin, did:ion generally fulfils regulations across the globe, although there may be some specific regulations that need to be attended to.	3
Per-DID constraints on visibility	As ION writes to the Bitcoin blockchain, it is completely public and visible to any interested party. There is no possibility to make it private.	2
Cross-DID Leakage	Each DID is generated based on its own set of cryptographic keys with high entropy, so they are isolated from one another and their relationships are difficult to infer.	4
Revocation of Consent	did:ion allows for deactivation of DIDs, and once deactivated it can no longer be resolved, but the history is permanent on the Bitcoin blockchain.	3
Total Score		70

Table 22: Resulting Evaluation Scoring Sheet of did:ion

C.8. did:ebasi

DID Method	did:ebasi	
Specification URL	https://hub.ebsi.eu/vc-framework/did/legal-entities https://hub.ebsi.eu/vc-framework/did/natural-person	
Current status of specification	Release (Last Updated 21 Feb 2024)	
DID Specification Registries?	No	
Underlying technology	<ol style="list-style-type: none"> 1. For legal entities: Relies on the European Blockchain Services Infrastructure (EBSI) as a public VDR, however can also rely on other EBSI-complying DID VDRs. 2. For natural person: Utilizes cryptographic keys generated for the DID Document as the identifier, similar to did:key. 	
Criteria	Evaluation	Score
Open contribution (participation)	The specification is not open-source. Contributions to governance of did:ebasi is conducted only by the European Commission and European Blockchain Partnership in private.	2
Transparency	At its current state, the did:ebasi specification rules are available to the general public on EBSI's website, however there are no signs of the ability to view a history of past changes.	2
Breadth of Authority	Only the EC and EBP actively participate in the governance authority. The list of individual contributors from the two organizations is also not visible.	2
Public vs private economies	At this point of the did:ebasi specification's maturity, it can be considered that the DID method was established for the interests of the EC, however it does not seem that profits are a main driver.	3
Cost	It is not possible to participate in the specification of did:ebasi, unless being a part of the EC or the EBP. Costs for using EBSI are not shared to the public for now.	1
Permissioned operation	<p>For legal entities: Reading a DID from the EBSI network is completely open to anyone, however registering the DID and DID Document requires permissions and an onboarding process to the EBSI Authorisation service by contacting EBSI's technical office.</p> <p>For natural person: The DID creation solely relies on the generated cryptographic keys, which is used as both the identifier and part of the DID Document. The DID Document itself is not stored on a VDR, but is regenerated each time a resolve operation is performed. Hence, there are no exclusive permissions for it.</p>	2/4
Interoperability	The EBSI website lists a number of wallets that are compliant with the did:ebasi specification. Development for the did:ebasi requires joining the EBSI community, currently through the Early Adopters Programme.	2

Scope of usage	<p>For legal entities: Created DIDs are registered on the EBSI network, which is a public blockchain, hence can be resolved by anyone universally.</p> <p>For natural person: The DIDs are purely generative based on cryptographic keys, hence by principle they can be used universally by anyone.</p>	4
Financial accountability	The did:ebis specification does not describe information related to financial operations.	1
Auditability	<p>For legal entities: The EBSI network stores all records of DID Documents in a chronological order including the time duration in which the DID Document is valid. Due to this, it is possible to audit the change history of a DID Document through the VDR.</p> <p>For natural person: Since the DID Document is purely generative, there is no way to view the change history of it as it is unchangeable.</p>	4/1
Robust Crypto	<p>Both did:ebis specifications require the usage of cryptographic algorithms with a minimum of 128 bits of security for DID creation (minimum ES256 and Elliptic Curve NIST P256, respectively).</p> <p>For legal entities: Communication with the EBSI network is performed via HTTPS, whose security level depends on the web SSL/TLS certificate that the web domain uses, generally at 128 bits.</p>	3
Expert Review	<p>Generally, both did:ebis specifications encourage the usage of cryptographic algorithms that are recognized as secure, with a security level of at least 128 bits.</p> <p>For legal entities: The communication with the EBSI network as the VDR is based on strong industry standards of HTTPS, hence no additional security is included.</p> <p>For natural person: Since the specification is for a generative did:ebis DID Document, there are no extra security measures in place. The security depends solely on trustworthiness.</p>	3
Future Proofing	The did:ebis specification advises users to implement cryptography in a modular manner, so that implementations are prepared for future specification changes, in the case where new mandatory cryptographic algorithms are introduced.	2
Self Certification	For legal entities: The did:ebis identifier is built using the EBSI subject identifier of the identity. It is created also using cryptography in a decentralized manner in EBSI, however it is distinct from the inception keys in the DID Document.	3/4

	For natural persons: The did:ebis identifier is built using the cryptographic keys generated to be used for the DID Document, hence there is high self-certification.	
Availability	The EBSI network abides by standard European protections in place to prevent malicious attacks. However, since it is a new concept, its credibility still needs to be built.	3
Evolution	For legal entities: The EBSI network records all change history of DID Documents, hence its evolution can be traced. For natural person: Updates and deactivations are not possible for this type of the specification, hence evolution is not supported.	4/1
Many Eyes	The code for the specification of both did:ebis types are completely private and not visible to the public.	1
Regulatory Compliance	As did:ebis is governed by the EC itself, naturally it explicitly complies with several EU regulations, such as eIDAS and GDPR which are mentioned on the specification.	4
Per-DID constraints on visibility	For legal entities: The EBSI network is a public blockchain that acts as the public VDR, hence it is available to be accessed by anyone through an HTTPS connection. For natural persons: Completely private is possible because it is based on using cryptographic keys only, so there is no public VDR.	2/4
Cross-DID Leakage	Although an EBSI subject identifier is used to create the DID in the specification for legal entities, it is based on a cryptographic method, hence there is high entropy and difficult to infer to other DIDs, similarly in the specification for natural persons.	4
Revocation of Consent	For legal entities: The DID Document cannot be removed from the EBSI network, but its keys can be revoked or can be set to expire at a certain date, which leads to deactivating the DID. For natural persons: It is not possible to revoke the validity of a DID because there is no VDR. A DID will continue to be resolvable (generated DID Document) as long as it is in possession.	3/1
Total Score		52/53

Table 23: Resulting Evaluation Scoring Sheet of did:ebis

C.9. did:sov

DID Method	did:sov	
Specification URL	https://sovrin-foundation.github.io/sovrin/spec/did-method-spec-template.html	
Current status of specification	Draft (W3C Editor's Draft – 22 April 2024)	
DID Specification Registries?	Yes	
Underlying technology	DIDs and DID Documents are registered on the Sovrin network, a public permissioned ledger exclusively designed for self-sovereign identity built using Hyperledger Indy.	
Criteria	Evaluation	Score
Open contribution (participation)	did:sov, like the Sovrin network that it relies on, is maintained under the Sovrin Foundation. Contributions are open to anyone as did:sov's GitHub page is public, currently with 38 contributors. However, approval of contributions are still governed by members of the Sovrin Foundation.	3
Transparency	All discussions regarding the specifications are visible on the public GitHub as issues and pull requests, although it is not so easy to follow it in real time. Considering the Sovrin network itself, the Sovrin Foundation holds regular meetings to discuss regarding progressions of the Sovrin network, and sometimes also mentioning about the did:sov specification.	4
Breadth of Authority	Although contributions are welcome, they are filtered and managed by the Sovrin Foundation, who makes the ultimate decision on whether the change should be accepted or not.	3
Public vs private economies	For did:sov, the fees involved when performing operations on the Sovrin network display that there is some monetary interests by the governing parties, however it is more acceptable by the fact that these fees are used for the operational costs to maintain the Sovrin Foundation.	3
Cost	Contributing to the did:sov specifications is free, as its open to the public via GitHub, it will only cost time and effort. However, creating did:sov DIDs means writing to the Sovrin network, which includes fees not only for one-time DID writes to the ledger, but also to even be allowed to make transactions on the ledger.	1
Permissioned operation	Reading DIDs is publicly available by anyone, however users need to register to the Sovrin network as a transaction endorser to be able to make the actual writes, otherwise they are intermediate writes that rely on a transaction endorser.	2
Interoperability	did:sov and the Sovrin network do not restrict the types of technology that can work with it.	4
Scope of usage	did:sov allows for universal usage as it relies on the public Sovrin network that is open to be viewed and read by any party.	4
Financial accountability	The Sovrin Foundation publishes the fees incurred when performing operations on the Sovrin network. However, the network itself does not hold any financial information as it	3

	focuses on storing data related to self-sovereign identity, such as DIDs and VCs.	
Auditability	The history of changes to a DID Document is stored on the Sovrin network, and anyone can view it since the Sovrin network is completely public.	4
Robust Crypto	The did:sov specification provides an appendix containing cryptographic algorithm types that are allowed to be used in the DID Document. The list of cryptographic algorithms is quite extensive, and includes algorithms with lower security levels less than 112 bits. Since there is no explicit minimum algorithm set, it can be concluded that these lowest security level algorithms are the minimum requirement.	1
Expert Review	did:sov supports the usage of a variety of popular cryptographic algorithms which are generally very secure. However, the Sovrin network also relies on CurveZMQ for secure communication, which although provides much security for mitigating attacks, also adds possibilities of dependent vulnerabilities.	3
Future Proofing	From a user's standpoint, it is easy to replace the cryptography from a lower level to a higher level that is already listed on the specification. However, adding more allowed cryptography algorithms to did:sov, or updating the security of the Sovrin network, would most likely require code changes.	2
Self Certification	Since there are various ways in which the identifier for did:sov can be generated, it is not always the case that the inception keys are related to the identifier. This is possible when the did:sov identifier is derived from the cryptographic algorithm used for the DID Document, however it's not the case if the did:sov identifier is created using standard UUID.	2
Availability	In general, did:sov and the Sovrin network imposes strong security measures that attempts to avoid any risks of malicious attacks, however it is still a new network and have reliance on external dependencies which could potentially lead to vulnerabilities.	3
Evolution	Each evolution of state of the DID Document is recorded on the public Sovrin network, hence can be visible by anyone who can resolve did:sov DIDs.	4
Many Eyes	The code for the DID specification is public, however there are only 38 contributors so far. Vulnerability reporting is visible as GitHub issues.	3
Regulatory Compliance	Since the Sovrin network was specifically established for self-sovereign identity, it should comply with regulations set surrounding SSI, DIDs and VCs.	4
Per-DID constraints on visibility	It is not possible to restrict the visibility of the VDR as the Sovrin network is public and available to be read by anyone.	2
Cross-DID Leakage	There is no way to infer relationships between multiple DIDs as each one is created with a high degree of entropy,	4

	either from using standard UUID methods or by deriving the identifier from the inception cryptography keys.	
Revocation of Consent	A did:sov DID can be revoked by deleting the verification key in the DID Document, terminating the ability to perform any other operations. However, history of the DID Document will still permanently exist on the network.	3
Total Score		62

Table 24: Resulting Evaluation Scoring Sheet of did:sov

C.10. did:ethr

DID Method	did:ethr	
Specification URL	https://github.com/decentralized-identity/ethr-did-resolver/blob/master/doc/did-method-spec.md	
Current status of specification	Release (10.1.5 – 14 Feb 2024)	
DID Specification Registries?	Yes	
Underlying technology	Uses Ethereum address (or secp256k1 public key) as the DID, and registers the DID Document to an off-chain registry (following ERC-1056 smart contract standard) that is related to the actual Ethereum network.	
Criteria	Evaluation	Score
Open contribution (participation)	did:ethr is officially maintained by the Decentralized Identity Foundation. Despite being publicly available on GitHub, the only contributions to the did:ethr specification are currently by the uPort Veramo team.	2
Transparency	All contributions to the did:ethr specification are visible on the public GitHub’s pull requests and issues, however it is difficult to observe the discussions in real time.	3
Breadth of Authority	In theory, changes on the specification are approved by members of the DIF. However in practice, currently only members of the uPort Veramo team are active contributors.	2
Public vs private economies	Although the did:ethr specification is stored under the DIF GitHub repository, in reality it is created for the special intentions of the uPort Veramo team, which is a private company.	2
Cost	Participating to the did:ethr specification is free of charge as it is available on a public GitHub, but it will consume the time and effort, albeit it may be difficult to contribute as the specification is currently only governed by the uPort Veramo team. Moreover, similarly to other blockchains, Ethereum charges transaction fees when operating on its network. However, did:ethr itself follows ERC-1056, which tries to minimize the transaction costs by enabling the usage of off-chain transactions leading to zero fees for creation.	2
Permissioned operation	The Ethereum blockchain network and the ERC-1056 registry allows anyone to participate since Ethereum is a public permissionless network.	4
Interoperability	There are no restrictions imposed on which technology can work with did:ethr and the Ethereum blockchain network.	4
Scope of usage	Since the Ethereum network is permissionless and public, in principle the did:ethr DID method can be used universally.	4
Financial accountability	The financial operations of Ethereum are transparent and visible to the public on the network.	4
Auditability	Initial creation of the DID Document is private, however all historical changes related to updates and deactivation are stored on the public Ethereum blockchain, where each transaction is linked to a previous block transaction containing the related DID operations. Hence, it is visible to anyone.	4

Robust Crypto	Building the method specific identifier either relies on using an Ethereum address, or using the secp256k1 cryptographic algorithm, which provides 128 bits of security. Aside from that, did:ethr relies on cryptographic algorithms utilized by Ethereum.	3
Expert Review	In general, with the popular adoption of the Ethereum network, its security is considered as well-vetted by experts. However, the sample space of the adoption of smart contract standard involved with did:ethr, ERC-1056, is still not very wide.	3
Future Proofing	Currently did:ethr only supports limited cryptographic methods, hence code changes will be required to allow new cryptography in the future. This also includes updating cryptography of the Ethereum blockchain.	2
Self Certification	In most cases, did:ethr DIDs uses an identifier that is based on the Ethereum address, instead of using a value from its inception keys. However, did:ethr does open up possibilities to use any secp256k1 public key to be used as its identifier.	3
Availability	So far, the Ethereum network has been stable and regarded as one of the most secure blockchain networks, however it is not completely immune from all forms of malicious attacks.	3
Evolution	The evolution history of a DID Document is stored on the Ethereum network in a series of block events, where each current DID Document has a reference to its last event.	4
Many Eyes	The source code of the DID method is public, however there is only one known group of contributors, which is the uPort Veramo team.	3
Regulatory Compliance	As a decentralized platform, Ethereum on its own does not specifically comply with specific regulations, as basically anyone from around the world can take part in it.	2
Per-DID constraints on visibility	The Ethereum network, as the VDR, is completely public and auditable by anyone, without any constraints. The main purpose of blockchain networks are to be transparent.	2
Cross-DID Leakage	A different Ethereum address is advised to be used when creating each new different DID. Although did:ethr identifiers depend on Ethereum addresses, these addresses are generated with high entropy, hence it is nearly impossible to infer relationships among did:ethr DIDs.	4
Revocation of Consent	Since did:ethr relies on a ledger-based VDR, once something is written to it, it cannot be removed. However, the DID Document can be updated to a revoked state. Its history will still exist on the network but it will no longer be valid and updatable for the future.	3
Total Score		63

Table 25: Resulting Evaluation Scoring Sheet of did:ethr

C.11. did:cheqd

DID Method	did:cheqd	
Specification URL	https://docs.cheqd.io/identity/architecture/adr-list/adr-001-cheqd-did-method https://github.com/cheqd/identity-docs/blob/main/architecture/adr-list/adr-001-cheqd-did-method.md	
Current status of specification	Implemented (Last Updated 6 Feb 2023)	
DID Specification Registries?	Yes	
Underlying technology	DID Documents are stored on the Cheqd network, a blockchain network built based on the Cosmos blockchain framework and created with the main purpose for self-sovereign identity. DIDs are generated mainly using UUID, or alternatively for compatibility purposes by encoding the inception key stored in the DID Document.	
Criteria	Evaluation	Score
Open contribution (participation)	The Cheqd company are the main owners and decision makers of the did:cheqd specification. The specification itself is open-source and publicly available on Cheqd’s website and GitHub, however contributions are maintained by Cheqd team members. Cheqd does invite anyone to join the Cheqd community to suggest improvements across all technologies and services provided by Cheqd, mainly through their Discord platform.	2
Transparency	It does not seem that there are regular meetings open to the public for contributing to the did:cheqd specification, although this is possibly because direct contributions from the community is not expected. Cheqd does promise daily discussions regarding various Cheqd technologies through its various social media streams. Moreover, the history of pull requests and issues from the Cheqd team can be viewed on Cheqd’s GitHub, hence the thought processes coming to the decisions are visible to any interested party.	3
Breadth of Authority	Since Cheqd is a for-profit technology company that provides its services to others, it seems only natural that the specification rules are only governed by the Cheqd team.	2
Public vs private economies	The Cheqd network is owned by the for-profit Cheqd company, hence usage of the did:cheqd method also contributes to revenue growth of the Cheqd company.	2
Cost	At this point of time, it is hardly possible to participate in the did:cheqd specification. Moreover, similarly to other blockchains, there are fees involved in order to perform DID operations on the Cheqd network, although with arguably more reasonable prices.	1
Permissioned operation	did:cheqd’s VDR, Cheqd, is a permissionless blockchain network, hence anyone can perform operations on it from any location.	4

Interoperability	There are no restrictions on which technologies that can be used with did:cheqd. It actually aims to be compatible with other similar solutions (such as those built based off Hyperledger Indy). Cheqd recommends some SSI wallets to easily get started with using did:cheqd, but does not specify that these are the only ones available to use.	4
Scope of usage	The VDR for did:cheqd (Cheqd blockchain network) is public, hence anyone can use it universally to perform DID operations.	4
Financial accountability	The usage fees for Cheqd are transparent, and financial transactions are stored on the blockchain.	4
Auditability	As each did:cheqd DID operation on its DID Document are stored on the Cheqd public blockchain network, anyone can access and audit the historical changes of the DID Document.	4
Robust Crypto	did:cheqd currently only supports three cryptographic algorithms to generate the cryptographic key formats stored in its DID Document, where there is a possibility for a cryptographic algorithm providing less than 128 bits of security to be used.	2
Expert Review	In general, the cryptographic algorithms supported by did:cheqd are regarded as secure, however since the Cheqd blockchain is still a growing network, it needs some time and experience to solidify its reputation.	3
Future Proofing	As did:cheqd specifies exactly which cryptographic algorithm formats it supports, code changes need to be carried out when adopting new cryptography. It is most likely also the case for the Cheqd blockchain network.	2
Self Certification	Identifiers used for the did:cheqd DIDs mainly take the form of generated UUIDs with an algorithm of the user's choice, providing a high degree of arbitrariness. However, building the identifier using the encoding of the DIDs inception keys is also possible, mainly to allow for compatibility with Hyperledger Indy-based DIDs. Although this alternative method will guarantee its self-certification, generating by UUID is still the preferred choice of method.	2
Availability	At its current state, the Cheqd network is generally robust and well-maintained by the Cheqd team, although it may not be completely immune from any risks of malicious attacks by motivated attackers.	3
Evolution	A did:cheqd DID Document stored on the Cheqd blockchain network can be traced back to its previous versions on the Cheqd blockchain, and even specific versions can be explicitly queried, where it is also linked to its next version.	4
Many Eyes	The code is published publicly on GitHub, however the number of contributors is small since it only involves the Cheqd team.	3
Regulatory Compliance	Being a blockchain network that was built with the purpose of adhering to self-sovereign identity needs, the Cheqd network generally complies with regulations related to SSI.	4

Per-DID constraints on visibility	did:cheqd depends on the Cheqd blockchain network as its VDR, which is public and permissionless. Hence, there are no restrictions to its visibility to the public.	2
Cross-DID Leakage	Both methods for generating DIDs (UUID-style and Indy-style) have high levels of entropy, hence each DID is isolated from one another and it is nearly impossible to infer relationships among them.	4
Revocation of Consent	It is possible to deactivate a DID so that a DID Document can no longer be updated, but its history and existence will remain on the Cheqd blockchain network forever.	3
Total Score		62

Table 26: Resulting Evaluation Scoring Sheet of did:cheqd

C.12. did:web

DID Method	did:web	
Specification URL	https://w3c-ccg.github.io/did-method-web/	
Current status of specification	Release (06 May 2023)	
DID Specification Registries?	Yes	
Underlying technology	Using an existing web domain resolved through a DNS as the DID, and publishing the DID Document under the /.well-known path of the web server.	
Criteria	Evaluation	Score
Open contribution (participation)	The did:web DID specification is maintained by the W3C Credentials Community Group. Contributions can be suggested by anyone via the public GitHub repository for did:web, however final decisions to accept the changes are made by maintainers from the W3C CCG.	3
Transparency	All discussions regarding the specifications carried out via open issues and pull requests are openly accessible via GitHub. However, there is limited possibility to audit it in actual real-time.	3
Breadth of Authority	The rules are decided by a small set of active maintainers from different organizations within the W3C CCG.	3
Public vs private economies	W3C CCG, as the governing body, established the did:web specification for open usage by the society.	4
Cost	It is cost-free to contribute to the specifications on GitHub, however it will take time and effort. Aside from that, the only other costs for utilizing did:web would be maintaining a web domain and server, which can be considered as negligible since most organizations would already have this setup.	4
Permissioned operation	As the owner of the web domain, there are no additional permissions required to use the web domain to create the DID and host the DID Document. There are also no special permissions needed for anyone to read did:web DIDs.	4
Interoperability	Since did:web is dependent on the web and DNS, it is possible to be utilized by any technology.	4
Scope of usage	The scope of usage of did:web is universal since the web is unrestricted and available to anyone with connection to the internet.	4
Financial accountability	The financials of any parties are by default not shared with each other.	1
Auditability	The change history of the DID Document is not inherently visible for did:web, as performing updates to DID Documents means replacing the existing JSON files with newer ones. However, other technologies such as an online version control repository allows the change history of the code to be auditable, though it is not a requirement to implement it.	1
Robust Crypto	The security of did:web depends on the SSL/TLS certificate of the web domain, where generally the level of security ranges from 112 to 256 bits.	2

Expert Review	Security of SSL/TLS certificates are mostly guaranteed and vetted by experts as the web has been reliant on it for a long time. They do have known vulnerabilities which can be mitigated.	3
Future Proofing	Modifying the security of did:web is quite difficult as it relied on current web standards.	1
Self Certification	The method specific identifier could be any random value that the DID owner chooses since it relies on web domain names. Moreover, domain names can expire and be reused by other parties.	1
Availability	In general, the security provided by SSL/TLS certificates are adequate, however there is always possibility of attacks on the web, such as DDoS, XSS, and MitM.	3
Evolution	did:web on its own cannot prove its history, but if changes to the contents of the web page are tracked and published on additional technologies such as an online version control repository, then the change history can be verified.	1
Many Eyes	The code for the DID specification is published online on GitHub for public visibility, but the list of contributors is still minimal. Vulnerability reporting exists in the form of GitHub issues.	3
Regulatory Compliance	Cryptographic mechanisms used for web security is generally compliant to any regulations worldwide.	4
Per-DID constraints on visibility	By default, there are no constraints on restricting the public visibility of did:web as web domains can be accessed online by anyone with an internet connection anywhere. However, special restrictions can be put in place such as specifically whitelisting or blacklisting certain IP addresses.	3
Cross-DID Leakage	Generally, it is difficult to infer other DIDs of the same owner as chances are the domain names used are different. However, in the case where different DID Documents are stored under different paths of the same domain, then there is a possibility that the DIDs can be deduced to the same owner.	3
Revocation of Consent	The DID can be revoked simply by removing the DID Document from the web path. It will no longer be resolvable after. A side effect of this is that it raises issues in differentiating whether the DID is actually deactivated, or whether the web server is just offline hence the DID is unresolvable.	4
Total Score		59

Table 27: Resulting Evaluation Scoring Sheet of did:web