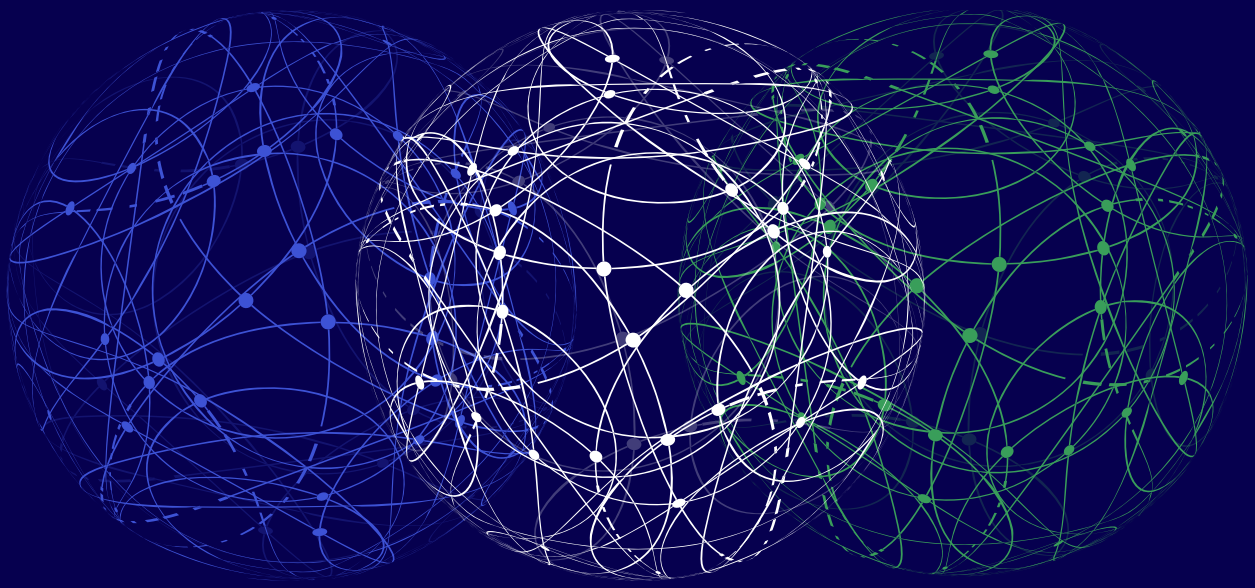
DATA SHARING COALITION

Use Case Implementation Guide



Version 1.0, March 2022

Table of Contents

Section A.

Introduction

01. Reading Guide

1.1. Intended Audience

People and organisations that are a stakeholder in the development of the future *Trust Framework* are the main audience of this document.
As a standalone document, the UCIG provides relevant insights for:

- Members of and people interested in the DSC in general,
- People that play a role in *Data Sharing* that want to learn how to achieve interoperability,
- People interested in (cross-*Domain*) *Data Sharing* in general,
- People that want to set up a cross-*Domain Data Sharing* use case.

1.2. Typography

The typography in this document follows the following rules:

- Regular text appears like this,
- *Defined Terms From The Glossary Appear Like This*,
- Requirements imposed by this document have this prefix: **REQ NN**, where NN (arabic numerals) refers to the requirement number,
- Scalability requirements suggested by this document within a specific context have this prefix: **sREQ MM**, where MM (roman numerals) refers to the requirement number, these requirements can be considered optional in all instances, but when adhered to, contribute to scalability,
- References to other resources appear like this, see **Table 18** for an overview of relevant resources,
- Examples of source code appears like this,

> - Boxes: are used to give examples of source code

- Greyed background indicated text quoted directly from another document/ source, the first line of the greyed box specifies the document/source,

1.3. Notational Conventions

The following key words are used to indicate requirement levels in accordance with IETF RFC 2119:

Table 1: Notational Conventions

Key word	Description
MUST	This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification
MUST NOT	This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.
SHOULD	This word, or the adjective "RECOMMENDED", mean that there can be valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
SHOULD NOT	This phrase, or the phrase "NOT RECOMMENDED", means that there can be valid reasons in particular circumstances when the particular behaviour is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behaviour described with this label.
MAY	This word, or the adjective "OPTIONAL", mean that an item is truly optional. A party may choose to include the item, another party may choose not to.

02. Context

2.1. About the Data Sharing Coalition

The *Data Sharing Coalition* (DSC) is an open and growing, international initiative in which a large variety of organisations collaborate to unlock the value of cross-*Domain Data Sharing*. The DSC aims to drive cross-*Domain Data Sharing* under control of the entitled party, by enabling *Interoperability* between *Domains*.

Several *Data Sharing Initiatives* exist (as of 2022), and these are often focused on a specific sector or *Domain*. Examples include Dutch *Initiatives* such as, Hypotheken Data Netwerk (HDN) for the mortgage *Domain*, Netbeheer Nederland (NB NL) for the energy *Domain*, MedMij for the healthcare *Domain*, and SURF for the higher education and research *Domain*. These facilitate Data Sharing for their Participants. Additionally, generic *Initiatives* such as GO FAIR, AMdEX, iSHARE, NEN, and the International Data Spaces Association provide overarching principles, standards or functionalities which can be used in new and existing *Data Sharing Initiatives*. The DSC aims to build on these existing *Data Sharing Initiatives* to strengthen them in unlocking the value of *Data Sharing* in and across their domain.

The DSC started in January 2020 with support of the Dutch Ministry of Economic Affairs and Climate Policy. The first phase of the DSC is a feasibility study to prove the value potential and practical feasibility of cross-*Domain Data Sharing*. In a, to be planned, subsequent phase, the DSC will continue developing use cases for cross-*Domain Data Sharing* and converge its results and activities to an entity that will operate and govern a future *Trust Framework*, covering all aspects of cross-*Domain Data Sharing*. The DSC believes that a future *Trust Framework* (based on all topics described in Data Sharing Canvas) for cross-*Domain Data Sharing* will enable the *Interoperability* and establish the Trust needed between actors to enable scalable data sharing across *Domains*. For more information on the DSC, visit: https://coe-dsc.nl/.

2.2. About the Use Case Implementation Guide

The DSC supports *Data Sharing* use cases and helps prepare the use cases for future scalability. To this end the *Use Case Implementation Guide* (UCIG) has been developed. The UCIG contains reference requirements which lay the foundation for scalable *Data Sharing* that can directly be implemented by actors in a use case. The requirements in the UCIG are based on insights captured in the Data Sharing Canvas and the DSC use cases.

The DSC is not liable for any damages incurred due to the implementation of the requirements presented here.

2.2.1. Issue Tracking

The UCIG has been initially developed by the DSC project team and version 1.0 was completed in March 2022. This document will be managed and may be updated in the future by the DSC project team. For any comments get in touch via info@coe-dsc.nl.

2.2.2. Change Management

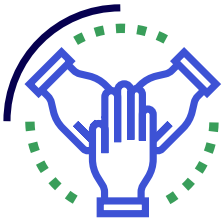
The UCIG does not have a formal change management process. For any suggestions for change get in touch via info@coe-dsc.nl.

03. Guiding Principles

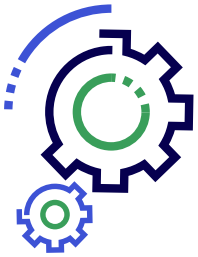
A number of guiding principles are set by the DSC and its vision of a future Trust Framework for cross-*Domain Data Sharing*. In turn, these have guided the development of the UCIG. These Guiding Principles provide a basis for decision-making but are do not represent the absolute truth or hard requirements. Therefore, the Guiding Principles need to be considered in the context of each decision. In no particular order, the following five principles have been identified:

Future proof

Trustworthy



Inclusive



As generic as possible, as specific as needed

Cost-efficient

3.1. Future proof

Statement

A *Trust Framework* for cross-*Domain Data Sharing* should be future proof and therefore extensible and adaptable.

Rationale

A future proof design entails a *Trust Framework* which supports different implementations and is, to some extent, able to cater for changes in technology, user behaviour, regulation and for a growing number of *Data service Transactions*. An adaptive, extensible, and non-static design enables scalability of the *Trust Framework*.

Objectives

1. Create a cooperative *Domain* that allows *Participants* to innovate their services.
2. Support scalable and fully *Interoperable Participant* implementation.

3.2. Trustworthy

Statement

A *Trust Framework* should be designed and maintained in a way that establishes *Trust* for all *Participants* and organisations, fitting the transaction context. *Trust* is defined as a situation between actors where (perceived) risks are sufficiently reduced to enable Data Sharing.

Rationale

Trust is required on all levels of the *Trust Framework* to achieve wide-reaching adoption. *Trust* is required across *Domains* and on a transactional level to facilitate cross-*Domain Data service Transactions*.

Objectives

1. Enable *Trust* between actors from different *Domains*.
2. Ensure that Data remains sovereign and is used for authorized purposes only, as controlled by *Entitled Party*.
3. Define levels of trust dependent on a transaction context to perform a transaction.
4. Facilitate the use of required Data security and privacy mechanisms.
5. Be transparent towards *Participants* and related organisations.
6. Be transparent in process and *Dispute* resolution.
7. Install measures/sanctions against *Participants* and related organisations violating trust.

3.3. Inclusive

Statement

A *Trust Framework* for cross-*Domain Data Sharing* should be generic, usable, and feasible to all organisations or *Domains*, regardless of sector and Data sharing context.

Rationale

Inclusivity is fundamental to enabling solution independent Data sharing across *Domains* and organisations. It ensures diversity by providing a level playing field and comparable opportunities for incomparable organisations. Inclusivity leads to collaborative advantages across all *Domains*.

Objectives

1. Neutrality by ensuring a non-discriminatory approach and Policies towards all organisations, users, and contexts.
2. Cater for different levels of maturity of *Domains* and their *Participants*.
3. Create a level playing field for *Participants*.

3.4. As generic as possible, as specific as needed

Statement

A *Trust Framework* for cross-*Domain Data Sharing* rules should be as generic as possible and as specific as needed, considering different transaction contexts.

Rationale

This principle is needed to keep the *Trust Framework* as lightweight as possible to drive adoption. It ensures that *Participants* are not held back by restricting agreements to keep implementation costs low. Furthermore, it ensures a broad reach amongst sectors and types of organisations.

Objectives

1. Maximise the competitive *Domain* by minimising the collaborative *Domain* requirements.
2. Keep the *Trust Framework* as lightweight as possible.
3. Minimise risk of over-engineering.
4. Ensure solutions are generic to enable as many use cases as possible.

3.5. Cost-efficient

Statement

A *Trust Framework* for cross-*Domain Data Sharing* should be cost-efficient.

Rationale

Controlling costs is essential in a collaborative *Domain* as it enables a fast and effective development. It lowers the threshold for organisations to participate and enables long-term sustainable participation.

Objectives

1. Enable cost savings at an ecosystem level, financially or in terms of effort.
2. Use proven and open standards where possible.
3. Learn from (inter)national best practices.
4. Ensure a transparent cost and benefit structure.
5. Minimise cost of entrance and impact of implementation.
6. Consider impact for *Participants* when changes occur in the future.

04. Compatibility overview

The UCIG is based on open standards and aligned with existing *Data Sharing* standards and implementations where applicable and relevant. This ensures alignment among Data Sharing implementations and prepares *Data Sharing* use cases for future scalability.

To that end, this chapter provides an overview and explanation of how the International Data Spaces Association (IDSA) and iSHARE are applied in the UCIG to provide context for the rest of the document.

4.1. International Data Spaces Association

*I*DSA is an organisation which aims to build data spaces which are characterised by uniform rules, certified data providers and recipients which ensures trusts among participants. To allow for future scalability and possibilities for future interoperability, the UCIG is inspired by IDSA and tailored to the specific context of the UCIG. Many IDSA elements become increasingly important for data sharing use cases as they increase in complexity. Therefore, for the relatively simple use case archetype considered in the UCIG, as introduced in **Chapter 5**, these may not be directly relevant.

4.2. iSHARE

iSHARE is an operational *Trust Framework* for *Identification*, *Authentication* and *Authorization*. As such iSHARE provides mechanisms for cross-*Domain Identification*, *Authentication* and *Authorization* of actors, with underlying adherence agreements and governance. To prepare for future interoperability, the requirements in the UCIG are based on the iSHARE implementation and infrastructure as much as possible, while tailoring to the specific context of the UCIG. Elements of the UCIG, such as Machine-to-machine *Authentication* and the OAuth flow are based on the implementation in iSHARE.

Section B.

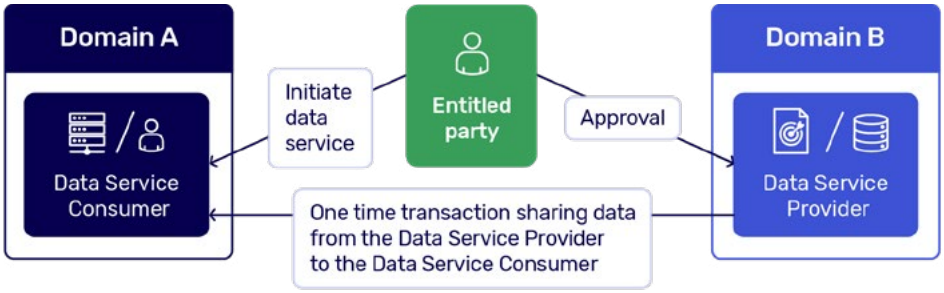
Functional Topics

05. Functional Scope

The scope of the UCIG is limited to a single, common cross-*Domain Data Sharing* use case archetype in which the DSC has experience in its **Green Loans use case**. Based on this specific use case archetype, the requirements in this document have been created. The implementation of use cases of other archetypes can benefit from parts of the UCIG and/or use the UCIG as inspiration. Note that in scoping this use case archetype, reference is made to specific roles and responsibilities which are detailed in **Chapter 6 Roles**.

The UCIG use case archetype can be described as: A *Data Service Consumer* in a *Domain* (*Domain* A) seeks to retrieve *Data* from a *Data Service Provider* in another *Domain* (*Domain* B) that requires approval given by an *Entitled Party*. This use case archetype is visualised in **Figure 1**. The UCIG focusses on enabling access control to the *Data Service* for this use case archetype, the subsequent usage control is not in scope. See **Chapter 7 Interaction model**, for a detailed description of this *Data Sharing* use case archetype.

Figure 1: Overview of the Data Sharing use case archetype in scope of the UCIG



Key characteristics of this *Data Sharing* use case archetype are:

- The *Data Service* is an ad hoc, one-time, transactional *Data Sharing* service
- *Data* is shared bilaterally between two *Domains*
- Approval given by the *Entitled Party* is a fundamental basis for *Data Sharing*

5.1. Domain Implementation

The DSC has a main goal to enable cross-*Domain Data Sharing*. To that end the UCIG aims to support *Data Sharing* across any *Domain*. In the Data Sharing Canvas, the DSC has defined a *Domain* as any number of organisations collaboratively working together to share *Data* to achieve a shared purpose. For example, this includes bilateral *Data Sharing* between two actors, or a complete *Data Sharing* ecosystem between many actors. To be applicable to all *Domains*, the UCIG is agnostic to the inner dynamics of *Domains*. This is achieved by harmonising the communication between *Domains*, without placing any requirements on the *Domain* specific implementations.

Proxies were introduced in the Data Sharing Canvas (see Chapter 4.3) as a method to enable scalability and harmonise all cross-*Domain* communications. Therefore, *Proxies* will be used as a basis for framing all the requirements described in the UCIG. However, it could be that a specific *Domain* implementation does not rely on *Proxies*. Therefore, it is important to understand the various implementation options.

Depending on the functional scope, number of actors, and maturity of a *Domain*, its implementation may be different to that of other *Domains*. To understand various *Domain* implementations, and how the UCIG may be applicable to a specific *Domain* a, non-exhaustive, overview of possible *Domain* implementations is presented here:

- **Integrated**: When a *Domain* consists of a single actor (*Data Service Consumer* or *Data Service Provider*), the actor can choose to implement all *Proxy* functionality integrated into their own systems to enable them to share data with other *Domains*.
- **Hybrid**: As the scale and scope of a *Domain* increases, it may become beneficial to decouple the modules for specific functionalities from the actors in a *Domain*. For example, the *Identification*, *Authentication* and *Authorisation* (IAA) implementation in a *Domain* can be centrally implemented and used by all actors in the *Domain*.
- **Proxy**: For a *Domain* with existing implementations and many actors, a separate *Proxy* can facilitate all required functionalities for *Data Sharing* to other *Domains*. This is achieved by translating between the existing intra-*Domain* implementations and the harmonised cross *Domain* implementation. See the Data Sharing Canvas Chapter 4.3 The Proxy Model for a detailed introduction of Proxies.

The UCIG can be applied to any Domain implementation to enable *Data Sharing* across *Domains*. Most requirements in the UCIG are formulated from the perspective of a *Proxy*. If the *Domain* implementation of one or more *Domain* is integrated, or hybrid as described above, these requirements may be directly applied to the *Data Service Consumer* or *Data Service Provider* respectively.

5.2. Additional Domains

The UCIG assumes that relevant stakeholders from *Domain* A and *Domain* B are closely collaborating to enable a bilateral *Data Sharing* use case of this archetype. Given this bilateral context, scalability of the use case to multiple *Domains* is not necessarily the priority. Therefore, the UCIG primarily describes what requirements are needed in this setting and does not require a scalable solution to all topics (such as operational, legal and governance). Only when additional overhead is minimal, scalable solutions are prescribed for the bilateral use case. Optional considerations for scalability to other *Domains* are explicitly presented in separate sub-chapters.

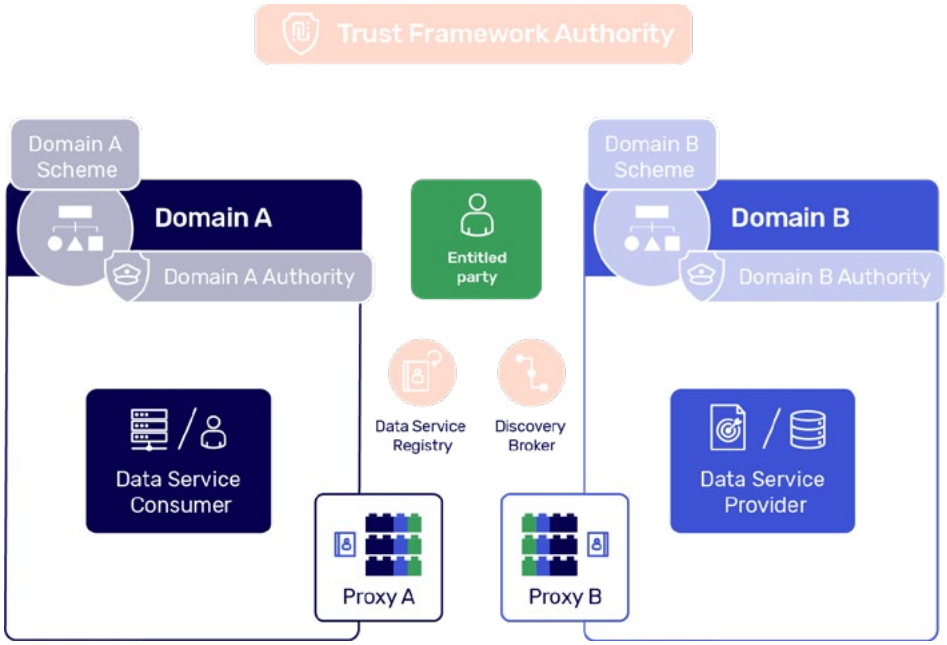
5.3. Data Service Transaction Agreement

The *Data Sharing* use case archetype presented in the UCIG describes a *Data Service Transaction* between the *Data Service Provider* and *Data Service Consumer*. To be able to execute a *Data Service Transaction*, an agreement between the actors must be established. The Data Sharing Canvas Chapter 4.1 introduces the concept of the *Data Service Transaction Agreement*. The *Data Service Transaction Agreement* is specific to a transaction and its context and can be considered the final handshake between the actors to confirm *Trust* and the mutual acceptance of the *Data Service* and its *Terms And Conditions*. More information about the *Data Service Transaction Agreement* and how it is established can be found in **Chapter 17**.

06. Roles

In the Data Sharing Canvas, the roles which play a role in *Data Sharing* use cases in general are introduced. This chapter focusses on the roles which play a role in the context of the UCIG, see **Chapter 5 Functional scope**. Roles are defined as the set of rights, obligations and expected behaviour patterns associated with an entity within a *Data Sharing* ecosystem. This chapter presents the roles and responsibilities which are in scope of the UCIG and then goes on to describe the roles that are not in scope of the UCIG but are important to understand within its context. All roles within the context of the UCIG are depicted in **Figure 2**.

Figure 2: Overview of roles within the context of the UCIG



6.1. Roles In Scope

This chapter presents the four roles which are in scope of the UCIG: the *Data Service Provider*, the *Data Service Consumer*, the *Entitled party* and *Domain Proxies*.

Data Service Provide: A *Data Service Provider* is defined as the actor that offers a *Data Service* to the *Data service consumer*. For the use case archetype, described in the UCIG, The *Data Service Provider* may only provide the *Data Service* when the *Entitled Party* has explicitly given approval. The *Data Service Provider* defines the data service and its corresponding *Terms And Conditions*.

Data Service Consumer: A *Data Service Consumer* is defined as the actor that uses a *Data Service* offered by the *Data service provider*. The *Data Service Consumer* must become aware of the *Data Service* offered by the *Data Service Provider* and accept its *Terms and conditions* before it can initiate the *Data Service*.

Entitled party: The *Entitled Party* is defined as the entity which has rights over data. The *Entitled Party* approves the *Data Service*, enabling the *Data* they have rights over to be shared from the *Data Service Provider* to the *Data Service Consumer*.

Domain Proxy: To enable multilateral *Interoperability* across *Domains*, as well as prepare for future scalability, each *Domain* requires a *Proxy*. *Proxies* are systems which are to be used by every *Domain* with the function of translating between *Domain* specific specifications and common, *Harmonised* inter-*Domain* specifications to achieve Interoperability and trust across Domains. In the remainder of this document Proxy A will refer to the *Proxy* of the *Domain* of the *Data Service Consumer*, *Proxy* B will refer to the *Proxy* of the *Domain* of the *Data Service Provider*.

6.2. Roles Out Of Scope

In **Figure 2** several roles are presented which have been greyed out. This indicates that they have been introduced in the complete *Data Sharing* ecosystem, and have been introduced in the Data Sharing Canvas, but do not play a significant role in the *Data Sharing* use case archetype of the UCIG as introduced in **Chapter 5**. These roles are briefly introduced (alphabetically) and explained why they are not in scope of this document. For more information regarding these roles, see the Data Sharing Canvas.

Data Service Broker: A *Data Service Broker* supports a *Data Service Discovery* mechanism. As the use case archetype considered in the UCIG consists of a single bilateral *Data Service*, there is no need for a technical implementation of a *Data Service Discovery* mechanism, and therefore a *Data Service Broker* is out of scope.

Data Service Registry: A *Data Service Registry* contains the necessary information to perform *Data Service Discovery* As the use case archetype considered in the UCIG consists of a single bilateral *Data Service*, there is no need for a technical implementation of a *Data Service Discovery* mechanism, and therefore a *Data Service Registry* is out of scope.

Domain Specific Scheme: A *Domain* specific *Scheme* enables *Data Sharing* within a specific *Domain*. As the use case archetype considered in the UCIG focusses on interoperability between *Domains*, *Domain* related affairs, and the *Domain* specific *Scheme* specifically are not directly relevant for the UCIG.

Domain Specific Scheme Authority: A *Domain* specific *Scheme Authority* defines and manages *Domain Specific Schemes*. As the use case archetype considered in the UCIG focusses on interoperability between *Domains*, *Domain* related affairs, and the *Domain* specific *Scheme Authority* specifically are not directly relevant for the UCIG

Trust Framework Authority: A *Trust Framework Authority* defines and manages the future *Trust Framework* for cross-*Domain Data Sharing*. The *Trust Framework Authority* is out of scope for the UCIG as this document is aimed at a single bilateral use cases which does not require a third party to enable scalable *Trust*.

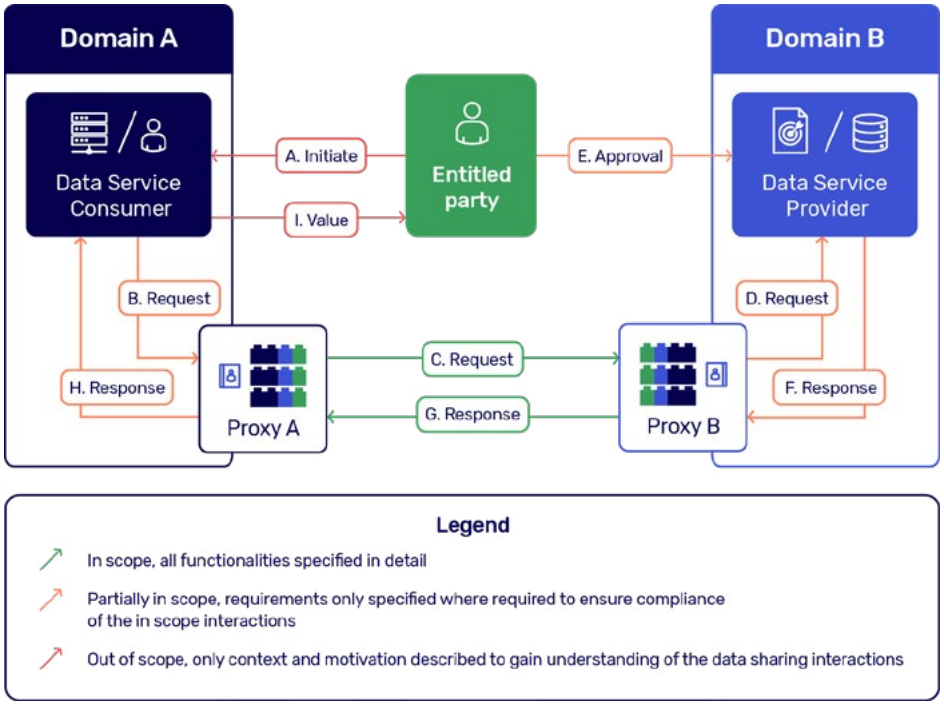
07. Interaction Model

This chapter describes the high-level functional interactions between all actors involved in the focus use case archetype of the UCIG. The interactions in scope of the UCIG are then further detailed.

7.1. Generic Functional Interaction Model

As introduced in **Chapter 5**, the UCIG focusses on a single data sharing use case archetype that can be described as: An *Entitled Party* initiates a *Data Service* at a *Data Service Consumer* in *Domain* A which in turn initiates a transaction where data can be retrieved from a *Data Service Provider* in *Domain* B based on approval given by the *Entitled Party*. The interaction model is depicted in **Figure 3** and this chapter goes on to describe each step and the scope of the UCIG.

Figure 3: Graphical overview of the interaction model



Step A Initiate: The *Entitled Party* initiates the *Data Service* at a *Data Service Consumer* and the *Data Service Consumer* are involved. As this step is specific to the *Data Service* and involved parties, it is out of scope of the UCIG.

Step B Request: The *Data Service Consumer* sends a request for *Data* to its *Domain Proxy* (*Proxy* A) to be forwarded to the target *Data Service Provider*. This step is realised within a *Domain* and may contain *Domain* specific implementations; therefore, it is partially out of scope of the UCIG.

Step C Request: *Proxy* A translates the *Data* request from a *Domain* specific request to a harmonised request and forwards it to *Proxy* B.

Step D Request: *Proxy* B translates the *Data* request from a harmonised request to a *Domain* specific request and forwards it to the *Data Service Provider*. This step is realised within a *Domain* and may contain *Domain* specific implementations; therefore, it is partially out of scope of the UCIG.

Step E Approval: The *Entitled Party* authenticates themselves and authorizes the *Data Service* to approve the *Data Service*. It is the responsibility of the *Data Service Provider* to ensure the mechanism for *Authentication* and *Authorization* of the *Entitled Party* is sufficient for the specific *Data Service* offered. However, to enable the required functionalities, some requirements are defined for the results of the approval and the user experience. Therefore, this is partially out of scope for the UCIG.

Step F Response: The *Data Service Provider* sends a response containing data to its *Domain Proxy* (*Proxy* B) to be forwarded to the source *Data Service Consumer*. This step is realised within a *Domain* and may contain *Domain* specific implementations; therefore, it is partially out of scope of the UCIG.

Step G Response: *Proxy* B translates the *Data* response from a *Domain* specific response to a harmonised request and forwards it to *Proxy* A.

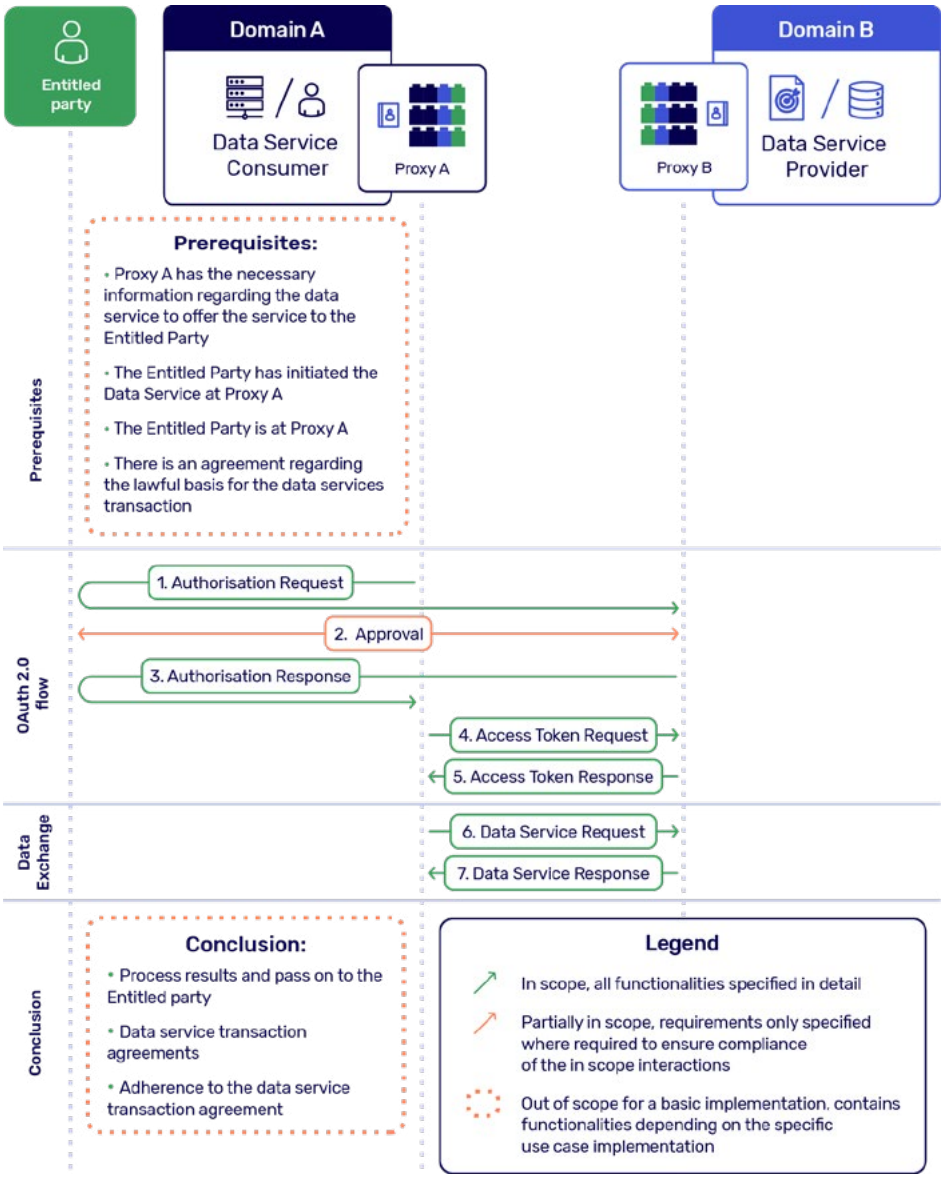
Step H Response: *Proxy* A translates the *Data* response from a harmonised response to a *Domain* specific response and forwards it to the *Data Service Consumer*. This step is realised within a *Domain* and may contain *Domain* specific implementations; therefore, it is partially out of scope of the UCIG.

Step I Value: The *Data Service Consumer* uses the result of the *Data Service* to deliver a service to the *Entitled Party*. As this step is specific to the *Data Service* and involved parties, it is out of scope of the UCIG.

7.2 Detailed Interactions

Figure 4 gives an overview of the detailed interactions (in scope) which are required to enable the functionalities introduced in Chapter 7.1. In general, four steps: Prerequisites, OAuth 2.0 flow, Data Exchange, and Conclusion are defined to enable the functionalities.

Figure 4: Detailed interaction model for a Data Service in the context of the Data Sharing use case archetype of the UCIG



7.2.1. Prerequisites

Before a *Data Service* can be consumed, several prerequisites must be met. The exact implementation of these functionalities is dependent on the use case.

Appendix I.I presents additional agreements for enabling scalability in *Domain* A for these prerequisites.

REQ 1. *Proxy* B SHALL define all relevant aspects of the *Data Service* on behalf of the *Data Service Provider*

REQ 2. *Proxy* A SHALL fully understand all relevant aspects of the defined *Data Service* such that *Proxy* A can ensure adherence to the *Data Service Transaction Agreement*

REQ 3. There SHALL be an agreement regarding the lawful basis for the *Data Service* between *Proxies*

REQ 4. Proxy A SHALL have all relevant information regarding the *Data Service* to inform the *Entitled Party* about their role in consuming the *Data Service*

REQ 5. The *Entitled Party* SHALL trigger Proxy A to invoke the *Data Service* at *Proxy* A

REQ 6. Proxy A SHALL be able to control the interaction with the user agent of the Entitled Party

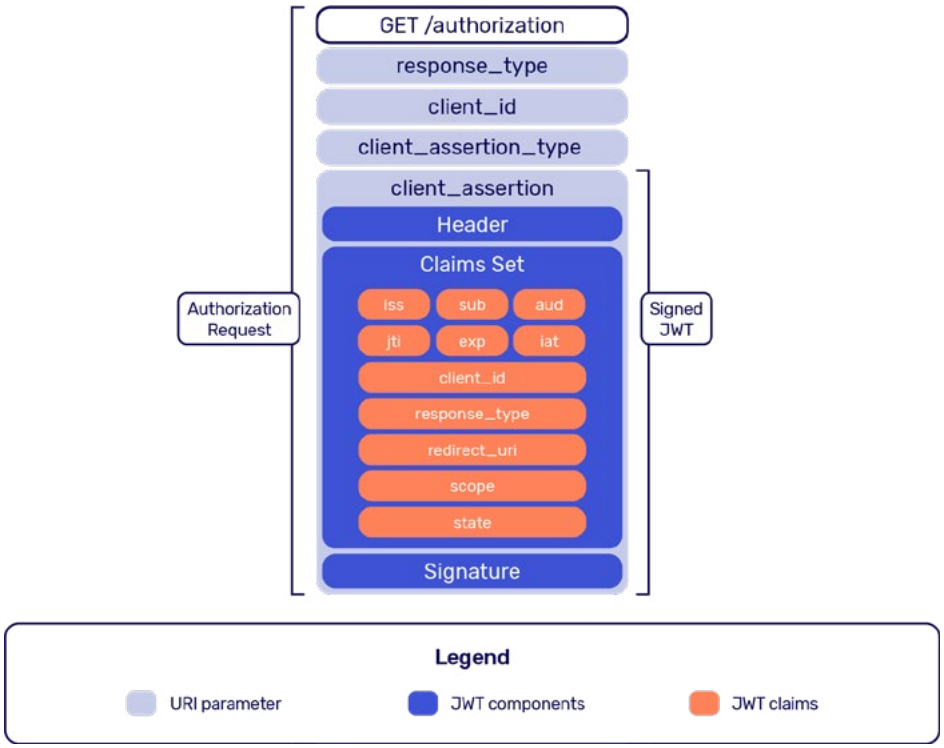
7.2.2. Step 1. Authorization Request

The Authorization Request is required such that the *Entitled Party* can *Authorise* the *Data Service* by approving the *Data Service*. The Authorization Request consists of the *Entitled Party* (specifically, their user agent) being redirected to *Proxy* B with a request including specific parameters as defined below.

REQ 7. *Proxy* A SHALL redirect the *Entitled Party* to the *Proxy* B /authorization endpoint according to OAuth 2.0 Authorization Code Grant, Chapter 4.1.1, Authorization Request, using a signed JWT according to the JWT bearer profile

Specific requirements on top of OAuth 2.0 and the JWT bearer profile, or requirements that are deemed useful to include for clarity are included below. **Figure 5** gives an overview of an Authorization Request and the contained signed JWT.

Figure 5: Overview of an Authorization Request



As a result of the choice made in **REQ 7** and applying **REQ 37** to the /authorization endpoint, *Proxy* B must support the URI parameters as described in **Table 2**. Similarly, *Proxy* A must include the URI parameters in the Authorization Request as described in **Table 2**. When *Proxy* B receives an Authorization Request, it must validate the signed JWT received in the client_assertion parameter.

Table 2: Summary of the URI parameters for an Authorization Request

Authorization Request URI parameters		Description
response_type	Required	must be set to "code"
client_id	Required	As stated in iSHARE, this must be an EORI identifier
client_assertion_type	Required	As stated in OAuth 2.0 JWT bearer profile this value must be set to "urn:ietf:params:oauth:client-assertion-type:jwt-bearer"
client_assertion	Required	As stated in OAuth 2.0 JWT bearer profile this value must contain a single signed JWT conform to the generic UCIG requirements (See **Chapter 10.2**) and additionally contain the JWT claims in **Table 2**

Table 3: Summary of the additional JWT claims in the client_assertion for an Authorization Request

Authorization Request JWT claims		Description
client_id	Required	As stated in iSHARE, this must be an EORI identifier
response_type	Required	must be set to "code"
redirect_uri	Required	As stated in OAuth 2.0
scope	Required	As stated in OAuth 2.0
state	Required	As stated in OAuth 2.0

In OAuth 2.0 scope values are used to specify what access privileges are being requested for *Access Tokens*. The scope values associated with the *Access Tokens* determine what functionalities will be available when they are used to access OAuth 2.0 protected endpoints. As the values for scope and the related functionalities are highly dependent on the *Data Service*, these are not defined in the UCIG, but should be defined by *Proxy* B as part of the Prerequisites (see **REQ 1**). As captured in **REQ 2**, the scope must be fully understood by *Proxy* A as this heavily influences the consequences of the *Data Service Transaction Agreement*.

Note: values of scope are statically defined for the *Data Service*, this may be limiting for some use cases. A draft RFC is under development enabling fine grained, dynamic parameters in the authorization request.

REQ 8. *Proxy* A SHALL only use scopes which are understood and supported by Proxy B

The relevant part of an example Authorization Request is presented below (with indents and extra line breaks for display purposes only):

```
GET /authorization?
   response_type=code&
   client_id=EU.EORI.NL123456789&
   client_assertion_type=urn:ietf:params:oauth:client-
   assertion.Type:jwt-bearer&
   client_assertion=eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
   eyJpc3Mi[...omitted for brevity...].
   cC4hiUPo[...omitted for brevity...] HTTP/1.1
Host: ProxyB.example.com
```

Note: A possible future alternative to the OAuth JWT bearer profile to enable client authentication is the recently released RFC9101. In the UCIG the above implementation was chosen for alignment with iSHARE.

7.2.3. Step 2. Approval

Approval is given by the *Entitled Party* at *Proxy* B to give provide part of the lawful basis of the *Data Service*, as detailed in **Chapter 16**. Thereby, the approval mechanism is an element of the decision-making process that *Proxy* B performs to decide to authorise the requested *Data Service*. As stated in **REQ 1**, *Proxy* B must define all relevant aspects of the Data Service. This includes a mechanism which is used by *Proxy* B to perform *Identification* and *Authentication* of the *Entitled Party* and verifying the *Authorisation* of the *Entitled Party* to approve the *Data Service*. All these mechanisms are specific to the use case specific situation and therefore, the UCIG does not specify detailed requirements for this step.

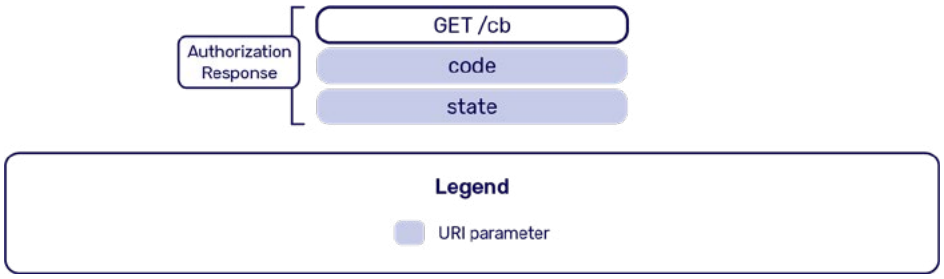
7.2.4. Step 3. Authorization Response

The *Authorization* Response is the response to the Authorization Request after approval given by the Entitled Party. The Authorization Response consists of the *Entitled Party* (specifically, their user agent) redirect including specific parameters as defined below.

REQ 9. *Proxy* B SHALL redirect the *Entitled Party* back to the *Proxy* A according to OAuth 2.0 Authorization Code Grant, Chapter 4.1.2, Authorization Response

Specific requirements on top of OAuth 2.0, or requirements that are deemed useful to include for clarity are included below. **Figure 6** gives an overview of the Authorization Response.

Figure 6: Overview of the Authorization Response



As a result of the choices made in **REQ 9**, *Proxy* A must support the URI parameters as described in **Table 4**, other URI parameters must not be used. Similarly, *Proxy* B must include the URI parameters in the Authorization Response as described in **Table 4**.

Proxy B must handle errors according to OAuth 2.0 chapter 4.1.2.1. *Proxy* B must not supply an *Access Token* in the *Authorization* Response (these must only be supplied via the designated /token endpoint as described in **Chapter 7.2.6 Step 5. Access Token**). *Proxy* A must validate the state response field is equal to that sent in an Authorization Request and can be related to a known session with an *Entitled Party* before *Proxy* A proceeds.

Table 4: Authorization Response URI parameters

Authorization Response URI parameters		Description
code	Required	As stated in OAuth 2.0
state	Required	As stated in OAuth 2.0

The relevant part of an example Authorization Response is presented below:

```
GET /cb?code=SplxlOBeZQQYbYS6WxSbIA&state=xyz
Host: ProxyA.example.com
```

7.2.5. Step 4. Access Token Request

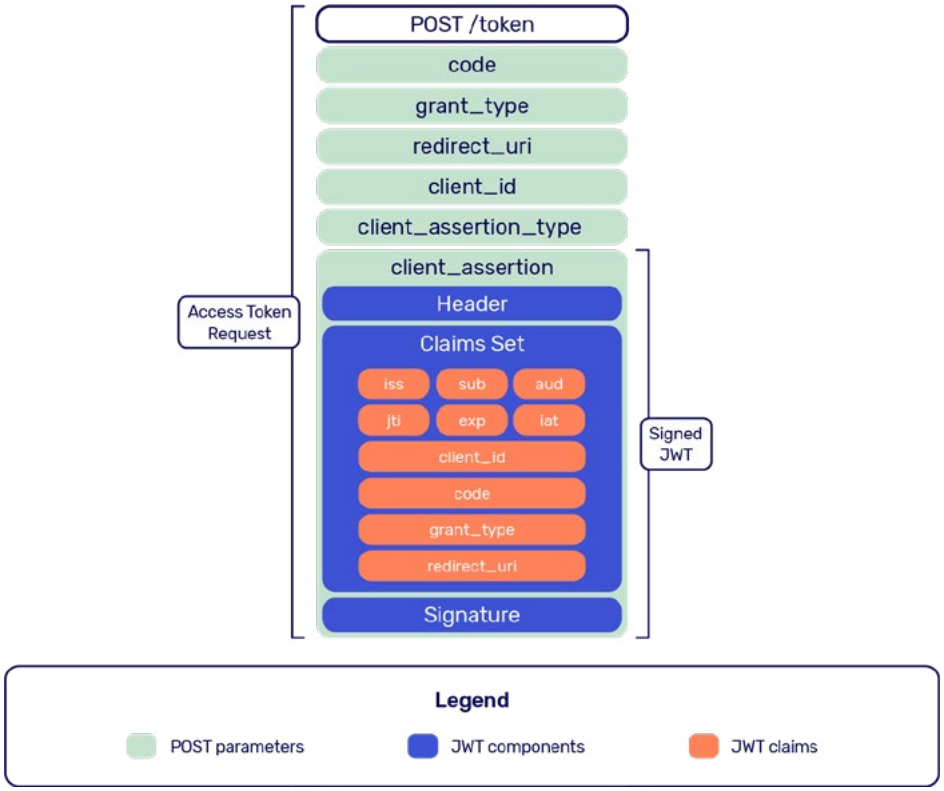
To obtain an *Access Token*, *Proxy* A sends an Access Token Request to the /token endpoint of *Proxy* B to obtain an Access Token Response.

Note: Similarly, to the Authorization Request, and Data request, the Access Token Request is signed. This is not strictly necessary from a security perspective (as non-repudiation is not a requirement), but is a choice made for convenience and consistency such that the security of all defined endpoints is identical.

REQ 10. *Proxy* A SHALL request an *Access Token* from Proxy B according to OAuth 2.0 Authorization Code Grant, Chapter 4.1.3, Access Token Request

Specific requirements on top of OAuth 2.0, or requirements that are deemed useful to include for clarity are included below. **Figure 7** gives an overview of an Access Token Request.

Figure 7: Overview of an Access Token Request



As a result of the choice made in **REQ 10** and applying **REQ 37** to the /token endpoint, *Proxy* B must support the HTTP headers as described in **Table 5**. Similarly, *Proxy* A must include the HTTP headers in the Access Token Request as described in **Table 5**. When *Proxy* B receives an Access Token Request, it must validate the client credentials received in the client_assertion.

Table 5: Summary of the request parameters for an Access Token Request

Access Token Request parameters		Description
code	Required	As stated in OAuth 2.0
grant_type	Required	must be set to "authorization code"
redirect_uri	Required	As stated in OAuth 2.0 this value must be identical to that in the Authorization Request
client_id	Required	As stated in iSHARE, this must be an EORI identifier
client_assertion_type	Required	As stated in OAuth 2.0 JWT bearer profile this value must be set to "urn:ietf:params:oauth:client-assertion-type:jwt-bearer"
client_assertion	Required	As stated in OAuth 2.0 JWT bearer profile this value must contain a single signed JWT conform to the generic UCIG requirements (See **Chapter 10.2**) and additionally contain the JWT claims in **Table 6**

Table 6: Summary of the additional JWT claims in the client_assertion for an Access Token Request

Access Token Request JWT claims		Description
client_id	Required	As stated in iSHARE, this must be an EORI identifier
code	Required	As stated in OAuth 2.0
grant_type	Required	must be set to "authorization code"
redirect_uri	Required	As stated in OAuth 2.0 this value must be identical to that in the Authorization Request

An example Access Token Request is presented below (with indentation and extra line breaks for display purposes only):

```
POST /token HTTP/1.1
Host: ProxyB.example.com
grant_type=authorization_code&
  code=SplxlOBeZQQYbYS6WxSbIA
  redirect_uri=https://ProxyA.example.com/cb
  client_id=EU.EORI.NL123456789&
  client_assertion_type=urn:ietf:params:oauth:client-
    assertion.Type:jwt-bearer&
  client_assertion=eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
    eyJpc3Mi[...omitted for brevity...].
    cC4hiUPo[...omitted for brevity...]
```

7.2.6. Step 5. Access Token Response

The Access Token Response is the response to the Access Token Request. In a happy flow scenario, *Proxy* B processes the Access Token Request and determines that an ACCESS CODE will be provided to give *Proxy* A the *Authorisation* to request the *Data Service*.

REQ 11. *Proxy* B SHALL issue an Access Token to Proxy A according to OAuth 2.0 Authorization Code Grant, Chapter 4.1.4, Access Token Response

Specific requirements on top of OAuth 2.0, or requirements that are deemed useful to include for clarity are included below.
Proxy B must include the HTTP headers in the Access Token Response as described in **Table 7** other headers must not be included. Proxy B must handle errors according to OAuth 2.0 Chapter 5.2. Furthermore, *Proxy* B must include an *Access Token* attribute in the HTTP payload, unless no Authorization was given by the *Entitled Party* in Step 2. Approval.

Table 7: HTTP headers in an Access Token Response

Access Token Response HTTP headers		Description
content-type	Required	As stated in OAuth 2.0
cache-control	Required	Must be set to "no-store" conform to the generic UCIG requirements (See **Chapter 9.2**)
pragma	Required	Must be set to "no-cache" conform to the generic UCIG requirements (See **Chapter 9.2**)

An example Access Token Response is presented below:

```
HTTP/1.1 200 OK
content-Type: application/json;charset=UTF-8
cache-Control: no-store
Pragma: no-cache
{
   "access_token":"2YotnFZFEjr1zCsicMWpAA",
   "token_type":"example",
   "expires_in":3600
}
```

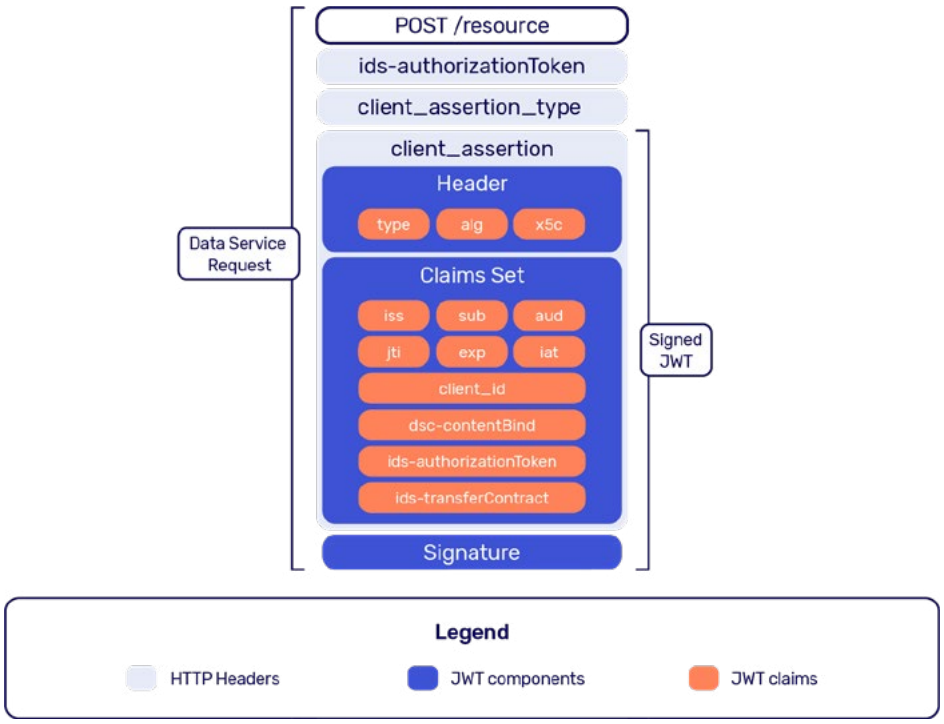
7.2.7. Step 6. Data Service Request

For *Proxy* A to make use of the *Data Service* provided by *Proxy* B, it must send a Data Service Request including the obtained *Access Token* to a /resource endpoint of *Proxy* B. The /resource endpoint is used as an example, depending on the use case implementation, this could also be multiple endpoints.

REQ 12. *Proxy* A SHALL request the *Data Service* at *Proxy* B in conformance with the specifications of *Proxy* B

Figure 8 gives an overview of the mandatory parts of a Data Service Request.

Figure 8: Overview of the HTTP headers of a Data Service Request



Applying REQ 37 to the /resource endpoint results in the following requirements.

REQ 13. *Proxy* A SHALL include at least the HTTP headers in the Authorization request as described in Table 8

REQ 14. *Proxy* B SHALL support at least the HTTP headers described in Table 8

REQ 15. Proxy B SHALL validate the client_assertion in the Data Service Request

Table 8: Summary of the mandatory HTTP headers for a Data Service Request endpoint

Data Service Request HTTP headers		Description
ids-authorizationToken	Required	This claim must be set to the value of the *Access Tokens* received in the Access Token Request Message. IDSA defines this as optional for the use of a separate *Authorisation* service. As this is the case here, this claim mandatory in the UCIG
client_id	Required	As stated in iSHARE, this must be an EORI identifier
client_assertion_type	Required	As stated in OAuth 2.0 JWT bearer profile this value must be set to "urn:ietf:params:oauth:client-assertion-type:jwt-bearer"
client_assertion	Required	As stated in OAuth 2.0 JWT bearer profile this value must contain a single signed JWT conform to the generic UCIG requirements (See **Chapter 10.2**), and additionally contain the JWT claims in **Table 6**

Table 9: Summary of the additional JWT claims in the client_assertion for a /resource endpoint

Data Service Request JWT claims		Description
client-id	Required	As stated in iSHARE, this must be an EORI identifier
dsc-contentBind	Required	This claim must contain a SHA256 hash over the HTTP body to bind the content of the request to the Data Service Request
ids-authorizationToken	Required	This claim must be set to the value of the *Access Token* received in the Access Token Request Message. IDSA defines this as optional for the use of a separate *Authorisation* service. As this is the case here, this claim mandatory
ids-transferContract	Optional	As stated in IDSA this value MAY contain a URI to the contract which is (or will be) the legal basis of the data transfer. For possible future scalability, See **Appendix I.III.II**, for options to enable scalable dynamic contracts.

Depending on the use case specific context, the Data Service Request may require use case specific fields. This should be determined during the design of the specific use case and can be included in the Data Service Request Payload.

REQ 16. *Proxy* A SHALL include any use case specific fields (as decided in the use case design) in the HTTP Body

REQ 17. *Proxy* B SHALL validate that the *Access Token* in the ids-authorizationToken field corresponds to that issued in the Access Token Response

REQ 18. *Proxy* B SHALL validate all aspects of the Data Service Request

An example Data Service Request is presented below (with indentation and extra line breaks for display purposes only):

```
POST server.ProxyB.com/resource
ids-authorizationToken: 2YotnFZFEjr1zCsicMWpAA
client_id: EU.EORI.NL123456789
client_assertion_type: urn:ietf:params:oauth:client-
   assertion.Type:jwt-bearer
client_assertion: eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
   eyJpc3Mi[...omitted for brevity...].
   cC4hiUPo[...omitted for brevity...] HTTP/1.1
Host: server.ProxyA.com
```

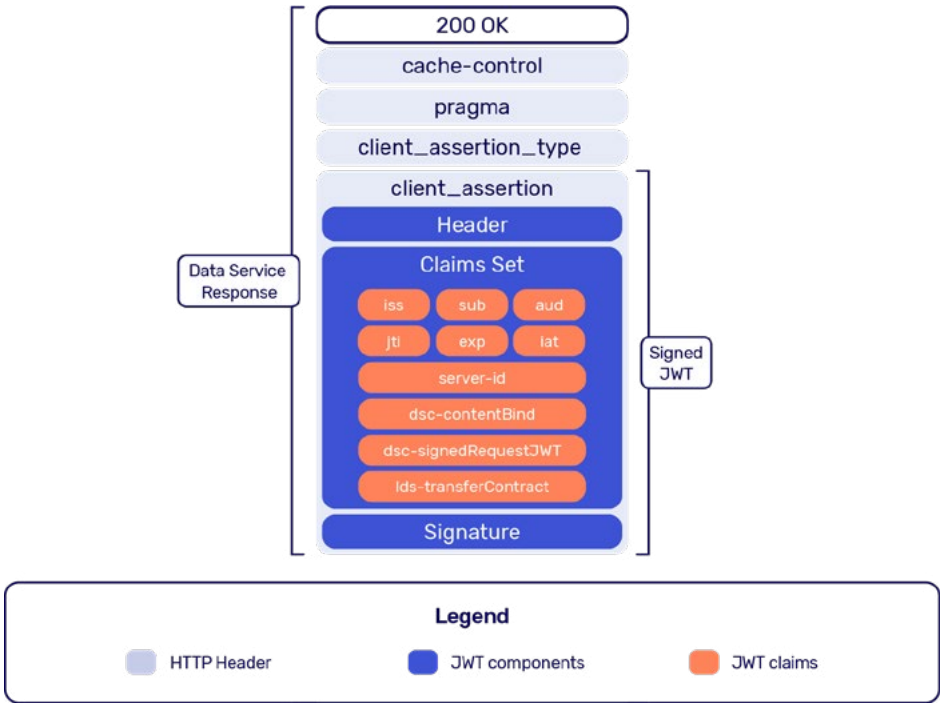
7.2.8. Step 7. Data Service Response

The Data Service Response is the response to the Data Service Request. Due to the value being shared in the Data Service Response based on the Data Service Request, it is required to irrevocably bind this Response to the Request and the *Data Service Provider*. Therefore, *Proxy* B must authenticate itself in sending the response such that *Proxy* A has the assurance in the source of the *Data Service*.

REQ 19. *Proxy* B SHALL respond to a Data Service Request in line with a Data Service Response

Figure 9 gives an overview of the HTTP header of a Data Service Response.

Figure 9: Overview of the HTTP headers of a Data Service Response



Error handling based on the IANA HTTP Status Code Registry is fully aligned with iSHARE.

REQ 20. Proxy B SHALL handle errors according to IANA HTTP Status Code Registry
REQ 21. *Proxy* B SHALL include the requested data in the HTTP body

At the time of writing, there is no standard mechanism to enable the non- repudiation of a HTTP Response. Therefore, as a proprietary implementation the UCIG suggests identical implementations to the Data Service Request, i.e. including a signed client assertion with a digest of the HTTP body. If new developments enable this functionality, or specific use cases implement their own mechanism for this, this is deemed sufficient. Furthermore, applying **REQ 37** to the Data Service Response, results in the following requirements.

REQ 22. Proxy A SHALL support the HTTP headers described in **Table 10**
REQ 23. Proxy A SHALL validate server credentials present in the Authorization Request
REQ 24. *Proxy* B SHALL include the HTTP headers in the Authorization Request as described in **Table 10**

Table 10: Summary of the HTTP headers for a Data Service Response

Data Service Response HTTP headers		Description
cache-control	Required	Must be set to "no-store" conform to the generic UCIG requirements (See **Chapter 9.2**)
pragma	Required	Must be set to "no-cache" conform to the generic UCIG requirements (See **Chapter 9.2**)
server_id	Required	In accordance with iSHARE, this must be an EORI identifier
server_assertion_type	Required	In line with the client_assertion_type, this value must be set to "urn:ietf:params:oauth:client-assertion-type:jwt-bearer"
server_assertion	Required	In line with the client_assertion, this value must contain a single signed JWT conform to the generic UCIG requirements (See **Chapter 10.2**) and additionally contain the JWT claims in **Table 11**

Table 11: Summary of the additional JWT claims in the server_assertion in a Data Service Response

Data Service Response JWT claims		Description
server-id	Required	Identical to as stated in iSHARE for the client-id, this must be an EORI identifier
dsc-contentBind	Required	This claim must contain a SHA256 hash over the HTTP body to bind the content of the request to the *Data Service* REQUEST. This irrevocably binds the HTTP header with the HTTP body to ensure non-repudiation.
dsc-signedRequestJWT	Required	This claim must contain a copy of the payload of the client_assertion received in the *Data Service* Request for the purpose of irrevocably binding the Data Service Response to the Data Service Request
ids-transferContract	Required	As stated in IDSA. This claim must minimally contain a URI reference to the agreement underlaying the data service. Depending on the use case, *Data Service Provider* specific details may need to be supplemented to the requested contract, see **Appendix I.III.II**, for more information

An example Data Service Response is presented below (with indentation and extra line breaks for display purposes only):

```
HTTP/1.1 200 OK
cache-control: no-store
pragma: no-cache
server_id: EU.EORI.NL987654321
server_assertion_type: urn:ietf:params:oauth:client-
   assertion.Type:jwt-bearer&
server_assertion: eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
   eyJpc3Mi[...omitted for brevity...].
   cC4hiUPo[...omitted for brevity...]
```

7.2.9. Conclusion

After a *Data Service* has taken place, depending on the use case implementation (see **Chapter 5**), any set of concluding actions may be performed. Most of this is out of scope of the UCIG, however the following requirement does apply to legally bind Proxy A to the *Data Service Transaction Agreement*.

REQ 25. *Proxy* A must ensure adherence to the *Data Service Transaction Agreement*, see **Chapter 17** for more information.

08. User Experience

8.1. Introduction

As described in **Chapter 7.2.3**, the approval mechanism is not in scope for the UCIG. It is likely that this will require Human to Machine (H2M) interaction. Therefore, a use case can reference the generic aspects of the iSHARE user interface requirements as guidelines to enable a consistent user experience.

Source: iSHARE User interface requirements

For all Human to Machine interactions, as in primary use case 2 and 3, an interface is required. This interface MUST comply with the following guidelines:

- The name of the legal entity that provides a broker service or identity provisioning service MUST be clearly visible;
- During the process of authentication, information not directly relating to the identity provision process or supporting the identity provision process MAY NOT be present. Links to websites irrelevant to the identity provisioning process or advertisements MAY NOT be present;
- Parties facilitating the identity provision process MAY use their own corporate styling and logos;
- The iSHARE brand MUST be shown during the identity provision process. Showing the iSHARE brand MUST be in line with iSHARE communication guidelines;
- Human Service Consumer that are being identified through the use of a browser MUST be able to verify the URL and used SSL certificate during all steps of identity provisioning process.

Section C.

Technical Topics

09. Generic Technical Standards

This chapter introduces the various generic technical components used in the UCIG. Where these are used throughout the document they are detailed and referenced to explicitly. The UCIG defines APIs to enable communication and in turn data sharing between actors. The APIs are RESTful, and JSON is used for structuring data. TLS is used for secured HTTP communication, and authorization mechanisms are based on X.509 certificates and OAuth 2.0.

9.1. RESTful APIs

An API (Application Programming Interface) is a technical interface, consisting of a set of protocols and data structuring standards ('API specifications') which enables computer systems to directly communicate with each other. APIs are used in to facilitate direct and real-time communication between different actors, eliminating the need for a central platform. Therefore, in the UCIG APIs are designed to be RESTful, and JSON is used for structuring data.

Representational State Transfer (REST) is an architectural style for building systems and services, systems adhering to this architectural style are commonly referred to as RESTful systems. REST itself is not a formal standard, but it is an architecture that applies various common technical standards such as HTTP, JSON and URI. RESTful systems can process common HTTP operations, such as GET, POST and DELETE.

A RESTful API indicates that the API architecture follows REST constraints. Constraints restrict the way that servers respond and process client requests, to preserve the design goals which are intended by applying REST. Goals of REST are, among others, performance, and scalability. It is possible to deviate from the RESTful architecture when required by a specific use case.

Following iSHARE and IDSA, the UCIG supports data services built upon RESTful APIs.

> **Source:** iSHARE API
>
> APIs are used in iSHARE to facilitate direct and realtime communication between different parties, eliminating the need for a central platform. iSHARE APIs are designed to be RESTful, and JSON is used for structuring data. iSHARE prescribes caching requirements relating to the use of APIs in various situations.

> **Source:** iSHARE RESTful
>
> Within iSHARE RESTful architectural principles MUST be applied to the APIs that are specified. A RESTful API indicates that the API architecture follows REST constraints. Constraints restrict the way that servers respond and process client requests, in order to preserve the design goals which are intended by applying REST. Goals of REST are, among others, performance, and scalability. Both are of utmost importance in iSHARE.

The UCIG is aligned with iSHARE on the topic of RESTful implementations. However, if a specific use case has a specific reason to deviate from a RESTful implementation to enable their data service (e.g. due to legacy constraints), this is possible. Therefore, the iSHARE requirement is lowered to a SHOULD.

REQ 26. RESTful architectural principles SHOULD be applied to specified APIs

9.2. Caching

Often data is temporarily stored on a medium different from the original data storage location, to enable faster access to the data. This is called caching and is a way to boost performance efficiency.

Source: iSHARE Caching

For every API exposed under iSHARE caching MUST be made explicit to the API consumer.

If a response is not cacheable it MUST contain the following headers:

Cache-Control: no-store
Pragma: no-cache

If a response is cacheable it MUST contain the following headers:

Cache-Control: max-age=31536000
Note: max-age MAY vary

The UCIG is fully aligned with iSHARE on the topic of caching. This is considered in the following generic requirements for the UCIG.

REQ 27. For every API exposed caching SHALL be made explicit to the API consumer.
REQ 28. If a response is not cacheable it SHALL contain the following headers:
cache-control: no-store
pragma: no-cache
REQ 29. If a response is cacheable, it SHALL contain the following headers:
cache-control: max-age=31536000
Note: max-age MAY vary

9.3. HTTP(S) And TLS

HyperText Transfer Protocol (HTTP) is a communication protocol for computer networks. The mocent version of the HTTP specification can be found at w3.org.

Transport Layer Security (TLS) is a cryptographic protocol that describes communication security for computer networks. It is used to secure the HTTP protocol, resulting in HTTPs (HTTP Secure). The most recent version of the specification can be found in RFC 5246.

Source: iSHARE HTTP(s)

iSHARE authentication/authorization data is generally transferred in HTTP Headers. These headers can become very large when containing multiple encrypted certificates or JWT's. iSHARE parties SHOULD configure their web servers to accept HTTP headers of 100K length to minimise implementation impact on current services.

After sending a HTTP request to a server, the server responds with (among others) a Status Code which indicates the outcome of the request made to the server.

HTTP verb	CRUD	Entire Collection (e.g. parties)	Specific Item (e.g. /parties/ {id})
POST	Create	201 (Created), 'Location' header with link to / customers/{id} containing new ID.	404 (Not Found), 409 (Conflict) if resource already exists.
GET	Read	200 (OK), list of customers. Use pagination, sorting and filtering to navigate big lists.	200 (OK), single customer. 404 (Not Found) if not found or invalid.
PUT	Update Replace	404 (Not Found), unless you want to update/ replace every resource in the entire collection.	200 (OK) or 204 (No Content). 404 (Not Found) if not found or invalid.

HTTP verb	CRUD	Entire Collection (e.g. parties)	Specific Item (e.g. /parties/ {id})
PATCH	Update Modify	404 (Not Found), unless you want to modify the collection itself.	200 (OK) or 204 (No Content). 404 (Not Found) if not found or invalid.
DELETE	Delete	404 (Not Found), unless you want to delete the whole collection, not often desirable.	200 (OK). 404 (Not Found) if not found or invalid.

The UCIG is fully aligned with iSHARE on the topic of HTTP(s). This is considered in the following generic requirement for the UCIG.

REQ 30. *Proxies* SHOULD configure their web servers to accept HTTP headers of 100K length to minimise implementation impact on current services.

9.4. JSON

JSON (JavaScript Object Notation) is a data formatting standard. JSON is an open standard data format that does not depend on a specific programming language. This compact data format makes use of human-readable text to exchange data objects (structured data) between applications and for data storage. The most recent version of the JSON specification can be found at json.org.

9.5. JWT

JWT (JSON Web Token) is an open standard that defines a compact and self- contained way for securely transmitting information between parties as a JSON object. JWTs can be signed using the JSON Web Signature (JWS) standard to ensure non-repudiation of these claims. **Chapter 10.2** describes the use of JWTs in the UGIC and the specifications for JWTs, while the official standard can be found in RFC 7519.

9.6. PKI And X.509

A PKI (Public Key Infrastructure) is a system for distribution and management of digital keys and certificates, which enables secure authentication of parties interacting with each other.

In cryptography, X.509 is a standard defining the format of public key certificates. X.509 certificates are used in many Internet protocols, including TLS/SSL, which is the basis for HTTPS, the secure protocol for browsing the web. The most recent version of the X.509 specification can be found at RFC 5280.

> **Source:** iSHARE X.509
>
> X.509 is used as a standard defining the format of public key certificates.

The UCIG is fully aligned with iSHARE and IDSA on the topic of certificates. This is considered in the following generic requirement for the UCIG.

REQ 31. X.509 SHALL be used as a standard for PKI certificates

9.7. OAuth 2.0 And Not OpenID Connect

OAuth is a widely used security standard that enables secure access to protected resources in a fashion that is friendly to web APIs. It is a delegation protocol that provides authorization across systems. The UCIG uses the OAuth 2.0 protocol for providing access tokens when requesting access to a service within a use case based on the UCIG. The most recent version of the OAuth 2.0 specification can be found in RFC 6749.

Source: iSHARE OAuth 2.0

iSHARE uses the OAuth 2.0 protocol for authenticating parties and providing access tokens when requesting access to a service within iSHARE.

Source: iSHARE OpenID Connect 1.0

Just as in OAuth 2.0, iSHARE deviates from the original standard to allow for information exchange with previously unknown parties. Identity Providers need to provide API access to iSHARE participants based on whitelisted PKI, clients need not to be pre-registered at an Identity Provider.

The *Data Service* in the context of the UCIG, requires an *Access Token* as proof of *Authorisation* before the service is provided. Proxy A must request an *Access Token* at Proxy B for this purpose. This is facilitated in a process based on OAuth 2.0.

The UCIG is aligned with iSHARE on the topic of OAuth 2.0, see the OAuth Flow in **Chapter 7.2** where the iSHARE OAuth implementation is tailored to the specific use case archetype of the UCIG as described in **Chapter 5**. In iSHARE elements from OpenID Connect were added on top of an OAuth 2.0 implementation to enable client authentication when requesting an Access Token. Since then, RFC 7523 (JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants) has come into place filling this gap in OAuth. Therefore, the UCIG makes use of this and OpenID Connect is out of scope for the UCIG. However, this implementation is still fully in line with iSHARE.

Proxy interfaces are built on top the OAuth 2.0 Authorization Framework, authorization code grant type. Therefore, *Proxies* should inherit the specifications as stated in the OAuth 2.0 standard for that flow.

REQ 32. The *Proxy* SHALL implement, use, and support an interface that complies with the OAuth 2.0 (RFC 6749) Chapter 4.1. Authorization Request, specifications for the purpose of requesting an access token

REQ 33. The *Proxy* SHALL implement, use, and support an interface that complies with the OAuth 2.0 JWT bearer profile (RFC 7523) specifications for the purpose of client authentication when requesting access tokens.

Note: Since the UCIG does not make assumptions on the approval mechanism or on how the *Entitled Party* is authenticated during that process, the UCIG does not rely on/ prescribe the use of OpenID Connect.

9.8. UTC

The UNIX timestamp is a way to track time as a running total of seconds. This Unix formatting of UTC point in time technically does not change no matter where you are located on the globe. This is very useful to computer systems for tracking and sorting dated information in dynamic and distributed applications both online and client side.

The UCIG is fully aligned with iSHARE on the topic of UTC. This is considered in the following generic requirements for the UCIG.

> **Source:** iSHARE UTC
>
> In iSHARE all dates and times MUST be communicated in UTC time. All dates and times MUST be formatted in the Unix timestamp format.

REQ 34. All dates and times SHALL be communicated in UTC time.

REQ 35. All dates and times SHALL be formatted in the Unix timestamp format.

Note: This choice deviates from the XSD DateTimeStamp used in IDS to communicate time.

10. IAA Mechanisms

10.1. Identifiers

For standardised identification for all organisations involved in a use case, an EORI identifier ("Economic Operators Registration and Identification number") must be used. This is a common type of identification number across the EU and is followed by iSHARE. Therefore, by using this in the UCIG, the UCIG is aligned with iSHARE.

REQ 36. All identifiers used SHALL be EORI numbers

10.2. Generic JWT Requirements

The use of JWTs in the UCIG is based on the implementation in iSHARE and are thereby fully aligned.

Source: iSHARE JWT

A JSON Web Token (JWT) is used when non-repudiation between parties is required. A statement, of which the data is encoded in JSON, is digitally signed to protect the authenticity and integrity of the statement. iSHARE uses signed JWTs in the following ways:

- In a request for an OAuth Access Token or an OpenID Connect ID token the client sends a signed JWT. The client is authenticated based on the verification of the JWT's signature.
- Delegation evidence is presented as a signed JWT. The signature of the Authorization Registry or Entitled Party provides proof to other parties.
- In a response from a server iSHARE metadata is presented as a signed JWT. The signature is used to bind the iSHARE metadata (such as license information) in the JWT to the content of the response.
- A service from an iSHARE Service Provider MAY require a request to be signed.

- If there is potential that a JWT could be intercepted by intermediaries, JSON Web Encryption (JWE) may be used. This is explicitly used in Human2Machine interaction in the OpenID flow.

The UCIG is aligned with iSHARE on the topic of JWT requirements for the OAuth flow and Data Exchange and tailored to the context of the UCIG data sharing archetype as described in **Chapter 5**:

- ID tokens and delegation are out of scope for the UCIG *Data Sharing* archetype
- Instead of iSHARE *Metadata*, the *Data Service Transaction Agreement* is used in the UCIG
- *Proxies* must require signed requests (instead of MAY) to authenticate the client.
- Binding of the content of a Data Service Response and the *Metadata* of a Data Service Response is described in section 7.2.8

REQ 37. *Proxies* SHALL require all JWTs to be signed for the purpose of *Authentication*, Integrity, and non-repudiation

REQ 38. *Proxies* SHALL use a signed JWT in all requests. The *Proxy* is authenticated based on the verification of the JWT's signature

REQ 39. *Proxies* MAY use JSON Web Encryption (JWE) if there is potential that a JWT could be intercepted by intermediaries

The following chapters describe the requirements for a signed JWT for the *Data Sharing* archetype covered in the UCIG.

10.2.1. JWT Signing

Source: iSHARE JWT

All iSHARE JWTs MUST be signed using the JSON Web Signature (JWS) standard which can be found at RFC 7515.

The UCIG is aligned with iSHARE on the topic of JWT signing.

REQ 40. *Proxies* SHALL sign all JWTs using the JSON Web Signature (JWS) standard as defined in RFC 7515

REQ 41. For the purpose of signing, *Proxies* SHALL allow the use certificates issued by a widely accepted certificate authority, however, if both *Proxies* agree, self-signed certificates MAY be used in a bilateral setup

10.2.2. JWT Header

Source: iSHARE JWT

- Signed JWTs MUST use and specify the RS256 algorithm in the alg header parameter.
- Signed JWTs MUST contain an array of the complete certificate chain that should be used for validating the JWT's signature in the x5c header parameter up until an Issuing CA is listed from the iSHARE Trusted List. Certificates MUST be formatted as base64 encoded DER.
- The certificate of the client MUST be the first in the array, the root certificate MUST be the last.
- Except from the alg, typ and x5c parameter, the JWT header SHALL NOT contain other header parameters.

The UCIG is aligned with iSHARE on the topic of JWT headers. However, certificate chains are omitted in the UCIG as this is out of scope. In addition to the iSHARE requirements, the UCIG defines a minimum key length for the signing ofJWTs

REQ 42. Signed JWTS SHALL use and specify the RS256 encryption for signing JWTs

REQ 43. Proxies SHALL only use X.509 certificates with a minimum key length of 2048 bits and a validity period of two years maximum, for the purpose of signing

REQ 44. The JWT headers shall contain the parameters as described in **Table 12**, other fields SHALL NOT be supported

Table 12: Summary JWT header fields

JWT header fields		Description
alg	Required	must be set to “RS256”
typ	Required	must be set to “JWT”
x5c	Required	As described in RFC7523

An example JWT Header is presented below:

```
{
   “alg”: “RS256”,
   “typ”: “JWT”,
   “x5c”: [“MIIGCDCCA/CgAwIBAgICEAQwDQYJKoZIhvcNAQELBQ…”]
}
```

10.2.3. JWT Payload

Source: iSHARE JWT

- The JWT payload MUST conform to the private_key_jwt method as specified in OpenID Connect 1.0 Chapter 9.
- The JWT MUST always contain the iat claim.
- The iss and sub claims MUST contain the valid iSHARE identifier (EORI) of the client.
- The aud claim MUST contain only the valid iSHARE identifier of the server. Including multiple audiences creates a risk of impersonation and is therefore not allowed.
- The JWT MUST be set to expire in 30 seconds. The combination of iat and exp claims MUST reflect that. Both iat and exp MUST be in seconds, NOT milliseconds. See UTC Time formatting for requirements.

- The JWT MUST contain the jti claim for audit trail purposes. The jti is not necessary a GUID/UUID.
- Depending on the use of the JWT other JWT payload data MAY be defined.

In OAuth 2.0 clients are generally pre-registered. Since in iSHARE servers interact with clients that have been previously unknown this is not a workable requirement. Therefore iSHARE implements a generic client identification and authentication scheme, based on iSHARE whitelisted PKIs.

Since OAuth 2.0 doesn't specify a PKI based authentication scheme, but OpenID Connect 1.0 does, iSHARE chooses to use the scheme specified by OpenID Connect in all use cases. This is preferred above defining a new proprietary scheme.

The UCIG is aligned with iSHARE on the topic of JWT payload. As described in **Chapter 9.7**, the UCIG makes use of RFC 7523 which is backward compatible with iSHARE.

REQ 45. The JWT Claims Set SHALL conform to the JWT Bearer profile as described in RFC 7523

REQ 46. The JWT Claims Set SHALL contain the claims as defined in **Table 13**. Other fields SHALL NOT be used.

Table 13: Summary of claims in a JWT payload

Claims		Description
iss	Required	As described in iSHARE, this must be an EORI identifier
sub	Required	As described in iSHARE, this must be an EORI identifier
aud	Required	As described in iSHARE, this must be an EORI identifier
exp	Required	As described in RFC7523
iat	Required	As described in RFC7523
jti	Required	As described in RFC7523

REQ 47. The JWT SHALL be set to expire in 30 seconds. The combination of iat and exp claims SHALL reflect that. Both iat and exp SHALL be in seconds, NOT milliseconds. See UTC Time formatting for requirements.

REQ 48. Depending on the use of the JWT other JWT Claims Set data MAY be defined.

An example JWT Claims Set is presented below:


```
{
   “iss”: “EU.EORI.NL123456789”,
   “sub”: “EU.EORI.NL123456789”,
   “aud”: “EU.EORI.NL987654321”,
   “jti”: “378a47c4-2822-4ca5-a49a-7e5a1cc7ea59”,
   “exp”: 1201957230,
   “iat”: 1201957200
}
```


10.2.4. JWT Processing

Source: iSHARE JWT

A server SHALL NOT accept a JWT more than once for authentication of the Client. However within it’s time to live a Service Provider MAY forward a JWT from a Service Consumer to one or more other servers (Entitled Party or Authorization Registry) to obtain additional evidence on behalf of the Service Consumer. These other servers SHALL accept the JWT for indirect authentication of the Service Consumer during the JWT’s complete time to live.

A server SHALL only accept a forwarded JWT if the aud claim of the forwarded JWT matches the iss claim of the JWT from the client that forwards the JWT. JWT contents that are not specified within the iSHARE scope SHOULD be ignored.

The UCIG is fully aligned with iSHARE on the topic of JWT processing

REQ 49. A server SHALL NOT accept a JWT more than once for authentication of the Client.

REQ 50. JWT contents that are not specified within the UCIG scope SHOULD be ignored.

REQ 51. *Proxies* SHALL only accept signed JWTs (for the purpose of *Authentication*, integrity, and non-repudiation) if:

- The signature is valid
- It is addressed to them, based on the `aud` claim
- It has not expired, based on the `exp` claim
- It has not been received before, based on the `jti` claim, considering the expiration period

10.2.5. JSON Web Encryption (JWE)

In the UCIG, all communication between *Proxies* is done over a TLS encrypted line. For most use cases, this is deemed secure enough for safe communication. Given the specific context of a *Data Sharing* use case, additional security may be required to prevent intermediaries from gaining access to information shared. This could be the case when the JWT contains sensitive information which should not be logged unencrypted.

JSON Web Encryption (JWE) can provide an encryption mechanism to secure communication if deemed necessary in a specific use case. iSHARE defines how JWE can be implemented for a *Data Sharing* use case.

11. Security

11.1. Transport Layer Security

The UCIG aims to secure the confidentiality of information for each communication step by putting strict requirements on the use of Transport Layer Security (TLS).

REQ 52. The *Proxy* SHALL only expose endpoints that are secured with TLS version 1.2 or higher

REQ 53. The Proxy SHALL reject all (TLS) unencrypted requests

REQ 54. For interaction between *Entitled Parties* and *Proxies*, the *Proxy* SHALL only use server-side TLS

REQ 55. For interaction between *Entitled Parties* and *Proxies*, the *Proxy* SHALL only use certificates issued by a widely accepted certificate authority

REQ 56. For interaction between *Proxies*, *Proxies* SHALL only use mutual TLS

REQ 57. For interaction between *Proxies*, *Proxies* SHALL allow the use certificates issued by a widely accepted certificate authority, however, if both *Proxies* agree, self-signed certificates MAY be used in a bilateral setup

REQ 58. For TLS, the *Proxy* SHALL only use X.509 certificates with a minimum key length of 2048 bits and a validity period of a maximum of two years

REQ 59. For TLS, the *Proxy* SHALL use one of these cipher suites:

- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

11.2. Risk Management

As introduced in the Data Sharing Canvas **Chapter 8**, all *Data Services* inherently expose actors involved to risks. To determine the risk of a *Data Service*, a risk analysis should be performed. In turn, the result of this analysis should determine the required security implementation of the use case.

REQ 60. In the design phase of implementing a use case, a risk analysis SHOULD be performed

REQ 61. Fit for purpose information security measures SHOULD be implemented to manage all risks to an acceptable level for all stakeholders involved implementing a use case

11.3. Information Security

In the bilateral setting of the UCIG, the assumption is that the security implementation (according to **REQ 61**), is determined between actors involved during the design phase of the *Data Sharing* use case. Therefore, there is no need for a technical implementation of the exposure or communication of specific security requirements. Specific security requirements are set by *Proxy* B when defining the *Data Service* definition. Therefore, the delivery of proof of adherence to these requirements by *Proxy* A (i.e. in the Data Service Request) is also not required. To prepare the use case for future scalability, see **Appendix I.I** for how (dynamic) security information can be communicated, and **Appendix I.IV**, for how adherence to specific security requirements can be realized.

12. Audit Trails

The UCIG aims to make sure that information obtained when using the UCIG requirements can serve as long lasting proof for the *Data Service Consumer* domain and that an indisputable audit trail can be constructed of all actions that have been carried out.

REQ 62. *Proxies* SHALL store and archive full messages, signatures and related certificates in order to be able to establish non-repudiation

REQ 63. Detailed archiving requirements, including archiving periods, should be specified within the specific use case

See **Appendix I.V**, for options on possible additional information which can be included in all proxy-to-proxy messaging to support auditing purposes.

Section D.

Operational Topics

13. Service Level Agreements

A service level measures the performance of a service. In defining a *Data Service*, the Service Level Agreements (SLAs) of the use case archetype of the UCIG (as described in **Chapter 5**) should be defined. As the SLAs are highly dependent on the specific *Data Service*, and there is no need for *Trust Framework* level agreements, the UCIG does not prescribe requirements for this topic.

iSHARE has defined norms and minimum levels for actors in a data service. These can be used as a reference of best practices for a specific use case when formalising their SLAs.

Source: iSHARE Service levels for Certified Parties

For Certified Parties, the following service levels apply:

- Availability
- Performance
- Incidents
- Support
- Reporting

Availability

Availability is a measure of the time a service is in a functioning condition. It includes the availability window and the maintenance window.

Availability window

The **availability windo**w includes the times at which Certified Parties guarantee the availability of their service.

Norm: 24 hours * all days of the year

** Planned maintenance does NOT count as unavailability*

Minimum level required: 99% availability* per calendar month, from 00:00-23:59h

Maintenance window

The **maintenance window** includes the times at which Certified Parties can perform planned maintenance, that is likely to result in downtime, to their service(s). If no downtime is expected, maintenance can take place outside of the maintenance window. Planned maintenance does NOT include incident resolution, as this can take place outside the maintenance window as described under incidents.

Norm:

- The maintenance window includes the nights from Friday to Saturday and from Saturday to Sunday, from 00:00-5.59h;
- Maintenance MUST be announced to the impacted parties directly as well as tothe Scheme Owner*;

  ** The Scheme Owner presents an overview of its own and Certified Parties' current and planned maintenance on its website*
- Announcements MUST be made at least 10 working days before the maintenance and MUST include date, time, and impacted service(s).

Performance

Performance includes the time it takes for a service to respond when requested or called upon; i.e. the time a Certified Party's service takes to respond to a received message.

Norm:

- 95% of messages MUST be responded within 2 seconds;
- 99% of the messages MUST be responded within 5 seconds;
- Each Certified Party MUST be able to process at least 100 simultaneous messages while meeting above requirements.

Incidents

An **incident** is an event, not part of the standard service operation, that results in a potential impact or risk with regards to the quality, availability, confidentiality and/or integrity of (data within) the iSHARE Scheme. This ONLY includes the data used for identification, authentication and authorisation purposes in the context of data exchange, but not the contents of the actual data exchange.

Three classifications of incidents are recognised within iSHARE, as explained in the incident management process:

- Minor incident;
- Calamity;
- Crisis.

Norm:

- All incidents MUST be communicated by the Certified Party(s) to the Scheme Owner directly after they are discovered;
- Communication MUST include date, time, incident level as estimated by the Certified Party(s), argumentation including impacted service(s), and a potential incident manager;
- In case of a calamity or crisis, the Certified Party MUST have an incident manager available during working days, and SHOULD have an incident manager available 24 * 7;
- An update on the incident MUST be communicated to the Scheme Owner*:

  ** In line with the incident management process, the Scheme Owner presents an overview of current calamities and crises on its website*

  - For minor incidents, at the end of each working day;
  - For calamities, within 2 hours of every significant update and at the end of each working day;
  - For crises, within 2 hours of every significant update and every 4 hours.
- All incidents SHOULD be handled by the Certified Party (in cooperation with the Scheme Owner as per the incident management process) within 3 working days after being appointed as the responsible party - unless agreed otherwise.

Support

Support by Certified Parties includes answering questions and requests from Adhering Parties.

Norm: Certified Parties are available for support via e-mail; they MUST confirm receiving a question/request within 1 working day. They SHOULD send an underpinned reaction (with an answer/solution or at the very least a direction) within 5 working days.

Reporting

Reports are meant to monitor both compliance to the service level agreements and the (growing) use of the iSHARE network, as described in the management reporting process. The following will be reported on (non-exhaustive):

- Availability;
- Number of relations with Adhering Parties;
- Number of transactions;
- Number of transactions per Adhering Party;
- Number of incidents.

Certified Parties are expected to collect management information about each month: 0:00h on the first day to 23:59h on the last.

Norm: each Certified Party MUST deliver the management information about the last month, conform the iSHARE template, before 23:59h on the 5th working day of the current month

14. Proxy Set-up

All information that is required to set-up the connection between *Proxies*, including, but not limited to *Data Service* definition, client_id, redirect_uris, scopes, TLS and signing certificates, will be exchanged between *Proxies*, bilaterally.

REQ 64. The *Proxy* SHALL offer a manual process for the registration of other *Proxies*. The exact process is up to the *Proxy*

REQ 65. The *Proxy* SHALL make all reasonable efforts to provide changes to its configuration to other *Proxies* as soon as possible, but no later than five working days prior to implementation

REQ 66. For migration purposes, the *Proxy* SHOULD allow the use of multiple configurations per actor

REQ 67. Each *Proxy* in a use case SHALL be able to reach, connect to and interact with all other Proxies involved in the use case

15. Use Case Change Management

The UCIG assumes that data sharing use cases are initially being collaboratively developed between all actors involved in the use case. During its initial development, the actors involved should proactively make agreements on how the implementation of the use case can be changed in the future to remain aligned with the wishes and requirements of all those involved. All use case implementations are based on bilateral agreements between the actors involved. These may refer to a specific version of the UCIG.

The UCIG itself does not have a formal change management process. In the future, to enable scalability through trust and interoperability between use cases these can be on a *Trust Framework* for cross-*Domain Data Sharing*, instead of bilateral agreements. A future *Trust Framework* for cross-*Domain Data Sharing* will have a formalised change management process.

Section E.

Legal Topics

16. Lawful Basis

As introduced in the Data Sharing Canvas Chapter 7.3.3, all *Data Sharing* transactions should have a lawful basis. For each type of *Data* involved (personal or non-personal, public, or restricted), different lawful bases exist for the processing of the *Data* and therefore, the sharing of the *Data*.

REQ 68. There SHALL be a lawful basis for sharing *Data*

Depending on the specific context of a *Data Sharing* use case, there are many possible lawful bases which could be appropriate. Examples are a bilateral contract between actors involved and a more complex mixture of legitimate interests and *Consent*.

Within the context of the use case archetype considered in the UCIG, as introduced in **Chapter 5**, it is assumed that at least approval given by the *Entitled Party* is required to form part of the lawful basis for sharing *Data*. Likely a contract between *Proxies*/*Domains* will additionally be required. In the interaction model, a reference to this additional contract is included during the *Data Service* through the ids-transferContract field. Given the bilateral context of the *Data Sharing* use case archetype, a single bilateral contract between *Proxies* shall be sufficient for creating the *Trust* needed for sharing *Data*. See **Appendix I.III.III**, for considerations regarding *Contracts* and *Domain* specific *Schemes* when a use case scales to multiple actors within a *Domain*.

Note: As the use case archetype considered in the UCIG consists of cross- *Domain Data Sharing*, actors should be aware that actors in other *Domains* may not be aware of *Domain* specific laws and regulations which apply to another *Domain*. To ensure a mutual understanding, these should be made explicit in any contracts between *Domains*.

17. Data Service Transaction Agreement

As part of each *Data Service Transaction* between a *Data Service Consumer* and a *Data Service Provider*, an agreement between them is established. As introduced in the Data Sharing Canvas, this agreement is defined as the *Data Service Transaction Agreement*. This *Data Service Transaction Agreement* is specific to a single transaction and can be considered a handshake between the *Actors* to confirm *Trust* and the mutual acceptance of the specific *Data Service Terms and Conditions* under which the *Data Service Transaction* takes place.

REQ 69. Every *Data Service Transaction* SHALL result in a *Data Service Transaction Agreement*, legally binding and covering the transaction specific context

A complete *Data Service Transaction Agreement* includes the following elements:

- A *Data Service* description including all *Policies* consisting of *Acces Control Rules* and Obligation and Advice (Created by the *Data Service Provider*)
- Proof of adherence to the *Acces Control Rules* (Provided by the *Data Service Consumer*)
- Evaluation of the proof of adherence (Evaluated by the *Data Service Provider*)
- Execution of the *Data Service Transaction* (Performed by the *Data Service Provider*)
- Proof of adherence to the *Obligation And Advice* (Provided by the *Data Service Consumer* and *Data Service Provider*)

Given the elements described, the signed Data Service Request and Data Service Response are an important part of (the capturing of) the *Data Service Transaction Agreement*. Therefore, these must be stored in case the *Data Service Transaction Agreement* needs to be reconstructed, see **Chapter 12** for more for information regarding logging.

For more information on how specific elements of the *Data Service Transaction Agreement* can be mapped onto concepts introduced in the UCIG, see **Appendix I.III.I.**

Section F.

Appendices

I. Considerations for scalability

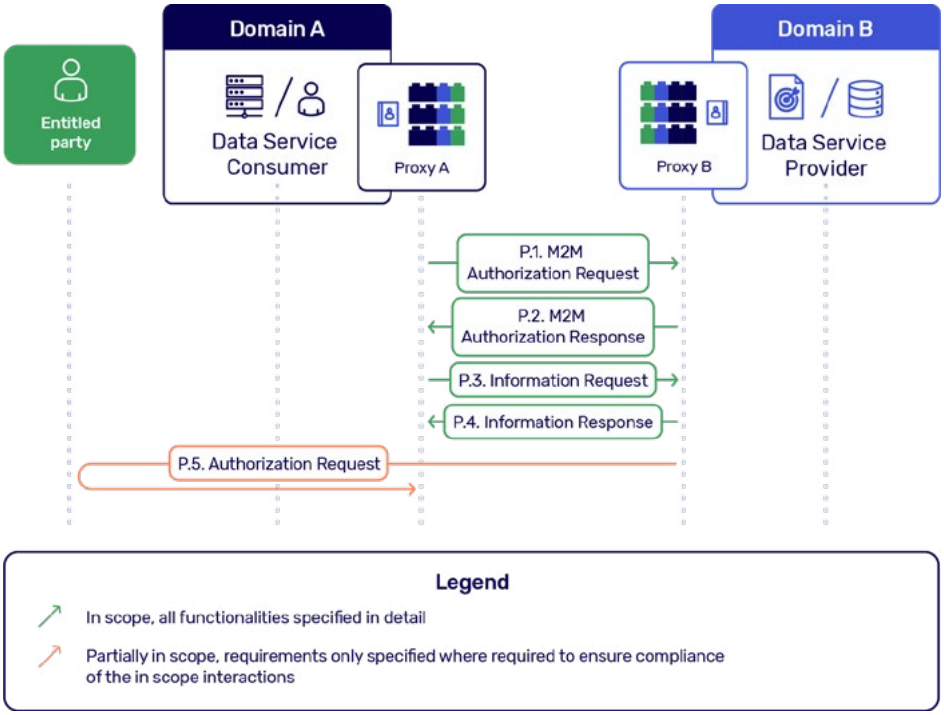
In this chapter an analysis of Chapter 7.2 is done and additional requirements are presented which can be considered for *Data Sharing* use cases scaling beyond the scope as detailed in Chapter 5. Parts of this chapter are applicable to varying *Domain* implementations (as described in Chapter 5.1) or to adding additional *Domains* (as described in Chapter 5.2). All requirements presented in this chapter are therefore optional within the context of a use case, and are indicated by **sREQ MM**, where MM (roman numerals) refers to the requirement number.

I.I. Prerequisites

For some use cases, it may be necessary to exchange information between *Domains* before the *Data Service Consumer* and *Data Service Provider* before the *Data Service Consumer* can determine whether they want to make use of the *Data Service*. This becomes especially relevant when a not all elements of a *Data Service* are static or can be agreed upon bilaterally between the *Data Service Provider* and all possible *Data Service Consumers*. Dynamic elements of the *Data Service* may differ per stakeholder or at the time of the transaction compared to prior made agreements. This could include elements such as pricing of the *Data Service*.

In Chapter 7.2.1, requirements are stated regarding the state of *Proxy* A before a transaction can take place. In this chapter, specific elements of this process are detailed and explained to enable a scalable use case. Figure 10 details interactions for *Domain* A to enable scale within the *Domain* implementation.

Figure 10: Overview of detailed interactions for the prerequisites



I.I.I. Step P.1. M2M Authorization Request

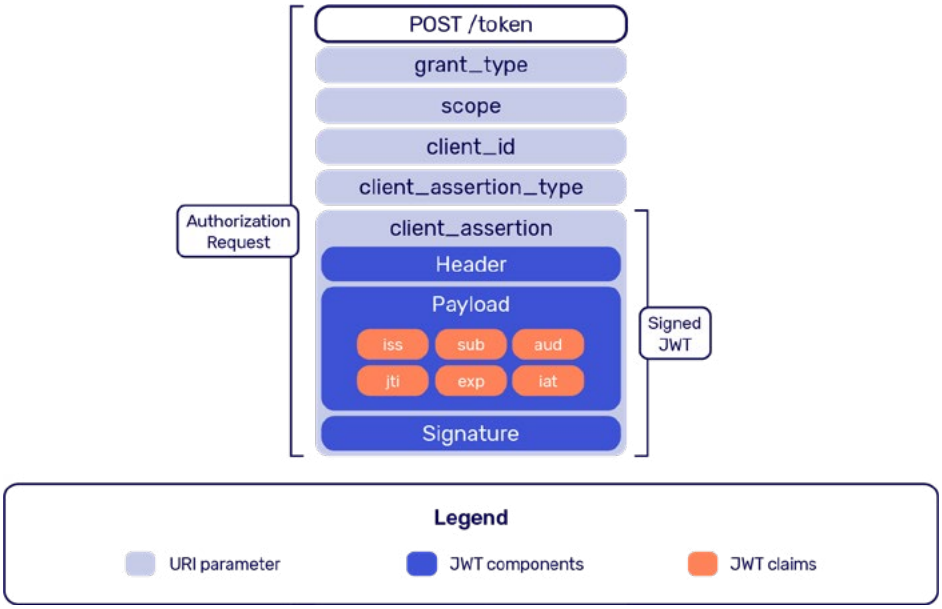
An M2M Authorization Request enables *Proxy* A to obtain an *Access Token* from *Proxy* B to Authorise *Proxy* A to perform the subsequent Information Request. This step is aligned with the iSHARE implementation and further tailored to the specific use case archetype of the UCIG as described in **Chapter 5**.

Note: If the *Data Service* information is publicly available, Step P.1 and Step P.2 can be skipped as no *Access Token* is required to access this information.

sREQ I. *Proxy* A MAY request authorization to request additional *Dataservice* information at *Proxy* B at the `/token` endpoint according to OAuth 2.0 Client Credentials Grant, Chapter 4.4.2, Authorization Request and the assertion framework for client authentication

Specific requirements on top of OAuth 2.0, or requirements that are deemed useful to include for clarity are included below. **Figure 11** gives an overview of the HTTP headers of an M2M Authorization Request and the contained signed JWT.

Figure 11: Overview of the HTTP URIs of an Authorization Request



As a result of the choices made in **sREQ I** and applying **REQ 37** to the /token endpoint, *Proxy* B must support the HTTP headers as described in **Table 14**. Similarly, *Proxy* A must include the HTTP headers in the Authorization Request as described in **Table 14**. When Proxy B receives an Authorization Request, it must validate the client credentials received in the client_assertion.

Table 14: Summary of the HTTP headers for a /token endpoint

M2M Authorization Request HTTP headers		Description
grant_type	Required	As stated in OAuth 2.0, MUST be set to "client_credentials"
scope	Required	As stated in OAuth 2.0
client_id	Required	As stated in iSHARE, this must be an EORI identifier
client_assertion_type	Required	As stated in OAuth 2.0 JWT bearer profile this value must be set to "urn:ietf:params:oauth:client-assertion-type:jwt-bearer"
client_assertion	Required	As stated in OAuth 2.0 JWT bearer profile this value must contain a single signed JWT conform to the generic UCIG requirements (See **Chapter 10.2**)

An example M2M Authorization Request is presented below (with extra line breaks for display purposes only):

```
POST /token
   grant_type=client_credentials&
   scope=xyz&
   client_id=EU.EORI.NL000000001&
   client_assertion_type=urn:ietf:params:oauth:client-
   assertion-type:jwt-bearer&
   client_assertion= eyJhbGciOiJSUzI1NiIsImtpZCI6IjIyIn0.
   eyJpc3Mi[...omitted for brevity...].
   cC4hiUPo[...omitted for brevity...] HTTP/1.1
Host: ProxyB.example.com
```

I.I.II. Step P.2. M2M Authorization Response

The M2M Authorization Response is the response to the M2M Authorization Request

sREQ II. *Proxy* B SHALL issue an *Access Token* to Proxy A according to OAuth 2.0 Client Credentials Grant, Chapter 4.4.3

Specific requirements on top of OAuth 2.0, or requirements that are deemed useful to include for clarity are included below.

As a result of the choice made in **sREQ II**, Proxy A must support the HTTP headers as described in **Table 7**. Similarly, *Proxy* B must include the HTTP headers in the Authorization Response as described in **Table 7** other headers must NOT be included. Proxy B must handle errors according to OAuth 2.0 Chapter 5.2.

Table 15: HTTP headers in an M2M Authorization Response

M2M Authorization Response HTTP headers		Description
content-type	Required	As stated in OAuth 2.0
cache-control	Required	Must be set to "no-store" conform to the generic UCIG requirements (See **Chapter 9.2**)
pragma	Required	Must be set to "no-cache" conform to the generic UCIG requirements (See **Chapter 9.2**)

An example M2M Authorization Response is presented below (with extra line breaks for display purposes only):

```
HTTP/1.1 200 OK
content-type: application/json;charset=UTF-8
cache-control: no-store
pragma: no-cache
{
   "access_token":"2YotnFZFEjr1zCsicMWpAA",
   "token_type":"bearer",
   "expires_in":3600
}
```

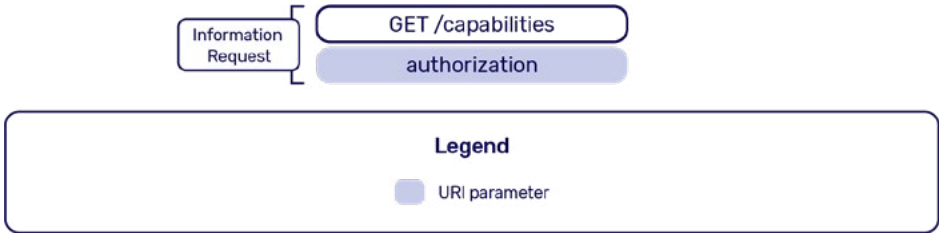
I.I.III. Step P.3. Information Request

Once Proxy A has received the *Access Token* for the Information Request in the Authorization Response, it can request the information regarding the *Data Service*. This is achieved via the iSHARE /capabilities endpoint.

sREQ III. III. *Proxy* A MAY request information from *Proxy* B at the /capabilities endpoint, conform the iSHARE definition

Figure 12 gives an overview of the Information Request and the contained signed JWT.

Figure 12: Overview of an Information Request



As a result of the choices made in sREQ III, *Proxy* B must support the URI parameters as described in Table 16. Similarly, *Proxy* A must include the URI parameters in the Authorization Request as described in Table 16. When Proxy B receives an Information Request, it must validate the client credentials received in the client_assertion.

Table 16: Summary of the URI parameters for an /capabilities endpoint

Information Request URI parameters		Description
authorization	Optional	OAuth 2.0 authorization based on bearer token. must contain "Bearer + Access Token" value as received in the Authorization Request. This value is optional for if the *Data Service* information is publicly available

An example Information Request is presented below (with extra line breaks for display purposes only):

```
GET /capabilities?

Authorization: Bearer 2YotnFZFEjr1zCsicMWpAA
```

I.I.IV. Step P.4 Information Response

The Information Response is the response to the Information Request. The specific information that is included in the Information Response is highly dependent on the use case design. Both iSHARE and IDSA have defined a general structure which can be

used to provide the *Data Service* information. For more information, see the iSHARE capabilities_token, or the IDSA self- description (See the IDS Reference Architecture Model Chapter 3.4.5 or an example in the Data Space Connector).

sREQ IV. Proxy B SHALL respond to an Information Request with an Information Response according to the IDS REST implementation

sREQ V. Proxy B SHALL include the information requested (requested via the Information Request) in its payload

Specific requirements on top of IDS REST, or requirements that are deemed useful to include for clarity are included below.

As a result of the choices made in **sREQ IV** and **sREQ V**, Proxy A must support the HTTP headers as described in **Table 17**. Similarly, *Proxy* B must include the HTTP headers in the Information Response as described in **Table 17**, other headers must NOT be included.

Table 17: HTTP headers in an Information Response Access Token Response

M2M Authorization Response HTTP headers		Description
content-type	Required	Must be set to the type of information included in the HTTP body
cache-control	Required	Must be set to "no-store" conform to the generic UCIG requirements (**See Chapter 9.2**)
pragma	Required	Must be set to "no-cache" conform to the generic UCIG requirements (See **Chapter 9.2**)

Proxy B must include the requested information regarding the *Data Service* in the HTTP payload. As the *Data Service* description is highly dependent on the *Data Service*, this is not specified here, but either iSHARE or IDSA can be used for this purpose. In the use case design this should be decided.

An example Information Response is presented below (with extra line breaks for display purposes only). Note in this example, an IDSA self-description is used:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
   "@context" : {
      "ids" : "https://w3id.org/idsa/core/",
      "idsc" : "https://w3id.org/idsa/code/"
   },
   "@type" : "ids:Resource",
... }
```

I.I.V. Step P.5 Authorization Request

In **Chapter 7.2.1**, the prerequisites are given that *Proxy* A has all the information needed to make use of the *Data Service*, and the user agent of the *Entitled Party* can be controlled. This is enabled via an Authorization Request between the *Data Service Consumer* AND *Proxy* A. To facilitate this Authorization Request, Proxy A must implement an /authorization endpoint, in line with **Chapter 7.2.2**.

sREQ VI. The *Data Service Consumer* MAY forward Authorization Requests to Proxy A's /authorization endpoint for the purpose of enabling scalability by allowing multiple *Data Service Consumers* to connect to *Proxy* A

sREQ VII. The *Data Service Consumer* SHALL redirect the *Entitled Party* to the *Proxy* A /authorization endpoint according to OAuth 2.0 Authorization Code Grant, Chapter 4.1.1, Authorization Request, using a signed JWT according to the JWT bearer profile

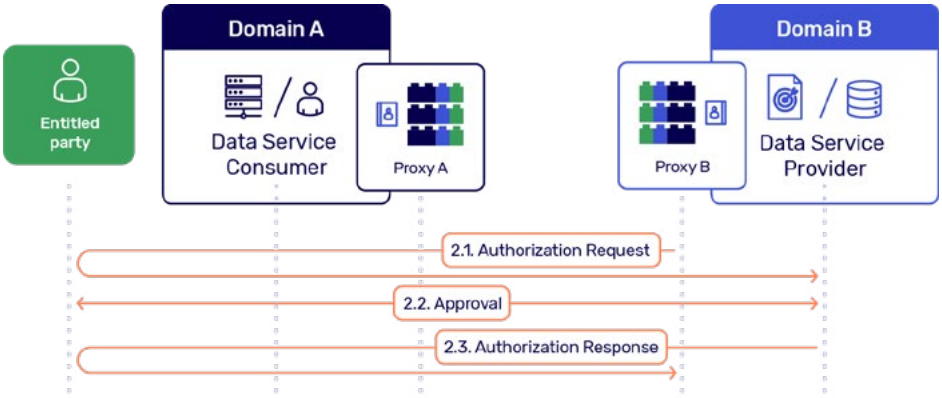
See **Chapter 7.2.2** for more detail on an Authorization Request, and **Figure 5** for an overview of the HTTP URIs of an Authorization Request and the contained signed JWT. *Proxy* A must support the URI parameters as described in **Table 2**. Similarly, the *Data Service Consumer* must include the URI parameters in the Authorization Request as described in **Table 2**. When Proxy A receives an Authorization Request, it must validate the complete Authorization Request, including client credentials received in the client_assertion.

sREQ VIII. *Proxy* A must forward valid Authorization Requests to *Proxy* B according to Chapter 7.2.2

I.II. Scalable Approval

In Chapter 7.2.2, requirements are stated for an Authorization Requests in the situation where the *Data Service Provider* is an Integrated *Domains* as defined in Chapter 5.1. This paragraph covers how the *Data Service Provider* can arrange approval when *Domain* B scales from Integrated *Domain* implementation to a *Proxy* domain implementation. Figure 13 details the required interactions.

Figure 13: Overview of detailed interactions for scalable approval



I.II.I. Step 2.1 Authorization Request

For *Domain* B to support multiple *Data Service Providers*, a mechanism must be implemented to redirect the *Entitled Party* from the *Proxy* B to the *Data Service Provider*. This is enabled via an Authorization Request between *Proxy* B and the *Data Service Provider*. To facilitate this Authorization Request, The *Data Service Provider* must implement an /authorization endpoint, in line with Chapter 7.2.2.

sREQ IX. *Proxy* B MAY forward Authorization Requests to a *Data Service Provider*'s /authorization endpoint for the purpose of enabling scalability by allowing multiple *Data Service Providers* to connect to *Proxy* B

sREQ X. *Proxy* B SHALL redirect the *Entitled Party* to the *Data Service Provider* / authorization endpoint according to OAuth 2.0 Authorization Code Grant, Chapter 4.1.1, Authorization Request, using a signed JWT according to the JWT bearer profile

See **Chapter 7.2.2** for more detail on an Authorization Request, and **Figure 5** for an overview of the HTTP URIs of an Authorization Request and the contained signed JWT. *Proxy* A must support the URI parameters as described in **Table 2**. Similarly, the *Data Service Consumer* must include the URI parameters in the Authorization Request as described in **Table 2**. When Proxy A receives an Authorization Request, it must validate the complete Authorization Request, including client credentials received in the client_assertion.

I.II.II. Step 2.2 Approval

As stated in **Chapter 7.2.3**, the UCIG does not specify requirements for this step.

I.II.III. Step 2.3 Authorization Response

When *Domain* B implements an Authorisation Request (**Chapter I.II.I**), and the *Entitled Party* has been redirected to the *Data Service Provider*, in the response, a mechanism must be implemented to redirect the *Entitled Party* from the *Data Service Provider* back to *Proxy* B, in line with **Chapter 7.2.4**.

sREQ XI. *Proxy* B SHALL redirect the *Entitled Party* back to the *Proxy* A according to OAuth 2.0 Authorization Code Grant, Chapter 4.1.2, Authorization Response

As a result of the choices made in **sREQ XI**, *Proxy* B must support the URI parameters as described in **Table 4**, other URI parameters SHALL NOT be used. Similarly, the *Data Service Provider* must include the URI parameters in the Authorization Response as described in **Table 4**. , the *Data Service Provider* must handle errors according to OAuth 2.0 chapter 4.1.2.1. , the *Data Service Provider* must NOT supply an *Access Token* in the Authorization Response (these must only be supplied via the designated /token endpoint as described in **Chapter 7.2.6 Step 5. Access Token**)

I.III. Lawful Basis

I.III.I. Data Service Transaction Agreement

The concept of a *Data Service Transaction Agreement* and elements that it should contain was introduced in **Chapter 17**. The Policies of the *Data Service* Transaction Agreement can be mapped to a contract defining the data service. The proof of adherence can be mapped to Dynamic Attribute Tokens, which can be provided by a Dynamic Attribute Provisioning Service as defined by IDSA.

I.III.II. Dynamic Contracts

In **Chapter 7.2**, a simple reference to contract underlaying the data service transaction was included in the ids.transferContract claim in the interaction model. For simple data services in a bilateral setting, this is likely sufficient. To enable the data service to be more scalable, instead of the ids.transferContract claim containing a URI, it can be replace by a ids.contractRequest in the Data Service Request, and the ids.contractOffer in the Data Service Response, both of which contain the complete contract. The IDS Information Model contains a standard for the contract, based on ODRL. This structure defined in IDS can be used instead of the reference URI to include the entire contract in the ids.contractRequest and ids.contractOffer claim. The Fraunhofer Data Space Connector can be used as a reference implementation of how this can be realised.

An assumption upon which the UCIG is based is that there is a single contract underlaying a data service between two actors. If the use case is to scale beyond these actors, it may be desirable to implement a technical contract negotiation which can be performed to come to an agreement before the Data Service takes place. IDS Usage Contract Negotiation defines such a contract negotiation sequence which can be followed to enable this functionality.

I.III.III. Domain Schemes

To enable multiple actors within a *Domain* to share data, a *Domain* should consider the implementation of a *Domain Scheme*. In a *Domain Scheme*, Participants have one contract with the *Domain Scheme* to enable *Data Sharing* with all other Participants. When the *Domain Scheme* has a single contract with the *Domain* a chain of bilateral contracts can enable scalability and provide a solution to enable multilateral *Trust*. See the Data Sharing Canvas Chapter 7.3.2. for more details. iSHARE satellites can serve as a basis for setting-up a *Domain Scheme*.

I.IV. Dynamic Attribute Provisioning

In the bilateral setting in the context of the UCIG, the security elements of *Proxies* can be determined offline. Therefore, there is no need for a mutual technical validation of *Proxy* security-related claims, technically during interactions. If a use case were to scale to multiple domains with many participating *Proxies*, it may be desirable to perform this. To this end, the ids.securityToken field using an IDS Dynamic Attribute Token may be used in all *Proxy* to *Proxy* communication.

sREQ XII. *Proxies* MAY use the ids.securityToken header field in all *Proxy*- to-*Proxy* communication, in line with the IDS for the purpose of providing dynamic security attributes

sREQ XIII. *Proxies* MAY validate Dynamic Attribute Tokens present in the ids.securityToken header field

sREQ XIV. *Proxies* MAY fill the the ids.securityToken header field with a valid Dynamic Attribute Token representing security-related claims.

sREQ XV. If both *Proxies* agree, the Dynamic Attribute Token used MAY be self-signed in a bilateral set up.

I.V. Additional Auditing Information Entities

For auditing, or internal processes such as debugging or the analysis of logs for disputes, it could be useful to store additional information regarding actors involved in communication. This should be captured in all *Proxy*-to-*Proxy* communication by following IDSA, and including the ids.senderAgent and ids.recipientAgent HTTP headers. These fields contain an identifier of the underlaying legal entities on behalf of which the communication takes place. For the basic context of the UCIG, the value of the ids.senderAgent is equal to the iss JWT claim and the ids.recipientAgent is equal to the aud JWT claims.

sREQ XVI. *Proxies* MAY use the ids.senderAgent header field in all *Proxy*-to- *Proxy* communication to reference to underlaying source legal entities

sREQ XVII. *Proxies* MAY use the ids.recipientAgent header field in all *Proxy*-to-*Proxy* communication to reference to underlaying receiving legal entities

I.VI. Persistent Authorization

For some use cases, it could be useful, or necessary, to deviate from the ”one- time, ad hoc transaction” part of the use case archetype described as in **Chapter 5**. This could be the case if the *Data Service* would be regularly performed over extended periods of time. For such a use case, refresh tokens can be used to enable the not require explicitly approval for every transaction.

Typically, the *Access Token* used to access the data service are short lived for security purposes. A refresh token allows Proxy A to retrieve new valid *Access Tokens* without requiring approval. The issuing of a refresh token is optional and at the discretion of *Proxy* B. See OAuth 2.0 Chapter 1.5 for more information on refresh tokens.

sREQ XVIII. Proxy B MAY provide a refresh token if the *Entitled Party* has given persistent *Authorization*.

II. Data Sharing Coalition Overview

Figure 14: Overview of participants of the Data Sharing Coalition as of February 2022

III. Relevant Resources

Table 18 contains an overview of the main resources which are relevant for this UCIG as they provide relevant context and reference. Where these are referenced throughout this document, it is explicitly mentioned.

Table 18: Relevant important resources

Resource name	Published by	Pub. date	Source
Assertion framework for OAuth 2.0 Client Authentication and Authorization Grants	Internet Engineering Task Force (IETF)	May 2015	Link
Data Sharing Canvas	Data Sharing Coalition	April 2021	Link
IDS Reference Architecture Model v3.0	International Data Spaces Association	April 2019	Link
IDS Information Model v4.1.0	International Data Spaces Association	June 2021	Link
IDS Connector REST Interaction	International Data Spaces Association	August 2021	Link
iSHARE scheme v1.11	iSHARE	November 2020	Link
JSON Web Token (JWT)	Internet Engineering Task Force (IETF)	May 2015	Link
JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants	Internet Engineering Task Force (IETF)	May 2015	Link
OAuth 2.0 Authorization Framework	Internet Engineering Task Force (IETF)	October 2012	Link

IV. Glossary

Table 19: Glossary

Glossary item	Definition
Access Control Rules	Policies that are assessed and enforced prior to the establishment of a *Data Service Agreement*, which regulate how *Data Services* can be accessed
Access Token	A string denoting a specific scope, lifetime, and other access attributes which represents the *Authorization* (on behalf of the entitled party) of a specific application
Authentication	The process where the validity of a claimed identity is verified
Authorization	The permissions or rights of an actor (humans, machines, proxies, etc.) to perform an action
Consent	Any freely given, specific, informed and unambiguous indication of the *Entitled Party's* wishes by which he or she, by a statement or by clear affirmative action, signifies agreement to the processing of personal data relating to him or her
Data	A reinterpretable representation of information in a machine-readable format, suitable for communication, interpretation, or processing
Data Service	Any service offered by a *Data Service Provider* aimed at exchanging or processing *Data* (for example, this includes basic *Data Services* such as *Data* pull, *Data* push, bringing an algorithm to the *Data* as well as complex use cases based on combinations of these basic types)
Data Service Consumer	The actor that makes use of a *Data Service* offered by the *Data Service Provider*
Data Service Discovery	The mechanism through which a *Data Service Consumer* and *Data Service Provider* can find each other across *Domains*
Data Service Provider	The actor that offers a *Data Service* to the *Data Service Consumer*

Glossary item	Definition
Data Service Transaction	The event of executing a *Data Service* between *Data Service Provider* and *Data Service Consumer*. Depending on the type of *Data Service* the *Data Service Transaction* can be a single moment or take place for a length of time
Data Service Transaction agreement	The agreement (handshake) between a *Data Service Consumer* and *Data Service Provider* to enable *Trust* and accept the terms on which the *Data Service Transaction* can take place
Data Sharing	The machine actionable exchange of structured *Data* through a *Data Service Transaction* between *Data Service Providers* and *Data Service Consumers*
Data Sharing Coalition (DSC)	A collaborative initiative that aims to enable organisations to easily share *Data* across *Domains*
Data Sharing Initiative	Organisation that enables *Data Sharing* in a certain *Domain* by providing a coherent set of specifications and requirements and by providing supervision
Data Standards	Provide the semantics, structure, and formatting of *Data*
Delegation	The provision of explicit rights (to perform an action) to a third party
Domain	Flexibly defined as any number of organisations collaboratively working together to share *Data* to achieve a shared purpose, goal or mutual benefits
Dispute	When actors within the *Trust Framework* cannot settle disagreements between them according to specific service level agreements
Dispute Management	The process of managing *Disputes* when they have been reported to the *Trust Framework Authority*
Entitled Party	The entity which has rights over *Data*. This may include the storage of the *Data* as well as the access and usage of the *Data*
Governance	The management and maintenance of a *Trust Framework* agreements and network

Glossary item	Definition
Harmonisation	Establishing agreements, standards, and requirements between actors to enable *Data Sharing* between them
Harmonisation Domain	Network of *Proxies*
Identification	The process of claiming an identity by a subject or the process of attributing/issuing an identity to a subject by an authority
Information Security	Preservation of the confidentiality, integrity, and availability of information though the implementation of technical or organisational information security measures
Interoperability	The ability of systems of different actors to exchange *Data* in a meaningful way that is mutually understandable and satisfactory
Logging	The recording of actions with goal to create a reliable overview of events that have occurred
Metadata	Describes everything about *Data*, *Data Services*, and *Data Service Transactions* in *Data Sharing* that cannot be assumed to be known
Obligations and Advice	Policies that are assessed and enforced after the establishment of a *Data Service Agreement*, on what must be carried out after a *Data Service* is approved. Advice is similar to obligation with the difference that enforcement of the advice is not mandatory
Policies	Define rules for access to and usage of *Data Services*, can be split into *Access* control rules and *Obligations* and *Advice*. *Terms and conditions* are formalised into *Policies*
Proxy model	Solution for multilateral Interoperability across *Domains* where different *Data Sharing Domains* implement *Proxies*.
Proxy	A module that translates between specifications and requirements from a *Data Sharing Domain* and *Harmonised* specifications and requirements (and vice versa) to achieve *Interoperability* and *Trust* across *Domains*
Scheme	Synonym for *Trust Framework*
Service Registry	Contains necessary *Data Service* information to perform *Data Service Discovery*. Can be considered similar to a telephone book

Glossary item	Definition
Terms and Conditions	Define the concepts as well as the duties and rights, the powers and liabilities that apply to the actors engaged in *Data Service Transactions*
Trust	A situation between actors where (perceived) risks are sufficiently reduced to enable *Data Sharing*. The amount of risk deemed as acceptably low is determined by each actor themselves and therefore varies between actors
Trust Framework	Enables many-to-many transactions though business, legal, operational, functional, and technical agreements, tools, and processes which facilitate *Trusted* transactions between *Participants*
Trust Framework Authority	The cross-*Domain Data Sharing* authority defines and manages the *Trust Framework*, monitors compliance, and settles disputes to facilitate cross-*Domain Data Sharing*
Trust Framework Governance	Needed to develop, manage, and maintain the *Trust Framework* agreements and network
Use Case Implementation Guide (UCIG)	This document

DATA SHARING COALITION

info@coe-dsc.nl
https://coe-dsc.nl/