FEDeRATED REFERENCE ARCHITECTURE

Serves as Addendum to the
FEDeRATED Master Plan

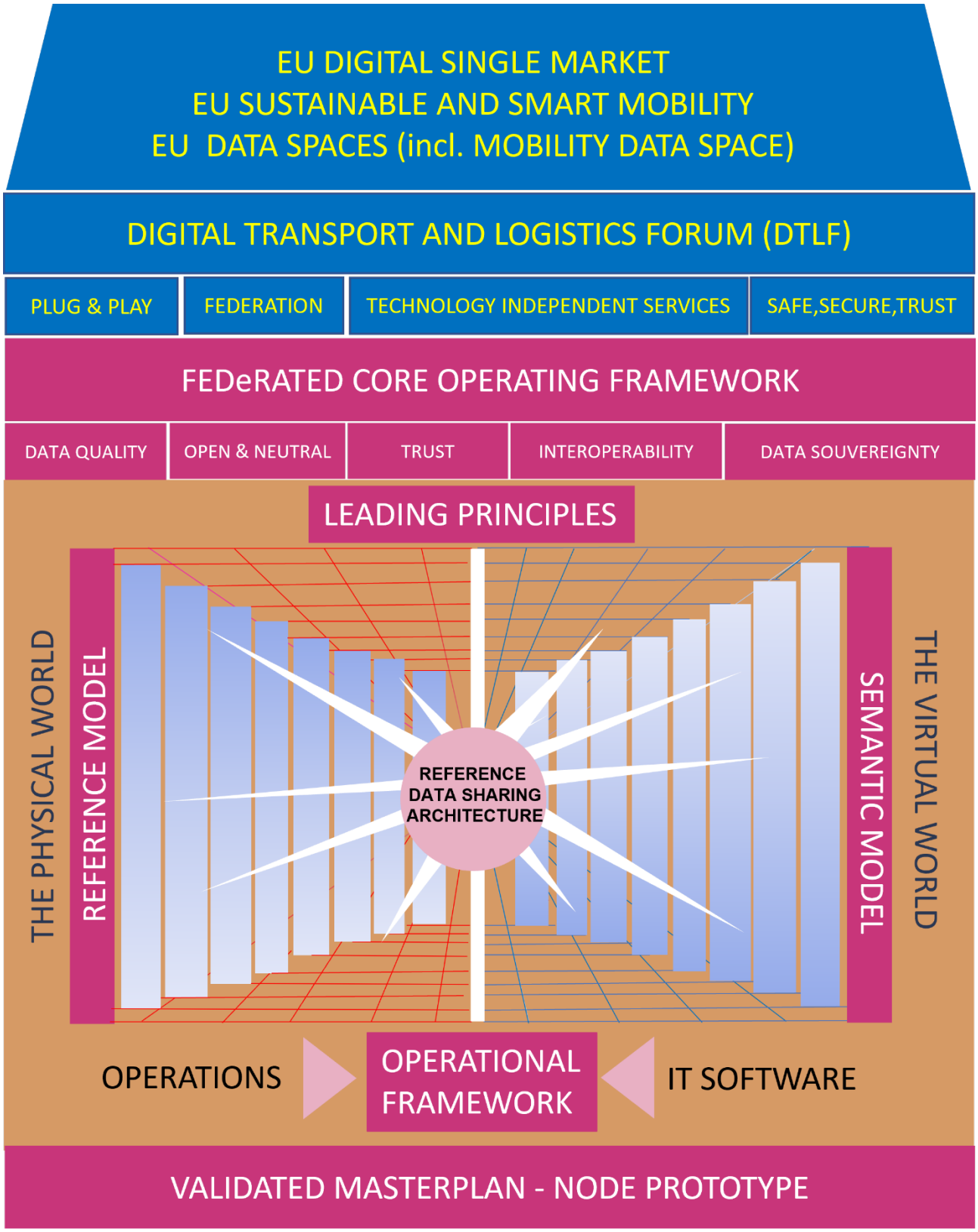
Milestone 14

February 2024

www.federatedplatforms.eu

FEDeRATED NETWORK OF PLATFORMS

Co-financed by the Connecting Europe Facility of the European Union



This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the FEDeRATED project consortium and can in no way be taken to reflect the views of the European Union.

EXECUTIVE SUMMARY

Towards the European Mobility Data Space (EMDS)

The FEDeRATED Reference Architecture aims to enable the DTLF (Digital Transport and Logistics Forum) concept of a federated network of platforms to be develop as a technology grid enabling all parties in freight transport and logistics to share data according to the European Interoperability Framework (EIF). The *DTLF federative network of platforms* policy approach is based on 4 Building Blocks: 1) plug & play, 2) federation, 3) independent technology services and 4) safe, secure, trust. In practical terms this policy approach can be explained as a policy impulse to develop a future proof *European Mobility Data Space (EMDS)* enabling Distributed Data Resources (DDR) - i.e. IT systems/platforms that provide or use data aimed at delivering business services - to share data with another for business operation and compliance. The *EMDS* would enable millions of IT systems/platforms to draw data at some times and supply data at other aimed at providing tailor made services to all participants, including compliance with legislation.

Overarching principle: Data sovereignty – data at source

The DDR embodies one of the basic principles of the EU Data Policy, the EU DTLF, and the FEDeRATED Core Operating Framework (COF)[1], being data sovereignty; - more in particular data at source, pull data made available through sharing events with links between data holders and users. This should be established in combination with the need for an open, neutral, and trusted digital grid, and enabling interoperable data distribution of high quality. The consequence being that on a local or national scale, DDR's will be empowered to scale their activities onto an overarching – interoperable - EMDS. Very complex to plan for, orchestrate and keep in balance. Innovative and transitional at the least.

Capabilities

To enable this EMDS, each participating DDR must implement certain capabilities. The main objective of these capabilities is to support digitization of any DDR for its business activities, compliant to regulations. As such, it reflects the current way of working of for instance a forwarder implementing various industry - and open standards, which is now addressed by the capabilities, especially semantics.

The capabilities required by a DDR are specified semantics and functional capabilities:

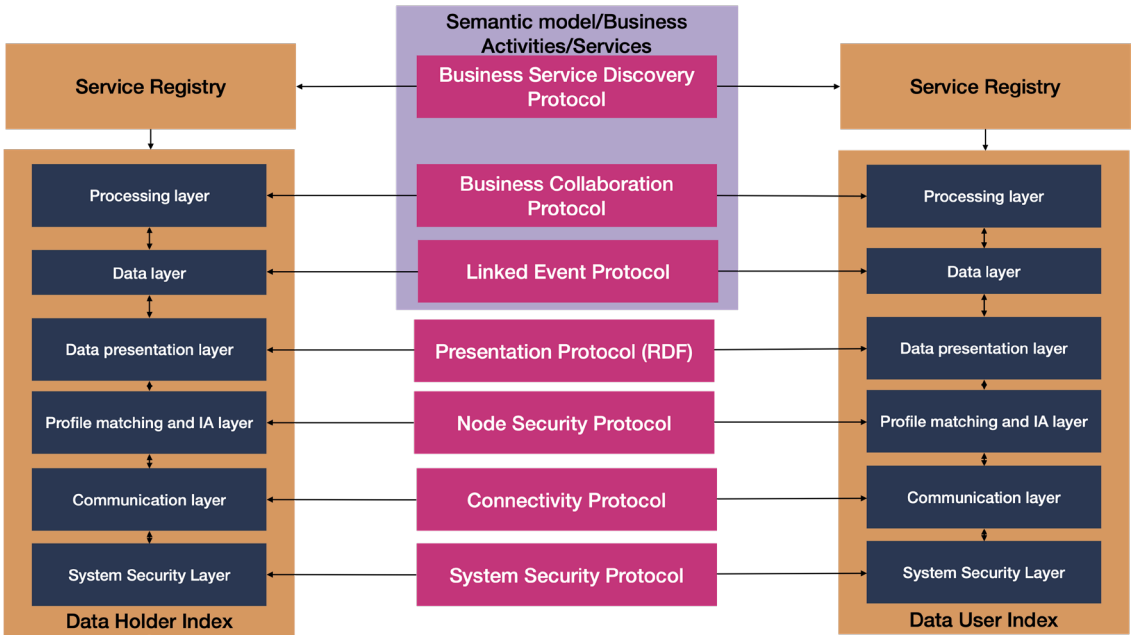
- **Semantics**. It is about data sharing semantics for business activities like transport and transshipment, compliant with regulations. The semantic model contains all concepts required for service development (e.g. ordering – and visibility services) and their customization by organisations. Service development results in Technology Independent Services; service customization in plug and play.
- **Functional capabilities**. These are the capabilities for service development, - customization, - and implementation. Their architecture and interfaces give the 'federated network of platforms' as one of the DTLF building blocks. The Service Registry supports service development and - customization, and business service discovery. It also supports matching

[1] *FEDeRATED Milestone 2 Interim MasterPlan*

and alignment with industry – and open standards. The Index functionality supports data sharing according to customized services, including a trust mechanism This is the third functional capability of participants: they must implement particular security requirements like a verifiable credential (VC) with claim(s), authorization, access control, and non-repudation. Data sharing claims refer to plug and play and competent authorities requiring data access for regulations.

The core approach to the functional capabilities is their applicability to any (logistics) business activity, services, and data structures (events and other data sets). The functional capabilities are agnostic of services and service customization. Thus, they can be applied to any use case and any organisation acting in (at least) supply and logistics if the concepts of semantics are properly applied.

The objective of these capabilities is the implementation of protocols. Eventually, a set of protocols must be implemented, with minimal and optional ones. This protocol stack is shown in the next figure with a relation to the capabilities of users or platforms.



Implementation variants

The architecture mentions three implementation variants, namely an adapter to existing IT systems implementing the protocols, a node implementing the protocols with standardized, configurable openAPIs to IT systems, and a gateway based on a node with a more flexible openAPIs interface to IT systems. The proposed protocols are a prerequisite for scaling. Data sharing interfaces between data holders and – users with openAPIs leads to an 'API forest'. A node or a gateway overcomes this forest by reducing it to 'local openAPIs' sharing RDF data amongst each other.

It is up to each organisation to decide the flexibility of their implementation. Preferably, it is completely configurable and thus extendible with new business functionality and flexible to share data with all types of peers.

Since there are not only different implementation variants, but there will be proprietary implementations (for instance by platform service providers and large enterprises) and various COTS (Commercial Off The Shelve) solutions (like a node or gateway). For rapid deployment, it is recommended to implement a conformance testing facility and consider certification testing. All should be web-based, thus enabling relevant users to perform the tests themselves.

Certification testing must be considered in the context of issuing identities. These should be trustworthy, not only based on transparent issuing policies, but also on correctness and completeness of claims. These claims refer to for instance a profile.

Minimal Viable Product

FEDeRATED has developed a validated Masterplan and prototype to showcase how any DDR can sustainably participate in the *EMDS*. The prototype is a Minimal Viable Product (MVP) that is a basis for further development into a product according to the architecture. It consists of two parts, namely a prototype Service Registry (called Semantic Treehouse) and a node to support data sharing between peers. **It is ready to support use cases!**

The Service Registry MVP generates settings for the node for any data structure constructed by the semantic model (a so-called named graph or tree structure). These settings are for data transformation (semantic adapter) and data validation. The Service Registry MVP is also able to generate openAPI specifications that can be transformed into openAPIs for a particular solution.

The node MVP supports the Linked Event Protocol, the Presentation Protocol, and a Connectivity Protocol with a System Security Protocol, where the latter two have a proprietary implementation based on peer-to-peer data sharing solution using freeware. Each organisation can download and implement a node (as Docker or Kubernetes). The proprietary solution implements a mechanism like VCs and needs to be migrated to or replaced by an open solution for connectivity.

The node MVP provides a set of openAPIs to IT backend systems of organisations. The MVP supports event sharing APIs.

The MVP of the Service Registry containing the semantic model has been used to configure the MVP of the node for a Multimodal Supply Chain Visibility Service. This is an example of service development and is specified in a separate document.

Experiments with alignment with other standards and ontologies have been made, e.g. alignment with an ontology for rail and ONE Record. These alignments may result in matches.

Opportunities

By identifying and specifying interfaces between the functional capabilities, a development strategy for each of them is provided. Identification and Authentication requires further attention for (preferably) basing it on eIDAS2.0 and using private developed solutions with standardized interfaces. This enables re-use of existing solutions and services provided by issuers. It also enables adoption by authorities with re-use by enterprises.

There are many other opportunities that require exploration in the context of these capabilities, like:

- Value added functionality. Especially the Index capability can be used to develop value added functionality like automatic event federation to customers and competent authorities stemming from events of physical actions generated by a physical operator (like a truck driver) or sensor. All types of exceptions can be detected like delays or early deliveries, etc. These are all input for constructing so-called Logistics Control Towers or Virtual Watch Towers.
- Core semantics. Applying the (core) semantic model (also called: upper ontology) for data sharing in the person mobility domain and other domains with physical objects (e.g. retail, wholesale, production). An alignment and demonstrator for Digital Product Passports has already been made.

- Verifiable Credentials (VCs) with claims. These are powerful capabilities that can be applied in many situations like physical access to a location, federated data access, permits, releases, certificates, etc.
- Capability Large Language Models (LLMs). This is about support of queries, construction of specifications (event and other data structure) and data transformation by LLMs. A first assessment learned that LLM chatbots for semantic query generation with natural language input already is available. Applying LLMs with ontologies hides complexity of technology even further.
- Training LLMs. Exploration of LLM chatbots for training employees in using the capabilities with a specific focus on data processing by the Index. A query chatbot is potentially a training tool for learning semantic standards. A logistics chatbot may support logistics persons in formulating their data requirements from a logistics perspective instead of an IT (systems) perspective.
- Decision support and planning LLMs. This is about LLMs that are applied for optimization of capacity utilization, agility, and resilience, whereby agility and resilience refer to dynamic chain (re)configuration. These LLMs require input data for learning, like historic events shared between all types of stakeholders.
- Runtime conformance. There are already many process mining tools that can validate conformance of an implementation to a specification or standard based on audit trails and logs. These tools could be applied for continuous conformance testing of an implementation.

Having a semantic model is core to developing and applying these new technologies. It provides flexibility, extendibility, and ease of use with these new technologies.

TABLE OF CONTENTS

Table of figures

1 INTRODUCTION

1.1 *Objective*

This architecture specifies the capabilities creating an infrastructure provision as defined by the Master Plan. In terms of the Master Plan, these are the so-called technical specifications. From an IT Architecture perspective, these are the functional specifications of the various components and their interfaces.

This document is geared to assist:

- EC DG Move to develop an EU Regulatory Framework for the European Mobility Data Space (EMDS) for freight (and potentially persons).
- The EU Digital Transport and Logistics Forum (DTLF) to develop a policy for governance.
- Communities or individual enterprises to develop and implement solutions for an infrastructure provision and innovate in supply and logistics with data sharing.

The document identifies the components for data sharing in supply and logistic chains from a business and functional perspective and the interfaces between the various components specified from a business, functional, and technical perspective. The technical perspective of the components, which aims to enable different stakeholders to implement the architecture using a technology of choice, is outside scope of this document.

As digital technology evolves, this architecture document should be considered as work in progress. It may also still lack sufficient details on some parts. These will have to be finalized by the DTLF. Furthermore, it will possibly also result in a separate protocol stack, potentially also further refined by the DTLF.

The current version of this architecture is an Annex 1 to the Master Plan, due April 2024.

The knowledge gathered and elaborated in this document is based on 5 years of intense discussions, knowledge gathering, and testing in Living Labs within the FEDeRATED project, especially the FEDeRATED IT Architecture Board and Semantic Modelling Group. Input was also provided by DTLF, subgroup 2 and in connection to the EU CEF FENIX project.

1.2 *Background and relevant input*

FEDeRATED has developed a Vision (Milestone 1) and a Master Plan (Milestone ?) to elaborate how the so-called federated network of platforms as identified by the DTLF can materialize. This network approach covers three perspectives:

- Business perspective – language and process. In brief: the condition being that data that are understandable for third parties are eligible to be shared for multiple purposes)
- Functional perspective - discoverability/searchability, access, Identification and Authentication (IA). In brief: browsing the network of platforms requires a trustworthy environment where every party can move freely, find a suitable partner and be found by others, make your preference known and shield away when you want to
- Technical perspective - basic functionality for data sharing (log, audit trail, security, etc.). In brief: the technology toolkit enabling you to browse the network in an accountable way.

These perspectives are more detailed in this document and can be illustrated as follow.

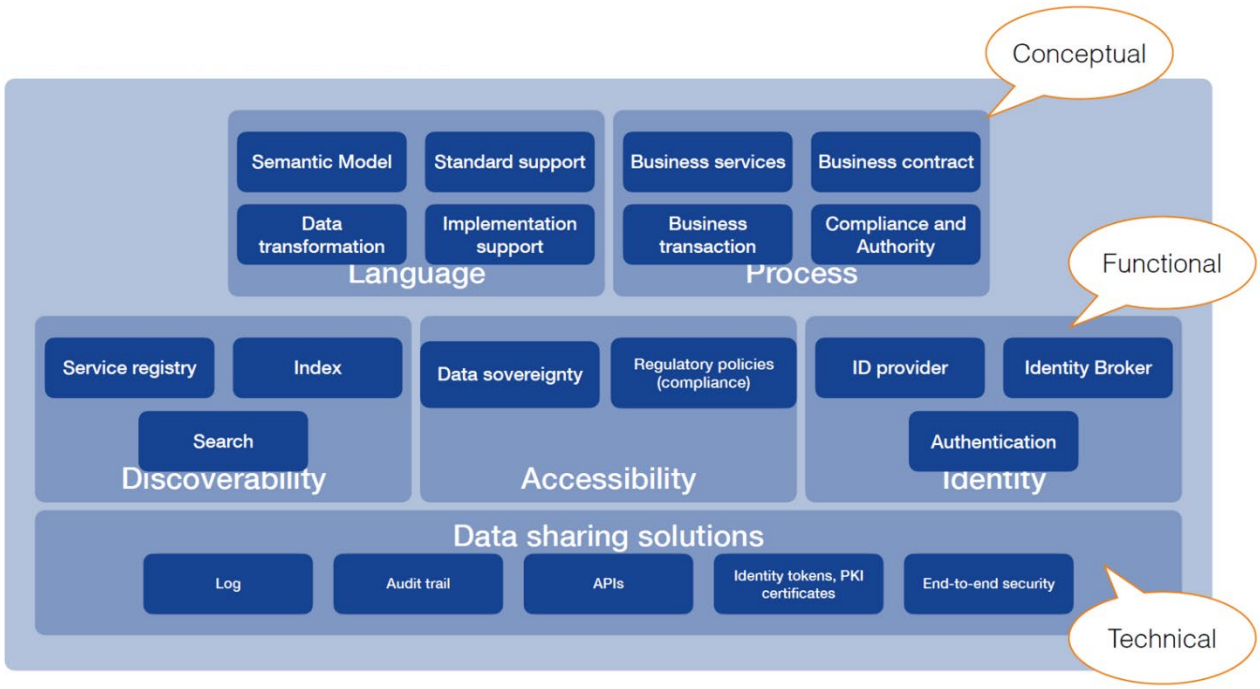


Figure 1-1 The different perspectives of a federated network of platforms

In this Reference Architecture document, the following has been taken in consideration:

- All relevant output produced by the various DTLF Subgroups is used.
- All business processes - Business-to-business (B2B), business-to-government (B2G), government-to-business (G2B), and government-to-government (G2G)
- All transport modes, cargo types, and relevant logistics activities in supply and logistic chains. This does not necessarily imply that FEDeRATED will develop all relevant aspects. For instance, part of the language can be developed by another stakeholder and linked to the FEDeRATED specification of the language.

Relevant international treaties, EU acts and regulations and national legislation effecting data sharing and (cyber-)security are applicable to the architecture presented, i.e. the Hague-Visby Rules, the EU Data Governance Act, and the Cyber-Security Act.

1.3 *Interoperability frameworks for logistics*

There are several interoperability frameworks, each of which are touched upon by the CEF funded FEDeRATED Action.

The European Interoperability Framework (EIF) defines four interoperability layers, namely technical, semantical, organisational, and legal. All layers are required to achieve full interoperability.

Another interoperability framework (reference) has six interoperability layers. Technical interoperability of the EIF is for instance decomposed in connectivity and syntax. Furthermore, they included conceptual interoperability as a way to share data without prior agreements. This framework does not address legal interoperability, since it stems from another application area.

FEDeRATED addresses all four interoperability layers of the EIF, where this architecture specifies concepts (and solutions) for technical, semantical, and organisational interoperability and proposes legal interoperability. Furthermore, FEDeRATED addresses conceptual interoperability. This is required from a logistics perspective to enable for instance resilience, agility, and improved capacity

utilization. All contribute to sustainability by lowering the carbon footprint and reducing waste.

The concept of 'service' and its customization is the core of the proposed solution, together with semantics for multimodal supply and logistics. The challenge is to reduce design time, i.e. the time needed to configure data sharing, and do as much as possible at runtime. Eventually, the objective is to do everything at runtime, based on the capabilities (and new ones) similar to how Google Translate works by using an intermediate language. This results in what is called seamless interoperability. The document presents further research required for seamless interoperability.

1.4 *From architecture to interfaces*

The FEDeRATED Architecture aims to enable the free flow of data between all relevant stakeholders in supply and logistic chains to support the physical flow of goods. The architecture is based on the core DTLF and FEDeRATED principles. These principles set the proposed framework – Reference Architecture - for the development of the advocated digital technology. The requirements are:

- All parties – organisations (IT systems/platforms) involved in supply and logistic chains should be able to participate and profit from it.
- Participating organisations are accountable for their actions. When interfacing, organisations should adhere to a protocol stack:
    - Connectivity protocol
    - Security protocol
    - Presentation protocol
    - Linked Event data protocol
    - Business protocol
- The proposed Reference Architecture will be constantly updated (change management)
- A governance perspective will identify what aspects of the Reference Architecture should be developed and maintained on an EU/international scale and national scale, including the accompanying certification procedures.

The participating organisations are free to implement the Reference Architecture under the condition they comply with the above requirements.

1.5 *Intended audience*

The further development of this reference Architecture will require interaction and commitment of:

- Business analyst – to specify interfaces and interaction patterns in supply and logistic chains applying the 'language' and processing rules, and to analyze their impact on business processes.
- IT architects – to analyze the impact of the protocol stack on existing IT infrastructures and system of an organisation.
- IT developers – to develop and deploy software supporting the relevant parts of the protocol stack for an organisation.

1.6 *Architectural approach*

TOGAF, an open standard, is the common approach for architectures. It consists of various perspectives that must be addressed. The next figure shows the requirements at the core. These start from a vision and are complemented by change management.

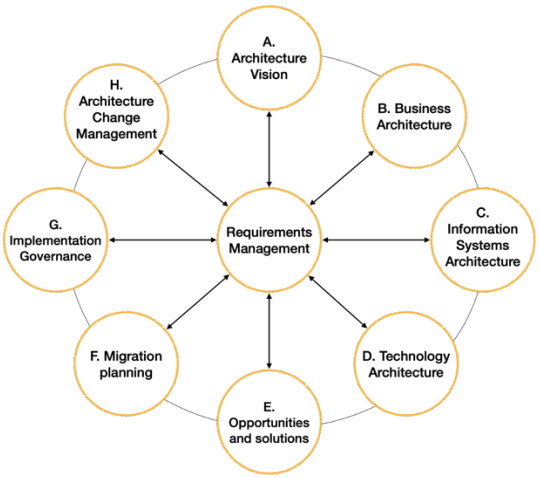


Figure 1-2 Architectural perspectives

The vision is given in the FEDeRATED Vision document, namely the concept of an infrastructure provision or a federated network of platforms. The leading principles for this infrastructure provision are given in the Master Plan; these will not be repeated in this document.

This Reference Architecture will mainly focus on a business – and information systems architecture since the technology architecture can be specific to organisations. However, opportunities, migration, governance, and change management are touched upon.

Whenever an architecture is developed, various alternatives supporting a vision, principles, and requirements are elaborated, resulting in selected alternative. This selected alternative is also based on state-of-the-art technology, which can change over time. This requires change management of both the architecture and its implementation with software components.

Whereas most architectures consider IT software solutions for a single organisation, this architecture addresses data sharing between many organisations. Each of these organisations must implement certain capabilities to use the infrastructure provision. As such, the provision itself is virtual: it only exists if two or more organisations implement capabilities. It is like the Internet consisting of protocols between its components enabling different services.

The infrastructure provision capabilities can be developed and provided by various software and solution providers. Eventually, their interfaces must be standardized to enable interoperability between all types of solutions.

1.7 *Structure of this document*

This document is structured as follows

- Chapter 2 – the core concept
- Chapter 3 – design choices
- Chapter 4 – overview of capabilities, technical representation, and interfaces
- Chapter 5 – the semantic model capability
- Chapter 6 – service development (TIS – Technology Independent Services) and – customization (plug & play) with the semantic model
- Chapter 7 – the Service Registry capability
- Chapter 8 – security capabilities (Identification and Authentication, authorization, access control, etc.
- Chapter 9 – the Index capability
- Chapter 10 – final remarks and future extensions.

2 THE CORE CONCEPT AND ITS APPLICATION

This section illustrates the objective that is to be reached by implementing the FEDeRATED architecture. A reference – serving as an illustration of the core concept - is made the 'intelligent' box[2] The example provides input to requirements to 'language' (see next sections).

2.1 The 'intelligent' box – the story

The core concept is illustrated by a box delivered to a warehouse or depot. This box has several identifiers such as shipper and carrier numbers. These identifiers have little meaning to the warehouse operator. Instead, what if we gave the pallet a universal identifier: one that can be accessed simple, as a two-dimensional barcode on the box that links to additional information such as its content and the action to be performed by the warehouse operator: i.e. unpack and store the content in the warehouse. It just so happens that the W3C (World Wide Web consortium) has defined such a system: the URL (Uniform Resource Locator) or web address linking to data on a webserver, as shown in Figure 1.

Figure 2-1 uniform identifiers (URIs) as links to data

The previous examples shows that the URL of the box consists of a hostname and a UUID, a Universal Unique Identifier. A UUID is a means to generate global unique identifiers, applied in many (modern) IT solutions.

The warehouse operator scans the barcode of the URL to retrieve and update information such as goods receipt at warehouse, storage location in the warehouse, or similar.

Another example concerns trucks involved in road accidents. Emergency services such as the police and fire brigade, as well as many other entities, need to be informed of the nature of the cargo onboard the truck. Is the cargo dangerous, is it livestock, or of any other nature that will influence the work of these emergency services? Where can these emergency services find data pertaining to the truck? The only available data at the time of an accident might be the license plate of the truck and trailer and this can be linked to a URI of the truck; this then enables a link to eCMR (electronic version of consignment information) data, for example, which will provide access to the necessary information.

In all examples it is all about applying the FAIR principles: Findable, Accessible, Interpretable, and Re-usable. Applying URLs as identifiers for accessing data, common security aspects addressing

[2] *is also used in the documentation produced by the DTLF (DTLF Interim report, dtlf.eu).*

for instance identities, and common language and interaction set for business process collaboration are required. Data remains at its source, only links are shared amongst the various entities. All examples have two roles in common as defined by the Data Governance Act, namely a data holder and data user. These issues will all be addressed by the architecture.

2.2 Data browsing – towards innovative applications

Think of all types of goods, assets, and locations having a unique link to additional data. For instance, a terminal can have unique link to its opening hours, geo-coordinates, gates, and any other data that terminal operator wants to disclose. Quays, roads, all types of assets, and packaged products can have such a link. Only bulk cargo like sand or grain will not have a barcode with the link.

Browsing these links provides data of those physical objects and locations. Browsing will become stronger if also links between these objects and locations are available. We enter the area of linked event data where events have a meaning in supply and logistics chains: the ETA (Estimated Time of Arrival) of a truck at a terminal, a vessel ETA in a port of call or alongside quay, and a container loaded on a vessel. Events construct these links: they can have a link of a container and one of a vessel. Events need to include 'time': when are these links constructed, i.e. at which time is a container loaded.

Business documents will also have a unique identifier as a link to its electronic data. Physical objects like goods, containers, and transport means are 'linked' to these documents via (administrative) events.

Thus, having these events makes it possible to browse through all types of data. One can simply access a vessel - or container track or access all business documents in which a container has been linked.

2.3 Distributed data management

Data browsing of links and events gives opportunities but can also bring threats. Using a smart phone, everyone can read barcodes and directly link to a website with additional data and start browsing.

In supply and logistic chain, open access to data is an issue. Data transparency increases vulnerability and liability, resulting in theft, and thus an increase of logistics costs. Access to event data will reveal the content of a container. Data is also commercial sensitive; it shows trade relations, prices, conditions, etc. Even the 'hostname' in the previous example already reveals information. On the other hand, authorities also require data browsing to access the content of a container for instance for customs risk assessment or emergency handling in case the container is involved in a road accident.

Thus, controlled access to links and events combining links is required. It implies that any link on a physical object is just an identification without meaning. The identification can refer directly to source data, which would imply increase vulnerability to unauthorized data access. By combining a meaningless link with additional data that has been shared via events, like a host name of the one that has provided the link, to become meaningful. There can be a business rule like composing a (meaningful) URL is only feasible if one has received the meaningful - and meaningless identifier in advance.

There are cases where the meaningful identifier, i.e. a host name, is published as a web address on for instance a truck. It means that everyone (with the proper credentials) that is able to combine both

identifiers can access the data, unless the public web address differs from a data address (i.e. the so-called endpoint for a data query). The architecture will provide rules for handling these.

In supply and logistic chains, parts of orders can be subcontracted. For instance, these meaningful – and meaningless identifiers of a customer are thus shared by a service provider to its subcontractor. That customer must be sure that only an authorized subcontractor requires access to that link.

2.4 Data semantics and – quality – Digital Twins

When browsing through data, the meaning of that data needs to be clear for processing by IT systems and solutions. Like said, each physical object and location will have a link to its data. A digital representation of those physical objects, locations, etc. is called 'Digital Twin'.

A Digital Twin representing a part of the real-world needs to be specified by its properties. What is the data representation of a Digital Twin representing equipment like a container, i.e. which data properties describe a container and which can be derived? Container number, container size and type, etc. are properties of a container; an indicator of an empty container can be derived from the fact there is not a link to goods or there is a difference between container tare weight and (verified) gross mass.

A Digital Twin also has restrictions as to its association (link) to another Digital Twin. For instance, a container can only be used to carry pallets and boxes that fit, implying it cannot be used to carry another container or goods that are too large to fit. Ferries can be used to transport containers, but only if they are on a trailer. These types of rules reflect the real-world and thus also need to be reflected by the digital world.

To be able to process the data, its quality needs to be specified. Data quality is amongst others about 'correctness' and 'completeness' of data: a minimum event data set required to perform logistics activities, data formats, and association rules between Digital Twins. In terms of data correctness, a container number has for instance a prescribed structure and there are code values for container size and type. These restrictions can be validated before sharing and/or at reception of the data. Any deviations from (a minimal set of) these restrictions have impact on decisions made with this data.

2.5 Data requirements of users – flexibility and extendibility

It will not be possible to have a single organisation specifying these data requirements. There will be modality or sector specific data requirements, like for the chemical or automotive sector. Modalities and sectors can re-use common agreed semantics and extend it for their specific needs. These needs may already be known or will have to support any future applications.

Whenever (communities of) users formulate their specific data requirements, they can publish them and make them available to a larger community. They can become part of the common data requirements that can be used by all users. The latter requires a governance structure.

2.6 Data requirements of a single user

All users, like authorities and enterprises, differ from each other. It implies that each user will have different data requirements to support its business processes. However, many data requirements can also be common to user groups or are formulated by regulations. For instance, particular business document data sets like eCMR are common for a large group of users. Another example is a container track based on browsing the data of various sources.

Common data sharing requirements relate to initiating and completing business transactions, regulations, predictability of supply and logistic chains, and any data sets that reflect liability and responsibility (business document data sets in the current implementation). These requirements are formulated as follows:

- **Business transactions** – this includes digitization of all relevant individual process steps from finding and matching logistics services to their execution (visibility) and payment. There are various ways to implement these processes; best known are framework contracts and shipment-based booking and ordering.
- **Document data sets** – this is about digitization of existing business documents like electronic CMR, - Bill of Lading (eB/L), and – Airway Bill (eAWB). Data can be made available according to the architecture. The structure of these document data sets needs to be expressed by (modality specific) data requirements. Aspects like data integrity are of relevance for these types of data sets, since they reflect responsibilities and liabilities of service providers.
- **Visibility data (events)** – these represent the past, present, and foreseen progress of transport orders. Since there are also various ways events are perceived, implying we need some standardization that is adaptable to user requirements. They reflect predictability of supply and logistic chains.
- **Compliance to regulations** – data will have to be available to an authority, based on a (national implementation of a) EU Regulation. To optimize the implementation, the compliance data requirements must be expressed in the common data requirements, maybe with some extensions specific to a regulation. Please note that compliance can be based on digitization of document data sets, where an authority must be able to accept digitized document data (e.g. eFTI) or a logistics stakeholder has to provide data access digital.

Not all combinations realize interoperability of all business processes between any two stakeholders. For instance, if a user only implements sharing document data sets, it will not be fully interoperable with another one that implemented quotation and ordering as part of logistics services.

Any specific data requirements of a user must be expressed in the common ones, including any industry and compliance specific data requirements, and relate to data sharing capabilities of potential data holders. For instance, a user can never receive container data of a data holder that only transports bulk cargo.

Thus, data sharing capabilities and logistics services need to be known to fully make use of the capabilities provided by FEDeRATED.

2.7 Physical operation – reporting progress

Logistics is about moving, (re-)packing, and storing of cargo and its contents, products. These products can be various, like components, raw material (oil, sand, grain), livestock, and consumer products. To facilitate logistics, various types of equipment are applied. The most common is 'container' in global supply chains. Some (consumer) products require their specific equipment, e.g. car transport is facilitated by specific vessels and chemical require their own containers for transport.

In logistics, the following business activities are distinguished:

- *Logistics (core) activities*: e.g. transport, transshipment/cross-docking, (temporary) storage;
- *Value added activities*: e.g. (re-)packing/stuffing, unpacking/stripping, ironing (of textile), consignment grouping, vendor managed inventory;

- *Supporting physical activities*: e.g. vessel waste management, container cleaning, tugging, stowing, fueling (petrol, hydrogen, electricity), bunkering, parking, maintenance.

These activities have various properties that are represented by events associating (in time) Digital Twins with infrastructure capabilities. For instance, a transport activity is about cargo (goods, containers, bulk, etc.) to be moved from one location to another, based on a customers gaol (expected times), a service providers planning (planned or estimated times), and actual performance of the activity.

Each of these physical activities are controlled by data sharing and they will report their progress according to their instructions. Reporting can be manual, e.g. a truck driver reporting on its trip with pickup and dropoff of cargo, or automated with sensors, e.g. a RFID sensor identifying a train wagon that passed a location or a crane reporting on loading a container on a vessel.

Reporting of physical operations is first to the organisation responsible for the physical operation. These 'events' can also be forwarded to a customer of that organisation that can use them for synchronisation of logistics processes, e.g. different legs in a logistics chain. This is visualized in the next figure.

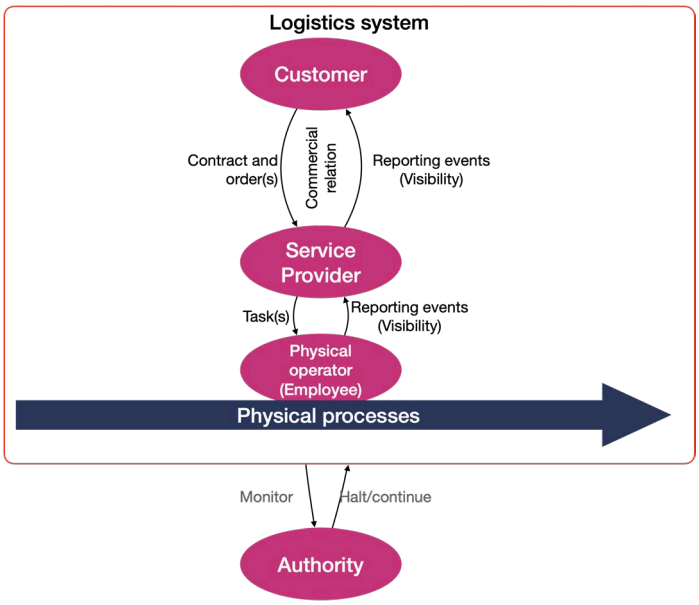


Figure 2-2: basic operation

2.8 Bilateral – versus multilateral (chain) support

This basic operation can be complex, it can be a complete supply chain from a shipper to a provider with various modalities. The Master Plan shows an example of a chain. Globally, there are many supply chains. It is therefore not feasible to specify them all. A common approach is required.

A common approach is only feasible by modelling bilateral data sharing. Multilateral data sharing or chains chains are constructed according to outsourcing strategies of stakeholders in these bilateral data sharing relations. Outsourcing strategies are outside scope; these are the internal processes and decisions of individual enterprises.

These bilateral data sharing relations are represented by 'services': the structured sequencing of interactions ('events') for business process collaboration. An infrastructure provision must provided the following services for business process collaboration:

Services	Definition
Agility service	The structured set of interactions supporting cancellation of an order due to unexpected conditions like delays, losses, or theft of cargo and/or vehicles and potentially triggering re-planning of a (leg of a) logistics chain.
Booking service	The structured set of interactions for negotiating of and concluding a (framework) contract encompassing an agreement of prices and conditions for performing one (or more) business activity(-ies) by a service provider meeting customer goals. A framework contract is an agreement for multiple orders in a period.
Ordering service	The structured set of interactions for actual execution and detailed planning of a business activity according to prices and conditions resulting from booking service.
Quotation and marketplace services	The structured set of interactions to discover (logistics) business services meeting customer goals. This service needs to implement a high precision and recall, all logistics services, timetables, and spare capacity meeting customer demands have to be found.
Resilience service	The structured set of interactions assessing risks in completion of supply – or logistics chains or individual transport legs based on Information Services. Resilience services implement supply chain resilience.
Visibility service	The structured set of interactions providing details of the execution of a business activity and its planning by a service provider according to an agreed transport plan.

These services need to be specified in the context of a business activity, e.g. transport, transship, and stuff/strip.

These B2B services must be supported by others like financial service (insurance and payment). These must be provided by financial institutions.

Not all these services are always required for all business activities and all customer – service provider relations. In some cases, business services are a commodity: there is no negotiation about prices and conditions, those provided by the service provider are applicable.

Like shown in the previous figure, supervising bodies can monitor these services, according to the regulations they must supervise. Logistics stakeholders can provide data access to supervising bodies based on their data requirements for compliance to regulations.

2.9 Autonomous operation

Each individual organisation in this organisational network is autonomous in making decisions. To improve decision making, for instance by selecting a certain service provider or taking a particular route, requires access to additional data and making data available to others. Examples of these data sets are traffic information, water depth, business services, flight – and voyage information, capacity availability, etc.

Sharing this data can be restricted by the data holder, e.g. capacity data is consider commercial sensitive and might influence prices. This is called ‘data sovereignty’: a data holder decides which data is accessible to others with transparant conditions.

Besides organisations, which are ‘legal persons or – entities’, also natural persons and autonomous

operating ‘things’ can make decisions based on data and decision algorithms. Autonomous barges and vessels are examples of these things. It is distributing processing capabilities to the edges. Other assets like containers and trucks can make autonomous decisions (soon). An infrastructure provision must enable these types of innovations.

3 Design choices of the infrastructure provision

This chapter identifies the various perspectives to the architecture. These perspectives give guidance to development, maintenance, and deployment of the various components by the different stakeholders. The constitute the basis for creating a protocol stack.

3.1 The context of an infrastructure provision

The objective of the EMDS (European Mobility Data Space) is to create an open, neutral environment that can be used by all logistics stakeholders, the so-called federated network of platforms. Since 99,8% of these logistic stakeholders are SMEs (Small and Medium sized Enterprises), standard solutions must become available. This standardized solution will only be developed for SMEs if the four building blocks of a federated network of platforms are addressed.

Underlying the leading principles of the Master Plan are the building blocks of the DTLF SG2 (Digital Transport and Logistics Forum, Sub Group 2 – Corridor Information Systems), the Core Operating Framework of the FEDeRATED Vision, and the interoperability layers of the European Interoperability Framework (EIF).

3.1.1 DTLF SG2 building blocks

The federated network of platforms consists of four building blocks as identified by the Digital Transport and Logistics Forum Subgroup 2 (DTLF SG2):

- **Plug and play** – each user should be able to register and connect to a platform of choice and select the services it needs.
- **Technology independent infrastructure services** - the services of the platform should be designed technology independent, thus enabling different providers to offer a solution that best fits their end-users and to support different technologies for realizing the services.
- **Federation** – the commodity will establish harmonized connectivity and interoperability of different solutions (platforms). It will consist of platforms of different service providers, whereas these platforms can also operate in an enterprise domain, thus creating so-called peer-to-peer solutions.
- **Trusted, safe and secure** – the commodity and its (integration with) end-users should be trusted, data sharing should be safe, secure and based on minimal central governance.

3.1.2 Core Operating Framework

FEDeRATED developed a Core Operating Framework guiding the further development of a federated network of platforms approach, constituting the following key principles:

1. **Ensure data sovereignty** – data that is exchanged is made available by the data owner through a pull/push mechanism. The data then consumed is based on a push/pull mechanism received as a push by an authorized recipient. A data owner grants Access to the data ten to the authorized recipient.
2. **Create trust among all stakeholders** - Any infrastructure provision[3] facilitating data exchange should contain various mechanisms, service and solutions that contribute to trust in using the infrastructure. Not only should users be identified, but also particular active

[3] The FEDeRATED Core Operting Framework is also based on the need for establishing an infrastructure provision, see chapter 4

attacks to the complete infrastructure should be prevented. Data privacy should also be respected.

3. **Provide a framework to enable interoperability** - to enable interoperability amongst all stakeholders, providing a level playing field, each user of the federated network of platforms should be able to do business digitally without making any additional data sharing agreements. There should be a mechanism during registration at which each organisation will be able to formulate its data sharing policies, expressed in their business services, timetables, voyage schemes, distribution patterns, and what you have. These mechanisms should be supported with results of the semantic interoperability layer (see before).
4. **Be open and neutral to[4] any participating party**. The infrastructure provision for data sharing in supply and logistics should be open for anyone to use, be easily accessible for any parties, where each party can make the choice to implement the existing infrastructure provision Agreements themselves or connect to a platform that implements these agreements.
5. **Ensure data quality** - various dimensions, e.i., completeness, correctness, and consistency. Data quality also needs to be considered from different views. Data quality is required to meet all goals in supply chain synchronization, innovation, and other opportunities. Data correctness might be enforced based on more technical representations that can be implemented by IT systems. Enforcing data correctness will increase data quality but might decrease the data volume that is shared and made available to for instance authorities.
6. **Rapid deployment of new services** – new services must become rapidly available to all organisations implementing capabilities of the infrastructure provision. It is like the distribution of new functionality (‘smart contracts’ or ‘apps’) with a (public) blockchain network or Appstore.

3.1.3 European Interoperability Framework

Digital transformation of individual organisations requires the use and integration of digital technologies into existing (and new) business processes as well the four layers of the European Interoperability Framework (EIF) that must be addressed, namely:

- **Technical interoperability,** covering applications and infrastructures linking systems and services. Including Interface specifications, data integration, exchange and interconnection services, and secure communication protocols.
- **Semantic interoperability,** ensuring that the precise format and meaning of exchanged data and information is preserved and understood throughout.
- **Organisational interoperability,** documenting and integrating or aligning business processes and relevant information exchanged.
- **Legal interoperability,** ensuring that organisations operating under different legal frameworks, policies and strategies can work together.

[4] See FEDeRATED Vision Milestone 1, pages 68 and 69

3.2 Stakeholder groups

The focus of FEDeRATED is on the following stakeholder groups:

- **Public authorities**, policy as well as law enforcements agencies, inspections, Port administrations, customs administrations, etc.;
- **Supply and Logistic chain operators** – terminal operators, transporters (seagoing maritime, rail, hauliers, inland navigation, aviation), forwarders, shippers, sellers, consignors and buyers, private port operators;
- **Industry Associations** – these often develop amongst other guidelines for data sharing with and on behalf of their members. They are organized by for instance modality (e.g. road, rail, air, inland waterways, and sea) and/or cargo type (e.g. chemical industry, commodities, container transport by sea).
- **IT solution/platform providers** – these stakeholders deploy services to enable data sharing by supply and logistic chain organisations. They will have their own business – and governance model and deploy a particular data sharing paradigm. As of currently, most of these providers support messaging and APIs. New entrants might support Linked (semantic) Data. Integrators and platform service providers are examples.

Recognized standardization bodies like ISO, CEN CENELEC, OASIS, W3C, and UN CEFACT can also be involved as stakeholder groups. However, these bodies are currently not facilitating heterogeneity and complexity as described hereafter.

3.3 Service development and - customization

The previous section introduced the concept of 'service' in the context of logistics business activities. A multimodal supply chain visibility service is an example of a 'service' for transportation of cargo. The following main assumptions are applicable to the infrastructure provision:

1. Service development: any (collaboration of) organisation(s) must be able to specify its (their) own services. These organisations are for instance platform providers, industry associations, and regulating bodies.
2. Service customization (and – implementation): any individual organisation (enterprise, authority) must be able to configure and operate these services to support their business activities according to their business strategy compliant with the FEDeRATED Vision (see also section 3 of this document).

Service customization can be considered as constructing implementation guides of services that are implemented by a single organisation. Each organisation has its implementation guide(s); these share the services they have in common. These services are implemented for data sharing.

Underlying these two assumptions is autonomous decision making by each organisation. Each organisation has its business strategy implemented by business processes and IT that requires services for data sharing in bilateral collaborations. To elaborate:

1. Service development illustrates that the services of the infrastructure provision can gradually be developed, for instance by a community. It also provides the capability to organisations for developing innovative services. To realize the FEDeRATED Vision, all services must be developed according to the same approach, i.e. using the same concepts, although services will cover different functionality. Service development must be supported

by the capabilities of the infrastructure provision and is about creating the Technology Independent Services as identified by DTLF SG2.

2. Service customization addresses the issue of onboarding to the infrastructure provision. Any organisation can join the infrastructure provision and implement those parts of relevant services for supporting its business activities. This refers to the 'plug and play' principle identified by DTLF SG2.

Whenever a platform develops services and acts as a participant in the infrastructure provision, its users must be discoverable according to the principles of 'open', 'neutral', and 'level playing field'.

The infrastructure provision consists of capabilities for service development and - customisation as shown in the next figure. Services and their customisation can require new capabilities and services will be changed when their customisation by individual organisations requires new functionality, i.e. additional data requirements.

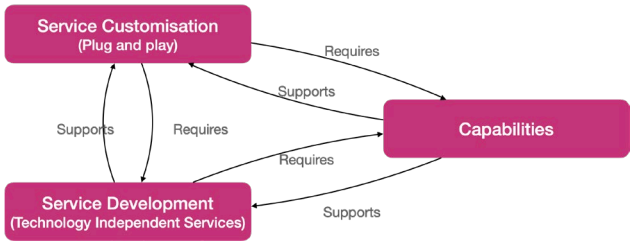


Figure 3-1: Main assumptions of the infrastructure provision

The capabilities must enable rapid service deployment as mentioned in the Core Operating Framework.

The Architecture distinguishes two types of services, namely business services as the way business activities of an organisation are available to their customers (marketing and sales perspective) and business process collaboration services (operational perspective). The latter are called 'service' in this document.

It is foreseen that individual organisations can deploy their services in the future if these services can automatically be integrated with IT backend systems of their peers. This is with the current state-of-the-art technology not yet feasible. Thus, services must be common to at least two organisations and preferably to all logistics stakeholders.

3.4 Design choices for functioning of the infrastructure provision

The infrastructure provision itself is a concept, in the sense that it exists if end-users implement and configure capabilities for services supporting their business activities. This is based on the following design choices:

- Any two organisations that share data must have a service in common.
- Data sharing of each organisation, and its implementation, is decoupled from that of others. This allows rapid onboarding of an organisation, with a customized service.
- Existing or new services must be configurable at runtime. This allows for reconfiguration of data sharing by the infrastructure provision.
- For integration with IT backend systems, a generic set of openAPIs configurable for (customized) services will be available.

These design choices enable onboarding: each organisation can join and configure the provision

independent of another. A service implementation is part of the organisational identity, which can be tested and certified and is the basis for supervision of legal agreements.

To enable that these organisations are still able to share data, i.e. their service implementation must be matched, services are related to business activities like transport and transhipment performed by those organisations and published according to their business services.

Any two organisations must have at least one service in common to enable data sharing. Whether they themselves have developed this service or re-use a service developed by another organisation is irrelevant to the infrastructure provision. Of course, the more common a service is, the more organisations will support that service.

The adoption of a service is by service development: what is the size of the community and are first movers involved in developing a service. The larger the community, the more flexibility a service will have, the more details must be given in a profile.

These design decisions enable rapid deployment of new – and all types of different services. It enables communities to develop and deploy innovative services with the same infrastructure provision they apply for other services, resulting in cost reduction. The infrastructure provision is extendible and flexible. Whenever a new service is configured and an organisation wants to support that service, its IT systems might have to be adopted. The capability for implementing such a change, specifies the flexibility of an organisation to cope with changes.

OpenAPIs are state-of-the-art for development and adaptation of IT backend systems, although there will always be IT backend systems with file-based interfaces. Since there will be different services and the vision states these services must be rapidly deployed, a generic set of openAPIs will be provided. These generic APIs will be configurable for services. Therefore, these openAPIs will be part of the capabilities.

The applicability of the infrastructure provision is defined by governance of the capabilities and service development with these capabilities, which refers to the legal agreements.

3.5 Heterogeneity and complexity

The semantic model includes all modalities and types of goods for both business relations and compliance to regulations. There are several aspects of interest with respect to this model (in analogy with 'language'):

1. **Extendibility** – the model will always evolve with new data sharing needs, i.e. new services. Companies, industry associations and legislators (at Member State and EC level) must have the means to further develop the model.
2. **Own 'dialects'** – it is not possible to support everyone from a central organisation with solutions to participate in the infrastructure provision. Two approaches must be supported:
   - Every organisation must be able to formulate which data it can make available and integrate with their internal systems (plug and play or service customization).
   - Organisations must be able to make agreements for their requirements themselves. This also depends on the scale; it is the intention of FEDeRATED to offer solutions for the European Union. Organisations must be given the means to specify their own data needs and data to be delivered. They can form communities.

   Note that some may call a community that implements its 'dialect' constitutes a data space. Via the common language, these data spaces are by nature interoperable. Thus, governance processes are required.

3. **Own 'language'** – there will always be organisations and associations that have their own language, for example their own standards. A translation with that language is needed, which is called 'matching'. An association having its own language is no problem if users of this language are only applying that one and not active in other industry areas.
4. **'Standard phrases'** – the model offers so much freedom that predefined settings are required, for example for electronic documents (eCMR, eB/L, etc.) and supply chain visibility (see before). Often these types of standard 'phrases' are specified by industry associations and regulatory bodies. For example, these standard phrases provide access to data in internal IT systems from a link that has been received. This is necessary for eFTI implementation, for example, but the same mechanism can also be used for EMSWe and for multimodal supply chain visibility.

These kinds of aspects of 'language' are related to governance – who has control over 'language' and how does it connect with other 'languages'. For this, resources, 'tools', must be developed. A governance should be involving all stakeholders.

3.6 Data sharing options

Whereas the FEDeRATED Architecture prescribes sharing of triples between any two implementations of the Index, there are still additional choice to make, including support of other data sharing paradigms. These will be introduced here. They require additional functionality.

There are basically three ways of data sharing, namely:

- **Messaging (e.g. EDI)** – data is sent by a data holder to a data user. A data user must be reachable by a data holder to share a message. In most cases, these messages support business process integration of a data holder and – user.
- **Application Programming Interfaces** (APIs) – (sets of) APIs are used to access (GET) or share data (PUT/POST). In case of data access, a data user can retrieve data when it is required. A data holder must be reachable. Otherwise, the mechanism is like with messaging.
- **Linked Data** – links to data are shared amongst stakeholders. These links can be evaluated to retrieve the data.

Linked Data can be implemented by (generic) APIs; these could also be semantic APIs (so-called SPARQL endpoints). Linked Data implements the FEDeRATED data pull principle, which implies that this is the design choice made by FEDeRATED for data sharing. Data is kept at the source and only access is distributed via links to that data.

Messaging is a push-based way of data sharing, where data is duplicated by a data holder to a data user. Data push can be implemented by accessing data (automatically) via a link that is received.

Implementation of openAPIs is useful if their number is (relative) low. Therefore, FEDeRATED recommends their implementation for integration of the Index capability with IT backend systems, instead of end-to-end implementation between different organisations. Making these openAPIs generic and configurable, implies that the software supporting an openAPIs has limited validation functionality. Additional validation functionality must be configured by data semantics.

3.7 Legacy integration and standard support

A solution must be able to integrate with existing IT systems (legacy systems) and support (defacto/open) standards. This will assist the migration of organisations and platforms towards the proposed solution.

Organisations, enterprises, authorities, and platform providers, have IT capabilities, either proprietary, Commercial Off The Shelve (COTS), or cloud-based solutions based on openAPIs of a cloud provider. They may have implemented messaging standards like EDI and these IT systems may support openAPIs or have a file structure interface (e.g. Comma Separated Value (CSV), XML (eXtensible Markup Language, and JSON (Java Script Object Notation). Cloud service providers may offer semantic web technology like triple stores or RDF plugins. This refers to heterogeneity of IT (legacy) systems.

Industry Associations, standardization bodies, and dominant players have specified and implemented interfaces, open standards, and/or guides for the implementation of open standards. Certain industries may have adopted interface specifications developed by providers that thus become defacto standards.

4 Capabilities constituting the infrastructure provision

This chapter shows the capabilities that are required to create the infrastructure provision. Details of each of the capabilities can be found in other chapters. The capabilities are implemented by roles.

4.1 The capabilities

To participate in and benefit from an infrastructure provision, its participants should be capable to comply with some technical specifications – capabilities - are:

1. **Semantics** of the data that is shared in multimodal logistics applications.
2. **Identification and Authentication** (IA – security perspective) providing trust according to agreed rules.
3. **Service Registry** for service development, - customization, and discoverability.
4. **Index** for data sharing and authorised access via events with links to data (data pull).

In addition, to these capabilities this section introduces a semantic adapter. Although it is not listed as a separate capability, a **semantic adapter** is required to adapt existing IT systems or (defacto) standards to semantics. One could call a semantic adapter a tool.

These four capabilities are specified in more detail in separate sections.

4.2 Relations between the capabilities

Semantics is at the core of the solution. It is fully implemented by the Index functionality, can be applied in the Service Registry for service development and service customization, and is used for discoverability and matching of data sharing capabilities of organisations based on their customization (their 'profile'). Authorization and access control relate to service development and a profile, based on sharing (links to) data.

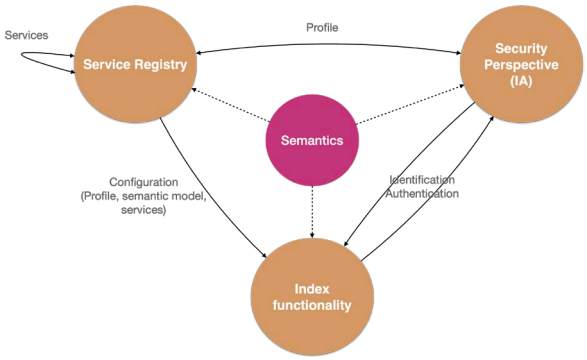


Figure 4-1: Relations between the various capabilities

This section also lists the maximum and minimum functionality of the Service Registry, Index and Semantic Adapter. Minimal functionality focusses on the need to fit into existing processes and IT systems that support part of the other functionality e.g., implementing framework contracts and/or (unstructured) person-to-person communication.

4.3 Basic design choices for the capabilities and their interfaces

To optimally meet requirements, semantic web technology for data and Business Process Modelling (BPM) for process aspects have been selected:

- **Data requirements representation (semantics)** – these are expressed by a semantic model represented as an ontology with various constraints. OWL (Ontology Web Language)

SPARQL (SPARQL Protocol and RDF Query Language), SHACL (SHApe Constraint Language), and RML (RDF Mapping Language) are some of the applicable open standards.

- **Data representation** – Resource Description Framework (RDF) or Java Script Object Notation – Linked Data (JSON-LD) are examples of a syntax that seamlessly integrates with its representation. Triple stores are used for storing data directly associated with their semantics.
- **Services** – services are modelled by Business Process Modelling notation (BPMn2.0) choreographies. This is an open standard for specification of the foreseen types of services since they only specify behavior of two stakeholders (section 2.8). Data representation relevant for services are represented by SHACL. There is not yet an open standard for the elements of a choreography.

The main reasons to apply semantic technology are:

1. it best fits the design choices: it is extendible and flexible, and supports distributed service development and - customization;
2. it can be applied for linking various data sets to supply and logistic chain data;
3. it creates mappings between all types of internal data sets and (implementation of) open standards;
4. the technology can be used for data preparation in data analytics; and
5. semantic technology can be interface with legacy data models via the semantic adapter.

For creating capabilities, all relevant aspects must be machine-readable constructs that can be referred to, updated, etc. for sharing data. Semantic technology is selected since it is supported by open standards and open-source tools, but also by freeware and commercial (cloud) service providers. A so-called upper ontology can be constructed representing data requirements for multimodal logistics operations as a basis for lower ontologies specifying details, e.g. for a modality. The upper ontology represents the main concepts that can be specialized by lower ontologies. Governance is still under development in the DTLF; the management perspective will support governance.

Another reason for applying semantic technology is in data preparation for data analytics. Data analytics considers three main aspects, namely data quality, technical – and business expertise. The business expertise is for applying data analytics in the context of business/logistics processes. Technical expertise is on the infrastructure and data management aspects, in relation to for instance data analytics applications.

The technical expertise refers amongst others to knowledge of data and its semantics. Studies have shown that data analytics involves data mining, data management, statistical analysis, and data presentation, before actual performing data analytics and interpreting the output of data analytics. The following figure gives an estimate of the time spent on various steps in the context of data analytics.

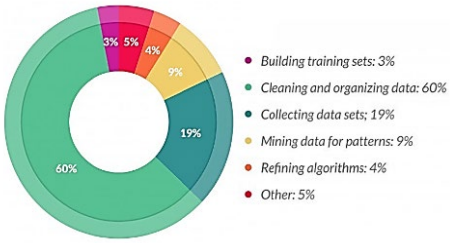


Figure 4-2: estimation of time spent on data analytics[5]

The previous is just an example, similar estimates are given by other sources. The objective of applying semantic technology and the architecture given in this document is to reduce the time of cleansing and organizing data and collecting data sets and thus have a better focus on the other activities (mining, refining algorithms, etc.). By applying a common semantic model that is applicable for any supply and logistic chain operators and not specific for customs, the technical expertise is also expected to increase.

In general, one could state that having a common language for multimodal logistics will contribute to innovation, since the potential market for applications increases. This is applicable to data analytics and all other types of applications that can be developed to support logistics service providers, their customers, and authorities.

At present, many data platforms use traditional data models and semantic technology can be used to interface with those models. This creates an opportunity for migration from traditional data models to semantic models and use both in parallel to provide both backward and forward compatibility.

4.4 Interfaces between capabilities

The capabilities can be decomposed further as shown in the next figure. The Service Registry is for instance decomposed into three components, namely a component for service development, another for service customatisation, and a third component for business service management and -matching. The following components will be elaborated in this architecture:

- Service Registry components – service development, query formulation, service customisation, and business service management.
- Index functionality.

The 'registration' -, conformance test component, and IT legacy systems are out of scope, although the interfaces must be specified. For IT legacy, these interfaces can be identified from a set of potential interfaces. The interfaces with the registration component will depend on the implementation of an SSI/VC based infrastructure (see Identity and Authentication), which will result in the implementation of an organisational wallet as part of the Index.

A conformance test component is the reference implementation for business activities and their services implemented by a profile. The objective is to test and validate the implementation of a profile. A conformance test component can also be used for certification purposes by validating an implementation of the Index against certain mandatory test runs. Another use is a continuous monitoring of an Index.

[5] Sarih, H. et al – Data preparation and preprocessing for broadcast systems monitoring in PHM framework, 6th Internetional Conference on Control, Decision, and Information Technologies (CoDIT'19), April 2019, Paris, France.

The figure shows that an Index interfaces with local APIs to internal IT legacy. This is one way of implementation; others are described later in this document.

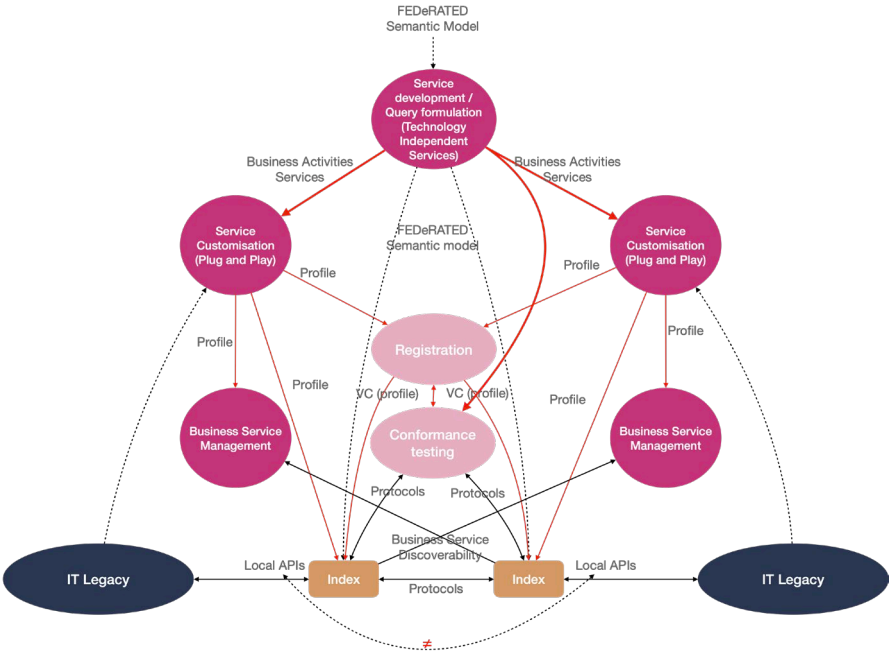


Figure 4-3: interfaces between the various components

Although the decomposition is not visualized, an Index includes (amongst others) data validation and data transformation by the semantic adapter.

4.5 Baseline standards for interfaces between capabilities

The semantic web standards of the World Wide Web Consortium (W3C) are the baseline standards for the multimodal ontology and interfaces between the capabilities of the infrastructure provision. They are applied as follows:

Interface	Description	Interfaces between capabilities	Baseline standard
FEDeRATED semantic model	Ontologies and their constraints (the semantic model or 'language')	Service Registry – Service Registry Service Registry – Index	OWL (Ontology Web Language) SHACL (SHApe Constraint Language) SKOS (Simple Knowledge Organisation System) – for code lists only.
Business activity (data)	The data set specifying all data that can be shared for a business activity	Service Registry – Service Registry Service Registry – Conformance Testing (Service Registry – Index to optimize the operation of the Index)	SHACL (supported by a triple store or graph database in the Index)
Service	Specification of event	Service Registry –	No standard available yet;

Interface	Description	Interfaces between capabilities	Baseline standard
(process)	sequencing	Service Registry Service Registry – Conformance Testing Service Registry – Index	modelling by BPMn2.0 - choreography
Service (data)	Specification of event structures	Service Registry – Service Registry Service Registry – Index	SHACL (configured for data validation in the Index)
Profile	Customization of a business activity and its services by an end-user	Service Registry – Registration Registration – Conformance Testing Service Registry – Index (this may optimize the Index replacing a configuration of a complete business activity)	SHACL (part of a VC and used for matching data sharing capabilities between Indexes)
Business service	Discoverability of an end-user for the business services it can provide	Service Registry – Index	RDF (business services and goals; as part of the discoverability protocol)
Data transformation	Configuration of the semantic adapter for data transformation	Service Registry – Index (semantic adapter)	RML (RDF Mapping Language; the semantic adapter is part of the Index)
Events and data	Sharing of events with links to data and accessing the data for those links	Index – Index	RDF (Resource Description Framework)
Queries	(complex) queries for data retrieval	Index – Index	SPARQL (SPARQL Protocol and RDF Query Language)
Information services	Data made available by a data holder	Index – Index	Metadata standard (e.g. DCAT and/or Dublin Core) SHACL (data) ODRL (Open Document Rights Language, access rights).
Verifiable Credentials	Identity of an end-user that can be authenticated (verified)	Registration Authority – Index	Open standards (under development)

Interface	Description	Interfaces between capabilities	Baseline standard
	by another end-user	Index – Index	
Authentication	A means to get authorized access to data that can be verified (an alternative for VCs)	Index – Index	OAUTH2
Local interface	The integration of an IT backend system with an Index	IT backend – Index	OAS (openAPI Specification) Any others (JSON, XML, CSV or other syntaxes used for file exchange)

There is a wide range of tooling, triple stores, and graph databases supporting RDF and OWL/SHACL.

For implementation of an SSI/VC based infrastructure open standards must be applied, preferably those relevant to eIDAS2.0, since that is expected to be implemented for business-to-government data sharing. The Architectural Reference Framework (ARF2.0) of eIDAS2.0 still needs to be extended with functionality required by supply and logistics (this is called the EMDS – European Mobility Data Space, see before).

4.6 Roles and responsibilities

FEDeRATED distinguishes serveral roles and responsibilities in the context of capabilities, service development and – customization, and data sharing. The tasks of these roles may be extended towards implementation of the capabilities, e.g. by generating configurations as will be explained later.

The roles are:

- Regulator
  The authority (or community) that sets the rules for operation of the infrastructure provision as a basis for issuing policies of VCs. These can be private – or public rules. The latter is an infrastructure provision Regulation and is the preferred option to prevent any incompatibilities between private rules that prevent the construction of an open environment. In case of a Regulation, the EC is the Regulator. Such a Regulation is voluntary for business-to-business data sharing but can be made mandatory for business-to-government and government-to-business. Any rules may address the implementation of the capabilities by other roles and the internal processes and IT systems of organisations implementing these roles.
- Service Developer
  Any organisation, community, or regulator body that develops and publishes service(s) for business activity(-ies) or in the context of a regulation with the multimodal semantic model. A Service Developer can voluntarily implement the infrastructure provision Regulation and receive an Identification (VC). A Regulator specifying data requirements for a regulation can also act as Service Developer.

A Service Developer develops and publishes services and queries with the module of the Service Registry that supports service development. Service development is about:

- Existing standards into services. A Service Developer may already have standards specified in a particular format (e.g. XSD or JSON structures). These formats must be transformed into an ontology and aligned with all relevant concepts of the multimodal (upper) ontology, thus creating a service for these standards.
- New services. This can be done by specializing upper ontology concepts and properties.
- Queries on the infrastructure provision – creation of queries on existing services for business activities specified by other Service Developers. This may result for instance in the eFTI data set as part of a visibility service for road transport.

- Data holder / - user

  The organisations that make data available (data holder) to others (acting as data user). These roles are specified by the Data Act. Any organisation acting as data holder or – user and (voluntary) implement the infrastructure provision Regulation for the capabilities must implement relevant (parts of) services developed by recognized Service Developers and publish these as 'profile(s)' that are certified. There are two types of data holder / - user, namely supervising bodies and enterprises:

  - Supervising bodies

    Any supervising body implementing the governance of one or more (parts of) a Regulation(s).
  - Customer / Service Provider (enterprise roles)

    Any enterprise can take the role of customer requiring meeting particular (logistics) goals by business (logistic) services of one or more Service Providers, compliant with relevant Regulations. Business services are instantiations of a business activity(-ies) given by the profile(s) of an organisation. Business services are for discovery of potential Service Providers.

  A data holder and -user are able

  - to develop a profile for a service or services (those for business process collaboration and compliance) as a basis for data sharing. A profile specifies the behaviour of an organisation in the infrastructure and is used by enterprises to formulate their business services. A profile is also part of a Verifiable Credential after it (a profile) has been validated.
  - to specify and publish business services for discoverability an enterprise based on a goal formulated by a potential customer.
  - to obtain an Identity of a trusted Registration Authority for one or more profiles.
  - to implement Index functionality with its Identity and profile(s).
- Registration Authority

  Any organisation that voluntarily implements the infrastructure provision Regulation for providing Identities (Verifiable Credential) to data holders and – users after certification by a recognized Certification Authority. A Registration Authority can be acknowledged by a community (or country administration) implementing the infrastructure provision Regulation and considered to be mutual recognized if it implements that Regulation.
- Certification Authority

  Any organisation that can certify the implementation of profile(s) by the Index functionality of an organisation and any other aspects of an infrastructure provision Regulation.

Preferably, all roles are separated (separation of concerns) to increase trust. However, some roles are combined (customer/service provider and data holder/-user) whereas others might be combined (e.g. Registration – and Certification Authority) depending on potential risk for decreasing trust.

Data users and – holders can also take the role of Service Developer, either as a community or as

strong players specifying their services. The latter can be useful for organisations that want to develop and test innovative services, although they should preferably include others for adoption.

The size of the community of Service Developers defines the applicability and degrees of freedom of those services and thus the complexity of a profile. For instance, a multimodal visibility service will support all types of cargo and modalities, whereas a road carrier for container transport will only require a subset of that service. This subset is specified by its profile that is the basis for business services.

To start simple, an organisation may want to develop a single business service reflecting its profile. For instance, a barge operator for container transport will provide that service along the Rhine-Alp corridor.

4.7 Layering of the overall functionality

The various capabilities will implement functionality according to the FEDeRATED Vision. 'Discoverability', 'access' and 'IA' together constitute '**data sovereignty**', being a (main) principles of data sharing infrastructures in general as identified in the FEDeRATED Vision further elaborated by the FEDeRATED Master Plan. A common language is needed to process data.

The functionality provided by the FEDeRATED capabilities have also been identified by DTLF SG2, as follows (see also next figure Figure 4-4):

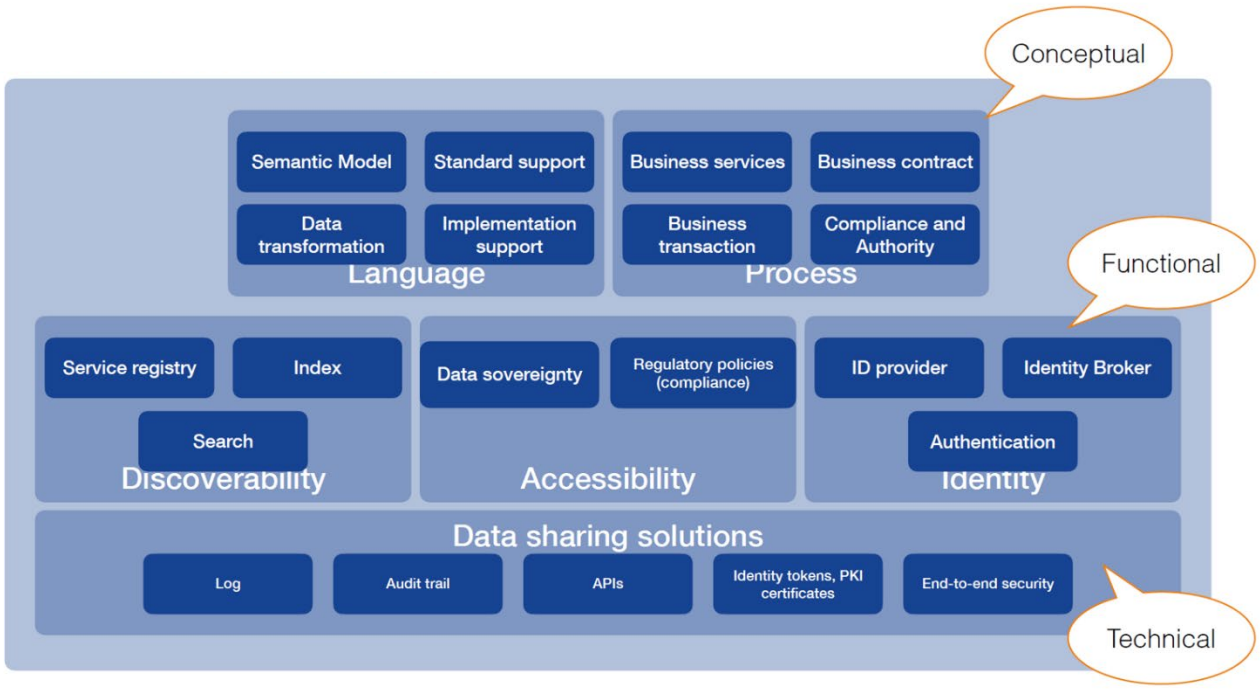


Figure 4-4: DTLF II SG2 building elements of the architecture

1. Conceptual level. These are conceptual building elements supporting semantic – and organisational interoperability. They are:

    - Language – a semantic model supporting data sharing in supply and logistic chain operations (multimodal) supported by tools and algorithms for its implementation by individual organisation, supporting required standards.
    - Process – data sharing between business processes of collaborating organisations, business-to-business, business-to-administration, and administration-to-busines.

2. Functional components – the required components to realize data sharing in a technology independent way. These are grouped into discoverability of data and business services, data sovereignty for B2B and B2A based on access control compliant with regulations, and IA mechanisms aligning with existing solutions like eIDAS2.0 and other token-based (defacto) standards like OAUTH and JWT).

3. Data sharing solutions – these are all types of solutions (proprietary, COTS – Commercial Off The Shelve, and open source) and third party (platform) services. They must implement functionality, independent of any application area. These components are said to be data agnostic. The functionality basically supports non-repudiation (immutable log and audit trail), technical standards like openAPIs, IA solutions (like organisational wallets for VCs or identity tokens), and end-to-end security mechanism to share commercial - or privacy sensitive data.

The conceptual level is independent of any implementation. It is used to configure the discoverability and accessibility components. Whenever an organisation selects a data sharing solution, the identified functionality needs to be provided. If a solution does not provide it, an organisation needs to implement it additional to that solution. The DTLF SG2 figure shows some additional features of the architecture, especially the generic requirements of data sharing solutions. These will be addressed by the architecture in this document.

5 Language – semantic model

This section elaborates the semantic model. The model itself can be found at the developer portal of federatedplatforms.eu.

The semantic model is the core of the architecture. All other capabilities are specified for handling the structure and syntax of the semantic model, either by specifying settings like business services or by processing events and data.

The section is structured as follows:

- Requirements to the model – listing the types of data that must be implemented by the model.
- Base structure of the model – the base concepts for implementing the requirements.
- Examples of applying the model – some examples of using the model.
- Events and date/times – the relevance of date/times and events in the context of data sharing for physical activities.
- Cargo perspective – operations on goods and equipment.
- Transport means perspective – representation of itineraries.
- Details of the semantic model – a high level overview of the main concepts of the model.

Service development by applying the semantic model and service customization for implementing services are specified in the next section.

5.1 Requirements to the model

The FEDeRATED semantic model must support sharing data of various real-world objects for basic physical activities like transport and transhipment of cargo. various stakeholders share data to coordinate their various activities. Figure 5-1 shows these main aspects that must be represent by the semantic model.

Figure 5-1: The main concepts for composing the semantic model

These main concepts are described as follows:

- Area of interest (hub/node/place/..) – a terminal, location, port, city centre, etc. where a cargo physical activity like transhipment or storage can take place. It can also be a border crossing, facilities in the infrastructure like locks or bridges, or a city centre with certain access restrictions. A node will have a name in the context of a particular transport activity, e.g. a

Port of Call for a vessel, a Port of Loading to indicate where a container is (to be) loaded onto a vessel, or a Place of Acceptance to indicate the origin of the cargo, i.e. the place of acceptance is known as the place where a carrier or forwarder takes over responsibility of the cargo from a shipper.

- Cargo – the goods that are transported from origin to destination. Cargo may be bulk or containerized. Cargo is defined by generic description of products that are transported such as fruit or textile. Cargo can be packed, repacked, consolidated, reconsolidated etc. Trailers may be a transport means but can also be cargo itself, on a vessel for example.
- Transport means – these are the vehicles that transport the cargo, such as tricks, vessels, trains, airplanes, barges etc. Each transport means has a specific transport mode; some might have more than one mode.
- Business services - the way a service provider makes its capabilities known to potential customers. A business service refers to a business activity like 'transport' and business services result in business transactions, for instance a shipper will have several business transactions with a forwarder over time. Data that is shared may form the basis of documents that are required for regulatory or legal reasons.
- Products – the actual objects that change ownership between a seller to a buyer. For transportation purposes, products are packaged as general cargo that can be loaded into containers. Note that 'product' is a construct from the perspective of its use. A transport means like a 'truck' or equipment like 'container' are also products.
- Customs item – the classification of products or cargo for customs purposes according to the Harmonised Systems codes (HS). One can basically have three classifications: export, import, and incoming/transit. Export and import relate to products and incoming/transit to cargo.
- Equipment – any re-usable asset for facilitating transport and handling of cargo.
- Person – any individual that is a crew on board of a transport means. A crew, a person can have a role, e.g. a truck driver or captain. These roles also relate to qualifications.
- Events represent actions, milestones, transactions, or any other real-world activity. These will typically involve one or more interactions or associations between location, transport means, and cargo. For instance, a vessel is expected to arrive at a given time in a port. Similarly, for a container: which will be loaded on and discharged from a vessel in ports.

This transport model needs further refinement:

- Area of interest – these are refined according to their logistic function, e.g. port, storage, transhipment, production, or infrastructure function, e.g. a road, a city center, a river, railway track.
- Cargo –these are refined into general cargo (pallets, packages, etc.), bulk cargo (oil, grain, etc.), and containerized cargo. Note that also a transport means can be cargo, e.g. trucks on a ferry (ro-ro).
- Transport means – these are refined into vessel, truck, barge, airplane, and train for each of the transport modes. Further refinements can be made such as for vessels into deep sea and shortsea vessels, and ferries.
- Business activities – these are refined into transport, transhipment, storage, administrative services, etc. Additionally, these include the process aspects like business transactions (booking, ordering).
- Equipment - can be refined in for instance container, ULD, trailer, and wagon. These refinements also are of a type like the types of containers.

These refinements imply additional business constraints. For example, containers can only be transported by vessels equipped for container transport.

5.2 Base structure of the semantic model – the upper ontology

A semantic model, represented by SHACL/OWL, must reflect these logistics concepts. There are different ways to model them and there are already many best practices from standardization that can be applied in constructing the semantic model. These will be discussed in this section.

5.2.1 Design choices for constructing the model

There are different approaches for representing the required logistics concepts in an ontology. It is about representation of concepts and their associations, where the following choices exist:

- **Logistics concepts** – all concepts are represented explicitly or via spacialisation (so-called super – and subtypes, where the subtypes also have data properties of the supertypes). For instance, a truck, container, and barge are modelled as concepts or truck and barge are a spacialisation of 'transport means' and container is a spacialisation of 'equipment'.
- **Associations** – all associations between concepts are modelled explicitly or as rules formulating constraints. For instance, a container can have an association with a barge and a truck ('it can be transported by barge and truck), giving at least two associations already when modelled explicit and one rule considering the rule approach.

The preferred option is to choose (1) spacialisation since that is the easiest extendible with new logistics (or other) concepts and (2) the rules-based approach for modelling associations since that makes the model better understandable, extendible for supporting new rules, and more concise. These choices allow the construction of a so-called upper ontology with generic logistics and data sharing concepts that can be refinement for service development. The upper ontology is a core that can gradually be extended for supporting new data sharing applications.

Combining both choices will for instance result in a 'cargo' rule stating that an association between equipment (with its subtypes) and transport means (and all its subtypes) can exist with potential limitations of the type of equipment and restrictions given by a transport means (e.g. maximum load).

5.2.2 The upper ontology

These modelling choices are transformed into common to supply and logistics as a basis for refinement. Since all communities will have different (implementation guides of) standards with different structures, the multimodal ontology provides an alignment framework consisting of 'Digital Twin' and 'event':

- Digital Twin is a taxonomy of real-world objects (container, truck, barge, goods, livestock, etc.). The taxonomy is constructed by spacialisation, i.e. creating subtypes for a supertype like a truck is a subtype of a transport means. Goods distinguish themselves from other types of Digital Twin by their lifetime: they only exist during logistics activities.
- Event is the association between at least two Digital Twins in time and space (past, present, and future, where future is 'expected', 'planned/estimated', and 'required' and present is 'actual'). 'Event' reflects the state of the physical world ('where are my goods', 'container track', etc.) controlled by data sharing ('my ETA is …', 'this container must be loaded on that vessel', etc.). Only those associations, i.e. events, can exist as specified by rules.

These two concepts are crucial, since they enable to represent all types of associations between

any physical objects, in time and place. 'Event' enables the representation of a network of physical objects, e.g. a container loaded on a truck and transported between two locations, its goods, and the details of those goods like finished products. 'Event' represents a truck in its freight transport and its components with their physical health from a safety and maintenance perspective.

Any constraints between subtypes of Digital Twins are formulated on 'event' level and specified in SHACL. By doing so, the model is easy to understand, extendible, flexible, and explainable. New physical objects can be included as subtypes of Digital Twins with their event constraints as rules representing associations to other physical objects.

Any value constraints on data like those of weights or the format for representing a date and time are formulated by SHACL. Code lists are modelled separately.

The semantic model is modularized to increase its maintainability and re-use of existing ontologies. An infrastructure like that of the road – or rail infrastructure can for instance also be represented as 'Digital Twin' but FEDeRATED choose to make this a separate module and align with existing infrastructure ontologies developed and maintained by others. The same is for 'person' (legal or natural) that is independent of supply and logistics.

Figure 5-2 visualizes the result of decomposition, including 'legal/natural person' and relevant financial and compliance concepts.

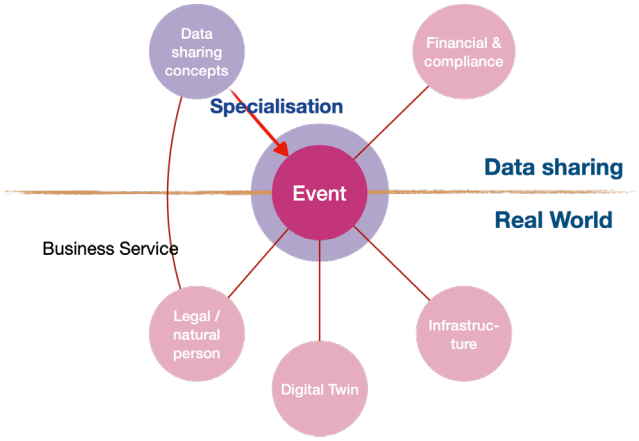


Figure 5-2: the multimodal ontology with details of business data sharing concepts

Figure 5-2 introduces data sharing concepts as a separate model. This is about the interactions of a 'service' and their sequencing for business activities. An interaction or a business document is for instance represented by a subtype of event associating Digital Twins, locations, and organisations for a business activity. The data sharing concepts are also an ontology. The 'Multimodal supply chain visibility' document provides an example of these interactions for a service. The data sharing concepts will be described in more detail in the next chapter.

5.2.3 Baseline standards for semantics

The following baseline standards are identified for constructing the multimodal ontology:

- The United Nations Trade Data Elements Directory (UNTDED) is a baseline vocabulary;
- The UN CEFACT Core Components Library provides a set of (composite) data types with formats;
- UN ECE Recommendations for all types of code values like packaging.
- ISO standards like the ones for country codes and date/time formats

- International adopted encoding schemes like those for container size and type, vessel types, etc.

Standards like UN CEFACT MMT (Multi Modal Transport model), WCO Data Model, GS1, EU Customs Data Model (EU-CDM), IATA ONE Record, Sea Traffic Management (STM), RIS (River Information Services), Port Collaborative Decision Making (PortCDM), and many others build upon these baseline standards. Others like TAF TSI (rail) and Datex II (road traffic management) do not share these baseline standards (or only a very small subset).

Any data formats and their constraints are the basis for encoding schemes for data sharing and validation of this data.

5.2.4 Best practices

UN CEFACT and other standardization bodies have best practices that are adopted by the multimodal ontology and services, like:

- Measure unit specifiers – each measure must have a unit specifier of the SI system.
- Date/time format – although much software can automatically process a date/time format, it must be mentioned during data sharing. ISO date/time formats are applicable.
- Time zone – a date/time is linked to a time zone that must be mentioned. Time zones start at GMT (Greenwich Mean Time).
- Currencies – each currency must be mentioned, including an exchange rate with its date when a common currency is applied.
- Geo-coding – international coordinate system for encoding a location. Geo-coding can be attached to other location code values like the UN Location Codes and port numbering schemes for a port. Since there are different ways for geo-coding, the applicable way must be mentioned during data sharing.

Whereas the multimodal ontology covers all potential values, selection of these values must be done at service development. This selection is also required for all code values like those of location codes. This reduces the number of options for service customization, thus contributing to interoperability.

5.3 Examples of use of the semantic model

The semantic model can be viewed from different perspectives, based on evaluating the event association between various concepts. Examples are:

- Shipment data set – any data set (i.e. links) shared between a customer and service provider providing details of cargo to be transported from one location to another at the same time.
- Document data sets – links to data that is normally contained by a particular document relevant to a shipment, e.g. a business document like a CMR or a document issued by an authority like a Certificate of Origin. This data set may include links to other data sets like cargo and transport means.
- Itinerary data set – a data set combining operations on cargo and a transport means at locations. An itinerary has links to cargo data, transport means, and nodes; It may have a unique identification stored by the event link between a transport means and locations.
- Route data set – any physical route of a transport means during an itinerary. A route links to a physical infrastructure, for instance by means of physical coordinates or identification of a road.

- Reporting data sets – reporting data sets like eFTI and eMSW are shared as a set of links to one or more of the other data sets, e.g. a link to the cargo loaded at a node and (to be) discharged at another node and the crew of a means of transport.
- Train data set – an association between a locomotive as a means of transport and wagons that are a spacialisation of 'equipment'. The association has a unique identification during a particular path or composition of wagons, the train number.

One can also consider including nodes or hubs as specific data sets, where enterprises operating these nodes provide services, e.g. a stevedore providing transhipment services at a terminal. Other types of nodes might contain storage facilities (e.g. warehouse or distribution centre).

5.4 Events in the reference model

Events are an important concept. Collectively, these events form the lifetime of associations and reflect physical operations with their planning. For example, an association between a container and a vessel is created after loading the container on a vessel and ends after its discharge. These associations start and end with milestones, which are physical action. High level milestones are:

Generic milestone	Description
Arrive	A transport means arrives at an area of interest or location at a time. In case this area is a terminal, the milestone could be given as 'gate in'. In case a transport means arrives in a country, it will be called 'border crossing' and the location is called 'border crossing place'.
Depart	A transport means departs at a location at a time. In case this area is a terminal, the milestone could be given as 'gate out'.
Load	Cargo is loaded into a transport means or equipment like a container. Loading indicates that the owner/operator of the transport means accepts responsibility for the cargo.
Unload/ Discharge	Cargo is discharged or unloaded from a transport means or taken from an equipment, like a container. The receiving party such as a freight forwarder will now take responsibility for the goods.

Although events are physical, they will be registered by and shared between IT systems with interactions. On many occasions, registration of events is like the generic once given in the previous table e.g., a container that has been loaded on a vessel, interactions may combine events, e.g. like a load list with all containers that are loaded on a vessel. These types of combinations of events must be supported by the data sharing ontology of the model.

The associations between the main transport concepts define business constraints. For instance, a container cannot be at different places at the same time. Another rule would be that when transport means after loading of the cargo leaves a location it implies that it has departed and that the association (from a data perspective) between a location and cargo has ended. This also implies a handover of responsibility to a carrier after loading cargo. These types of business logic rules will be specified in a detailed architecture.

Events create a trace of the life of a transport concept, i.e. cargo, location, business service or transport means. For instance, the sequence of events between a transport means and arrivals and departures to and from locations, show the itinerary of that transport means. This itinerary can be

according to a timetable like a voyage scheme or a flight schedule provided by a third party service provider. Similarly tracking and tracing a container can be represented by the locations and the transport means between any of the locations on its route.

5.5 Dates and times in data sharing

Data sharing is about synchronizing logistics activities, reduction of waiting times, etc. Therefore, four types of 'dates' for event associations between any two concepts of the Semantic Model are specified:

- Expected – the time provided by a customer. It is provided by an order. It may be time windows for earliest – and latest pickup and delivery.
- Planned – the time at which a service provider will execute an order. In case of a voyage scheme, it is the time of call of a vessel in the Port of Loading. This can also be a time window. The planned time can be updated with new values, the so-called Estimated Times (e.g. Estimated Time of Arrival or ETA).
- Requested – the time at which a service provider is able to perform a logistics activity and requests a customer to be available. This can also be a time window.
- Actual – the time at which a relevant milestone took place, e.g. actual loading. This is provided by for instance container status data.

Additionally, event association between Digital Twin and place have different properties depending on the type of business activity:

- Transport has typically eight properties: two locations with their dates and times. Transport is always between at least two locations.
- In case of transhipment -, storage -, and groupage activities, there are seven values: one location with six values for time split into arrival/departure and load/unload. These activities always take place at a single location.

In addition to these activities, there is a multitude of milestones referring to physical activities that must be performed with their associated times (like security inspection, quarantine, and dangerous goods inspection). These are all specific instructions of authorities with a requested time (window) and location at which such an activity must take place.

The requested time can be used by a node/hub operator or infrastructure manager to optimize the handling and flow of transport means.

Process synchronization requires that these values of dates and times of adjacent legs must be identical or overlapping. The actual dates and times must be identical for adjacent legs, e.g. the actual arrival time of a transport means at a hub provided by a carrier should be identical to that provided by a hub/terminal operator.

Typically, each business activity has its own characteristics, associated times from the list mentioned above, and potentially derived values like a turn-around time at a terminal. These will also be developed and harmonized by the Living Labs.

The associations between the various subtypes of Digital Twin have two concepts of 'time': the time at which the association is created and at which it is expired (e.g. loading - and discharge time respectively). Time might not always be given but is relevant.

Additionally, there can be a split of the goods (pallets, packages) of a shipper over different equipment of a service provider. If for instance goods are stuffed in two containers, there are two

instances of those goods created that associate to the original goods instance of the shipper, and the two created instances each associate to their container. This is called the split of a goods item in UN CEFACT standards. It is applicable to goods and bulk cargo for their association to other goods (repacking), equipment, and transport means.

5.6 Cargo perspective

As stated before, the following types of cargo are distinguished:

- Containerized cargo – cargo transported in containers. These are normally expressed as Twenty feet Equivalent Units (TEU) for sea and Uniform Load Devices (ULDs) for air transport. Thus, a 40-foot container is 2 TEU. ULDs come in many shapes and sizes to fit an airplane contour.
- Containers can transported by vessels, trucks with trailers, that can be transported by vessels (ro-ro or roll-on roll-off), and trains consisting of railway wagons (that can also carry trailers with containers).
- General cargo – this can be goods that are packaged with some type of disposable (or maybe re-usable) packaging material. These can be pallets with boxes, drums, postal bags, or anything else. There is a UN Recommendation on Package types. As the example shows, packages can contain other packages, eventual leading to the (commercial) products that are transported between a shipper and consignee.
- Bulk cargo – this is any type of cargo that can only be expressed in weight and volume. A distinction between dry – and liquid bulk is made, e.g. oil and chemical are liquid bulk, sand, coal, and grain are dry bulk. Bulk cargo can also be transported in containers.

Depending on its way of packaging, agricultural products like fruit can also be transported as dry bulk in containers or directly in vessels. It all depends on logistics operations at a shipper and agreements with a consignee for transporting products. This must be reflected by customs declarations.

At each handling location, but also during movement, the cargo of consignments and shipments can change. Therefore, it is relevant for customs to distinguish individual transport legs, detect any delays, and have additional information of the parties involved. Basically, two types of operations can take place on cargo:

- Convergent operation – cargo is packed together with other cargo into larger units. The identifications of the packed units is lost; the identification of the larger unit is used in data sharing. This operation is also known as ‘stuffing’, ‘packing’, and ‘loading’. It can be on various levels, like stuffing pallets in a container, but also on loading trailers on a train or containers in a vessel.
- Divergent operation – the individual units are taken from the larger unit. This operation is known as ‘stripping’ or ‘degrouping’ (containers), ‘discharging’ (transport means like vessels), and ‘unpacking’ (boxes from pallets). The identifications of the smaller cargo units become available.

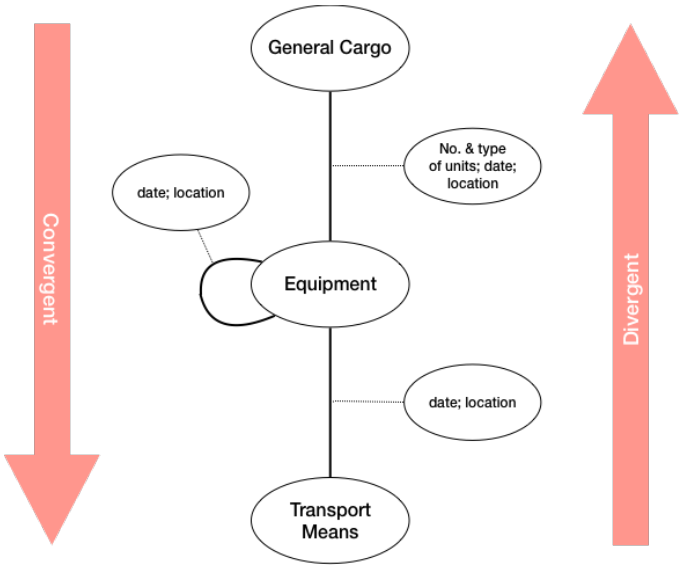


Figure 5-3: the Particular operations are reflected by their data perspective.

Figure 5-3 shows that general cargo, e.g. pallets with boxes, can be grouped into containers. One cargo item of one consignment can be split into two or more containers (equipment). Therefore, an association between general cargo and equipment contains the number and type of units that is stuffed in a container.

Each consignment has a unique identification between a customer and its service provider; each cargo item or package will have a unique serial number within that consignment. In practice, cargo items themselves do not necessarily have unique identifications, e.g. boxes don't have unique numbers. Cargo items are uniquely identified by their type and marks and numbers, which is anything written on these cargo items. Grouping of cargo items in a container can be retrieved from a container stuffing list, linking to a consignment.

In most cases, a stuffing list is not present or there is only an indication that particular consignment(s) are stored in containers. Thus, only an association between those containers and the consignment(s) can be constructed.

The association between two pieces of equipment, e.g. a container on a trailer or a trailer on a wagon, is based on their unique identification. The association is constructed at a location and time (convergent) and deleted at another location and time (divergent).

5.7 Transport means perspective - itineraries

A transport means is an asset that moves from one location to another. The movement can have different names e.g., a trip for a truck, voyage for a vessel, or flight for an airplane. These are called itineraries. There are two types of itineraries, namely:

- Itinerary-to-order – the itinerary is established by planning software assigning orders to a transport means, resulting in an itinerary of that transport means. This type of itinerary is completed when all cargo has been transported. It might never end, if the planning is updated during its execution, meaning for instance a truck just drives (indefinitely) and transports cargo. This would be relevant to especially autonomous transport means. A similar situation can be foreseen with barges, where any free capacity due to discharging cargo can be booked.
- Order-to-itinerary. An itinerary like a voyage scheme, flight or timetable of a train is published by a service provider like a shipping line, airline or railway undertaking to the

ledger. During booking and ordering, the itinerary is available to customers. An order is therefore linked to an itinerary.
This itinerary has a lifetime spanning either geographical or in time. For instance, a voyage ends at its destination. A vessel can have a new voyage at that destination. A flight or timetable, which is scheduled periodically like daily or weekly, will end at a certain time. Whether or not it is replaced by another one depends on the agreements of the service provider with one or more infrastructure managers and/or hubs, like slot allocation at an airport or path allocation for trains.
Itineraries might change during execution, due to unforeseen circumstances or for commercial reasons. These changes might impact authorities. For instance, if a vessel had Rotterdam as first port of call in the EU, but changes its voyage to Antwerp as first port of call, Belgium customs should have the Entry Summary Declaration data (ENS) of all containers discharged in EU ports.

A milk-run is a special type of itinerary to order: a transport means has a specific route where known shippers can indicate whether or not they have cargo to be picked up.

In many cases, forwarders may sequentially combine these itineraries, Forwarders may construct multi-modal chains with multiple hub transhipments based on timetables and using back-to-back business services of multiple transport means operators.

5.8 Details of the semantic model

The 'Digital Twin' concept is the core of the model: a data representation of any real-world physical object. A physical object can be viewed from different contexts, e.g. its user (in logistics), - owner, - producer, - maintenance operator, and supervising authorities. Each context requires potential other data, e.g. maintenance requires data of use (like mileage and cargo load for a truck) and an OEM may need to provide details of its components (like its battery for an electrical truck). All contexts add and share properties. Therefore, 'product', which refers to production, ownership, and use, is not modelled separate, but will add additional data properties to what is currently modelled.

Since physical objects can also be transported without using packaging or equipment, e.g. livestock and bulk (raw) material, these are modelled separately. At a later stage, the taxonomy of Digital Twins may require revision when representing all types physical objects as Digital Twin.

Details of the model (like its Turtle representation) can be found at the developer portal of federatedplatforms.eu. The following terminology is used:

Semantic Element	Definition	Identification
Customs Item	An item of cargo that is imported, exported, transited or (temporarily) stored under customs regime in a customs zone. A customs item	HS (Harmonized Systems) code
Parent: **Event**	Associations between Digital Twins and relevant locations	
Transaction event	All state data that can be shared between customer and service provider in the context of a service of a business activity.	Any applicable identification in addition to a UUID, like consignment identifier,

Semantic Element	Definition	Identification
	References to other concepts (Digital Twins, locations and their roles, and legal/natural persons and their roles) with user defined identifiers (e.g. identification of equipment, see further)	shipment reference number, Movement Reference Number. Since transactions can be split in long-term contracts and orders, there can also be different identifications for transaction events.
(update) event	Any event shared in the context of a transaction event and reflecting changes from physical operation(s).	UUID with reference to a transaction event and/or via sender/recipient combination and UUIDs of Digital Twins as specified by state transitions.
Parent: **Business Service and - Activity**	All relevant data shared by enterprises for commercial reasons based on common logistics activities	
Business Activity	Any physical or supporting activity that can be outsourced by a customer to a service provider	
Business Service	Any service that is provided to a customer by a logistics service provider for a given business activity. A business service is an instance of a business activity. Business services are required for discoverability. The Service Registry has a module functioning as business service catalogue.	Defined at more detailed level
Business Transaction	All data shared by interactions to synchronise states of a service for a business activity.	
Parent: **Digital Twin**	Any real-world physical object	
Bulk product	Unpackaged raw material (solid, liquid, chemicals, etc.)	Identified by quantity or volume (and quality) in the context of a business transaction
Livestock	Living animals	Mostly identified by their quantity and animal classification; sometimes by giving identifications (like earmarks) of individual animals.

Goods	That which is handled by logistics activities, like transport, storage, and transhipment and packaged accordingly	Identifying properties are in the context of a business transaction: no. and type of packages; or each package has a unique identification (e.g. SSCN of GS1) or a way bill reference besides a UUID.
Equipment	Re-usable equipment to facilitate transport of cargo, for example a wagon or a container	Defined at more detailed level
Transport Means	An asset that can move on its own power and that can carry cargo or equipment.	Defined at more detailed level
Location	Any physical location relevant for logistics	Defined at more detailed level
Person	A legal – (organisation) or natural person	A person can have a role in a transaction and/or physical operation, like 'shipper', 'forwarder', 'crew', 'driver'.

Parent: **Equipment**		
Container	Transport containers serve to containerize products. They have standardized dimensions and can be loaded and unloaded and stacked	Container number
ULD	Unit Load Devices are light weight containers for air transport and facilitate loading cargo into aircraft	ULD type and ID code
Railway wagon	Unpowered railway vehicles that are used for the transportation of cargo	Wagon number
Trailer	Unpowered road vehicles that are used for the transportation of cargo	Trailer license plate
Swap body	Types of standard freight containers for road and rail transport	Identifier

Parent: **Transport means**		
Vessel (sea)	Any transport means used for transporting cargo by water.	Vessel name or Radio Call Sign, or AIS (Automatic Identification System)

	• These may be further classified as Deepsea vessels for ocean transport, • Feeders (short range via sea) • Any of the two above for a particular cargo type (e.g. container – or bulk vessels) • Ferries for transport of other transport means like trucks and/or trailers.	
Barge	Vessel used for transporting cargo on inland waterways	Vessel name or AIS (Automatic Identification System)
Truck	Any transport means for road transport.	License plate (issued by national authority)
Locomotive	Any traction for a train composed of one or more wagons	Loc identifier (unique per owner, might be by a transponder).
Airplane	Transport means by air. This may be a dedicated freighter aircraft or belly space on a passenger aircraft.	Flight number or aircraft registration

Parent: **Location or area of interest**		
Logistical function	Any location where a logistical activity is performed (see also business activities)	Defined at more detailed level
Infrastructural function	Any part of an infrastructure used for performing logistical activities	Defined at more detailed level

Parent: **Logistical function (role)**		
Hub	A place cargo is exchanged between vehicles or/and between transport modes.	Location Code
Port	Location where vessels pick or drop of freight. Freight will be delivered or picked up by other transport modes. This may also be a hub where freight is transited from one airplane to the next. There may also be temporary storage facilities.	Seaport code or Location code
Airport	Location where airplanes pick or drop of freight. Freight will be delivered or picked up by other	Airport codes or Location code

	transport modes. This may also be a hub where freight is transited from one airplane to the next. There may also be temporary storage facilities.	
Terminal	A freight terminal for different modes is a processing node for freight	Location code
Transhipment	An intermediate destination for transiting cargo to another destination.	Location code
Parent: **Infrastructural function**		
City	A generic geographical location that may have ports, seaports, terminals etc.	City codes or Location code
Trajectory	Any part of an infrastructure identified by its infrastructure manager	Road number, stretch number, river name, …
Area	Any area relevant to logistics operations	City centre, region, country code
Parent: **Business activities (see previous list; this is to be extended)**		
Transport	An activity to move cargo from one location to another.	Company name and related identifications Company name and related identifications
Transhipment	An activity to tranship cargo from one transport means to another. It consists of two sub-activities: load/unload.	Company name and related identifications
Storage	An activity to (temporary) store goods or their content (unpacking goods). This may include special goods storage such as cool storage, animal hotels, dangerous goods storage.	Company name and related identifications
Groupage	An activity to group goods into equipment. Degroupage is the inverse activity. Other naming: stuffing and stripping of containers	Company name and related identifications

	or (un)packing of goods.	

The list has several physical objects. Combinations of these objects into larger ones are based on associations between those objects. For instance, a train is the composition of a locomotive and several wagons with a unique train number. The train number can be applicable to for instance that composition or a path on the railway infrastructure.

Business services are stored in a catalogue, which is a module of the Service Registry, and can have a unique identification.

Although not mentioned, wholesale, retail, manufacturing, and production are business activities triggering logistics operations, e.g. warehousing of products with Vendor Managed Inventory. These business activities have transaction events with subtypes of 'product'. A product catalogue is now defined as the instances of subtypes of products for commercial (buy/sell) and production business activities, where each product has a unique identifier in the product catalogue. The business service is mostly not part of a product catalogue, but implicit by the fact that products and their identifications are given.

These business activity related products are not the actual physical instances. Those are the ones that are sold or manufactured via similar services like those of logistics business services (e.g. quotation and ordering). They refer to the identification in the product catalogue and can have an identification themselves, e.g. a VIN (Vehicle Identification Number). In their turn, these products with their unique identifications can be transported, e.g. car transport, and can become assets of equipment for logistics purposes, e.g. a truck with its unique identification registered for logistics by a license plate.

Thus, the semantic model can also be specialized for services of these buy/sell and manufacturing/production business activities.

6 Service development and - customization

This section is about applying the semantic model in data sharing from two perspectives, namely service development and – customization. Service development comprises a high level data structure of 'service' for a business activity and how to specify processing rules for 'service'. Service customization is about constructing a 'profile' to support services.

First, different approaches to service development are given that have impact on service development and customization. Based on these approaches, service development and – customization are detailed.

6.1 Service development approaches

Currently, organisations must deal with a variety of (open or defacto) standards and their implementation guides. Dominant players impose their interfaces and technology to their suppliers or customers, communities agree on standards and their guides. The result is that individual organisations construct a single interface that supports standards and guides of their customers and suppliers/service providers, thus being able to develop (relatively) stable IT systems.

This issue remains the same when utilizing semantic technology. Therefore, the following stakeholders can take the role of Service Developer (section 4.5):

- **Bilateral**. Two collaborating organisations develop their services. They act as Service Developer. Customization is not required since service are tailored to their data requirements and business processes.
- **Dominant player**. A supervising body or dominant player acts as Service Developer. They themselves do not require a profile, but others that must implement the services may need to have one. In case a regulator acts as service developer on behalf of supervising bodies, it can be compared to a community or industry association.
- **Community/Industry Association**. A community like a port community, regulator, or an industry association acts as Service Developer on behalf of its members/supervising bodies. These types of services may contain optional elements to address data requirements of part of that community, which requires service customization by individual members.
- **Evolutionary**. Architects (business and IT architects) act as Service Developers and the architecture comes with services that can be implemented by organisations according to their profile of these services. An initial set of services can be provided addressing all relevant business activities for logistics. This set is based on best practices and existing standards. The set can gradually evolve by creating subtypes of business activities with their services. For instance, a transport activity can have subtypes of container -, commodity -, and parcel transport, and sea- and air transport. As an example, a multimodal visibility service is available. Each business activity subtype encompasses different data sets (see section 6.3).

The evolutionary approach reflects the current way of implementation of different guides of standards by for instance forwarders and Logistics Service Providers (LSPs), where they specify a single interface of their IT systems with some type of gateway implementing those standards and guides. It allows development of IT systems without them being affected by changes in standards or requirements of new standards.

Especially communities like a port community will have to take an evolutionary approach, since their members have to interface with service developed by modality and/or cargo centric industry associations.

Bilateral or dominant player services do not necessarily scale, meaning that they are only applicable for the developer(s) or imposed by a dominant player to its business relations. If a service is applicable for a wider community and application area, it will contain mandatory and optional features (constraints, properties, value constraints, and concepts). Each organisation can select those optional features that it is able to support. It also must indicate whether it is still optional or mandatory for them.

An example is a 'cargo rule' that specifies that at least one of a list of subtypes of equipment or goods must be present. Subtypes of equipment are for instance 'container', 'ULD', 'trailer', and 'wagon'. An organisation can select which to support. In fact, a 'profile' with the options supported by an organisation adds constraints to the model by omitting options.

Since a service has a geographical application area, that application area can also be reduced.

Whenever an evolutionary approach is taken, a profile not only supports the business of an organisation, but also services and standards of others. Of course, a community can also develop services and implement these with existing messaging standards and/or technology like Open API Specifications (OAS).

6.2 An evolutionary approach

The evolutionary approach is taken as guiding for service customization of business activities, although the other approaches are also supported.[6] The evolutionary approach is by specifying data shared with services in the context of a business activity outsourced by a customer to a service provider. Outsourcing has an identifiable end – and start state. All data shared in the context of a business activity is represented by its end state. The end represents the maximal data set that can be shared. Data sharing is by continuously updating each other's state.

An **organisational profile** is of a business activity as required or performed by an organisation, including the (sub)set of services and its sequencing. A business activity data set and the ability to create subtypes of business activities is expected to contribute to 'ease-of-use' of the solution, since an organisation will select the appropriate business activity subtype(s) to create a profile for its operation. This will also fit with organisations with specific business units for these business activity subtypes, e.g. a parcel delivery unit and an air freight unit.

Services represent processing aspects by modelling all steps from the start to the end state with data sharing. Each interaction must contain sufficient data for the next step in planning and execution of a business activity. This is represented by the data structure of these interactions or events.

The processing aspects of services, i.e. their interaction sequencing or choreography, represent best practices aligned with standards in supply and logistics and can differ per modality or way of operation. For instance, using timetables for flights or itineraries for vessels requires interaction sequencing different from incidental use of a carrier. Preferably, these processing aspects are being standardized by their users (as (sequences of) interaction patterns). If not, they can be developed based on existing best practices and be offered to organisations for implementation.

The evolutionary approach specifies business activity data and services that have the most degrees of freedom and thus require a complex profile. **Modular services** for a business activity provide the

[6] Note that authorities will always take the dominant player or community/industry association approach.

most flexibility in configuring a profile, for instance supporting payment before or after delivery. A proposal for modularization is given in this section. An example of a multimodal visibility service is given in a separate document.

The process aspect of a service is modelled by Business Process Modelling 2.0 (BPMn2.0) choreographies. The data aspects of a service are specified by SHACL constraints to the semantic model.

This part of the architecture provides templates for specifying all relevant aspects of state transitions; these templates can be supported by a module of the Service Registry supporting service development. It also specifies the data structures for data sharing and states with respect to business activities.

6.3 Business activity data

In logistics, the following business activities are distinguished that will all have their data sets:

- *Logistics (core) activities*: e.g. transport, transhipment/cross-docking, (temporary) storage;
- *Value added activities*: e.g. (re-)packing/stuffing, unpacking/stripping, ironing (of textile), consignment grouping, vendor managed inventory;
- *Supporting physical activities*: e.g. vessel waste management, container cleaning.

A business activity defines the data that is shared, including its start and end state. Data sharing between any two organisations is by synchronizing business activity data. For instance, a load event of goods onto a truck provides an update for transport.

The business activity data set is the total data set accessible to any customer and service provider involved in a commercial relation and is the basis for supervision. Part of this data set is stored by a customer, another part by the service provider (pull principle keeping the data at the source). Mainly a customer will have cargo details and a service provider, cost and delivery conditions and planning and progress of an activity.

As such, the business activity data set specifies all data that is accessible to a customer and service provider. Thus, it defines the access control policy of both stakeholders involved in a business transaction.

The business activity data set is visualized by the following figure.

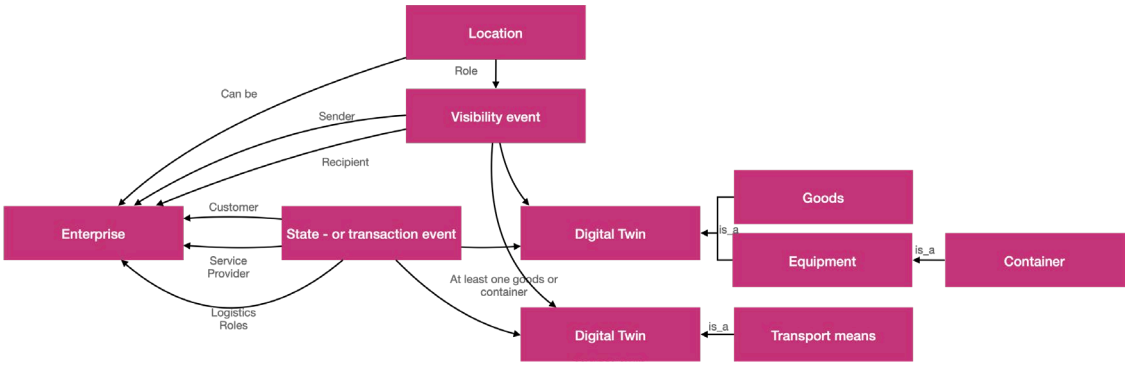


Figure 6-1: business activity data structure

Roles of locations and logistics roles are represented by associations, for instance ‘shipper’, ‘forwarder’, place of acceptance’, ‘border crossing place’, and ‘place of discharge’.

Figure 6-1 shows that a business activity consists of a transaction event and multiple visibility events,

each of them providing an update of the progress in planning and delivery of a transport service, namely expected, planned/estimated, and actual times of load/unload or arrive/depart. The transaction event represents general information also known as 'header data'. This is for instance about delivery- and payment conditions and cost components for service delivery. The visibility events represent the association between two Digital Twins (load/unload, stuf/strip, pack/unpack) or a Digital Twin and a location (arrive/depart). These visibility events are part of a business activity outsourced by a customer to a service provider, if they have the proper sender/recipient as specified by a service.

The number of visibility events differs per business activity. A **transport activity** requires at least two visibility events, namely where and when transport starts and ends. Additional events can be given to distinguish for instance a place of delivery from a port of discharge. These additional visibility events can also be used by a service provider to inform a customer on the legs of a logistics chain.

A **transhipment activity** requires two transport means with their visibility events for a transhipment location. One of these transport means is arriving for unloading, the other for loading and departure. In this case, the association between a visibility – and transaction event does not exist.

Stuffing and stripping (or (re-)packing) **activities** are like those of transhipment but are between two Digital Twins that represent 'cargo'.

6.4 General concepts for services

Choreographies only specify the behaviour of any two collaborating actors and not their internal processing and decisions.

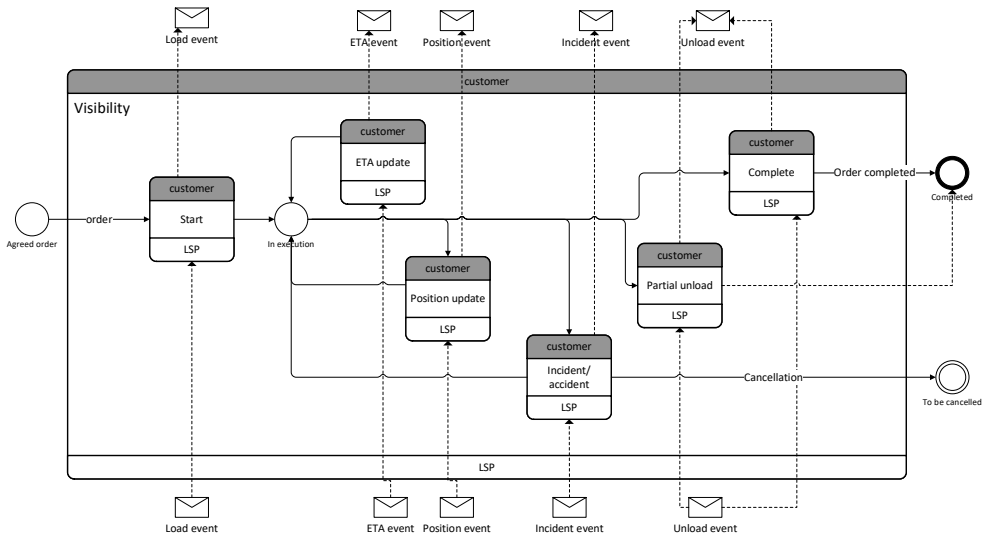


Figure 6-2: a choreography example: multimodal supply chain visibility

The concepts of choreographies are the basis for service development. The Service Registry will implement these concepts thus enabling service development by various stakeholders.

A service always refers to a business activity e.g., an ordering - and visibility service are for transport. The first ones e.g., ordering, are called transaction events; the other ones are just events[7]. Where

[7] The use of the term 'event' might seem a bit dubious since event is also an association concept. In the context of interactions, an event may contain additional data of Digital Twins or locations that are referred to by the event association.

transaction events can have references to all types of Digital Twins, physical events, and/or locations, the other events just state a change of one or more properties of a transaction event and its associated concepts (e.g. change of an ETA as compared with the planning or damage of goods).

The following data sharing concepts are defined:

- Service – the combination of interactions with a clear start – and end state in the context of a business activity. A service is modular, in the sense that it can be implemented independent of other services. A service has metadata like name, description, creator, date of creation, application area (like a particular port, country, or EC), and reference to a base service. The base service might be a generic one e.g., a global one, that is specialized for an area like a port. A synonym for service is interaction pattern.
- State – the aggregation of data that has been shared between any two actors. This definition of 'state' is specific to choreography. Other definitions of state reflect for instance that of a complete system like all data that is shared at a moment.
  The state data structure equals the business activity data structure (section 6.3).
- State transition – the update of an existing state or transfer to another state upon an external trigger. Thus, a state transition has an input state, a trigger, and an output state.
- Interaction type – the type of interaction send by one of the actors to the other one, triggering a state transition. This can be a transaction – or physical event.

Service, state, and interaction type are part of the data sharing ontology module. State transition requires attention, as is shown hereafter (section 6.9). They can contain processing fucntionality.

In the example, a Logistics Service Provider (LSP) always sends, and a customer receives an interaction. The example shows four states: agreed order, in execution, completed, and to be cancelled. This latter state is the end state for the example and the start state of another service, a cancellation service.

A service must always be completed, meaning that it will always go from its start to end states. In the example, it always must go from order created to completed or to be cancelled by sharing at least a load – and an unload event. All other events are optional.

Any logistics details of a service are specified by its business activity. This one can also be specialized e.g., for a transport modality and cargo type. Thus, a business activity is not only defined by its state structure underlying services (section 0), but also additional data properties: name, description, creator, date of creation, application area (like a particular port, country, or EC), and reference to a business activity that served as a base. The metadata data of a service may differ from that of its services, but the services must be contained by its related business activity. For example, the application area of a service must be inside or identical to that of its referred business activity.

The example of Figure 6-2 shows interaction types that represent physical events like load and unload. Other interaction types represent for instance bookings and orders. These have physical events and transaction specific events like conditions, prices, etc.

The concept 'state' can also be applied for logistics operations, like the 'state of a container' or the 'state of a port', where a container state reflects its current position and contents and a port state are all transport means (with their cargo), equipment, and goods in a port area. This physical concept of state is like a (SPARQL) query on relevant data.

6.5 Data sharing structures

Events are used for sharing data between a customer and service provider and synchronizing states of a business activity. Events with links to data are used to implement a data pull mechanism. Basically, only links are shared and no data. However, a data user receiving those links must have some ways to selecting links for accessing data of a data holder. This is given by the logistics context, like location codes, container numbers, vessel identifications, flight numbers, and license plates of trucks.

Two types of events are specified, namely 'transaction events' and 'visibility events'. Transaction events will have visibility events and refer to a transaction data set.

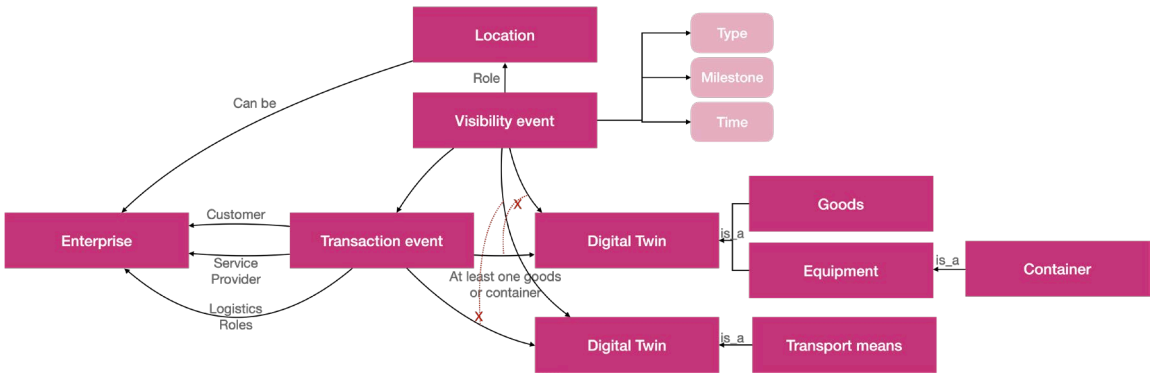


Figure 6-3: Main concepts of transaction events.

Figure 6-3 shows an example of a transaction event (order) generated with the initial version of the Service Registry using the semantic model. It is applied in a FEDeRATED Living Lab with a carrier. Figure 6-3 does not show the visibility events of this transaction event. The example reflects part of the order data (eCMR).

6.5.1 Transaction events

Transaction events reflect sharing part of the business activity data, e.g. booking – and order data. Figure 6-4 (next page) shows the main concepts of the semantic model for transaction events for transport also illustrating that in sharing transaction events:

- a visibility event refers to a transaction event.
- a transport means is either contained by a transaction - or a visibility event.

Also 'cargo' is contained by a transaction - or a visibility event. This is represented in the figure 6-4 example as container equipment and goods. This allows mentioning all details of transport means and cargo at transaction level and include visibility events with links to locations and details of time.

Mult	Message tree	Element name	Type	Base datatype
1..1	Order	Order	Event	
0..n	|-- involvedCargo	involvedCargo	Container	
0..1	|--|-- numberOfUnits	numberOfUnits	integer	integer
0..1	|--|-- transportEquipmentSizeType	transportEquipmentSizeType		
0..1	|--|-- digitalTwinID	digitalTwinID	string	string
0..1	|--|-- goodsDescription	goodsDescription	string	string
0..1	|--|-- grossVolume	grossVolume	integer	integer
0..1	|--|-- netMass	netMass	integer	integer
0..1	|--|-- grossMass	grossMass	integer	integer
0..1	|--|-- digitalTwinType	digitalTwinType	string	string
0..1	|--|-- size	size	Size	
0..1	|--|--|-- width	width	integer	integer
0..1	|--|--|-- length	length	integer	integer
0..1	|--|--|-- height	height	integer	integer
0..1	|-- involvesTimeClassification	involvesTimeClassification	TimeClassification	
0..1	|-- involvesTimestamp	involvesTimestamp	string	string
0..n	|-- involvedActors	involvedActors	Actor	
1..1	|--|-- involvesLegalPerson	involvesLegalPerson	LegalPerson	
0..1	|--|--|-- postalAddress	postalAddress	string	string
0..1	|--|--|-- locatedAtStreetName	locatedAtStreetName	string	string
0..1	|--|--|-- legalPersonID	legalPersonID	string	string
0..1	|--|--|-- postalCode	postalCode	string	string
0..1	|--|--|-- locatedInCity	locatedInCity	string	string
0..1	|--|--|-- locatedInCountry	locatedInCountry	string	string
0..1	|--|--|-- legalPersonName	legalPersonName	string	string
0..1	|--|-- actorRole	actorRole	string	string
0..n	|-- transportMeans	transportMeans	Truck	
0..1	|--|-- hasTransportmeansNationality	hasTransportmeansNationality	string	string
0..1	|--|-- UUID	UUID	string	string
0..1	|--|-- transportMeansMode	transportMeansMode	string	string

Figure 6-4: Example of a transaction event.

In case of data sharing, it is sufficient to share only the identification of a transaction event (its UUID) and a meaningful reference for users (like a consignment – or shipment reference number). Based on the UUID, all other data can be retrieved. However, it is useful to also share identifications and meaningful references to other concepts (like container numbers) to optimize logistics planning and data sharing and have additional querying capabilities.

6.5.2 Visibility events

Visibility events are contained by transaction events.Figure 6-5 shows the visibility structure for starting and ending (milestone) physical relations like ‘load’, ‘unload’, ‘arrive’, and ‘depart’.

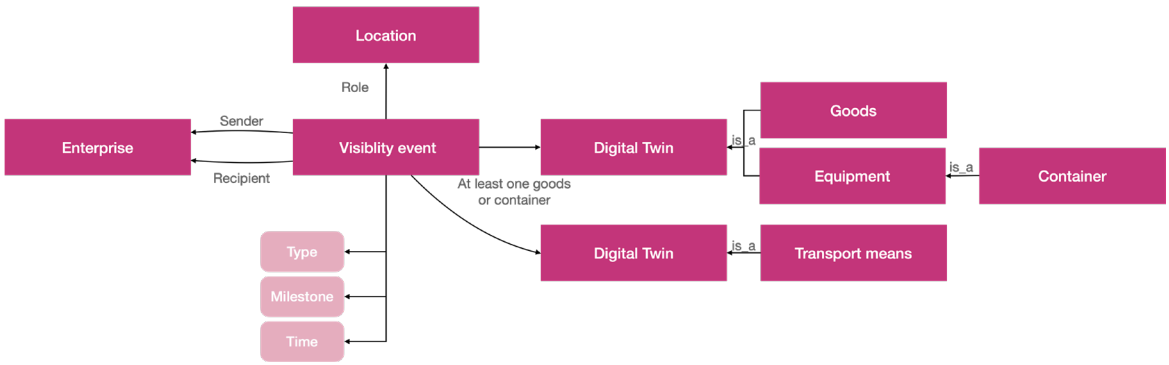


Figure 6-5: Main concepts of visibility (or other supporting) events.

To the event data structure, similar rules for business activities as for transaction events are given. For instance, transhipment requires a transport means for arrival and another for departure, which are, however, mostly represented by two separate events. Stuffing and stripping events don’t require a transport means.

In addition to these events, others like ‘border crossing’ can be given, where a visibility event refers to a transport means and a location of border crossing.

There are also cancellation events required for agility. They support cancellation and replanning of

logistics operations. These cancellation events refer to a transaction event and do not have to contain additional data (except a cancellation reason like 'cargo loss').

The table pictured in figure 6-6 gives an example of a visibility event. It is output of the initial version of the Service Registry with the semantic model.

Mult	Message tree	Element name	Type	Base datatype	Definition	Is attibute?	Path
1..1	Load	Load	PrimitiveEvent				Load
1..1	|-- involvesTimeClassification	involvesTimeClassification	TimeClassification		Every event should have a time classification. The time classifications ma		Load/involvesTimeClassification
1..1	|-- involvesMilestone	involvesMilestone	Milestone		The milestone may be start or end.		Load/involvesMilestone
1..n	|-- involvesTransportMeans	involvesTransportMeans	Vessel		This relation establishes an association between an Event and a Transpor		Load/involvesTransportMeans
0..1	|--|-- externalIdentifier	externalIdentifier	string	string			Load/involvesTransportMeans/externalIdentifier
0..1	|--|-- hasTransportmeansNationality	hasTransportmeansNationality	string	string			Load/involvesTransportMeans/hasTransportmeansNatio
0..1	|--|-- name	name	string	string			Load/involvesTransportMeans/name
1..1	|--|-- UUID	UUID	string	string			Load/involvesTransportMeans/UUID
1..n	|-- involvesActor	involvesActor	Actor		This relation establishes an association between an Event and an Actor.		Load/involvesActor
1..1	|--|-- actorLegalPerson	actorLegalPerson	LegalPerson				Load/involvesActor/actorLegalPerson
1..1	|--|--|-- UUID	UUID	integer	integer	ID of the legal person		Load/involvesActor/actorLegalPerson/UUID
1..1	|--|--|-- legalPersonName	legalPersonName	string	string	Name of the legal person		Load/involvesActor/actorLegalPerson/legalPersonName
1..1	|--|-- actorRole	actorRole	LogisticsRoles				Load/involvesActor/actorRole
0..1	|-- UUID	UUID	string	string	Every event has a UUID to be distinguishable when stored in a database.		Load/UUID
1..n	|-- involvesCargo	involvesCargo	Container		This relation establishes an association between an Event and a Cargo Di		Load/involvesCargo
0..1	|--|-- containerType	containerType					Load/involvesCargo/containerType
0..1	|--|-- externalIdentifier	externalIdentifier	string	string			Load/involvesCargo/externalIdentifier
1..1	|--|-- UUID	UUID	string	string			Load/involvesCargo/UUID

Figure 6-6: example of a visibility event (load).

6.6 Identifications and data sharing

Applying the pull principle, means that minimal data is shared and both stakeholders have access to additional data. The minimal data consists of the identifications of concepts of the business transaction data set. These are the globally unique software generated identifications, the Universal Unique Identifiers (UUIDs), and real-world identifications, i.e. those that are applied in logistics operations. Container numbers, license plates, and IMO codes are examples of these real-world identifications.

Goods are identified by their packaging type, the number of packages, and any description on the packages (marks and numbers). Each combination of packaging type and number of packages requires a UUID that is associated to a transaction event with its own UUID. Marks and numbers can be transaction event information attached to or printed on packages like shipper and consignee addresses. Note that a UUID can also be printed on packages, which make these packages (globally) unique.

This differs from the current way of working where goods are unique in the context of a business transaction, i.e. a consignment or shipment, each with their identifiers that are unique in the context of a customer and service provider relation.

A UUID cannot be used on its own to access additional data of a real-world object represented by a concept and its properties. It is no URI (Uniform Resource Identifier). Data access is only feasible in the context of transaction – and visibility events shared between two stakeholders. Accessing a UUID will give details of a transaction event with associations to these stakeholders, where each stakeholder must have a unique endpoint.

6.7 State data

States aggregate data shared by all types of events. This is about the state of a business activity that is gradually composed via sharing transaction – and visibility events. These events are accumulated to retrieve the progress of services and physical activities for state synchronisation of a customer and service provider.

The accumulation of transaction – and visibility events is called the 'state' of a business activity as specified by the data sharing module of the multimodal ontology. The 'state' of a service can be a subset of the relevant state data of a business activity since that is based on service chaining (see before). Those Service Developers that intend to develop services of two or more events must

specify ‘state’.

State data is gradually accumulated starting with business service discovery and a booking or request for quotation. A booking can contain for instance only totals of the cargo to be transported (e.g. total bulk volume or total number of containers), resulting in a quote and a contract with prices and delivery – and contract and carriage conditions. An order provides more details of the cargo, resulting in a plan and execution of the physical operation supported by a visibility service.

A state only contains the identifications (UUIDs and real-world identifications) of Digital Twins, persons, and locations, where humans can find and access data by using the real-world identifications. This is the data shared by transaction – and visibility events. The associated business activity data set specifies all data that is accessible by two stakeholders sharing and synchronising state information.

6.8 Electronic documents – state data integrity

Many of the current logistics operations are driven by documents. Gradually, these documents are replaced by electronic versions. These electronic versions of documents have the same data structure as ‘state’ for the business activity ‘transport’.

Electronic documents can be specific to a modality, implying that it is a service of a business activity with that modality. Examples are eCMR for road - and eB/L for sea transport.

Electronic documents represent the data of a state of a service for a modality of which its integrity can be assured, for instance in a state where cargo is handed over from one enterprise to another. The state ‘in execution’ of the multimodal supply chain visibility service (Figure 6-2) represent a data set of an electronic document.

6.9 A template to specify state transitions

State transitions specify the processing triggered by sharing data from an input – to an output state. They are the core of a ‘service’. They specify how states of two collaborating organisations that implement the same service can be synchronized. Each state transition is about validating event data with state data and if required perform calculations before updating the state information.

A state transition is specified as follows:

Specification item	Description
State transition	Name of the transition. This is the header of the template
Input state	Name of the input state. The input state contains data with a given structure (see before).
Event primitive	The interaction type that triggers the state transition. An interaction or event contains data with a given structure.
Pre-condition	Validation rules for the trigger versus the input state. The simplest validation is that an event has a UUID of an event in the input state data. A pre-condition can be expressed in SPARQL where event data is compared with state data, resulting in true (the transition can be executed), false (the transition will not take place or an error is detected).
Exception	A list of issues detected at the pre-condition. This is not necessarily an error, since a trigger event can reflect the actual physical operation that differs from the planned or expected one contained by the state. In this case, an exception is signaled.

Specification item	Description
Firing rule	Any additional processing that takes place like an ETA calculation via a service.
Post-condition (output state)	The update of the input state to the output state. This can be expressed as a SPARQL insert operation. For instance, a visibility event with an ETA for a transport means is stored in addition to what is already contained by the input state of a transition.

6.10 Supervision of business activities

Supervising bodies, responsible for supervising one or more regulations, have two approaches for sharing data with enterprises, namely the push-based approach with for example declarations and the pull-based approach. Both approaches and data requirements are specified by a regulator (according to its procedures) and supervising bodies can implement a profile for these data requirements.

Selecting a pull- or push-based approach depends on the supervision procedures defined for a regulation. Technically, a pull- and push-based approach can combined.

A push-based approach is the specification of a service. It clearly specifies when and where particular data must be pushed by an enterprise to an authority, for instance an entry summary declaration must be presented to a customs authority of the country in the port of discharge of cargo, 24-hours prior to its loading in a port of loading. As a service, it consists of an interaction sequencing between an enterprise and a supervising body. An enterprise will have roles like 'economic operator' or 'declarant'.

A pull-based approach differs from a pull-based approach in the sense that events are shared with UUIDs and identifications of real-world objects. These can be transaction – and visibility events. They provide data access according to the rules specified before. A regulator will specify a data set that can be called 'regulatory data set', which is like a business activity data set.

The eFTI (electronic Freight Transport Information) Regulation takes for instance a pull-based approach, where a regulator specifies the eFTI data set and the visibility events that must be shared. The eFTI data set refers to the eCMR data set, which represents a state data set (see before).

6.11 Service chaining - modularisation

Each service must be modular in the sense that it specifies a structured sequence of interaction with an identifiable start – and end state that can function in a service chain. A service chain is the sequence of services, where this state is the end state of one service and the start state of another service. An initial service has a unique start state and the final service a unique end state. The following figure shows and example of a sequencing of services where ordering and visibility have 'agreed orders' as common state.

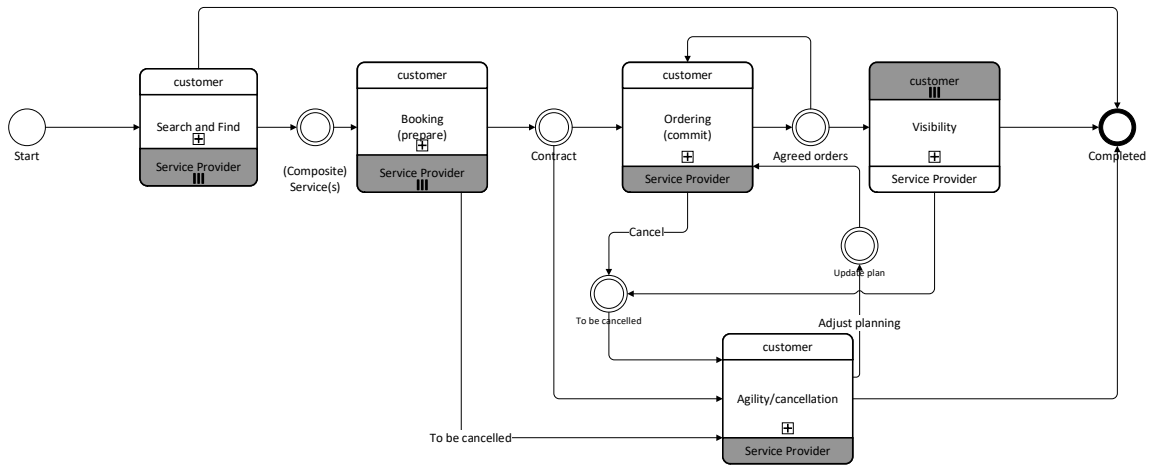


Figure 6-7: an example of sequencing of services for a business activity

The example pictured in figure 6-7 is one supporting a transactional relation. A customer discovers a business service, books and orders the service for its goal, and receives visibility data for the progress. The contract is implicit by ordering according to a booking confirmation of a service provider. A customer can also synchronise different legs of a supply and logistics chain by sharing order updates and having the capability to cancel an order (in case of delays or loss/damage of cargo).

The following services can be specified based on best practices and be made available to organisations:

Services	Definition
Agility service	Cancellation of an order due to unexpected conditions like delays, losses, or theft of cargo and/or vehicles and potentially triggering re-planning of a (leg of a) logistics chain.
Booking service	A service for negotiating of and concluding a (framework) contract encompassing an agreement of prices and conditions for performing one (or more) business activity(-ies) by a service provider meeting customer goals. A framework contract is an agreement for multiple orders in a period.
Ordering service	A service for actual execution and detailed planning of a business activity according to prices and conditions resulting from booking service.
Quotation and marketplace services	A discovery service for (logistics) business services meeting customer goals. This service needs to implement a high precision and recall, all logistics services, timetables, and spare capacity meeting customer demands have to be found.
Visibility service	A service providing details of the execution of a business activity and its planning by a service provider according to an agreed transport plan.

The base assumption behind these services is the reservation and use of 'resources' of a service provider by a customer. Transport means, containers, etc. are examples of those resources represented by Digital Twins. Internal equipment like cranes and personnel are other resources. Quotation and marketplace services support discovering these resources, booking services are about their reservation, and ordering services about their use, where visibility service show how they are used.

These services are specified in the context of a business activity, e.g. transport, transship, and stuff/strip. These business activities specify the total data set that can be shared, where individual services synchronize state data (see before).

These B2B services must be supported by others like financial service (insurance and payment). These must be provided by financial institutions.

Not all these services are always required for all business activities and all customer – service provider relations. In some cases, business services are a commodity: there is no negotiation about prices and conditions, those provided by the service provider are applicable.

Not all services must be digital. For instance, service discovery and booking can result in a framework contract. This is most probably according to internal procurement procedures of a customer. It results in a contract confirmed by customer and service provider. Ordering and visibility can be digitized in the context of a framework contract.

The previous two examples show that modular service is either flexible to support transactional relations and framework contracts, or different services for these types of relations are specified. Both is feasible, it all depends on the specification of state transitions.

Another example is where prices and conditions are properties of a business service, like in for instance parcel delivery services. Search and find results is directly followed by ordering. Planning can be selected by a customer, only visibility information is provided.

Service chaining is only feasible if these services have the same application area and, in case of spacialisation, have the same base service.

6.12 Profile

Taking a evolutionary approach, a profile limits the options of data, services, and their interactions (events) for an enterprise, either in its role of customer and/or service provider. A profile thus supports:

- The **business activities** of an organisation. This can be a list of one or more activities that basically identify the complete data set that can be shared.
- Per business activity:
    - The structured sequencing of **services**. This can be a standard pattern like the example shown before (Figure 6-7) or a selection of a service in such a pattern. Multiple services and/or service chains may be selected.
    - For each service (chain), a selection of the **optional elements** of a business activity, which refers to state data of that service (chain):
        - Subtypes of **Digital Twins** supported by an organisation. The optional ones can be selected according to the rules specified by the business activity.
        - Any **optional elements** (associated concepts and data properties) of the selected subtypes, for instance ‘dangerous’ or ‘reefer’. These optional elements must be specified as such by a business activity.
        - Relevant **locations** and **areas** for the organisation, i.e. where the business activity is performed.
- Per service a selection of the interactions (**events**) that are supported. At least the events where the end state of a service can be reached from its start state must be selected (mandatory events). This leads to a selection of states and state transitions, where the data stored by states and processed by transitions is defined by the data set selected for the

business activity.

- Resulting from the selection of a business activity, any **applicable regulations** and their data requirements that must be supported. Data requirements of authorities operating as supervising bodies must be selected. This can lead to additional data requirements for the profile. Furthermore, any events of services that can support regulatory data requirements are selected for implementation. These might be the ones that are already selected or may lead to selection of additional events and state transitions.

It is up to an organisation to specifies its profile for multiple – or a single business activity or for each subtype of a business activity since this will reflect its organisational structure. This is comparable with a community specifying its profile for its members and each member implementing a profile based on the community profile.

6.13 Overall structure for service development and - customization

The following figure visualizes the various concepts and their associations introduced in the previous pages.

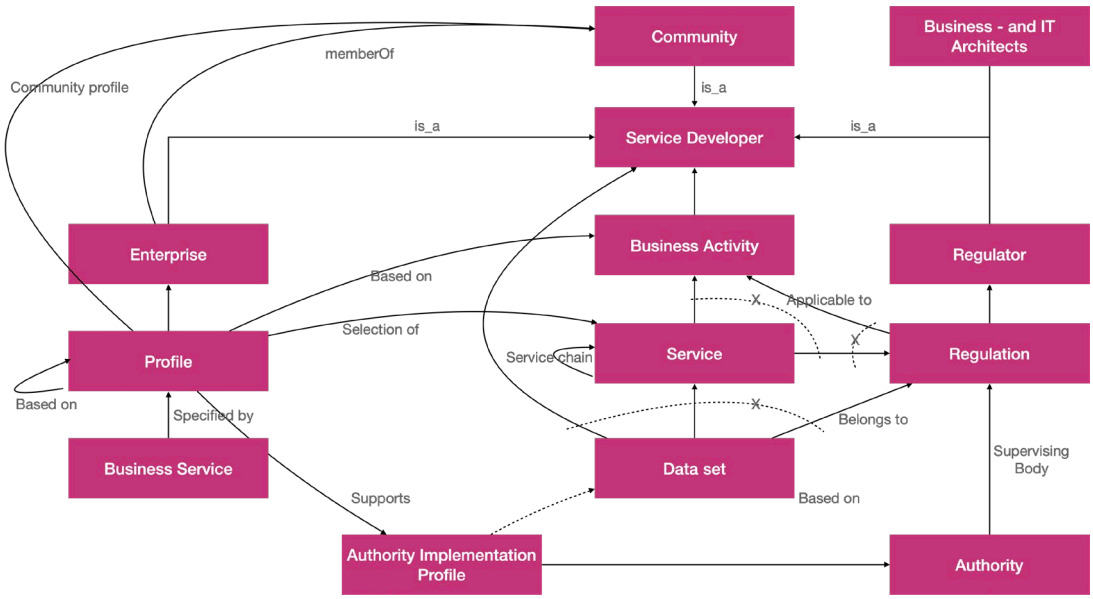


Figure 6-8: concepts and associations for service development and - customization ('X' – mutual exclusive)

The previous figure shows the role of 'Service Developer' that can be taken by an individual enterprise, a regulator, a community, or business and IT architects.

All associations between concepts shown in this figure are implemented by URIs. These links are required for implementing consistency rules (see Service Registry). A profile is thus represented by a SHACL document with a URI to the SHACLs of a business activity and authority profile(s), and a list of services and events of those services that are implemented for that business activity. The SHACL specifies the total data set of an organisation for a business activity.

All concepts, associations, and constraints to these associations are part of the data sharing ontology.

Each concept shown in the previous figure has metadata like (examples are given for the multimodal visibility service):

- Identification – unique identification of an instance of a concept.
- Short name – a name describing the instance (e.g. multimodal visibility service).

- Description – a (brief) description of the instance (e.g. a service that supports visibility of all modalities and cargo types).
- Creator – reference (UUID) to the creator of the instance (e.g. FEDeRATED business – and IT architects).
- Creation date – the date at which the instance is created (e.g. end date of the FEDeRATED action).
- Start validity date – the date at which instance can be applied (can be identical to the creation date).
- End validity date – the date at which the lifetime of the instance ends.
- Successor – the instance that succeeds this one.

The following constraints are applicable:

- Business activity constraints:
  - A business activity must at least have one service; a service is unique in the context of a business activity.
  - A business activity may be specialized into subtypes.
  - Subtypes of a business activity may partly match, but are also spacialisations of its supertype (this is very normal for sub- and supertyping).
- Service constraints, also input for correctness and completeness of service development:
  - A service must at least consist of a start and an end state and one state transition. There can be an additional rule where a user selects to have at most one state transition, e.g. for production of an electronic business document data set.
  - A service must specify a path of state transitions from its start – to its end state.
  - Service chaining is by making the end state one service as a start state of another service.
  - In a service chain, there is always a service where its start state is not an end state of another service and a service where its end state is the start state of another one.
  - The states of a service must contain a subset of at least the mandatory elements (concepts, data properties, value constraints) of its business activity data set; optional elements of a business activity data set may become mandatory for a service and its states.
- Regulation constraints:
  - A regulation is either specified by services or a data requirement formulated for a business activity. If a regulation is specified by services, it is like a business activity.
  - A regulation is applicable in an area.
  - A regulation can have additional supervision constraints.
- Data set constraints:
  - Data sets must specify events of a service that are shared by business to authorities.
  - Data sets can be based on a state data set of a service (e.g. an eCMR for eFTI).
  - A data set is either formulated as data requirements for a regulation or as a data set that can be made available.
  - If a data set represents data requirements of a regulation, the query arguments (and filter) must be specified upon which data must be made available.
  - If a data set represents an information service, conditions for accessing an information service must be specified.
  - Correctness and completeness constraints – all selected concepts have one or more data properties and each data property has value constraints. These can be

selections of code values and/or code lists. This constraint is implemented by the tree editor Service Development – and Customization module. These constraints are applicable for all tree structures created with the tree editor.

- Profile constraints:
  - A profile must at least refer to a single business activity or to another (community) profile; it can refer to multiple ones.
  - A profile must contain the mandatory elements (concepts, data properties, value constraints) of a business activity data set and its selected services.
  - A profile must refer at least to one service of a business activity; it may refer to multiple ones available for a business activity.
  - A service in a profile must at least contain those state transitions that specify a path from its start – to its end state.
  - A profile may contain the optional elements (concepts, data properties, value constraints) of a business activity data set and its selected services.
  - Any constraints formulated by a profile on element level can be on detailed value constraints, like selection of a code subset from a codelist.
  - A profile is applicable in an area/region.
- Authority profile constraints:
  - An authority profile must refer to at least one Regulation for which the authority acts as supervising body.
  - An authority profile must at least contain the mandatory elements (concepts, data properties, value constraints) of one data set for a Regulation for which the authority acts as supervising body.
  - An authority profile can contain multiple, combined data sets for those Regulations for which the authority acts as supervising body. This can be an authority profile for multiple Regulations of a business activity in a country or community thus implementing the principle of single data provision for multiple regulations.
  - An authority profile is applicable in an area/region.

These constraints are related to ‘roles’:

- Service developer – business activity – and service constraints are applicable.
- Regulator – Service Developer – the regulation -, business activity –, and service constraints are applicable.
- Regulator – data set – the regulation – and data set constraints are applicable.
- Service customiser - supervising body – the authority profile constraints are applicable.
- Service customiser - enterprise – the profile constraints are applicable.
- Service customiser – community – the profile constraints are applicable.

7 The Service Registry

The Service Registry is to configure the Index and enable discoverability with business services. Index configuration is based on constructing profiles for business activities and services, as specified in the previous section, identification and authentication by a VC(s), the integration of a separate Index implementation with existing IT legacy systems, and the support of (guides of) standards.

In this section, the Service Registry is decomposed in modules with a specific function. Although the objective is to re-use as much as possible business activity and service specifications, there can be many instances of these modules. Specifications and matches made for those specifications (and with open standards) must be trusted. These modules require identification and authentication (see the following chapter).

Being able to configure the Index enables the construction of a software component that supports all types of data sharing applications.

7.1 Overall functionality and interfaces

Thus, the Service Registry is decomposed into the following functionality (see also section 4.4, extended with additional functionality):

- Service development – specification of services for business activity(-ies).
- Data set / Information Service (IS) development – specification of standard queries (like an eFTI data set) or Information Services that support decision making.
- Service customization – construction of a profile.
- Business service management – specification, publication, and management of business services for a profile.

By applying open standards for the various interfaces of the functionalities of the Service Registry and other components (like the Index), the Service Registry becomes generic. Restrictions are given by support of the concepts of the data sharing ontology as given in section 6.4 and associations and data properties of concepts for service development and – customization (Figure 6-8).

The various interfaces of the Service Registry and its functionalities are already identified before (section 4.4). They are:

Interface	Description	Module	Interface from Service Registry to	Baseline standard
FEDeRATED semantic model	Ontologies and their constraints (the semantic model or 'language')	Service Development	Service Registry Index	OWL (Ontology Web Language) SHACL (SHApe Constraint Language) SKOS (Simple Knowledge Organisation System) – for code lists only.
Business activity (data)	The data set specifying all data	Service Development	Service Registry Conformance	SHACL (supported by a triple store or graph

Interface	Description	Module	Interface from Service Registry to	Baseline standard
	that can be shared for a business activity		Testing (Index to optimize the operation of the Index)	database in the Index)
Service (process)	Specification of event sequencing	Service Development	Service Registry Conformance Testing Index	No standard available yet; modelling by BPMn2.0 - choreography
Service (data)	Specification of event structures	Service Development	Service Registry Index	SHACL (configured for data validation in the Index)
Profile	Customization of a business activity and its services by an end-user	Service Customization	Registration Index (this may optimize the Index replacing a configuration of a complete business activity)	SHACL (part of a VC and used for matching data sharing capabilities between Indexes)
Business service	Discoverability of an end-user for the business services it can provide	Business Service Management	Index	RDF (business services and goals; as part of the discoverability protocol)
Data transformation	Configuration of the semantic adapter for data transformation	Service Customization	Index (semantic adapter)	RML (RDF Mapping Language; the semantic adapter is part of the Index)
Data set / query	(complex) queries for data retrieval	Query / IS development	Index	SPARQL (SPARQL Protocol and RDF Query Language)
Information services	Data made available by a data holder	Query / IS development	Index	Metadata standard (e.g. DCAT and/or Dublin Core) SHACL (data) ODRL (Open Document Rights Language, access rights).
Local interface	The integration of	Service	Index	OAS (openAPI

Interface	Description	Module	Interface from Service Registry to	Baseline standard
	an IT backend system with an Index	Customization (and/or Service Development)		Specification) Any others (JSON, XML, CSV or other syntaxes used for file exchange)

The functionality can be expressed by use case diagrams as a basis for identifying software components and packages. This section identifies the various software components for a Service Registry.

7.2 Decomposition in components

The Service Registry consists of the following components:

- Service development and – customization – a module to support a user in developing and/or customizing services for configuration of the Index and obtaining a Verifiable Credential (VC).
- Matching module – a module to support the matching of two tree structures for interoperability with IT legacy systems. This module must also have a VC.
- Business Service Management – a module for creation, publication, and finding business services.

Each module will have its settings like the organisation with its characteristics. These can be relevant for discoverability. These modules have interfaces.

The following figure shows the components and their interfaces.

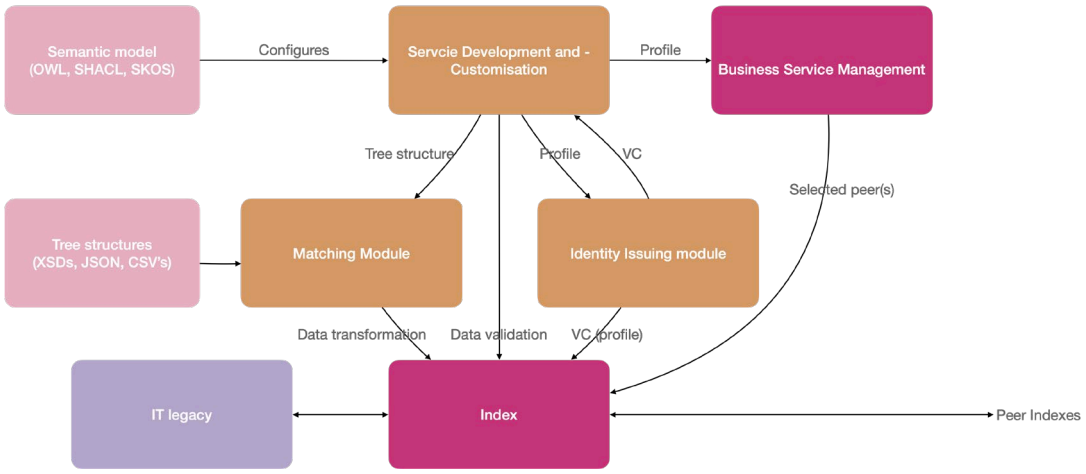


Figure 7-1: decomposition of the Service Registry into modules

The assumption is that the FEDeRATED semantic model contains all value constraints of data properties, including those represented by code lists, as SHACL and SKOS.

The functionality of each module will be introduced. The decomposition of each module reflects a client-server architecture, where the GUI, process logic, and data storage are separated, and the GUI is browser based. The Index and the Business Service Management modules are used at runtime; the others can be used at design time. The Identity Issuing module and the Index functionality are presented in separate sections.

Each implementation of (part of) the functionality of a module used for service development and matching must have a unique identification that can be authenticated (section 8.1). This is the mechanism as used for implementation of the Index. A VC contains a claim made by a service developer or someone constructing matches (section 8.2). These claims can be combined by one organisation or by separate ones. Thus, an instance of a module used for service development or matching must have an organisational wallet. This wallet will not be shown in the following figures, but has to be included in an implementation.

Implementation of the Service Customization functionality does not require a separate identity and authentication mechanism since an Identity Issuer will accept a profile according to an agreed issuing policy. Preferably, the data stored by this functionality is not accessible to external stakeholders.

Implementation of the Business Service Management module also does not require a separate identification and authentication mechanism since its output is a selected peer(s) that will be identified (and authenticated) at Index functionality level.

7.2.1 Service Registry for service development and – customization

This module consists of the following components (see also next figure):

- GUI (Graphical User Interface) for service development and – customization – a component to support a user in its role. The GUI is configured by the data sharing ontology.
- Process logic-development – three components to create, read, and update instances of concepts stored in some data store:
    - Rule set – the constraints on concepts and their associations of the data sharing ontology.
    - Tree editor – a component to construct tree structures on an ontology and its SHACL/SKOS data.
    - Process editor – a component to support service development for business activities.
    - Discoverability – a component to discover implementations of this module applied for service development.
- Data store – a component to store services and profiles. It is distributed and needs to be trusted in the case of storing services.
- Sample data generator – a component to generate samples of data sets for the tree structures that have been developed.
- Output generator – a component to transform tree structures into required configurations for the index.

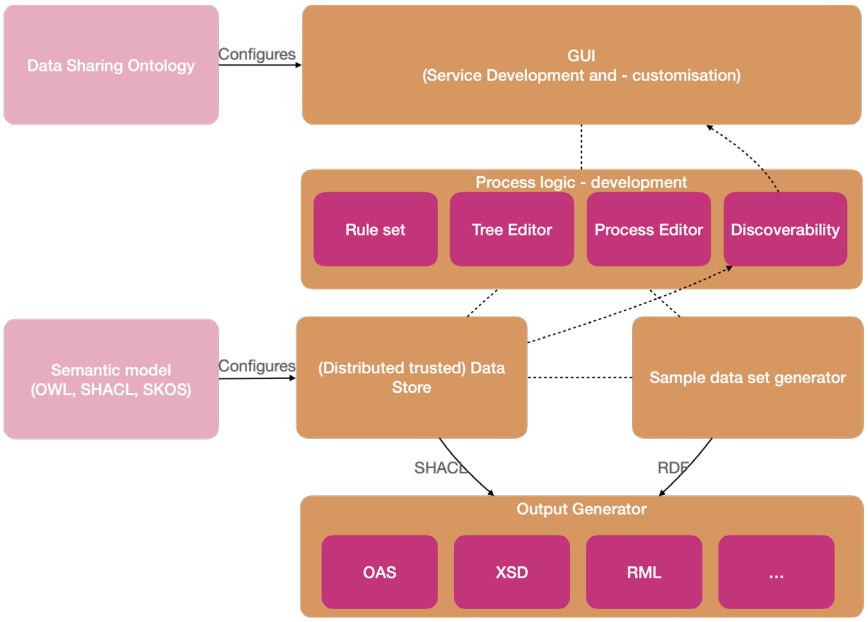


Figure 7-2: decomposition of the Service Development and – Customization Module

The individual components are briefly introduced.

7.2.1.1 GUI for development and customization

The GUI of the Service Registry supports a user in its role:

- Service development – specification of business activity data sets and services with events.
- Data requirements formulation – specifying queries for accessing data.
- Service customization – specification of a profile for a business activity.
- Information Service specification – specifying IS and their access policies.

The GUI is configured by data sharing ontology, its constraints, and potentially additional rules specific to a role taken by an organisation. Depending on a role of a user of the Service Registry instances of concepts can be manipulated (CRU – Create, Retrieve, Update). For instance, an employee of an enterprise with a service customization role can create, retrieve, and update a profile; this employee can retrieve business activities with their data sets and services. When creating or updating a profile, the profile constraints are applicable.

Thus, depending on a role, a user can browse through instances and add or update existing instances of the data sharing ontology.

The GUI uses the discoverability routine for accessing instances of the data store. The GUI can browse through these stores for discovering business activity – and service specifications that can be re-used. In case of business service management, the search function enables a user to formulate a goal. A user may decide to access a subset of the existing data stores; this subset could be configured as a setting.

Another function of the GUI is to select a required output format. This could be a setting contained by the rules, for instance a tree structure representing an event must always configure a generic eventAPI.

7.2.1.2 Rules

Each role may have additional constraints (rules) formulated by the organisation taking this role. This may reduce the functionality supported by the GUI. For instance, a service developer of an industry

association may want to specify a tree structure for a specific business document data set. By its business – and service constraints this is a business activity with one service of only one state transition (section 6.13). Business activity data set and state data are combined into one data set to enable the construction of a single data set.

Another specific role might be that only a subset of the subtypes of Digital Twin are supported. For instance, a service developer (or regulator) is only able to specify business activities, services, or data sets for road transport. These rules are additional constraints to the generic ones.

These rules are formulated by an organisation and stored as additional rules. They are used on the user interface.

7.2.1.3 Tree structure editor

The tree structure editor enables a user to create, retrieve, and update tree structures of a concept and its associated concepts. Creation and updating of tree structures is according to the constraints of a concept. For instance, a profile of a business activity cannot contain more elements (concepts, data properties, and value constraints) than those of that business activity data set. It can contain less but must contain the mandatory elements.

A tree structure editor is applied to create hierarchical structures of concepts and data properties with the semantic model ('named graphs'). A tree structure editor must support the following functionality (depending on concept constraints of the data sharing ontology):

- Selecting a concept that is the root of the tree.
- Selecting all other concepts that are associated to the root concept and for each of these concepts select other associated concepts.
- Concepts can be subtypes of others. A user must be able to select the applicable subtypes.
- If a user selects a subtype, all data properties of its supertype(s) can be selected.
- Selecting for each concept in a tree structure the data property.
- Including the minimal and maximal occurrence for each selected concept and data property. If the minimal and maximal occurrence are 1, a concept or data property must have at least and most one instance. If the minimal occurrence is '0' the concept or data property can have an instance. If the maximal occurrence is more than one, this indicates the maximal number of instances that can be available in a data set.
- Validation option – validating correctness and completeness of a tree structure (section 6.13). Tree structures can be stored during development and validated for including them in a claim (section 8.2).

7.2.1.4 Process editor

A process editor is a component for visual development of services. It supports the constraints for business activities, services, and profiles. It also interfaces with the tree editor to construct event types and state data sets. In terms of service customization, a user can select a business activity, one or more (chained) services, and the mandatory path for each service.

7.2.1.5 Data store

The structure of the data store is specified by the data sharing ontology. The concepts (and data properties) of this ontology provide access to SHACL documents that are based on the FEDeRATED semantic model.

The data store can be implemented by a triple store or graph database.

The data store is distributed. The discoverability routine is used to access proper instances of the data store for accessing data.

7.2.1.6 Discoverability

Discoverability is a resolver for finding sources, i.e. instances of the Service Registry. It is like DNS (Domain Name Server) for matching a URL to an IP-address. It could be implemented by a public Distributed Ledger like the European Blockchain Services Initiative (EBSI). Such a ledger may also contain information of a profile of an organisation, thus providing a rapid assessment of its capabilities in terms of data sharing.

In case applying a DNS type of resolver, such a resolver must distinguish service -, query -, and Information Service developers; profiles cannot be shared. The proper resolver mechanism is for further study.

7.2.1.7 Output generator

The output generator transforms a SHACL document into a required output format. These can be the following:

- OAS APIs – these are openAPIs according to the Open API Specification. This output must be transformed into code supported by the implementation of the Index. Eventually, standard output can be generated, for instance Java API-code.
  In case OAS APIs are generated, the required operations must be selected (POST/PUT, GET).
  OAS APIs can be used to integrate with IT legacy systems or construct an API based infrastructure.
- XSDs – XML Schema Definitions. These are used to directly share data between two systems based on a trusted, secure, and reliable protocol.
  Two XSDs may represent a simple query: one for formulating access to data and another for the query result.
- JSON – Java Script Object Notation. A most common format for openAPIs. Additionally, an option might be to support JSON-LD for representing events with links to data.
- SPARQL queries – a query is represented as a SPARQL. This represents standard queries. It is especially useful for complex queries.
- RML – RDF Mapping Language. This is applied for simple data transformations, basically from one syntax to another (e.g. JSON to RDF). Complex data transformations can be constructed with the Matching Module.
- ODRL – Open Document Rights Language. This can be applied to represent access rights of an Information Service.
- Other formats that will be required in the future.

The output generator for processes (event logic) is for further study. This is either a configuration of event processing or an executable specification requiring a runtime module as part of the index.

7.2.1.8 Sample data set generator

This generator takes a tree structure with its SHACL document and generates data for each property in the tree structure. Selections are possible with generating the minimal – and maximal occurrences of properties and data sets and minimal – and maximal values.

These sample data sets are RDF and can be in other formats based on the RML output (e.g. JSON).

7.2.2 Matching module

The matching module supports development of complex matches and thus also alignments. Matching is from an input to a required output format as required for data transformation. Matching between input and output is always via the semantic model, which thus serves as intermediate (internal) format. Matching is facilitated by re-use of matches between some format and the semantic model.

Business activities, services, and events must be matched (if these are specified). Business activity matching is by the business activity data structure matching. Service matching is by analyzing the states and state transitions triggered by events. This can become complex if also the state transitions need to be matched. A first version could focus on matching states with transitions triggered by events with identical functionality in both services, i.e. figure out if there are identical events in both services.

For matching business activity -, state -, and event data structures, the proposal is to support tree matching, where identical functionality represented by different trees is matched. The more advanced would be to match two ontologies, which may lead to alignment issues and lack of inverse matches. Furthermore, events and queries are used to share (links to) data and have tree structures. Ontology matching between the semantic model and an internal data structure might have advantages for instance in constructing RDF plugins and supporting various events.

The matching module consists of the following components:

- Matching GUI – a component for creating, retrieving (view), updating, and deleting matches between two tree structures.
- Import formatter – a component to import a tree structure with its syntax (e.g. XML, JSON) and format it as a SHACL.
- Tree matching editor – a component for visualizing two tree structures and their matches.
- Output generator – a component to transform a matching into a configuration of an algorithm or a matching algorithm itself.
- Data store – this contains matches between tree structures.
- Matching validation – transforming a test data set with a given input format and semantics into a required output format (and its semantics).

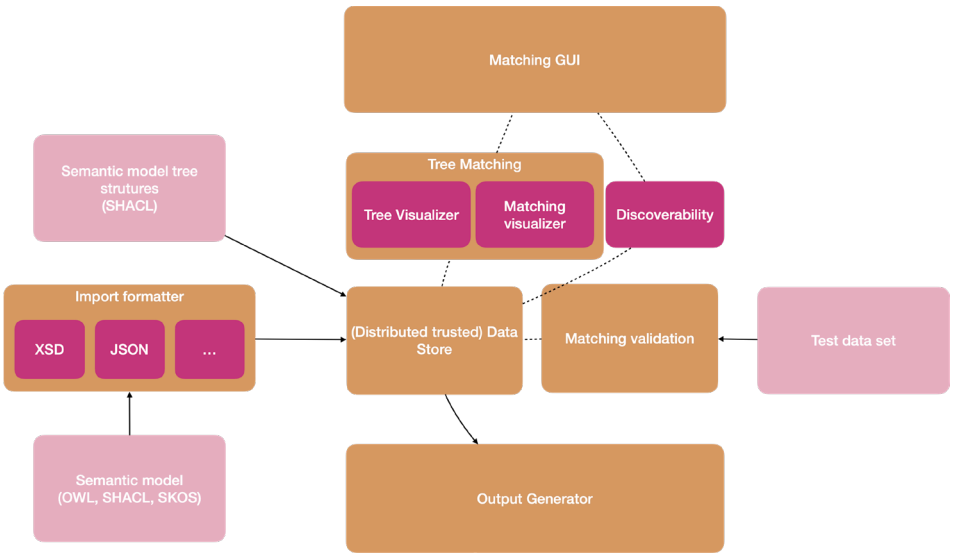


Figure 7-3: decomposition of the Matching Module

These components are briefly described. Discoverability is similar to that of the Service Development – and customization module and is thus not described here.

7.2.2.1 Matching GUI

There can be different GUIs, namely one for matching service functionality and another for tree matching. The service matching GUI supports a user in managing all its matches. These also must be stored. The tree matching GUI enables a user to construct matches between various tree structures selected by the other GUI, like business activity -, state -, and event data structures.

The tree structure matching GUI supports a user in:

- Selection of an input format and transforming the input semantics into a (SHACL) tree structure.
- Selecting a tree structure specified by the semantic model (SHACL)
- Creating (or updating) a match between all concepts and data properties of the selected tree structure.
- Generate an overview of concepts and properties that cannot be matched (this would be a basis for alignment).
- Select the required output format.
- Discovering existing matchings for any format with the semantic model.
- Select an input data set and transform it into RDF of the semantic model.
- Select an input data set and transform it into the required output format (via the intermediate semantic model).

7.2.2.2 Tree matching

This component consists of tree visualizer and a visualizer for matching. The matching visualizer must support matching operations, like

- Identical – a concept or property in one tree structure is identical to one in another tree structure.
- Concept split – a concept and its properties in one tree structure are mapped to two or more concepts in another tree structure.
- Data property split – a data property in one tree structure is split into two or more data properties in another tree structure.
- Combine (concept and data property) – the inverse operation of a split.
- Constant – fixed values are assigned to a property associated with another property (for instance inclusion of measure unit specifiers and so-called qualifiers used in EDI (Electronic Data Interchange) standards).
- Value transformation – input values are transformed into output values, for instance change of a measure unit specifier (from ‘g’ to ‘kg’) or changing time zones.
- Calculations – all types of calculations where values of input data properties are used to calculate values of output properties.

Not all data properties are necessarily matched between two tree structures. These missing matches are input for alignment. Alignment, which results in extending one of these tree structures, may not always be necessary. For instance, extending the semantic model functioning as intermediate model is not necessary if the extension is only supported by a single tree structure matched with the model.

Matching can have an inverse, meaning that a data set of a tree structure can be transformed into another tree structure, and back to the original one. Inverse matching does not lead to data loss.

Association matching, where associations are defined by the tree structure, is automatically via applying the semantic model. An example of association matching is where one tree structure represents an itinerary for picking up goods at locations, whereas another tree structure has an order type of structure where goods have to be pickup at one location and dropped off at another. This is a type of data sorting.

7.2.2.3 Data store

This stores all matches between tree structures. It can contain matches with open or defacto standards that can be used to transform an input format to a standard implemented by a peer. This enables migration of standards to the proposed capabilities. It also enables local data transformation for these open standards by re-using matches.

The data store is distributed, whereby for instance the matching with open standards is provided once for re-use by many others.

7.2.2.4 Import formatter

The semantics of a data set is imported and transformed into SHACL (tree structure). In most cases, the semantics of a data set is specified in its syntax. For instance, an XSD represents the semantics of an XML data set. In CSV (Comma Separated Value), the semantics might be limited to column headings; semantics of values in cells of a row might be absent.

The following formats might be supported: XSD, JSON, CSV. Some domains may have specific formats, which then can be included. These may lead to additional matching rules.

7.2.2.5 Output generator

The required output must be executable, either as a configuration of a data transformation tool or executable code generated by the Matching Module. In case of configuration, this could be XSLT used for XML data transformation. This output is for further study.

7.2.2.6 Matching validation

Matching validation is about the ability to transform input data in a required output format. This can be in two steps, namely from input to RDF of the semantic model and from RDF of the semantic model to an output format.

A user must be able to inspect the data transformation to detect any missing or wrong matches. This requires knowledge of the format of an input – and an output data set.

7.2.3 Business service management

Business Service Management supports creating and publication of business service for a profile and finding business services matching a goal.

Where a profile is a SHACL of a business activity (and its services) applicable in an area, a business service takes several data properties and select values for these. To enable discovery of business services, these data properties must be known to a customer formulating a goal. Therefore, they must be selected at Service Development and only those properties are shown in Business Service Management that are relevant for a profile.

Thus, the module consists of the following components:

- GUI for Business Service Management – a component to support a user in the specification and publication of business services and posting goals to find matching business services.

- Matching Algorithm – matching of a business goal with business services.
- Discoverability - a component to discover implementations of this module containing business services.
- Data store – a component to store business services.

It is not necessary to elaborate the functionality of these components in more detail.

7.2.3.1 Business service editor

Business services come in three types, namely:

- Business services based on business activities. These provide classifications of conditions, duration, and prices based on the cargo type and region. For instance, they can have a subdivision in weight ranges of goods related to prices and evolving in different ways of organizing logistics. These options should be selectable by a user.
- Timetables. These are fixed times at which a transport means is at a location and moves to another location. It can be a timetable of a train, flights, or voyages of vessels with a list of ports of call.
- Ad hoc capacity. This is an option whereby a logistics service provider publishes available transport capacity on (legs of) an itinerary.

Each of these business service types has its own editor.

7.2.3.2 Matching algorithm

Matching is in two ways, namely:

- Direct match – a goal matches a business service (or more). A goal is distributed to one or more service providers that locally match that goal in their Business Service Management module.
- Chain match – a sequence of business services matches a goal. These can be business services of different providers and may require a query strategy.

Decomposition of a customer goal in multiple business service can be done manual, i.e. by a human, or can be supported by a (complex) matching algorithm.

In the latter case, the matching algorithm may already have collected all business services of service providers that it wants to use. This can be a setting of the Business Service Management module for business service discoverability.

7.2.3.3 Data store

Business services can be stored in two ways, namely as SHACL files for a profile of a business activity or as RDF with values of selected data properties. In both ways, the data properties and their values are constrained by a profile of a business activity.

Storing business services as RDF can have advantages in terms of discoverability. A goal can be expressed as a (SPARQL) query of business services that is distributed to implementations of the Business Service Management Module. The query result is a list of service providers and identifications of the business services that match the query.

7.3 Implementation aspects

A first consideration for implementation is the decomposition of the functionality into various components supporting the roles. The constraints and roles can be made specific to these roles,

which makes the GUI functionality simpler and contributes to ease of use.

If this is approach is taken, it is recommended to decompose the Service Development and – customization module into three modules, namely a module for support Service Development, one for Service Customization and another for specification of queries and Information Services.

All modules can be developed in parallel, with the following dependencies:

- The module for service customization requires services that can be developed. Therefore, Service Development module must start.
- The module for business service management requires a profile for a business activity. Therefore, it can only be developed if such a profile can be constructed.
- The matching module requires at least a single event of a service to be represented as tree structure for matching it to another tree structure.

In parallel with these modules, a discovery mechanism for all modules must be developed. This could be DNS-based (Domain Name System) or any other mechanism. A discovery mechanism is required in case of implementing a distributed data store. Whereas this is most likely applicable to service customization and business service management, one could envisage a single (or a limited number of) data store for service development. All functionalities can be available in a cloud with access rights for services assigned to services developers.

Furthermore, an organisational wallet and VC mechanism must be developed for the Service Development module.

A stepwise development of the service development and – customization is as follows:

1. **Tree structure development** – development of tree structures for interactions (events). Generation of various settings for data sharing like openAPI specifications for integrating with IT legacy, RML for the semantic adapter, and SHACL for data validation. Documentation may also be required. Business activity data sets can be developed, but simply reflect for example order events. States and state transitions are not supported. The sample data set generator is part of this version to illustrate data sets of a developed tree structure.
   This functionality enables the specification of event data structures and the configuration of implementation of the index. Thus, data can be shared.
   The next figure shows a print screen of the visibility event example as managed by Semantic Treehouse, the initial version of the Service Registry.

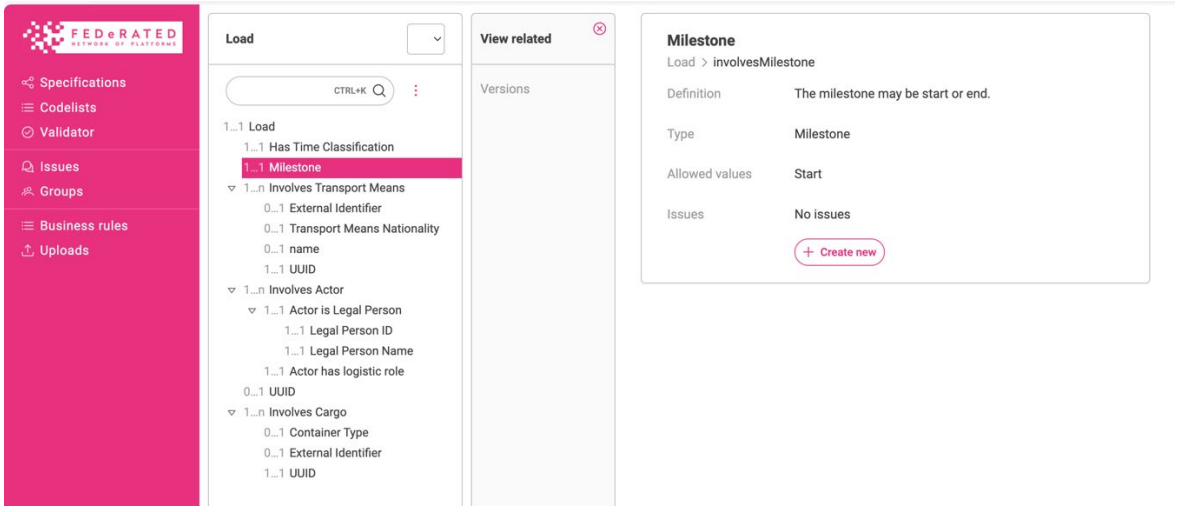


Figure 7-4: Tree view of a load event (Semantic Treehouse)

The next figure shows the sample output for this load event as an OAS document with an example data set (JSON). It shows that various types of output can be generated, for instance also RDF and XML. The tool can also be applied for data validation (the current version only supports XML validation, not yet SHACL or others).

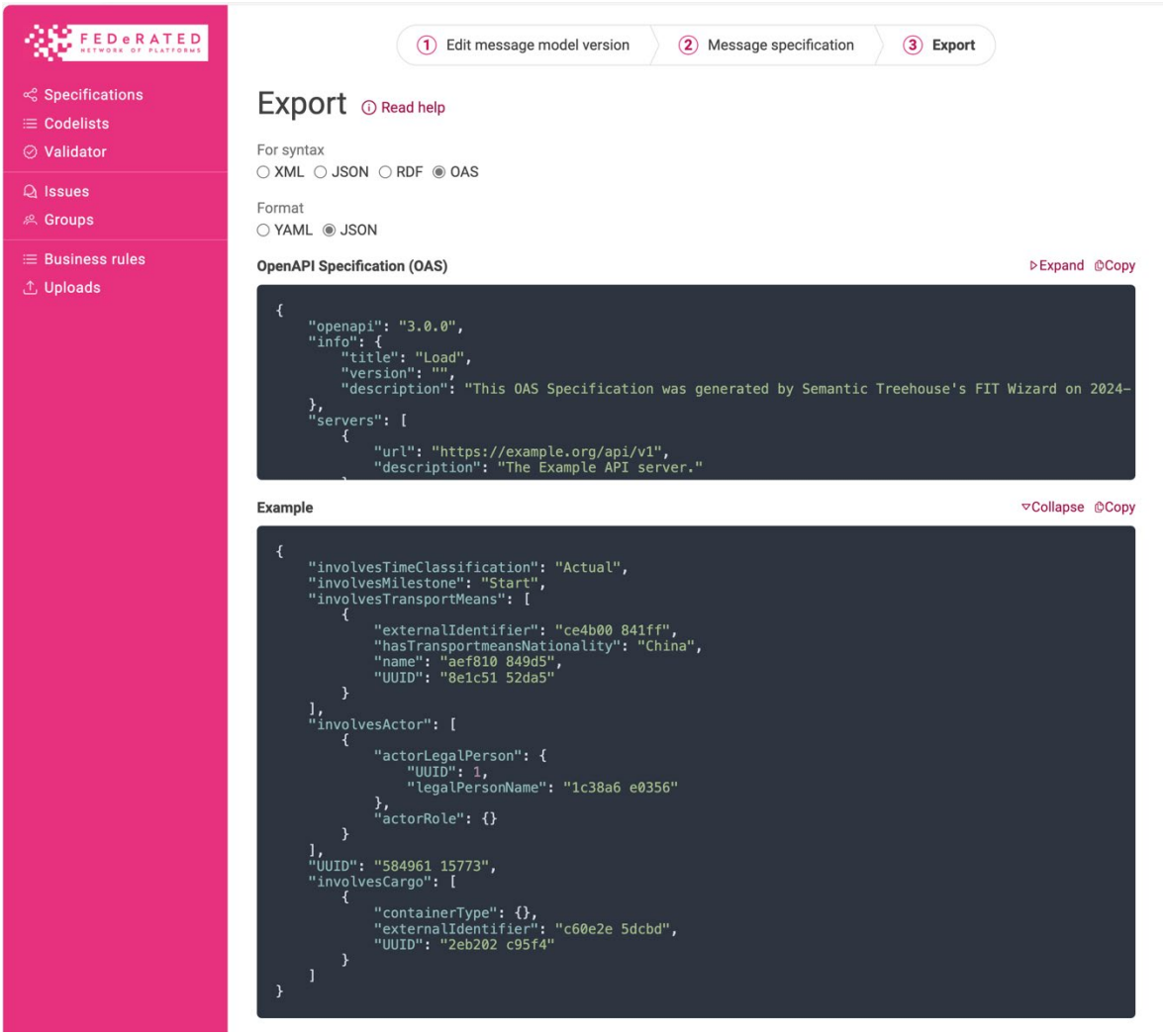


Figure 7-5: Output generated by Semantic Treehouse of a load event

2. **Guide development (profile)** – the tree structures may already have mandatory and optional elements. This functionality supports an organisation in constructing its profile. In this phase, a profile equals a guide for a standards. Such a guide can also be constructed for a community.
   A profile can be implemented as an openAPI or a configuration of generic, eventAPIs, which is up to an organisation. In the latter case, RML and SHACL are generated. Generating APIs each guide may (eventually) lead to a large number of APIs, which adds complexity to the implementation.
3. **Service development (simple)** – this is about support of the development of services. States and state transitions triggered by events are supported. States and state transitions are simple: status have a name (or number) and a pre- and post-condition simply refer to this name. No validation of event data with state data at this moment.
   Potentially, a type of subscription is required, for instance a customer subscribing to visibility events of its order. By having references from visibility events to a transaction event, a customer can simply subscribe via the unique identification of that transaction event.

This type of functionality supports the implementation of visibility services, including (simple) event logic.

4. In parallel, two types of functionalities can be developed, namely:
    a. **Service development (complex)** – states can have (complex) data structures and state transitions validate event – with input state data. This functionality can gradually be developed, starting with a visibility services. At this stage, an index implementation must support the stateAPIs. It is about development of the process editor, which can start simple and gradually provide more support to a user.
    This functionality enables exception signaling (like 'too late' or 'missing packages') required for business transaction management. Receiving a signal, a user can more rapidly access relevant data and decide for action.
    b. **Profile development (complex)** – this is about selecting mandatory elements of a service, both events and their data structure, and combining two or more services (with similar or overlapping functionality) into a profile. An initial version will support simple services, a next version more complex ones, and a later version the combination of services into a profile. The initial version supports an organisation in generating openAPIs for its selected events or configurations for a generic eventAPI.
5. **Business activity development** – this is about creating a business activity data set ('transport') with at least two (chained) services and support the development of a profile for a business activity. Existing services created by earlier versions of this functionality may be used in the context of a business activity.
    This type of functionality enables the support of multiple technology independent services for business activities, including the required plug and play functionality.

Tree structure development and output generation is already available (as shown by examples of print screens and output). A user of this tool is of course able to model any type of setting for the Index functionality that has a tree structure: state data structures, business activity data structures, transaction event data structures, etc. Consistency between these data structures is not yet supported. Taking this approach leads to a large variety of events, as shown in the next figure. Only experienced users can handle these, service development (third proposed step) may provide clustering of events and thus improve user experience.

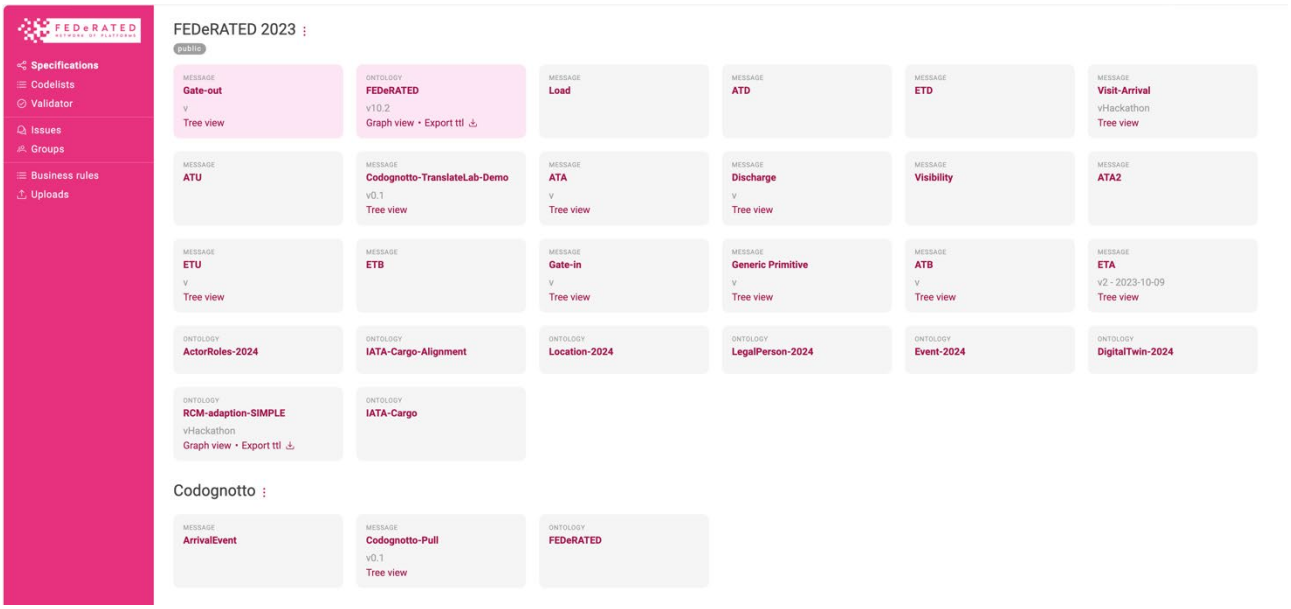


Figure 7-6: managing specifications

The query and Information Service module must contain an additional output generator for SPARQL query generation and support of rights standards like Open Document Rights Language (ODDRL) respectively.

The business service management module can be specified according to its components:

1. Specify and publish business services. This requires a profile of a business activity.
2. Specify a goal. Like business services, based on a business activity that can be requested by an organisation. Goals are not stored; they can be published (to organisations) to discover matching business services.
3. Matching algorithm. This could be simple or complex like matching a goal for a business activity to all business services based on a profile of that business activity of a service provider. Complexity comes when multiple organisations share a business service management module.

8 Security perspective - Identity and Authentication

Identity and Authentication is part of the broader security perspective of a federated network of platforms. Security encompasses certain capabilities of an organisation like a trusted and verifiable identity of an organisation, proper internal security procedures for data processing and handling, and safe and secure data sharing capabilities with others. Cyber-security as a capability of being able to deal with all types of data – and availability attacks is addressed by various European Acts like the Cyber-security Act and is outside scope. Organisations and their employees, IT systems, and procedures must implement the necessary measures.

From a security perspective, the following capabilities must be implemented:

- Identity and Authentication – a verifiable identification as a basis for authorization.
- Authorization – a verifiable claim to share events and access data.
- Access control – the rules that specify which data is accessible after authorization.
- End-to-end security – this refers to authorization of employees using IT backend systems.
- Link security – data shared of links between system components cannot be accessed or changed by unauthorized actors.
- Non-repudiation – the immutable proof that data is shared or accessed in an authorized manner.

Issuing policies implemented by trusted issuers are core to a solution for Identity and Authentication. Having an identity that can be authenticated (or verified) is one thing, trusting an issuer is required. Trust is not only based on private data sharing agreements or a public Regulatory Framework, but also on the proper implementation of capabilities. Identities are issued if the implementation of a profile is verified. A conformance testing module is required as a basis for certification.

Another aspect of such an issuing policy is to create trust in the implementation of the Index functionality, i.e. does it behave according to the protocols (section 9.1.5). This is specified as conformance testing and certification.

8.1 Identification and Authentication (IA)

This section gives a conceptual model for IA. The requirements are mapped to existing solutions. One of the requirements to a solution is that the identity of an employee of a data user accessing data of a data holder needs to be hidden to the data holder. The assumption must be that any stakeholder has sufficient security measures implemented in its own organisation.

First, a conceptual model of IA is given as currently under development based on SSI (Self Sovereign Identities) with DIDs (Decentralized IDentities) is discussed. Secondly, the model is mapped to existing solutions and the state of the art is assessed, resulting in a solution that can currently be deployed.

8.1.1 Conceptual model IA

For IA, the following requirements are applied:

- **Employee Identity** - an employee can use its internal identification and authorization to access data (and IT systems) of its organisation and any other organisation (based on access control), whereby the other organisation acting as data holder cannot refer to the identity of the employee.

- **Organisational Identity** – each organisation has a unique Identity that can be authenticated. This identity is used by any employee that is authorized to do so for data sharing with other organisations.
- **Organisation credential or claim** – the organisational identity refers to or contains the claim for data sharing capabilities. This claim is for instance the profile of an organisation (section 6.12). This credential or claim is independent of those of employees, which can for instance be role based (e.g. a procurement role for sharing transport order events).

There are basically three roles in this scheme, namely:

1. An owner or holder (natural or legal person or 'thing') requiring an Identity for actively participating in data sharing.
2. An issuer of Identities and their credentials/claims according to an agreed issuing policy,
3. A verifier (an IT system of an organisation being accessed) that verifies the identity (and its claim(s)) by trusting the issuer.

The following figure depicts the trust model.

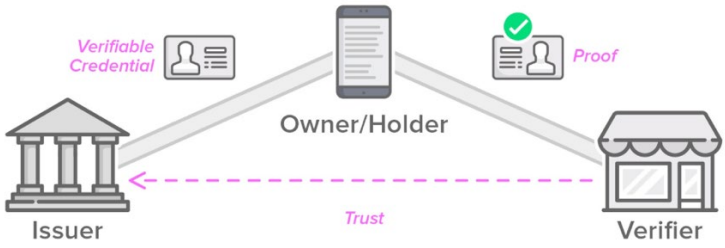


Figure 8-1: trust model for verifiable credentials

What is called 'Verifiable Credential' consists of two parts, where the first part generates the second part. The first part is called the VC, the second part the Verifiable Presentation. The Verifiable Presentation is generated from the VC and can be context dependent. It acts as a proof.

A VC consists of three parts, namely:

- **Metadata** like issuer expiration date, public keys, revocation mechanism, and other data requirements.
- **Claim(s)**: attestations about the holder/owner of a VC.
- **Proof(s)** like a digital signature of an issuer.

A VC does not necessarily contain a claim, but in many cases identities are issued based on criteria. These are input to the claim(s).

The Verifiable Presentation is the part of the VC that is used in a context. It can be considered as a token. In current implementations, such an authorization token provides can be generated from existing Identity and Access Management (IAM) Registries, that act as issuer. Thus, using a single IAM Registry or federation of these Registries is required to create trust. Such tokens are based on open standards like OAUTH2 or defacto mechanisms like JWT (Java Web Tokens). Such a token must contain or refer to a claim.

An additional role is that of a Register storing and evidence that credentials are valid. This is basically relevant for revocation of credentials. Having claims makes it possible to apply a VC in different contexts and allows for large scale application. A Register is included in the following figure.

Verification (5) is done in the data perspective (see later). 'Things' also require IA, although another

legal framework for liability and responsibility might be applicable. There are 'things' that are owned by a platform/service provider, in which case the provider organizes identity of those things, e.g. sensors.

This model is shown by the following figure (it is the model implemented by the European Blockchain Services Infrastructure – EBSI – where EBSI acts as Register). The 'token' shown in the figure can be an OAUTH2.0 token. As such, EBSI provides federation of IAM Registries.

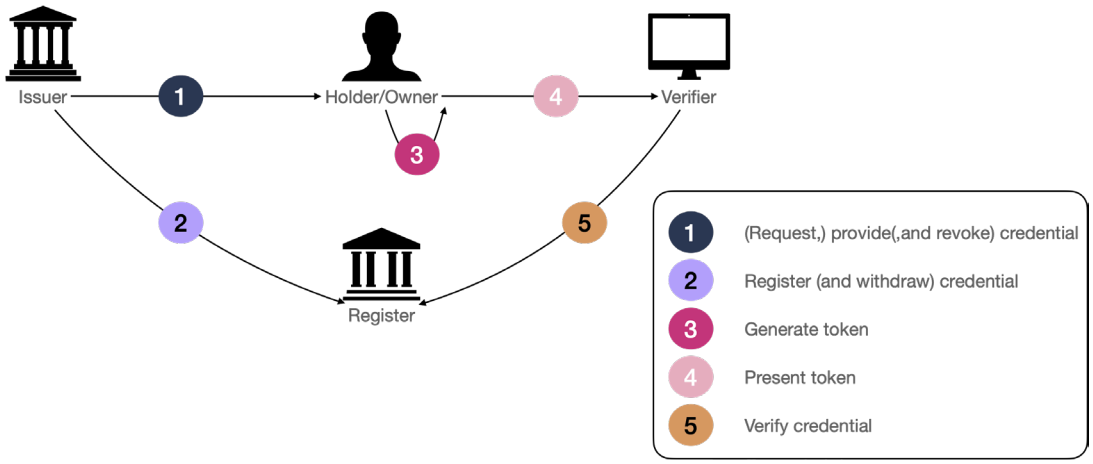


Figure 8-2: roles and mechanisms for applying tokens and verifiable credentials

Of course, a natural person can obtain a VC with multiple claims. However, a natural person may also use a trusted IAM Registry for generating a token. When doing so (step 3), two-factor authentication can be implemented. A natural person provides his username/password combination and must enter a code received via for instance a mobile phone linked to that username/password.

In this model, an issuer can register itself by a DID document as a trusted organisation, whereby the DID contains all types of metadata and key information, but also its issuing policy(-ies). Again, this also requires a type of verification and revocation mechanism.

8.1.2 Public and private initiatives in the EU – brief overview

Supervising, competent authorities must apply identity mechanisms enforced by regulations. They will not implement privately governed identity mechanisms for data sharing with natural – and legal persons. eIDAS is the public regulation for access to online government services by citizens and enterprises with an eID (electronic IDentity) and by mid 2025 enabling the use of VCs. eIDAS is available for egovernment services and business-to-business (B2B) data sharing, although this latter feature may not be implemented by all EU Member States. First VC providers are already operational in the EU implementing eIDAS2.0.

eIDAS is expected to cover the following aspects:

- **Issuing policy** – as part of the eIDAS Delegating Act, each Member State has its own issuing policy for identities.
- **Data Sharing Agreements** – these are specified by the case in which eIDAS is applied by a MS authority. These can be local (municipality) or EU (e.g. customs, based on the UCC (Unified Customs Code), or an EMDS Regulation (not available (yet))). These data sharing agreements must implement all generic data and privacy related acts like the Data Act, GDPR, etc.
- **Technical scheme** – specification of the technical infrastructure supporting eIDAS. This is part of the eIDAS Implementing Act.

- **Credentials** – the structure of credentials is part of the Architectural Reference Framework (ARF). This structure can be expanded to support the European Mobility Data Space (EMDS) and other initiatives in for instance healthcare and industry.
- **Issuers** – identity providers operating according to the eIDAS issuing policy of a Member State, resulting in different mechanisms per EU Member States (like Digid for natural persons and eHerkenning for legal persons in the Netherlands).
- **Federation** – re-use of an identity across multiple MSs is supported by eIDAS(1.0/2.0).

The EU develops an EU wallet by which a European citizen can store various credentials (e.g. driver license, passport, identity card, diploma,) issued by an authority, linked to its eID or part of its VC as a claim. Information of its eID is also stored on the blockchain. The blockchain solution supporting the 'register' role is developed by EBSI. It can also be used to store trusted IAM Registry APIs and thus serve as Identity Broker.

In parallel to public initiatives, private initiatives are established and under development. For instance private initiatives aim to develop organisational wallets for storing organisational VCs. Not all wallets for storing VCs have the same interfaces, these still need to be standardized. On top of these, private initiatives organize themselves into so-called dataspaces; these are governance frameworks with horizontal aspects (horizontal aspects can be applied by multiple dataspaces and also require governance). An example of such a private initiative is iSHARE addressing the following aspects:

- **Foundation** - governance of the Trust Framework (type of VICAL for dataspace authorities, creating trust). This is a horizontal aspect.
- **Trust Framework** – data sharing agreements (legal, functional, operational, and technical; independent of the data that is shared). This is also a horizontal aspect.
- **Dataspace Authority** – certified organisation acting as an issuer and storing credentials of holders. A dataspace authority acts as an issuer and a register. A dataspace (like EMDS) can have multiple dataspace authorities (in this approach, EMDS can also consist of multiple dataspaces).
- **Credentials** – the structure of credentials is specified in a dataspace and stored by a dataspace authority.
- **Authorisation Registry** - referred to via a holder credential, having its own credential, storing authorisations for data/service access, given to third parties by a holder. An Authorisation Registry can act as agent on behalf of an entitled party (data owner). It implements delegation.
- **Federation** – re-use of an identity across multiple dataspaces, supported by a ledger.

To fully comply with the Reference data sharing Architecture, all Identity Brokers should be federated in order to support a network of interactions. In practical terms this would enable an officer of an authority in one MS accessing an IT system like an eFTI platform that has registered its IAM Registry in another MS with another Identity Broker.

8.1.3 The proposed way forward IA

VCs with multiple claims are the proposed solution for implementing IA at a large scale in a dataspace like EMDS. eIDAS2.0 and issuers conforming to eIDAS2.0 will be operational by 2025. This implies that any Regulatory Framework for data sharing in supply and logistics that must be implemented after this period can be based on this regulation. Any claim required for such a Regulatory Framework must become part of the ARF of eIDAS2.0.

Issuers must also be able to provide VCs that can be applied in a B2B context. There can also be

issuers operating in a B2B context only using the ARF developed for eIDAS2.0. A Regulatory Framework for issuing VCs with such claims is required for all types of issuers in supply and logistics.

In the context of a federated network of platforms, organisations participating in data sharing must have capabilities in terms of VCs. They also must have the capability for an organisational wallet that can be applied in an open, neutral way, meaning that it must support standardized interfaces for managing VCs and their presentations.

Since there is not a (voluntary) Regulatory Framework for an EMDS (B2B, B2G and G2B) and many data sharing applications in supply and logistics still have a limited scale and/or are managed by trusted (platform) providers, it is recommended to use existing mechanisms like single or federated IAM Registries and authorization tokens (OAUTH or JWT). This might give different solutions for B2B and B2G/G2B like currently is the case. For large scale application of data sharing, VCs are required.

8.2 Authorization – claim

Authorization consists of two parts, namely authorization of employees to perform certain tasks and authorization of an organisation to access data of another organisation. Authorization of employees must be organized by each organisation participating in data sharing. A matching with internal employee authorization and organisational authorization must also be managed by each organisation. An example of the latter is to associate the role 'procurement employee' to openAPIs implementing an ordering service (see section 9.3.3 for examples).

Authorization at organisational level is based on a claim that can be verified by a verifier. Four types of claims are distinguished:

- **Service development claim** – this is a claim made by a service developer for one or more business activities and its service(s). It can consist of for instance:
    - Name of the organisation
    - Identifier – any relevant identifier issued by a trusted issuer like a registration by the Chamber of Commerce.
    - Type of organisation – a distinction of 'regulator/regulating body', 'industry association', and 'enterprise' is made. This assumes that natural persons cannot act as service developers. The options need further review when implementing this claim.
    - Classification – the primary industry sector of an industry association or legal person, e.g. 'logistics', 'health', 'industry', and 'retail'. This is a basis for business activities addressed by a service developer.
    - Application area – a list of business activities that are covered by a service developer. These can be more than the primary industry sector, it can cover for instance 'chemical industry' and 'logistics', whereby services for logistics are re-used from others. For a regulating body, this is the set of business activities for policy making and/or supervision.
    - Business activity – the business activity addressed in the claim. This can be multiple business activities. This is optional in case a service developer provides a specification of a single data structure.
    - Service – the service addressed by the claim. This can be multiple ones. This is optional like for business activity.
    - Data structures – a list of data structures (events) that are addressed by the claim.
    - Any other fields. The claim structure needs to be reviewed at implementation, which

can lead to additional fields to be included.

- **Matching claim** – this a claim made by an organisation providing matches between services and/or services and standards. It can consist of for instance:
  - Name of the organisation
  - Identifier – any relevant identifier issued by a trusted issuer like a registration by the Chamber of Commerce.
  - Type of organisation – a classification of the role of an organisation in data sharing like 'industry association' and 'ISS provider' (section 9.3.3.5). This list needs elaboration.
  - Application area – identification of the application area for which matches are provided. This refers to both regulations and business activities (see service development claim).
  - Type of matches – a classification the standards that are matched. The classification refers to industry standards for an application area, open standards (i.e. standards provided by a standardization body), and semantic models (and concepts) part of the capabilities of FEDeRATED.
  - Matching pairs – an indication of the pairs of matches that are available, e.g. semantic models ←→ open standards and semantic models ←→ industry standards (not necessarily a two-way matching).
- **Data sharing claim (profile)** – these are the capabilities claimed by an organisation with respect to data sharing. The structure of this claim is given in section 6.12. It is generated by the Service Development – and Customization module.
- **Competency claim** – this is the claim of a competent authority that supervises regulation(s) in an area. Besides details of the competent authority, the claim can consist of:
  - Regulation or authority decision – reference to the regulation or any decision made by an authority.
  - Competency area – the area where the competent authority is authorized to perform supervision.
  - Data capabilities – a reference to data requirements as published by the competent authority. This can be a reference to a single data set or a profile of a service identifying events required for data sharing.

A Service Development – and Matching claim are used by a verifier to assess trust in specifications. These specifications are open, public data, implying that a verifier does not have to hand over a VC. A data sharing claim is used to negotiate authorization as to which data can be shared and accessed. A Competency claim is handed over by a competent authority to access enterprise data. The competent authority requires a data sharing claim stating that the enterprise has implemented the data requirements of the applicable regulation.

Issuing these claims requires an issuing policy. Competency claims must be issued by governing authorities like ministries.

Service development claims are for a business activity -, service -, or event specifications (data structures). There are two aspects to issuing a claim, a soft aspect like assessing if a service developer can be trusted and another to validate correctness and completeness of the specification referred by the claim. These rules are given by the constraints of service development (section 6.13). They are to be implemented by a Service Development Module and can be validated once for that module, in which case the organisation operating the module can issue the claim itself (delegated

issuing).

Data sharing – and matching claims can be supported by conformance testing and/or certification (section 8.7). This assures the proper implementation of a profile and a match.

Authorization is given via the Index functionality. UUIDs are the basis for links to data, they are stored by the Index functionality. Having received this UUID is the authorization to additional data.

UUIDs can also be implemented as two-dimensional barcodes on the goods. In case a barcode is read by a competent authority and the authority requires access to additional data, it needs to retrieve a URL related to that barcode of the enterprise providing the barcode. If this URL or endpoint cannot be retrieved via the Index functionality, it must be provided otherwise, for instance as part of the barcode. Since VCs with claims must be applied (see before), only competent authorities are able to access data (or others if their claim can be matched).

In case a barcode has another identification, e.g. a GSIN (GS1 Identification Number), a resolver is required to retrieve the URL of the enterprise that printed the identification.

Whenever an enterprise receives a request to access data, this is only allowed for those data sets that reflect physical operations in the competency area of that authority. For example, a competent authority is only able to access consignment data according to the eFTI Regulation for consignments in or through its competence area (incoming and outgoing may also be required).

8.3 Access control

Access control assumes that IA is implemented, and a data user is authorized to access data (see the data flows section 9.2). It implies that an authenticated employee of a data user can be given authorized access to data of a data holder via Index functionality. There are various types of access control like policy -, attribute -, and role-based access control. These refer to claims made by an organisation.

A data user requires access to data of a data holder based on their relation:

- **Commercial** – access control is based on the business activity data set (section 6.3) and its Service Agreements made on profile matching (section 9.1.4). This refers to the data sharing claim of an enterprise. These access control data set can be implemented by a special openAPIs (GET Data implemented by an IT backend system, section 9.3.3.1).
- **Compliance** – access control is specified sharing events with links to data and/or by a data structure provided by a supervising authority implementing a regulation. This data set structure be the complete or a subset of the data set specified by a regulating body for that regulation.

There are extensions to these two assumptions (grey-scales):

- Seamless goods flow: an enterprise provides a supervising authority access to additional data risk assessment improvement with the aim to optimize physical inspections and improve seamless goods flow.
- Situational awareness: data is accessible (or can be analyzed applying Privacy Enhanced Technologies (PET)) to other organisations for optimization of (scarce) resource utilization. This is of interest in for instance traffic optimization and reduction of waiting times at hubs. Most often, a third party (algorithm) is used as a trusted service.
- Open data: data is available to anyone, I is not required. Open data can be implemented by specific Information Service APIs (section 9.3.3.6) or predefined SPARQL queries.

Access control for compliance, seamless goods flow, and open data is expressed by a SPARQL query formulated by a data user. For instance, a supervising authority formulates its data requirements on the semantic model, publishes it as a SPARQL query, and refers to the SPARQL query that needs to be executed when requesting data. This SPARQL query can be the implementation of an eFTI data subset.

A SPARQL query for compliance and seamless goods flow will always have an identification (for instance a UUID) that is provided by a data holder to a data user via an event. A container number and consignment reference number are examples of such identifications. As said, the competent authority has formulated and published the query and thus the required access is known. This is a type of ABAC.

A SPARQL query for open data can be more generic and not require a particular identification. In this case, the query is formulated by the data holder.

In a commercial relation, identifications to data that is accessible by a data user are shared by a data holder with that user in the context of a business transaction for a business service, i.e. a transport order. Thus, when a data user request for instance access to container details, the UUID (and container number) of that data is shared within a transport order for that container. There needs to be an agreement on which data will be accessible, i.e. the queries on Digital Twins or other types of objects need to be predefined in commercial relations. This can be on the level of a business service, for instance by formulating that a query on a Digital Twin like a container in a transport transaction always results in container size and type, tare weight, etc., namely all data relevant for executing a transport service of a container.

Any SPARQL query formulated on the semantic model can be transformed into another format, like XACML (XML Access Control Markup Language) or a JSON REST APIs for implementing an IS (section 9.3.3.6), e.g. for an eFTI IS. Organisations need to indicate the formats that must be supported and provide a SHACL for data validation as part of the Index functionality (section 9.1.4).

8.4 End-to-end application security

End-to-end application security is about access to data by an authorized employee of a data holder via its application to data managed by an application of a data user. It is on IAA, but it may also require end-to-end encryption and authentication between the two applications that share the data. The end-to-end security is independent of any intermediate functionality like a node.

End-to-end encryption seems currently not feasible. It requires key management procedures like that of public and secret (private) keys of users. SSI could provide this type of security. Otherwise, the mechanism addressed in the previous section and that presented in the next section are to be applied.

8.5 Link security

Link security is about establishing secure data between system components. In the context of a federated network of platforms, it is about secure links between a node of a data holder and that of a data user.

There are different aspects to link security, namely:

- the trust of a node sending/receiving data.
- data integrity of the data that is shared between both nodes.

There are different mechanisms to provide data integrity and trust. Data integrity can be addressed for instance by immutable, transparent availability of a hash of the data that is shared, for instance via a permissionless blockchain network. Asymetric encryption is another solution. Another way of creating trust is to register each participating node with a trusted registration authority. This requires extra functionality (and governance) which is not recommended.

Combinations of encryption and authentication of trusted nodes and data integrity is supported by protocols like Transport Link Security (TLS), using for instance eIDAS certified PKI certificates. This currently seems the safest way to implement link security between two nodes of the network. Https can also provide link security but does not involve trust in a client accessing a server. Therefore, https is considered more vulnerable.

8.6 Non-repudiation

Non-repudiation is the ability to proof that data has been send to or received from another organisation. Both a sender and recipient need to be able to provide such proof, where the proof must be immutable and identical. Non-repudiation is applicable to all data that is shared: linked event data and query/response data and especially:

1. Relevant in case of conflicts caused by for instance accidents or incidents or claims made by one organisation that activities are not performed in time.
2. Required as proof of compliance to regulation, e.g. access to data sets is timely shared with competent authorities.

The Index functionality must implement non-repudation.

An immutable **log** and **audit trail** are a means for implementing non-repudiation. A log contains all data that is shared: events, queries, links for accessing data, and data sets. An audit trail contains the timestamps for sharing (receiving or sending) and the peer (sender or recipient).

Additionally, an implementation of the Index functionality by a node or gateway could contain a log an audit trail of the openAPIs with the IT backend systems (section 9.3.3). This audit trail could contain the employee and its role initiating an openAPI via its IT backend system.

Immutability of a log and audit trail can be achieved by storing a hash on a permissionless public blockchain. Such a blockchain is a type of notary function that can provide an immutable proof of the truth in case of conflicts. Another type of immutability is achieved by encryption of a log and audit trail, where decryption is properly organized.

An immutable log and audit trail can also be provided by a third-party **notary service**, called a 'Clearing house' by IDSA. It is up to individual organisations to set up functionality for non-repudiation and to clearly specify when it is required. Many IT applications already have this type of functionality, implemented by open-source software components. It can be part of a node or gateway implementing the Index functionality. The current implementation of a node contains a notary network as part of the software used to implement the functionality (Corda).

8.7 Conformance testing and certification.

Conformance testing and/or certification may be required for issuing claims as indicated before. Certification is more rigid, some predefined tests must be passed successfully, whereas conformance testing is done by an organisation itself. Conformance – and certification testing for the Index functionality and their protocols is presented in this section.

8.7.1 Conformance versus certification

Conformance testing and certification are two separate coins of the same functionality:

- **Conformance testing** is a testing activity that assesses whether a product, service, or process complies with the requirements of a standard or specification.
- **Certification testing** is like conformance testing but for the most relevant requirements resulting in a certificate issued by an independent third party.

Such a certificate may be included in a claim, thus providing details about the certification authority. This can contribute to trust.

Where conformance testing is performed by the organisation that has implemented a standard or specification and is able to select test scripts and data sets, certification is driven by an independent certification authority providing predefined test scripts and data sets. During conformance testing, the certification scripts and data set may be used.

8.7.2 Test setup

Conformance – and certification testing is hereafter simply referred to 'testing'. A reference implementation of the Business Service Management module and an Index must be available, whereby the Index implementation contains a triple store with additional data.

The reference implementation is called the Test Application (TA), the implementation to be tested is called the Implementation Under Test (IUT).

The reference implementation also must provide a web based GUI with the following functionality:

- Test script and data set management – management of test scripts and example data sets that can be used for testing and are stored by the TA. In case of certification, only an employee of a certification authority is allowed to manage test scripts and data sets.
- Selection of test scripts and data sets – a user wanting to test an implementation can select a test script and optionally a data set. Not all tests require a data set: those tests whereby an IUT sends data itself.
- Inspection of test results – the results of running a test script are available. In case of successfully passing a test script for certification, this will be stored by TA. Running all test scripts for certification will be flagged to a user.

Since these GUIs are web based, each user can run the test scripts for their own IUT. No support is required of a certification authority.

The test scripts must be configurable for an IUT profile. Thus, data and events must be filtered to test a proper match and its implementation.

Testing is bottom up for each of the protocols (Figure 9-4), starting with the connectivity protocol. This implies separate test scripts for each protocol layer, whereby the lower layers must be tested before any upper layer can be tested.

Only those protocols can be tested that are implemented by an IUT. Thus, if claims with profiles are not yet supported, but configurations are matched manually, this does not require testing. Complexity of testing is reduced in case only data sets with given structures are shared. These are tested at presentation protocol layer.

An IUT must take a role in a protocol layer. The role of lower layer – and higher protocols differs, for instance the lower layers have the role 'sender'/'recipient' and the higher layers 'customer'/'service

provider' or 'operator/trader'/'competent authority'. This role will be given as part of the test scripts.

8.7.3 Connectivity protocol

It is the assumption that there are sufficient products for various connectivity protocols available and certified. Unless in pilot situations, the recommendation is to use these. Testing connectivity protocols will not be further detailed.

8.7.4 Presentation protocol

This is about testing an IUT on its capability to support RDF for events and data. The TA implements a SHACL validator implementing the required data structures. These are the data structures that are supported by the profile of an IUT, based on its related business activity and service(s).

An IUT can take two roles, according to the test scripts:

- **Sender** – data structures supported by the profile are shared by the IUT to the TA. These are only those data structures of the profile by which an IUT functions as sender. Three variants are tested for each data structure:
    - Minimal data set – repeat factors are set to one and only mandatory data is shared.
    - Maximal data set – optional and mandatory data is shared with high repeat factors. The repeat factors are either the maximal ones allowed or some high figure (like 99 repeats in case of 999 is allowed).
    - Some normal data sets – these are data sets that reflect normal operation as much as possible.

  Minimal – and maximal data sets may contain machine generated data, which can be based on operational data. If so, GDPR requirements must be adhered to (only virtual names of for instance truck drivers).
- **Recipient** – the IUT must be able to process several events and data sets that it is able to receive according to a profile. Like for the sender role it is about minimal –, maximal – and normal data sets. Additionally, the IUT can be tested on errors like:
    - Invalid concept – the data structure contains a data property of a concept that is not part of a data structure supported by the IUT.
    - Invalid value – a data property has an invalid value. These can be various like data type, exceeding maximal values, code values that are not supported, etc.
    - Repeat factors – mandatory data properties are missing or maximal repeats are exceeded.

  These types of errors can be based on the events and data sets formulated for services using concepts and properties that are not supported by a profile.

8.7.5 Node security protocol

This is about profile matching using claims of a VC. Verification of a VC must be tested separately and depends on protocols that are in progress. At this moment, only the profile matching is supported.

The IUT can take two roles, namely a customer initiating data sharing and a service provider responding to this initiative.

The following test scripts can be run:

- **Service support** – each profile has at least a service in common with the other profile. In its role of service provider, an IUT receives a profile and identifies the common parts of the

profile. In its role of customer, an IUT has issued a claim with a profile and receives a profile of the TA reflecting that of a service provider.
The following options are tested:
    - Complete match - the common services match completely, both on events that can be shared and their structure.
    - Partly event match – one of the profiles contains more events than the other. This will lead to a partly match or an error in case a mandatory event is not supported by the IUT or TA profile.
    - Partly data match – the data structure of one or more events or data sets of a profile do not match with those of the other profile. This type of match must be tested for an IUT and TA profile with a more extensive data set, whereby an error is generated if the TA profile contains mandatory data that is not supported by the IUT profile. The IUT profile must be amended. The other way, the TA profile must be amended.

  The test can be repeated for each service of an IUT and TA profile that match, eventually leading potentially to a Service Agreement.
- **No match** – there is no service match in the IUT and TA profile. In this case, two situations can be tested, for instance in the following way:
    - TA provides a match of services. Two test scripts: (1) the TA detects a match of its service to one of the IUT profile and proposes this to the IUT and (2) the TA does not discover a match. The first test results in a new profile matching using the matched services.
    - The TA does not provide a match. The IUT is tested on its capability to discover a service match. This may result in finding a service match and proceeding with profile matching. If no service match can be found by the IUT, the process must be aborted and no data is shared.

Successful profile matching results in a Service Agreement with data – and event structures that are a subset of those of an IUTs' profile. These can be tested again by running the presentation protocol tests for the Service Agreement.

8.7.6 Linked event protocol

The IUT takes two roles with a focus on testing authorization and access control. The assumption is that data structures of events and data sets are tested. The two roles are:

- Data holder. The IUT generates events and processes requests for or queries on data. The following cases can be tested:
    - Request for data of a link referred to by an event. This must result in a complete data set. The TA can take a role of enterprise and validating that business activity data is retrieved. The TA can take a role of competent authority and test if a required data set is retrieved, for instance an eFTI data set.
    - Request for data of a link of a Digital Twin referred to via an event. All data of this Digital Twin as specified by the business activity data structure is retrieved.
    - Request for data of a link that has not been received. This must lead to an error.
    - Submission of a query by the TA over several (transaction and/or visibility) events (with links) where the query has a filter. An example would be to retrieve all transaction events and – data of a particular shipper.

  Testing of queries with filters is on a sample basis. It must not be part of certification.
- Data user. The IUT receives events and generates requests or queries. This has already

been tested by the presentation protocol and does not need additional testing.

8.7.7 Business collaboration protocol

This is about testing the event logic, where it is based on a Service Agreement. In this protocol, the IUT has the role customer or service provider. The following test scripts can be run:

- **Event logic testing**. This is about testing the state transitions, whereby the event data that is shared is according to the start state. It start by synchronizing this start state between IUT and TA and from there on run the following test scripts:
  - Minimal sequence of events – the minimal path from start – to end state can be tested.
  - Maximal sequence of events – all events are shared at least once, whereby multiple instances of events that can be shared more than once, are shared.
  - Missing events – the TA does not share all events with the IUT. The IUT must detect a missing event. This test depends on the role in a service taken by the IUT.
  - Not able to process an event. The TA indicates that an event cannot be processed. An error is shared with the IUT leading to state synchronization triggered by the IUT.
  - Not supported event. The TA submits an event that is part of the service used for matching but not part of the Service Agreement.
- **Exception detection**. This is about sharing events with minimal data aimed to detect exceptions specified by the state transitions of a service. The following exceptions can for instance be tested:
  - Time – too late or too early. This is applicable to for instance load, unload, and ETA events. The TA generates events enabling an IUT to detect these exceptions.
  - Cargo – missing or damaged cargo, too much cargo. At unload, any missing cargo can be detected by an IUT. A test should also be run to detect if more cargo is unloaded than loaded.
  - Locations – the wrong locations are given for loading or unloading of cargo.

8.7.8 Business service discovery protocol

The IUT can take two roles, namely a customer role by posting a goal or a service provider role by providing business services matching a goal.

Validating goals posted by a an IUT can be with a SHACL validator as for the presentation protocol. A goal is represented as RDF with a data structure and the IUT acts as sender for SHACL validation.

When an IUT takes the role of service provider, goals are posted to that IUT by the TA. The IUT acts as recipient for validation of these goals according to the presentation protocol. The TA supports similar test scripts for testing the IUT.

8.8 Summarizing security requirements

With respect to security, the following minimal functionality must be implemented:

- All participants must implement the necessary security measures to meet the EC Cyber Security Act. They must use IAM Registries for authorization of their employees linked to the use of VCs or authorization tokens.
- Identity and Authentication has a long-term and short-term solution:
  - Long- term – this is about applying VCs embedded in public regulations (eIDAS2.0) for large scale applicability and adoption by competent authorities. Having VC with claims requires a testing environment for scalability and issuing these VCs.

    - Short term – use of authorization tokens based on federation of IAM Registries. This solution can be applied by relatively small communities operating as pilots or Living Labs.
- Implementation of claim-based authorization with claims for competent authorities and enterprise (their profiles). Claims for providing data like matches and specifications provide trust in completeness and correctness of these matches and specifications. Claims can only be made on links to data that are shared between two indexes.
- Access control is based on business transaction data structures limited to Service Agreements between two enterprises or data requirements formulated by a competent authority as referred to in its claim.
- Transport link security between data holders and – users must be implemented.
- Non-repudiation must be supported at least for compliance and preferably for commercial relations.

9 Index

This section presents a data sharing deployment solution specifies the functionality of what is called the index: the behavior that any two participants of the FEDeRATED should implement, independent of a solution and technology that has been chosen. The concept is that of a data 'pull' based on available 'links', supported by access control and Identity and Authentication (IA – security perspective), and Authorization based on shared links.

This section also presents implementation options. One of these options is a node that provides separate functionality for external data sharing and interoperability. Another is an adapter to existing systems, which may take some more time instead of implementing the required functionality separately.

A federated network of platforms is a set of interoperable indexes, where each index behaves according to agreed protocols and implements the semantic model and minimal required functionality.

First, the index functionality is introduced. Secondly, the underlying concepts for implementing the semantic model are discussed, leading into a decomposition of the concept of a 'node'. Multiple indexes form a network, interfacing with the other capabilities of participants in that network.

9.1 Index functionality

An index of an organisation is about sharing, processing, and storing events and access to data shared by these events. Events are shared between a data holder and – user, containing links with access to additional data. It is also about processing complex queries combining data from different data holders. Data and events are shared with RDF between two implementations of the Index functionality.

An index is a capability that has to be implemented according to the principles of the master plan. This section introduces a network of indexes, the index functionality itself, and the data sharing pattern it supports.

9.1.1 Federated network of Indexes

Each user must implement the Index functionality for its profile. Data is shared between these Indexes, in RDF format. Each Index is configured by a profile and registered with a VC containing the profile. The profile is specified by each user and a VC is issued by a Registration Authority according to an agreed issuing policy. A profile is based on services for business activities that are provided by Service Developers.

An **Index** has the following features:

- It provides the required **functionality** and **behavior** of an actor in its roles (data holder/-user)
- It interfaces with or is part of internal **IT system(s)** of an actor. There are different implementation options that will be discussed later in this section.
- By **registering** a Index, that Index is part of the network of all existing Indexes. It enables an actor to share events (booking -, order -, and visibility events) with all other Indexes.
- It is discovered via the Business Service Management module. This module enables business service discovery resulting in endpoints of indexes of potential business service providers.

The following figure visualizes this network with the various functionalities (or roles).

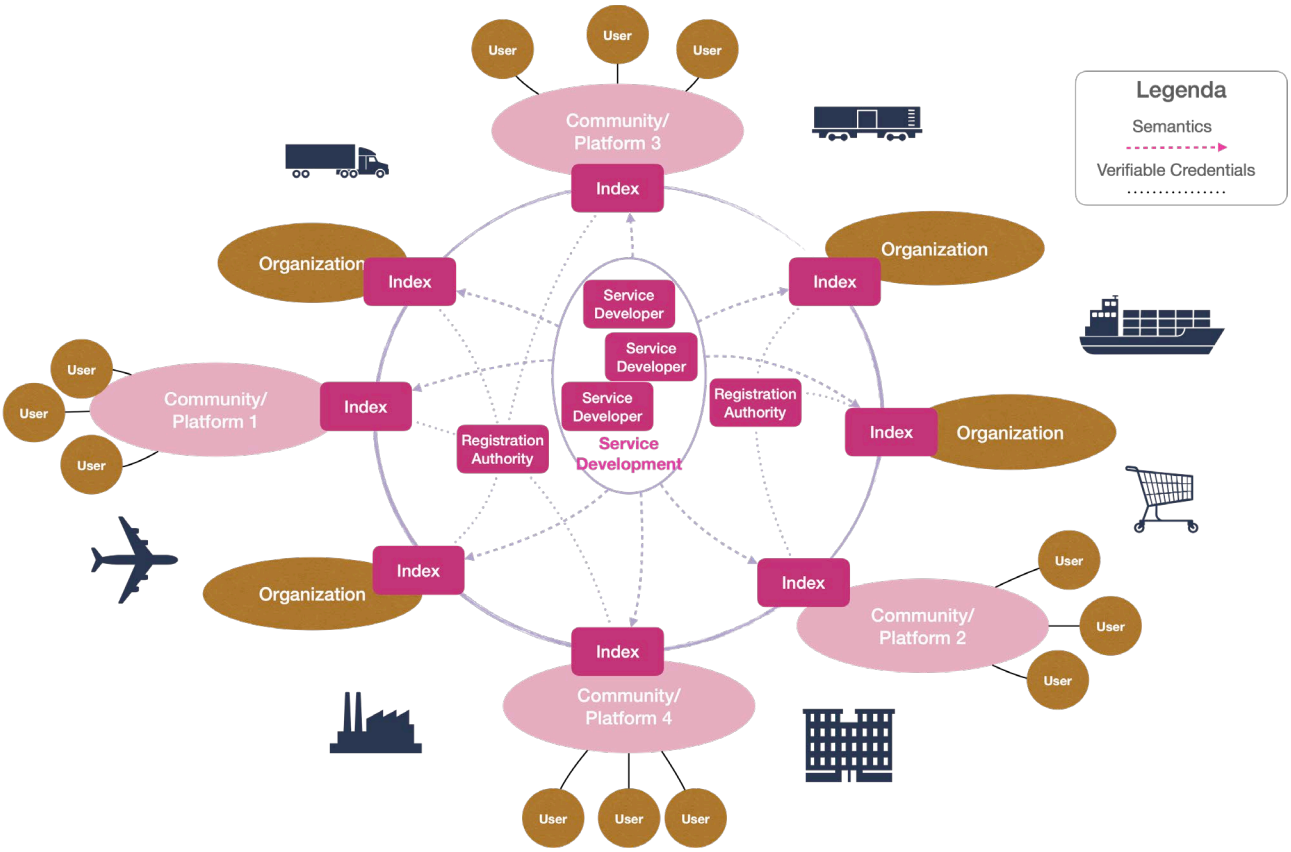


Figure 9-1: network of Indexes

Service Development by developers results is input to configuring the Index and constructing a profile. This is the flow of 'semantics'. The Indexes themselves are all able to share data with other Indexes.

9.1.2 Platform support of the functionality

The following implementation rules apply to platforms:

- **Index**. A platform can have a single Index for all its users that want to share data with others that either implement their own Index or make use of another platform that also implements an Index for its users. Those platform users that only want to share data with users of the same platform, don't have to use the Index.
- **Business Service Management** module. Those platform users that want to be discoverable, need to have a Business Service Management Module. A platform may operate this functionality for them.
- **Profiles**. The profiles of these platform users are based on the platform services and transformed by the platform. Each platform user can have its own profile.

As the general rule is that an Index is discovered by discoverability of business services, there are exceptions. The general rule is applicable to transactional relations. Other long term relations will have framework contracts based on human interaction. During contract negotiation, an Index endpoint can be shared. This rule is applicable to all users, thus also platform users. It may result in the absence of a Business Service Management module of a platform.

9.1.3 Data sharing pattern

The basic data sharing pattern is given by a sequence diagram (Figure 9-2). This figure visualizes three phases in data sharing:

- **Initiation**. This consists of two steps, namely business service discovery resulting in an Index endpoint and matching profiles. Profile matching results in a list of events that two Indexes can share, which is called 'Service Agreement'. Service Agreements can be stored by both Indexes, thus omitting profile matching for each transaction. Service Agreements may have to be updated once a profile changes.
- **Data sharing**. Sharing of events with links to data, where these links are accessible to a data user receiving them. Data access via a link is just sharing the link to a data holder. The data that is accessible is specified by the business activity profile of a data holder, that is based on the state data structure of a service (section 6.7).
- **Querying**. These are queries with filters on the data, not just data access via links. Such a query may have to be forwarded by a data holder to a third party (federated querying). An example of such a query is 'container track'.

The data sharing phase is always preceded by the initiation phase. Querying is only feasible if at least one link is shared between a data holder and – user.

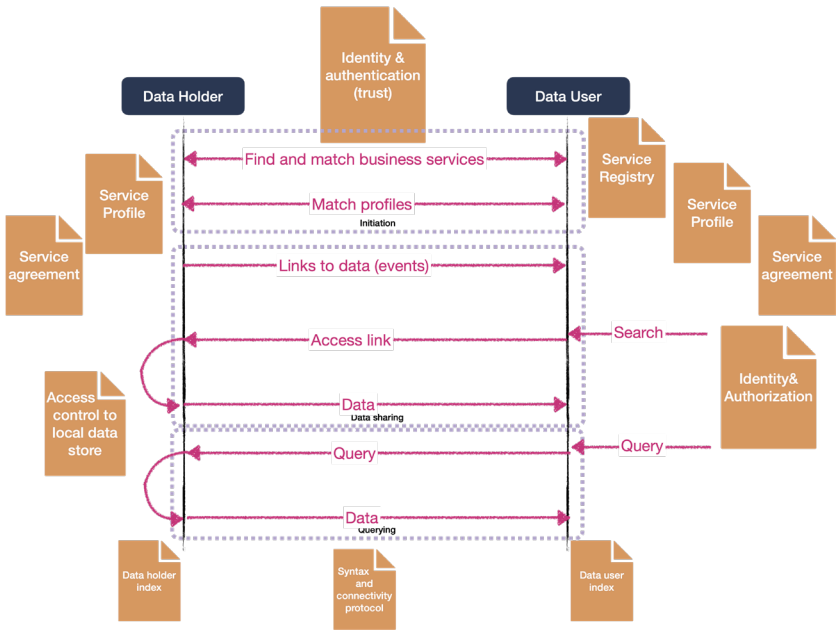


Figure 9-2: Data sharing pattern

Matching of profiles is via sharing VCs that can be verified. In case of a platform, each user with a profile must have its VC linking it to the Index of a platform. The latter one also must have a VC.

In case profile matching is replaced by Service Agreements that can be used for a longer period, those Service Agreements could be part of VCs of both participants issued by one of them to the other. They trust each other and can thus become an issuer. In that case, it is only about Identification and Authentication.

Each participant themselves must manage Service Agreements with others. Thus, if a Service Agreement is based on a profile that is changed, the Service Agreement may have to be replaced.

It can of course be simpler just to match profiles each time a transactional relation is established.

In case of a framework contract, the initiation phase is a bit different. It consists of two steps:

- **Profile matching** – automatically negotiate a profile as part of Identification and Authentication, where services leading to a contract can not be part of the matching.
- **State synchronisation** – the start state of both participating actors must be synchronised in

the Index. This is required for implementation of event logic. It is a type of 'publish/subscribe' mechanism where for instance a customer can subscribe to visibility events related to a transaction event with a service provider.

Whenever events with links to data have been received, complex queries considering various links of those events like 'give me the container track for container x' or 'give me the amount of cargo transhipped at a terminal for a particular recipient in the last month'. These (complex) queries are either predefined and published (design time queries), like those for regulations, or can be formulated by a data user at runtime. The eFTI data set can be implemented by sharing a transaction event with a link to an eCMR data set as a basis for providing the data (section 6.5.1). In case eFTI requires more details like only the cargo in a truck, visibility events with links to their transaction event have to be shared (section 6.5.2).

The figure shows in comments the various capabilities that are touched upon like the Service Registry with its business services. Internally, an organisation must have its Identity and Authorisation implemented to allow only for authorised access of links. Identity and Authentication of a data holder and – user is based on its VC.

9.1.4 Functionality of an Index

Two Indexes have the interaction pattern described before. The functionality of each Index can be grouped according to data – and processing aspects:

- **Profile matching and IA** – this is a function trigger by business service discovery. It includes Identification and Authentication (IA). VCs with a profile are stored in an organisational wallet, preferably a separate module interfacing with an Index.
  The input to this function is an endpoint of a related Index and a start state (which could refer to a contract or a business service that has been found), where the start state identifies the services of profiles that can be matched.
  Matching may not give a positive result, in the case the profiles don't have a service in common. The following strategy must be implemented:
  - Use a service match. There may be a match between the services of the profile (section 7.2.2). This service match can be applied in matching the profiles and leads to a Service Agreement.
  - Design time extension of a profile. One of the organisations involved extends its profile with the service in the profile of the other. This requires time before a new profile match can be made, since the profile also needs to be implemented and validated.

  The result of this function is either a requirement for additional configuration (at design time) or a Service Agreement with:
  - **Profiles** – a link to both profiles used as a basis for creating a Service Agreement.
  - **Common services and their events** – a list of common services and events per service that are applied during a business transaction. This can range from booking/quotation and ordering to visibility.
  - **Common data set** – this is the business activity data set that is common to both profiles. It is a type of match of their SHACL representations and the basis for data access for this Service Agreement.
- **Event logic** – validating the allowed sequence of events of a service, as implemented by a profile. Additional configuration is by a Service Agreement, thus enabling sharing only those events that are the result of profile matching.

Events are triggered by humans (transaction events) and/or sensors (visibility events). In case a human trigger an event, it is based on business service discovery or some type of long-term relationship with a framework contract. A sender/recipient pair of these events are known. Visibility events represent physical activities and are distributed according to existing transactional relations (i.e. an order exists).
Event logic supports B2B services and B2G ones. Event logic is configured by state synchronization (see previous pages), thus enabling automatic distribution of events from a service provider to a customer and to a supervising authority. State synchronization is thus a type of event distribution, that can be configured ones for compliance to regulations (design time).
In case event logic is not implemented, a publish/subscribe mechanism can be applied (see before) and **event distribution** rules can be implemented.

- **Query processing** – a query is on one or more links (UUIDs) with or without a filter. The UUIDs refer to for instance Digital Twins, locations, and transaction events. There are three basic functions implemented by query processing:
    - **Authorization** – only queries will be answered for links shared with others. Thus, the data store must contain an event with a link that is part of the query (simple or complex). This functions as a type of Policy Decision Point.
    - **Federation** – the query cannot be fully result in the requested output; another Index of which events with the same link are shared, must received a query for additional data. Federation requires a strategy, that is potentially complex.
    - **Response** – the proper output is generated as RDF according to the business activity data structure relevant to a profile or data requirements of an authority in the context of a regulation (e.g. an eFTI data set). A response may require combining data of a federated query with ones own data.
- **Data store** – a store containing all relevant data that are shared with other nodes with a SPARQL endpoint.
- **Data validation (SHACL validator)** – all data that is shared is validated. Validation is based on SHACL that is generated from a profile. This prevents receiving any events or data sets that are not supported. When sending, they are validated on completeness and correctness according to a profile.
- **Connectivity layer.** This is about actually sharing of data (events, links (for accessing data), queries, and query results) with other Indexes. Two functions must be available:
    - **Log and audit trail (non-repudiation)** – everything that is shared with or accessed by other Indexes is logged and an audit trail with timestamps and actions is constructed for implementing non-repudiation. A log or audit trail must interpret the data that is shared.
    - **Send/receive** – a agnostic protocol implementation for sharing data with other Indexes.

This functionality is visualized in the following figure, including its interfaces with other capabilities. The functionality is grouped in layers, where these layers implement specific protocols. Although not shown, the data presentation layer also interfaces directly with the processing layer.

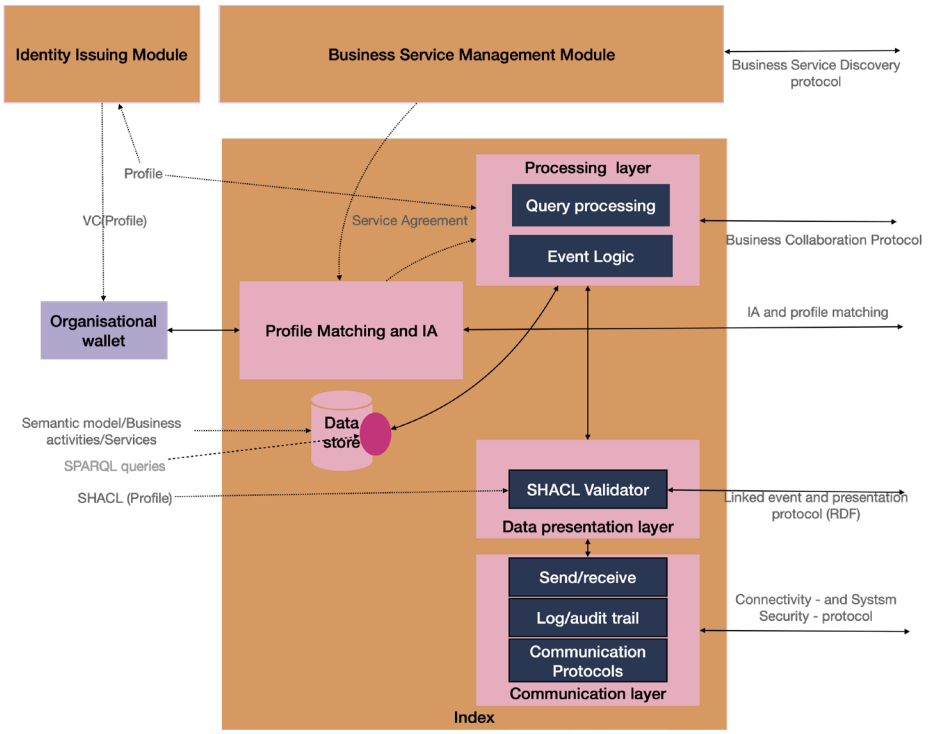


Figure 9-3: Components and interfaces of the Index

Conceptually, the data store contains all relevant data: state information, transaction events, visibility events, and all associations between transaction events relevant to chain composition and bundling/splitting of cargo. At implementation level, this conceptual data store maybe split into a data store with (transaction and visibility) events linking to data contained by IT back office systems of a user. Thus, the pink rectangles visualize the additional functionality required by IT legacy systems for implementing FEDeRATED, including of course a SPARQL endpoint to their data store.

9.1.5 Index protocols

The previous figure distinguishes several protocols. These can be visualized by various layers:

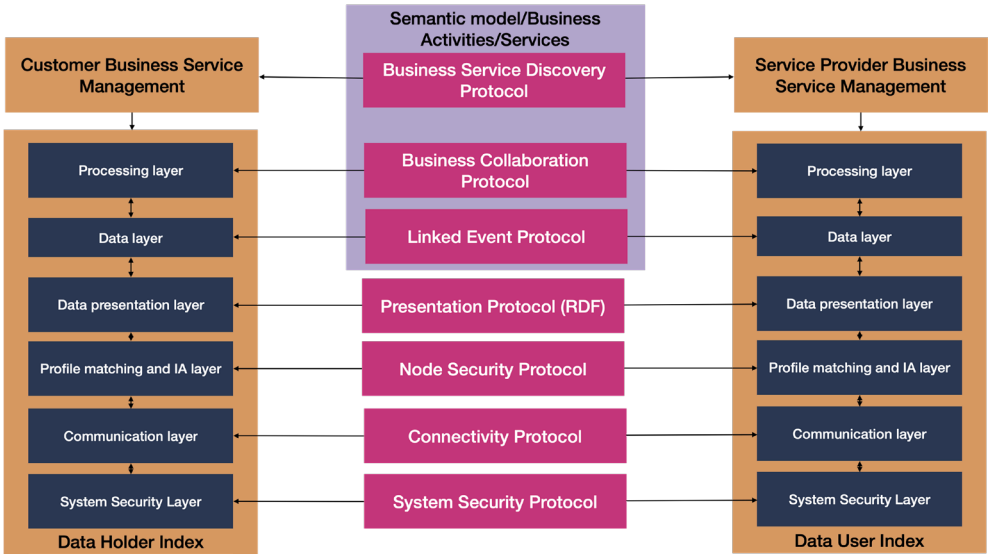


Figure 9-4: layering of protocols between two Indexes

The layering is not completely correct, since the Node Security Protocol, which is based on applying VC-based protocols with a profile for matching as performed by each Index, is independent of the

other layers. However, it requires a communication layer and does not require a specific data presentation layer (presentation protocol).

By distinguishing these protocols, different implementations of an Index can be developed, independent of each other. Furthermore, these protocols can be tested (and certified) by a conformance test module (see the previous section).

The protocols are defined as follows:

- **Business Service Discovery protocol** – this is about finding business service matching a goal. The goal is formulated by a customer, initiating the process and acting as data holder. Of course, a service provider is also a data holder, namely of its business services. This protocol is described in the section of the Service Registry.
- **Business collaboration protocol** – data sharing with events according to a Service Agreement between a customer and service provider (both can act as data holder and – user in this agreement). The Service Agreement stems from profile matching, where a customer and service provider profile are based on services. This also defines its data structures, i.e. the data that is accessible by both actors.
  An agreed business collaboration protocol is specific for two actors in their roles of customer and service provider and is thus not specified in detail. The only specification that can be given is by the rules for generating that are implement by the Service Customization module and the profile matching function. All data is shared by events.
  Since a business collaboration protocol is flexible, only two types of interactions are required:
  - **State synchronization** – the states of two implementations of the event logic may be out of sync. This interaction allows them to sync. The complete state data between two implementations is shared (see section 6.7 for the state data structure).
  - **Service Agreement synchronization** – any service agreement as the result of profile matching is shared. This interaction is also applied as the result of profile matching. Its purpose is to assure that a common data set is accessible by sharing the common events.
- **Linked Event protocol (pull)** – sharing of links based on logistics events. The links are the basis for accessing additional data (e.g. eFTI data set) and (SPARQL) queries with filters, for instance to access details of a container or a business document data set.
  This layer implements the events (transaction – and visibility events) of a service for a business activity. Since these are specified by the Service Development Module, they are also flexible.
- **Node security protocol** – a protocol for authentication of (verifying) the identity of nodes and matching the profiles that are part of a VC of each node. This protocol is based on VC verification protocols. It could potentially be supported by eIDAS2.0 – Architectural Reference Framework (ARF) or a similar private solution. This protocol can be specified including rules of adding a profile to a VC. Profiles, and thus claims of a VC, are different but have common concepts like 'business activity' and 'service'. After protocol matching, the Service Agreement Synchronization interaction(s) is (are) applied.
- **Presentation protocol(s)** – the syntax and technology used for sharing data. Standard, Indexes share RDF and SPARQL.
- **Connectivity protocol(s)** – the technical capability for reliable data sharing between two implementations of the protocol stack, over a System Security Protocol. These are existing protocols like eSens eDelivery and the connectivity protocols underlying the Data Space

Protocol (under development).

- **System Security protocol(s)** – the safe and secure sharing of data, utilizing standard protocols (e.g. https, TLS).

9.1.6 Robustness

Various types of issues may occur that must be handled by Index functionality. These issues are shared between data holder and – user. It leads to extension of the protocols with the following primitives:

- Receipt acknowledgement – data has been successfully shared. This is mostly implemented by a connectivity protocol.
- Validation error – an error at SHACL validation of an incoming event or data set. This must be part of the presentation protocol.
- Processing error – events have not been processed according to the agreed event logic.

The connectivity protocols may implement resending data since the receiving index is offline (not available). A profile of an organisation must be extended with indicators like MTBF (Mean Time Between Failure – the average time that a recipient is unavailable) and maximal number of resends.

Event logic may have to be supported for receiving events out of order, e.g. an unload is received before an ETA. This can be due to resending of events (data) by the communication layer. It can also be due to detection of an error at data validation, which must be signalled to as validation error to the sender.

Whenever an event cannot be processed by a recipient in due time, an error is signalled to a sender. This takes the assumption that whenever an event is processed successfully, this is not shared. Sharing a processing error must be solved by synchronizing the state data of sender and recipient. It may lead to omission of the event that could not be processed, since its data may already be part of the state data of the sender of that event.

9.1.7 Minimal and maximal functionality

As argued with the other capabilities, the total functionality does not have to be implemented (at once) by two stakeholders. A distinction between minimal – and maximal functionality is made:

- The **minimal** functionality is to share (visibility) events with optional links to additional data. This is only about progress validating the quality of event data. The Communication Layer functionality, data validation, a query processing mechanism with access policy evaluation based on links shared by events, and a data store with a SPARQL endpoint are implemented. Node security can be based on authentication tokens. There is no profile matching, the assumption is that any two Indexes sharing data implement the same service. This only works if they agree upfront. It requires additional functionality of the Business Collaboration Protocol to synchronize states (e.g. using a pub/sub mechanism).
- The **maximum** functionality is basically to support all identified functionality of an Index and any additional functionality that does not affect the identified protocols. An example is to include ‘event federation’ whereby received (visibility) events of service providers automatically generate customer – or authority (visibility) events or provide order updates to other service providers due to delays (this functionality is illustrated by the Multimodal Supply Chain Visibility Service).

9.2 Data flows

This section explains the data flows for sharing events and accessing data, applying the functionality identified in the previous parts. It specifies the initial flow, event sharing, and data retrieval.

9.2.1 Initial flow

The initial flow is about verifying the identity of two Indexes and matching a profile. Thus, also Service Agreement Synchronization is specified.

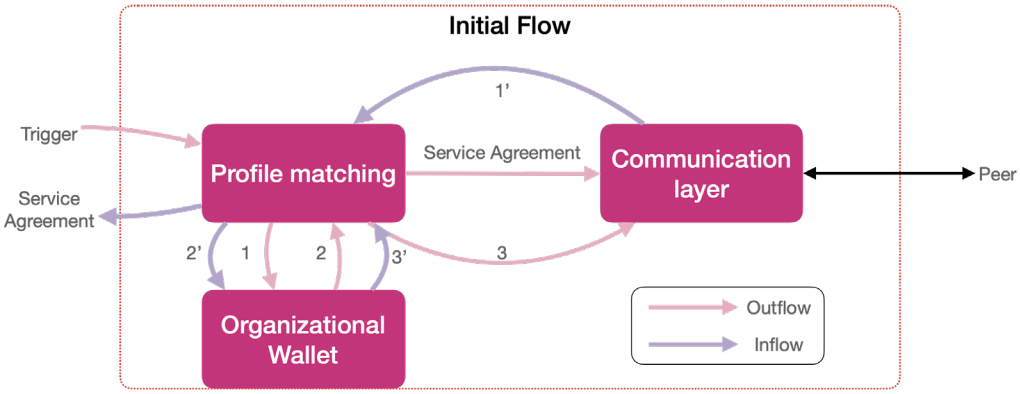


Figure 9-5: initial flow between two Indexes

Any two indexes can be triggered at the same time. This means that a peer can share a request for matching and identity verification at the same time the functionality is triggered. A trigger can for instance be manual or the result of business service discovery.

When a trigger is given, a VC with a profile is obtained from the organisational wallet and shared with a peer (steps 1, 2, and 3). At any time, a similar request for verification and matching is received from a peer (step 1'). The identity must be verified (step 2' and 3'). When both profiles are shared, profiles can be matched resulting in a Service Agreement. This is shared with a peer and if both Indexes approve on using a Service Agreement, it is passed on to the event logic.

9.2.2 Event sharing

Events are shared via a trigger or received from a peer. These are the two flows shown in the next figure. The figure shows that a Service Agreement is stored in the Data store. It must be associated with a transaction event (in its state) to select the applicable event logic.

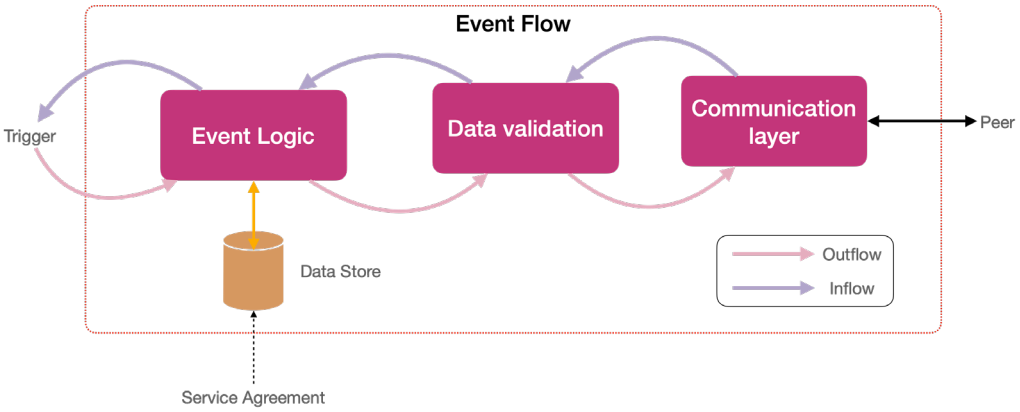


Figure 9-6: sharing events

A **trigger** can be the reception of another event or some internal trigger generated by a system. The trigger must contain the identifier (UUID) of the applicable transaction event. The event that is to be

shared is based on the state and next transition(s) identified by event logic. In case multiple transitions are feasible, the applicable one is selected by the pre-conditions (only one pre-condition is true, thus identifying the relevant transition and output event).

After generating the event, it is validated on its completeness and correctness, applying the relevant SHACL for the event. Next, it is shared via the communication layer.

At **reception** of an event, it is first validated with its SHACL and secondly with the applicable event logic of the Service Agreement. If correct, the event updates the state of a transaction and a trigger is generated to signal a new event (or an exception).

9.2.3 Data retrieval

Data retrieval is basically about data access by a data user to data of a data holder. The following cases are feasible:

1. Data at the data holder – the data store of a data holder contains all relevant data.
2. Data not stored by the holder – the query needs to be forwarded to a data holder known to the one that received the query.
3. Data partly stored by the data holder – the data holder must decompose a query into others that are shared with its peers.

These cases will be presented, starting with the **first case**. The flow is shown in the next figure, where a trigger initiates the process. A trigger can be a human pushing a link at its GUI. The trigger thus contains a link for which additional data is required. First, authorisation is validated and, secondly, 'response' stores that a query is formulated. The outgoing query is validated and shared with the peer selected from the data store.

When a response is received, it is validated against the expected data structure (response type), which is either contained by the common data structure of a Service Agreement or a data set formulated by a supervising body. The response can be stored by the data store and accessible to a human. In view of the principle 'data at the source' the response data is not stored.

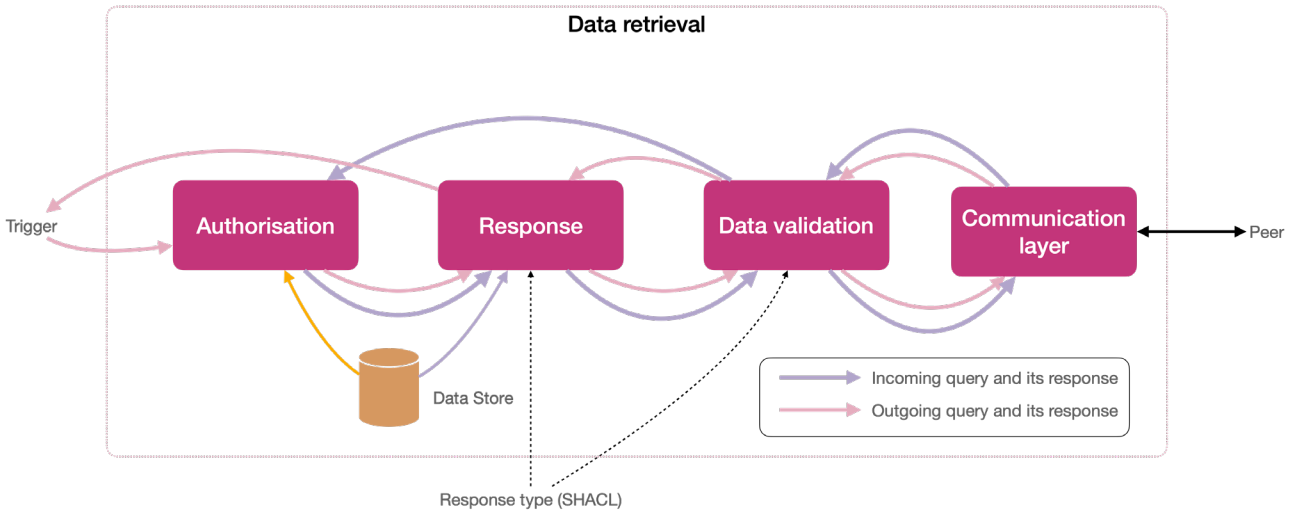


Figure 9-7: sharing events

At reception of a query and after its validation, it needs to be authorised. The link contained by the query must be validated by retrieving an event with that link that has been shared with the data user. If the authorisation is valid, a response is composed based on a data structure, the response type. This data structure can be a part of or the complete transaction event but can also be a data set

requested by an authority. In the first case, only the data is shared by the response that has been agreed in the Service Agreement. In the second case, it is a data set formulated by a supervising body based on a regulation.

This same data set is applied for data validation of the outgoing response.

The **second case** is a simple extension, whereby the Authorisation functionality of an incoming query produces a trigger for reception of a response. In this case, the incoming query must be stored to assure its timely response.

The third case is more complex than the previous one. A response from in incoming query may be retrieved from multiple data holders, with a dependency between the responses. This requires query decomposition and a query federation strategy. The 'federation' functionality is intended to support this functionality. It must be part of the flow of processing an incoming query. This flow and its implementation require additional research and prototyping.

9.3 Implementation of the Index

The four variants for implementation of the Index functionality are:

- Node – all functionality is implemented by a node.
- Gateway – closer integration of the index functionality with an internal IT system.
- Adapter – the index functionality is implemented by the IT systems of an organisation.
- Platform – functionality is implemented by platform provider for multiple organisations.

A node and a gateway support openAPIs to internal IT systems. Since a node or gateway can be developed as standardised products according to the Reference Architecture, these will have to be tested only once. One may also consider having them tested for various profile variants to assure that they will always operate for an organisation.

Whenever an organisation decides to implement a node or gateway, its openAPIs with that node and gateway must support the services implemented by its profile. This allows the processing of any query in the context of that service implementation. An adapter is assumed to support these queries.

A node or gateway solution that shares data with other Index implementations according to the protocols, acts as a type of API gateway: it decouples openAPIs of a data holder from those of a data user. This allows for large scale data sharing: the protocols assure that data can be shared between any two implementations of the Index.

Any queries that include data of two or more services must be specified at development, enabling an organisation with a node or gateway implementation to support these queries. The latter is only required if that organisation implements those services; otherwise, part of the query can be federated to another data holder via the node or gateway (federated querying).

9.3.1 Implementation aspects

This section briefly touches on some implementation aspects.

For migration purposes, an implementation of an Index may need to support **open standards**. These can be open standards or their implementation guides developed by a community (or data space). Data transformation with these open standards are developed via the Matching Module.

Another implementation aspect is that of **uniqueness of physical objects** in a federated network of platforms. Each physical object exists once in the real-world and should so in the virtual (IT) world. Physical objects are represented by concepts in the virtual world. These concepts have unique

identifiers, UUIDs, generated by a data holder that adds the concept to the virtual world. Since more than one data holder can create a virtual world concept of the same physical object, that physical object can exist more than once in the virtual world. This should be avoided. A means to avoid it is by storing identifiers of real-world objects with their UUID on a public ledger. Note that this may not be required for volatile physical objects like 'packages' that only exist during transport.

A **UUID** is just an **identifier**, it has no meaning. It is not a classifier. A UUID cannot linked to the organisation generating it. It is not a URL (Uniform Resource Locator). Indexes share these UUIDs, based on events. Via these events, sender and recipient are identified, thus linking a UUID to a data holder. As such, an Index acts as a type of **resolver**: it associates a UUID to an endpoint of a user based on events that are shared.

In case of information services, no events are shared implying that UUIDs are stored only by a data holder (data at the source) whereas others may have access to identifiers of physical objects (e.g. license plates of trucks). In those cases, there is an (delegated) authority acting as a resolver. If these information services are of open data, they can simply contain the URL where data can be found, for instance by scanning a two-dimensional barcode. In case of bankruptcy, the data may have to be accessible based on some type of regulation. This relates to archiving.

Another aspect is that of **archiving** and **historic data**. Where logistics is about business service provision, archiving is on actual data that is shared and made accessible in a commercial – or compliance relation. Archives contain historic data that can be relevant for future decision making. Historic data is input to deep learning, i.e. the application of Large Language Models (LLM). Supervising bodies need to store the data resulting in inspection, which means they must duplicate the data of a data holder.

Archiving laws are applicable to data of individual organisations, whereas they may have only links to data stored by a data holder. Mechanisms must be implemented by which a data holder and – user share the result of a business transaction, i.e. the end state of a business activity. By sharing these end states, LLMs can also be developed.

Data holders and – users may also agree to use an **archiving service**, which can be a cloud-based service provided by for instance an independent auditor. Such a service becomes relevant in case data must be accessible after for instance bankruptcy or a re-organisation, where especially product data (like its composition, its state of health, etc.) must be accessible. The archiving service acts as a type of resolver that also stores relevant data. This is applicable for the assets of a service provider like its vessels, barges, and trucks.

9.3.2 Implementation variants

9.3.2.1 Node

This variant is especially useful during piloting, for SMEs without any IT functionality, and other organisations that are not able to support the functionality in their internal IT systems. A node will be more detailed later in this section. For piloting and to suit SMEs, the node variant must also have a GUI.

This implementation variant requires additional functionality, namely:

- Semantic adapter – transformation of internal (JSON) data into RDF, where the JSON structure reflects a service profile.
- Service registry – an additional module to generate (configurations of) the semantic adapter.

- JSON enrichment – inclusion of UUIDs in outgoing event flows and matching these UUIDs to internal IDs of data.

Since querying existing IT systems with SPARQL may be (too) complex, data could temporarily be stored by a node in a triple store or graph database. This may also be done for handling federated queries: data of a data holder is temporarily stored by a data user and can be made available to another data user upon request.

Such a data duplication must be avoided since business operation is handled by employees with their existing IT systems.

9.3.2.2 Gateway

A gateway is an extension and an alternation of a node. There are two variants of a gateway, namely the variant where event logic is implemented by internal IT systems or by the gateway.

The extension of a gateway is its support of (open)APIs of internal IT systems and their transformation to (standard) openAPIs of a node. This requires a more complex semantic adapter and a matching module for mapping internal data structures to the ontologies of a service profile. It also requires handling internal identifications and UUIDs applied for data sharing.

It can also require additional functionality for instance filtering data retrieved from an openAPI or combining data retrieved by two or more openAPIs. This additional functionality must be developed for individual organisations; it may become part of a gateway solution in the future.

9.3.2.3 Adapter

All functionality is implemented as an adapter by the internal IT system(s) of an organisation. It is up to the software developer, i.e. the Commercial Off The Shelve (COTS) software – or service provider or the IT developer of a proprietary system to implement the functionality.

9.3.2.4 Platform

A platform provides its services to multiple users and can federated with other platforms or nodes on behalf of all or part of its users. Platforms are governed by a community of end-users or operate commercially. Semantic concepts for platform services may differ from the data sharing ontology, but services that are federated must be based on the data sharing ontology. A platform must have:

- A profile and Verified Credential (VC) for each federated service. This VC and profile is used for those users of the platform that implement that profile.
- An implementation of the index functionality by any of the three other implementation options, and
- A business service registry with business services of those platform users that support a particular profile for discoverability of these users.

Platform providers must define roaming agreements with other platforms and distributed implementations by end-users (node, gateway, or adapter).

9.3.3 openAPIs with IT backend systems

When using a node or gateway variant, openAPIs are one of the ways to implement the interface with internal IT backend systems. IndexAPIs will be detailed in section 9.3.3.1.

None the less, end-user or node/gateway provider may want to support other openAPIs. Of course, IT systems can have their own openAPIs. To do so, these open APIs must be mapped to the openAPIs of the infrastructure provision, resulting in (data) transformations.

OpenAPIs of the infrastructure are generated and must be configured by the Service Registry. IndexAPIs provide the generic structure, they are either configured by ServiceAPIs supporting a service, or ProfileAPIs with service customization. An organisation can also decide to implement ServiceAPIs or ProfileAPIs, where the latter ones optimal fit the organisational capabilities. Additionally, Infrastructure Support – and Information Services are supported.

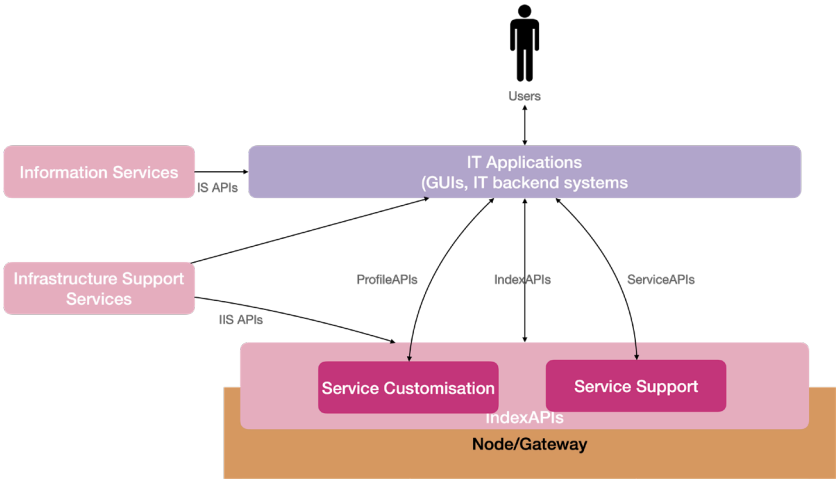


Figure 9-8: the various types of openAPIs

The previous figure shows that IndexAPIs and the ones for Service Support and – Customization are implemented by a node or gateway. The Infrastructure Support Services are external openAPIs provided as a service by some provider. Information Services are either public or restricted data services and discoverable via Service Registries. These can be implemented as data broker (IDSA Reference Architecture) and/or the control plane of the Eclipse Data Space Components. The functionality for providing these APIs is out of the FEDeRATED scope, but they will be briefly described hereafter.

Any openAPI contains code for data validation. Implementation of any of these three sets of openAPIs is based on the ability to configure those openAPIs with SHACL and RML from the Service Registry for data validation and - transformation.

Each organisation must configure **authorization** of their employees for these openAPIs. Authorization can role based in relation to a service. For instance, only employees with a specific authorization level ('procurement' role) can use POST OrderEvent for sharing access to order data with a service provider. This differs per organisation.

9.3.3.1 IndexAPIs

This is an implementation of the local interface between a node or gateway implementing Index functionality and an internal IT system of an end-user with openAPIs. These openAPIs are configurable for data transformation, data validation, and event logic by the Service Registry and can be applied for implementing any service or profile. They are agnostic of the type of events that are shared, they can be used to share and access all types of events and data.

Configurations of the IndexAPI can be provided by Service Developers (i.e. services) or end-users for Service Customization (i.e. profiles).

The Index APIs consist of three sets either provided by an Index or required for the integration of an Index with an IT backed system. One column shows the openAPIs supported by a node and the other those that must be implemented by IT backend systems.

Index APIs	Description	Node APIs	Backend APIs
eventAPIs	A set of openAPIs to share events and access data of links shared by these events.	POST Event GET Event GET Data GET State POST State	POST Event GET Data
serviceAPIs	A set of openAPIs supporting the implementation of a service by an Index (i.e. event logic)	GET State POST State GET Exception POST Service Agreement	POST Exception POST Complete
monitoring APIs	A set of openAPIs for monitoring the configurations and behavior of an Index	Examples: GET States GET Services GET Profiles GET Service Agreements	

The **eventAPIs** are APIs like POST Event and GET Event for publishing and retrieving events by a data holder and a GET Data for retrieving data from a node by a data user.

An IT backend system must support a POST Event API to receive events from the Index functionality and a GET Data API to retrieve data of a link. The GET Data API required by an IT backend system can only consider data of links that are shared by posting events, as specified by transaction events (section 6.5.1) and their support by a profile (or Service Agreement). Internal IT backend systems must support this.

The GET – and POST State APIs refer to the initial state required for event sharing. This is required in case a Service Agreement between a customer and service provider starts from a state where data has already been shared otherwise, for instance referring to a framework contract for an order or to an order in case of visibility.

These openAPIs a means of subscription for event distribution and its synchronisation of these subscriptions. Both customer and service provider can retrieve the initial state by a GET State. A customer can share this state with its service provider by calling the POST State API, thus subscribing to events. An enterprise can also configure this state as a subscription of a supervising body to events. The data structure of this initial state is given in chapter **Error! Reference source not found.**, since a customer (or supervising body) must have a subscription to Digital Twins and events for its order.

Event sharing APIs can be extended by webhook APIs signalling for instance that a new event is available.

The **serviceAPIs** are GET States, GET – and POST State, APIs for exceptions, and an API for

synchronising Service Agreements (POST Service Agreement) to be used in combination with a monitoring API. For accessing proper states of event logic, the GET States provides an overview of the states that are configured. POST State has two variants, namely, to change the state in one's own implementation and to synchronize it with a peer end-user for a service instance. The POST State for synchronization is a type of subscription. This might be required for synchronisation of transportation legs, e.g. by informing the service provider of a next leg of earlier or later arrival.

Exceptions are signalled by event logic like an arrival is too late. These exceptions are provided by a node or gateway to an IT backend system. That system must have a POST Exception. An implementation of the Index functionality can also have a GET Exception API to retrieve exceptions. This can also be part of the monitoring APIs.

An optional API is the signalling of service completion. An IT backend system must implement this API (POST Complete) that can be called by the Index functionality. It can be used to trigger internal business processes like invoicing or payment of a service.

The **monitoringAPIs** are the monitoring of the data sharing – and event APIs, the configurations, and accessing the log and audit trail. The GET States API is for instance an API for retrieving states; another is the GET Services to retrieve the list of services that are implemented and GET Profiles to retrieve the profiles of services. The monitor APIs to access the log and audit trail has variants, for instance to monitor data shared in a period, with a customer or service provider, combinations, or any other subset of the log and audit trail. Monitoring APIs of a log and audit trail provide information for payment of business – or (paid) Information services.

In addition to these openAPIs, another set of **archivingAPIs** can be included, for instance GET FinalStateData provided by a node by which all data is retrieved and can be stored in an archive. This example API will retrieve all data stored by a customer and service provider for a business transaction, where a business transaction is an instance of a business activity and its services.

9.3.3.2 ServiceAPIs

ServiceAPIs are openAPIs for a service. ServiceAPIs are generated by Service Developers. They are configurable for a profile of an end-user for its data transformation, data validation, and event logic. The ServiceAPIs encapsulate ProfileAPIs. The functionality of the ServiceAPIs is identical to those of the IndexAPIs, e.g. POST -/GET Event and others, but configured for a service.

In addition to the event logic APIs of the IndexAPIs, there is a GET Service states to retrieve the applicable service states.

The baseline of the Service APIs for data sharing in terms of GET and POST operations is identical to those of the Index APIs, but they reflect the event type they support. To support sharing ETA events this will give for instance a POST ETA and GET ETA supported by a node and gateway of a data holder and a POST ETA event with that of a data user. Furthermore, the data retrieval can be specific to that type of data that requires to be shared, for instance GET eFTI to retrieve the eFTI data set.

Implementing Service APIs results in (potentially) many openAPIs that must be tested and implemented by a node/gateway and IT backend system(s) of an end-user. An end-user may also require a single set of Service APIs that support various services with overlapping functionality.

9.3.3.3 Profile APIs

Profile APIs are openAPIs that support the implementation of a service by an end-user, based on its

profile. The set of openAPIs is identical to those of the Service APIs, but only support a profile of a service.

9.3.3.4 Query API

The query API supports queries of data of multiple services. These are queries like 'container track' or 'trip'. These queries are posed via a SPARQL endpoint of a node or gateway. A node or gateway can also pose this query to an IT backend system, which thus must support this query.

Since openAPI implementation by IT backend systems takes time (and costs), it is recommended that a community specifies and publishes several queries at runtime, based on the multimodal ontology.

9.3.3.5 Infrastructure Support Service (ISS) APIs

ISS APIs are computational services that can be embedded in the other services. These services are provided by third parties. There are two types of services, namely those to support:

- Data sharing. An example is a data transformation API. Data sharing ISS APIs are based on metadata, e.g. the business function and the input-/output semantics and syntax required for data transformation.
- Logistics operation Examples are an ETA calculation – and a chain composition API. A route planning API is also an example of a logistics ISS API, that can also be used for roads and person mobility. These ISS APIs must relate to logistics concepts and properties of the multimodal ontology to make them widely applicable. It is however up to a provider of this type of service to specify it.

9.3.3.6 Information Service (IS) APIs

IS services provide access to data to improve decision making and validation of operation. These are for instance:

- openAPIs to access traffic information, to share capacity, etc, or
- IS APIs for publishing accidents that might be accessed for validating delays that cause late delivery.

IS Services can have a classification, like public or restricted. Traffic information is for instance a public service. Restrictions can be formulated in for instance the Open Document Rights Langauge (ODRL). Restrictions can have classifications like:

- Public services – IS that are available to anyone.
- Paid services – public IS available under a price and payment conditions.
- Business activity services – IS available to all organisations utilizing or providing a particular business activity. This can be based on a profile as contained by a VC.
- Community services – IS available to members of a community. This can be an ad hoc community for capacity sharing or a port community whereby access to water depths is provided. Another example is the sharing of water depths as river information services for inland waterways.
- Transaction services – IS available in case there is a commercial, transactional relation.
- Internal services – IS only available within an organisation.

These classifications come with a claim of a data user that would like to access them, for instance a VC with its membership of a community. Any combination is feasible, e.g. paid services offered to organisations for a business activity in a community like a port.

It is up to a data user to identify the applicability of these IS APIs. The data holder has to specify and publish the IS APIs with the applicable conditions. For interoperability, these IS APIs must use the multimodal ontology. Their specification may result in specialisations (lower ontologies).

9.4 Implementation by a node

A node acts as a type of gateway for data sharing by an actor with all other actors, where the node contains functions as Index and Identity and Authentication. The data store of an Index is decomposed into two parts: the actual data stored by a database of an existing IT system and a triple store containing state data (for event logic) with links to data. The latter implies a type of mapping between links applied for data sharing and links used by IT systems.

One may choose to duplicate the database of an internal IT system to a temporary data store of a node. This temporary data store contains triples (RDF data). Performance is one of the criteria to do so, another is to avoid (potential complex) changes to an internal IT system. A drawback of such a temporary data store is that it can be out of sync with the IT systems' database. A polling mechanism must be implemented to avoid the latter issue.

This section introduces the functionality of a node and its openAPIs to internal IT systems that have to be integrated with a node. A node always shares data with an implementation of another node according to the protocols identified earlier.

9.4.1 Node functionality

Since existing IT systems have their own data(base) – and identification schemes, transformations must be supported. A communication – and data presentation layer is required:

- **Data presentation** layer. Since any two indexes share RDF (event) data with UUIDs, the following functionality is required for integrating with IT systems:
    - **Data transformation** – matching between data structures of IT systems and those specified by a profile. This data transformation is configured by the Matching Module (see the section of the Service Registry).
    - **Link management** – generating UUIDs for concepts and linking them to identifiers used in IT systems. These link mappings are not always required, since also user references (container numbers, consignment identifiers, etc.) are shared and can be used to access data in an internal IT system. Link management is at runtime level and does not require additional configuration.
    - **SHACL validation** – validating completeness and correctness of the data received from an internal IT system. SHACL validation is configured by a profile of an organisation.
- **Graphical User Interface (GUI)** – support of a user in formulating queries, data visualization, and business transaction progress. Business transaction progress could contain a function for visualizing visibility events by using for instance so-called 'metro maps'.
  Preferably, a GUI can be applied for any type of semantic data. There are different visualization frameworks supporting semantic data, for instance Obsidian for visualization of graph structures like stored by Neo4J and svelte.dev.

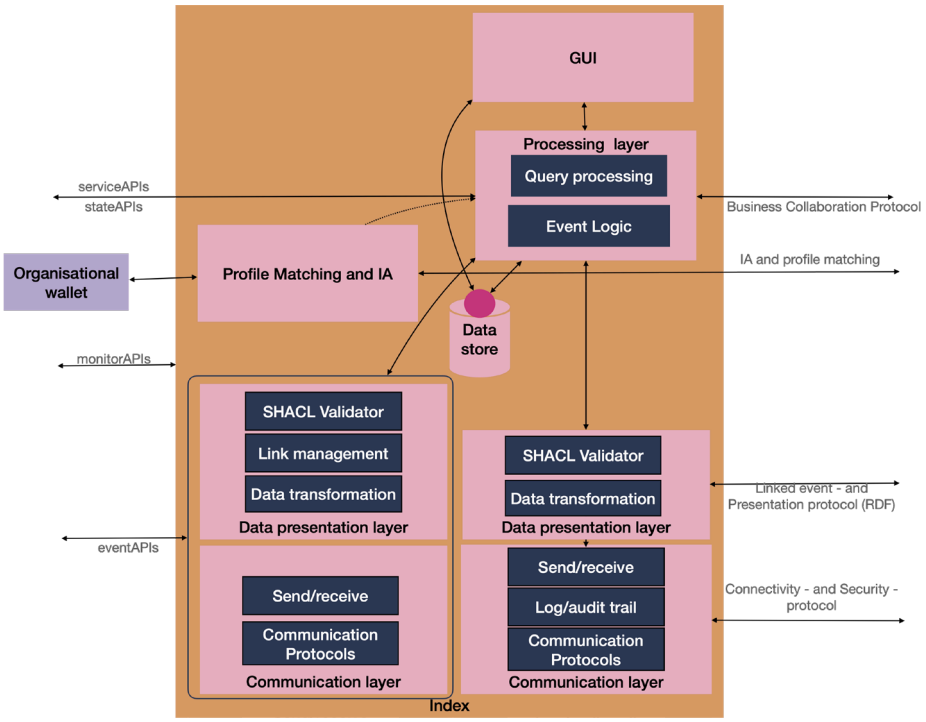


Figure 9-9: layered functionality of a node

The previous figure identifies several APIs for interfacing between a node and an internal IT system. These will be specified hereafter.

The figure shows two SHACL validators, one at the interface with an IT system and another for sharing event data with peers according to the protocols. Both are required, the one at the internal interface for validating data exchange with existing IT systems and the other for validating data shared with other implementations of an Index.

The implementation of data transformation is similar. Data transformation with peers is only required in case of those peers that implement a (community/data space) standard or a different service in which case service matching is required (section 9.1.4).

9.4.2 Implementation approach

A proposed implementation approach relates to development of the functionality of the Service Registry, where development of a tree structure editor generating RML and SHACL is proposed as first step. This configures the eventAPIs of a node, meaning that the processing layer is not implemented in a first variant.

Configuration of these eventAPIs can also be in steps, like:

- Link management – two versions like the generation of UUIDs for events to those of other concepts and eventually linking them to internal identifiers.
- Data transformation – simple transformation between a JSON data structure reflecting the RDF (tree) structure of the semantic model to more complex data transformations.

The following figure shows the simple implementation that is currently supported. It can be retrieved at Federated-BDI/Docker-BDI-Node (github.com). The configuration, implementation, and integration of this node is described in the github repository.

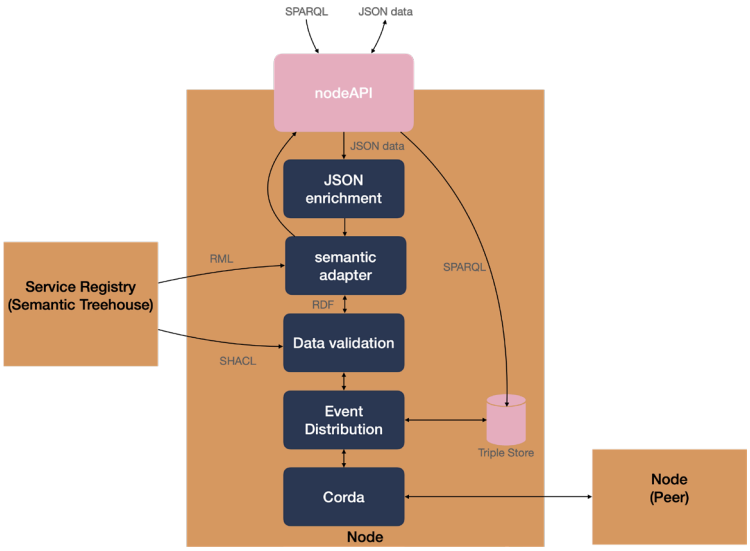


Figure 9-10: current implementation of a node

10 Final remarks

Business, physical assets, and IT will always evolve. Innovation is a key. Innovation cannot be managed centrally, it will be distributed like development. This results in for instance innovative business activities, the Physical Internet, different standards (and semantics), new technology (like Large Language Models (LLMs) and a VC-based infrastructure), etc. The architecture provides several means to incorporate these innovations by means of its flexibility and extensibility, whereby capabilities are configurable. This section elaborates on certain topics.

10.1 General

The concept is a set of capabilities that organisations and platforms can implement to be able to share data with others in a seamless way. Seamless implies no additional runtime configurations but rather negotiation for agreements as to which data can be shared. This concept allows for:

- smooth interaction between and among the different logistic chain operators and public administrations involved;
- a large variety of business cases in general based on improved decision making with better (and more data). Examples:
    - waste reduction and stock optimization due to improved visibility,
    - carbon footprint reduction by optimization of capacity utilization,
    - agility by dynamic planning,
    - resilience by better data, and
    - improved supervision by competent authorities;
- re-use of existing (partial) systems by integration with IT legacy and business processes;
- streamlining multimodal transport;
- decreasing or removing costs derived from lack of interoperability and reducing interoperability costs.

10.2 The Architecture

To support these business and authority objectives, the proposed architecture is to facilitate supply and logistic chain interoperability by providing capabilities – within a infrastructure provision (data sharing grid) - to any data holder and data user and providing them the opportunity to do business with another i.e., engaging and fulfilling any kind of contract, execute transactions, providing (Value Added) service and complying with legal obligations.

The Architecture complies with the DTLF Building Blocks – Plug&Play, Federation, Technology Independent Services and Trust, Safe and Secure – the FEDeRATED Core Operating Framework and 37 Leading Principles. A core requirement is data at source.

The functional requirements of the Reference Architecture refer to the need for:

- a “common” language (semantics)
- discoverability of organisations (level playing field)
- safe and secure data sharing for all participants
- access to data of all participants implementing data sovereignty

The proposed architecture results in the need for data users and data holders to technically:

- Utilizing an Identity and Authentication infrastructure.
- Applying a Service Registry - allowing for a distributed development of data sharing

according to common concepts and

- Applying an Index – enabling open-source data sharing, potentially with predefined interfaces to support stakeholder roles. The Index:
  - always interfaces with Service Registry and can thus always be configured and
  - allows each stakeholder to share data independent of any existing platform while implementing open source solutions.

The Service Registry and the Index will behave according to defined interfaces and protocols, ensuring safe and secure data sharing in the context of various EU legal settings. Semantics constitutes an integral part of these capabilities. They are fully configurable according to the (core) semantics. Protocols and their application in use cases (resulting in configurations) require governance and (potentially) standardization.

10.3 Considerations towards deployment

The architecture comprises of innovative features (the new), aim to solve existing interoperability bottlenecks. Deployment can only succeed when legacy (the existing) is recognized and can be dealt with through stakeholder commitment.

The new versus the existing:

- The ideal deployment of the architecture is based on semantic technology. However, many existing systems and solutions are based on other technology like Application Programming Interfaces (APIs) or Distributed Ledger Technology (DLT). They interface via existing commercial and community platforms with APIs and messaging.
- The objective is to facilitate the creation of a set of Technology Independent Services (TIS) enabling 'plug and play' by all stakeholders. When utilizing semantic technology for data sharing, TIS are completely configurable utilizing the Service Registry. This Service Registry can be applied also for other technology, but it will lead to a large amount of for instance APIs and additional data transformations (that will come with a cost and lead to potential data losses). The same is applicable to plug and play: organisations can implement so-called RDF plugins (or maybe duplicate their database to a triple store for external data sharing) or they need to implement APIs integrating with a component implementing the Index (a node or gateway).
- Whenever a platform supports TIS APIs, it is not required to have them interoperable. A user can call TIS APIs of any platform since they should all be identical. The latter will probably not always be the case since a platform will only implement a part of the TIS APIs relevant for its service. If, however, all platforms implement the same TIS APIs (or a minimal agreed set) according to IA interfaces, there is no need for platform interoperability if these platforms agree on some type roaming agreement where one platform user can utilize another platform to share data with a user of that other platform.

Various steps must be undertaken, not the least establishing collaboration with potential users and solution – and service providers, to complete this architecture to fly. The objective is to create a trusted, safe, and secure grid based on distributed capabilities. It is like implementing the phone network: how to scale from two phones to multiple ones creating a network.

Some necessary steps to be undertaken, also utilizing the existing prototypes Service Registry and Node, are:

- Pilot testing of the solutions for a large variety of use cases (i.e., LivingLabs).

- More instructions and examples illustrating how the two capabilities can be applied in practice.
- Development of a public data sharing agreements (i.e. a (voluntary) Regulatory Framework) with a governance perspective to provide more details on:
    - Rules and required capabilities for participation ('issuing policies) addressing data governance and digital identities;
    - Synergies with existing initiatives as part of the EU Data Strategy and developments like eSens (eIDAS2.0, eDelivery, etc.);
    - Service and operational governance (non-functional requirements);
    - Technical governance and standardization (architecture, semantics, etc.).

10.4 Applying the capabilities – regulatory data requirements driving adoption

One of the main concerns is the creation of a grid or federated network of platforms. Authorities and regulators can drive the development of this grid by applying the concepts and capabilities for their data requirements. As enterprises can use the node (and other capabilities), these enterprises are able to use the capabilities not only for regulatory data requirements, but also for business data sharing. Like stated before, it is like installing a phone (capability) and being able to talk (share data) with all others that also have a phone.

Regulators (and other authorities) will come with (new) data requirements like for instance those formulated by the electronic Freight Transport Information (eFTI) Regulation. Data requirements for supervision can be formulated as tree structures with the capabilities. The prototype Service Registry and the semantic model enables formulating the eFTI data requirements (although the semantic model may require (minor) extensions to cover all data requirements).

These regulatory data requirements can be published textual (spreadsheet, word document) and as configurations for the Index capability implemented by a node. The node provides event-based APIs for integration with IT systems and platforms like those of for instance eCMR providers and Transport Management Systems (TMS). Of course, these IT systems and platforms must have an interface (preferably also an openAPIs) for accessing the required data.

Where a regulator and its supervising authorities can use the prototype Service Registry for specifying regulatory data requirements (like eFTI data set) and their subsets, supervising authorities and enterprises can configure their node for these data sets.

As soon as an enterprise or platform decides in implementing the capabilities, it can apply them for other purposes. This will create a network effect, thus driving adoption.

10.5 Utilizes the capabilities - Value added services

Conceptually, the infrastructure consists of interoperable capabilities (nodes). A node implements the index, containing Linked Event Data that are received from or shared by a user with another user of the infrastructure, and a query for additional data by a data user to a data holder, based on links contained by the index. Each user (or user group, in which case a platform implements (party of) the node functionality) has its own node and each node contains different data. A node supports all required functionality for safe and secure data sharing, including functionality supporting non-repudiation (log and audit trail) and data integrity.

These capabilities and their protocols are a basis for developing value-added service (VAS). Any VAS is outside scope of the infrastructure. It rather uses the capabilities of the infrastructure. It can

run as a service on the infrastructure or can be implemented by one of the users of the data sharing infrastructure provision. A VAS is defined any (third party provided) service that utilizes links and access to data provided by those links to generate new data for one or more users of the infrastructure. Event federation, ETA prediction, dynamic routing, risk assessment, and maintenance prediction are examples of a VAS. A VAS can be based on data analytics.

The capabilities provided by an infrastructure deploying node functionality are as described before: common language, data sovereignty, discoverability (Service Registry, search, index), data quality, event distribution, and query federation. These are capabilities that can be deployed by a Value Added Service (VAS).

A VAS can be provided by a third party that does not have a function in logistics operations but provides optimization data for logistics stakeholders and/or authorities (users of the infrastructure). A third party may provide a VAS as a service to users of the infrastructure and/or as a facility used by one of the users for its own optimization.

A VAS always provides a function to one (or more) user(s) of the infrastructure, where this user needs to provide the necessary links to the data (**event federation**, which can be configure by the user to activate a VAS). A VAS will run a **search** on its node to collect all relevant data for calculating its output. Potentially, it requires **query federation** to assess the proper data set. The quality of the output data of a VAS depends on the **completeness** and **correctness** of its input data, which can be validated by a node (**data quality validation**).

A VAS can be provided to various users, based on their role, by applying the **semantic model** for developing its interfaces. It may be transport mode specific by using particular digital twins (e.g. vessel, truck), infrastructure (road, inland waterways, rail), and relevant features of the infrastructure (locks, traffic density, bridges crossing other modalities, water depths, weather conditions). These are all data holders. For optimization of a VAS, a copy of the infrastructure and its features might have to be stored at the VAS. These are all design decisions of a VAS. For instance,

- **ETA prediction** for arrival of a transport means at an intended location can be based on the location and speed of that transport means at a given time (an event with position and speed needs to be available), and the infrastructure composition (the available stretches of an infrastructure must be available). An improved ETA prediction can also make use traffic density, weather conditions, and estimated fluctuation of traffic density between the position and the location for which the ETA needs to be calculated.
- **Corridor management** is another example of a (more complex) VAS where agreed optimization rules provide an optimal speed for a transport means to reach its intended location in time (an ETA for this intended location is given). It requires details of more than one transport means, their position, their speed, and their intended destination. Corridor management is applied by inland waterways[8] and can be applied where infrastructures for different modalities cross each other (e.g. water with rail or road).

In fact, corridor management is part of what is called **situational awareness**. The latter implies that individual stakeholders optimize their goals with awareness and considering the goals of relevant other stakeholders. It means that they need to share their intention, e.g. their itinerary details, and need to agree on decision rules. Each participating stakeholder can implement these decision rules

[8] See for instance https://www.masterplandiwa.eu

with their own algorithms or use a service.

Large Language Models (LLMs) are a basis for many of these value-added services. By training them with (large) event data sets, these models are expected to support and improve decision making. Many experiments are already performed in this area, more are expected to take place. Preferably, these event data sets are not limited to those of a single organisation, since supply and logistics is a network where various participants use the same (scarce) resources. This is why most applications of LLMs focus on infrastructure optimization, which uses open (sometimes paid) data.

10.6 Improving the capabilities – Large Language Models

Large Language Models (LLMs) are not only applicable to support and improve decision making. They are already commonly applied in IT and boost productivity of software developers (with various pros and cons). Since there is still a lot of design time effort required in service development and -customization (which is configuration of software), LLMs are expected to reduce this and maybe even enable data sharing without any design time effort. The latter is feasible when data transformations can be generated by LLMs at runtime, utilizing the semantic model as some intermediate structure (this is how Google Translate works).

Applying LLMs for configuration and use of the capabilities is via a chatbot: a natural language interaction of a human with an LLM. The LLM must be trained for delivering the required output. The following opportunities are identified for development of these chatbots and training LLMs:

- **Ontology alignment** – joining two ontologies via common concepts and properties. This is like ontology matching, whereby ontology matching focusses on the common concepts and properties of two ontologies (the 'intersection').
- **Tree structure generation** – specifying all types of tree structures for data sets with the semantic model like for business activities, states, events, queries, etc. as indicated in this document.
- **Data transformation** – creating matches between two tree structures as basis for data transformation. Each of these tree structures has its own ontology. This is a special case of ontology matching: where ontology matching focusses on the intersection of two ontologies, tree structure matching focusses on the intersection of two tree structures. An LLM should not only be trained with ontologies, but also tree matches. By manually correcting these matches and feeding them to the LLM, this LLM is expected to improve its output. This can potentially result in runtime matching and data transformation.
- **Query generation** – the ability to formulate queries without prior knowledge of semantic web standards, in this case SPARQL. This type of chatbot is already available and has been tested. Experiments with users have shown that the chatbot contributes to training users in producing SPARQL.
- **Query federation** – the ability to decompose a query into smaller queries that can either be executed in parallel or must be serialized. This requires knowledge of data holders, i.e. their metadata that is accessible. It requires further exploration.

As the experiment with the query generation chatbot learned, this chatbot can be a basis for training. Two types of chatbots could be envisaged, one for training in using semantic technology and another for training in an application area like supply and logistics. Where the first chatbot contributes to productivity for software development, the second contributes to the ease-of-use of the capabilities.

Where LLMs improve the capabilities, other developments like standardization of many protocols contribute to cost reduction of these capabilities (development, maintenance, and operational costs).

By for instance implementing standardized connectivity protocols, a single connectivity component is required instead of multiple.

10.7 Extending the functionality – VCs as a means for further digitization

VCs and claims are a key capability for large scale data sharing as required in complex environments with many stakeholders. These VCs can be applied for other applications than the one given here, for instance:

- **Physical access** to a location for load/unload. A subcontractor of a contractor can load or unload cargo at a hub, whereby the customer does not know the subcontractor and the subcontractor does not know the customer. A scenario of adjusted VCs has been developed for this case.
- **Federated querying (data access)** – this is a use case whereby an upstream actor requests downstream data, whereby upstream and downstream actors don't have a direct relation (business or legal). An example is where a customs authority requires upstream data of a declarant, the latter needs to request data of its business partners and the declarant is not allowed to read the data provided by its business partners. This will allow for instance the querying of container content by a customs authority to a carrier, where the data of the content is provided by a forwarder. This is a type of chained delegation, potentially like the one of the second use case in this report.
  In business-to-business, this type of VC applications requires further attention, since a customer may not (want to) know a subcontractor (see second use case), whilst a subcontractor may require access to data of that customer.
- **Permits** – such a use case is quite straight forward. An authority issues a permit for a particular type of activity to a logistics stakeholder. A drivers' licence is already an example of such a permit, a permit for transportation of dangerous goods is another example. Examples like a drivers' licence are already developed for education.
  A (customs) **release** or **certificate of origin** are also a type of permit for goods, whereby it must be clear that an authority is the issuer of this type of permit and integrity of the data can be assured. These are claims issued by an authority about the goods. This differs from a VC that is irrevocably associated with a holder. The role of DIDs could be investigated for these types of cases.
- **Trade related statements** – there are all types of statements made by public – and private organisations like Certificate of Origin and Letter of Credit. These statements are issued for single goods flows and may potentially be held by different organisations; it has to be sure it is issued by a certain organisation.
- **Employee screening** (VGB – Verklaring Geen Bezwaar – or VOG – Verklaring Omtrent Gedrag) – this is a use case where a (natural) person receives a claim stating that an authority has investigated the behavior of this person and finds it compliant with a certain security level (VGB) or a claim stating that the person has not been convicted for any crime relevant to the performance of his or her duties (VOG). This would potentially reduce risks for illicit behavior of persons, both employees and others. The person can store this claim in a wallet in the form of a VC, besides other claims such as a VC that they can use to identifying themselves.

When identifying additional use cases for VCs, one must bear in mind to make a balance as to which data is in a VC and in IT systems. Conceptually, all data can be part of a claim in a VC (or its presentation), for instance a complete CMR or eB/L data set.

Thus, claims of VCs could potentially be data carriers. However, business collaboration requires adaptation to changes (resilience and agility), whereby data changes. This implies that it may be beneficial to share only linked event data between IT systems of organisations (data at the source) and only store the actual data in the form of VCs when it concerns limited non-volatile data, , e.g. the case of physical operation by a truck driver with a VC per shipment/consignment. Such types of applications for VCs must also be considered in the context of on-board units since these may already contain the data. Furthermore, such types of VC applications also require a revocation mechanism at delivery of the goods, where revocation is performed by a consignee (which is a business role, that is not necessarily a verifier).

With respect of eB/L data, such data sets are issued as a claim by a shipping line to its forwarder at exit. This forwarder will have to hand over the eB/L (thus the claim) to a forwarder at entry. This transfer either implies a type of chain delegation since a VC is associated to a single holder, or to apply another technology(-ies).

Annex Linked data

One of the main design principles given by the operational framework and the Master Plan is 'data at the source'. This means that links to data are shared. The concept of linked data is introduced in this section.

A.1 Data sharing roles

Data sharing processes distinguish two roles as specified by the Data Governance Act, namely a data holder and – user:

- A data holder is a legal person or data subject who, in accordance with applicable Union or national law, has the right to grant access to or to share certain personal or non-personal data under its control. (source: Data Governance Act). A data holder is the same as a data provider in the context of data sharing.
- A data user is a natural or legal person who has lawful access to certain personal or non-personal data and is authorized to use that data for commercial or non-commercial purposes (source: Data Governance Act).

Each data user can act as data holder for other data users. Thus, a chain of data holders and -users can be created. In this context, a platform also acts as data holder. In its role of data steward and/or – custodian, a platform acts as data user to its users. Thus, a chain of roles is created.

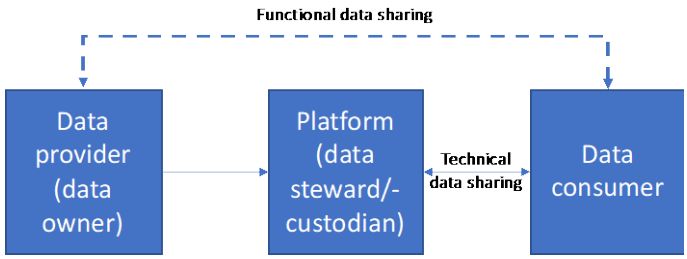


Figure A-1: the different roles in perspective

A.2 Linked data

There are many ways to share links to data. One way used in this section is to apply semantic web standards and technology like a triple store (next figure). The choice for this technology is argued in section 4.3.

A triple store or graph database can import a semantic model and link RDF data to the proper concept and data properties. Each instance of a concept, like a truck, can have a unique identification thus allowing to store data of that Digital Twin only once. This will be further elaborated. A triple store also supports SPARQL as a query language.

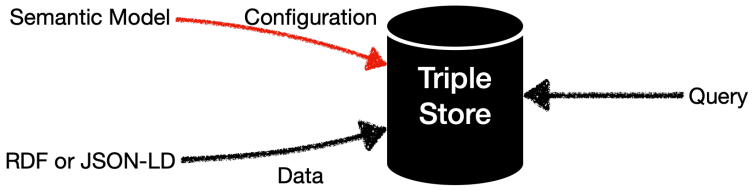


Figure A-2: basic implementation

Applying semantic web technology, brings additional functionality, namely the **unique (global) link** of any concept, where a concept can be anything, a box, a truck, a container, a document, or anything relevant to supply and logistic chain operations. The link is used in IT systems governing

physical flows; they are additional to any already known identification like for instance a container number or a license plate of a truck. These links can be printed as (two-dimensional) barcodes (also known as QR-codes) on physical objects like shown in the example.

The unique global link consists of two elements, namely a 'base URL' (URL -Universal Resource Locator or web-address, which is a unique identifier) and a software generated identifier, a Universal Unique Identifier (UUID).

Links provide (authorized) access to data of the concept. Sharing links implies a **data pull**, as required for instance to support of a targeting officer (see before). These links can be printed as QR-codes on physical assets. A combination of URL and UUID may reveal commercial sensitive information since a URL may refer for instance to a producer of high value products. Thus, part of the link is stored and shared between IT systems (the URL), where that part can be linked to an identifier used in the physical world. The identifier should preferably not contain any classification that would reveal information on the content or nature of goods.

Besides physical assets and boxes, also organisations, persons, locations, and infrastructural elements can have a unique link that can be used by IT systems. These unique links can refer to additional identifications like a port number for a terminal, geo-coordinates of the location (or area) where a pilot may board a vessel, or a chamber of commerce identifier of a legal entity.

The previous figure shows one triple store suggesting a central platform. The objective is to distribute this triple store to all stakeholders involved. The following figure shows the basic pattern that can be repeated. It consists of three triple stores that share links and can process federated queries.

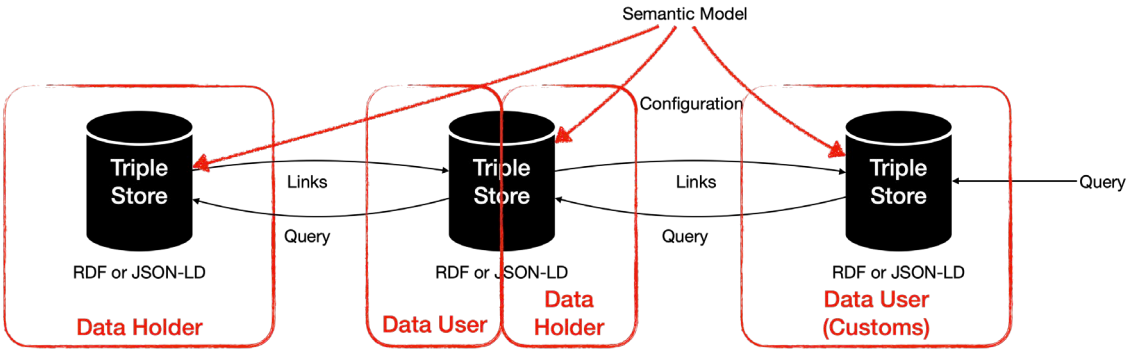


Figure A-3 multiple linked triple stores

The queries shared between two triple stores are simple: it are links that are shared for accessing data. In supply and logistics, it can be a link to access container data. More complex queries, i.e. queries with filters, stem from a human or IT system of a stakeholder, shown in the right side of the previous figure. A container track is such a complex query, since it must filter a single container from all container flows. A filter will give indications of a direction, a position relative to that direction, and the origin from which the track must be composed.

These triple stores all implement the same semantic model. The data is stored as RDF or JSON-LD. The data holder and – user roles are also mapped to this pattern, where a customs administration acts as data user. It shows that customs can pose a query to a known data holder, where this data holder can federate its query as data user to another data holder. Thus, customs can retrieve upstream data via browsing through links.

The previous figure shows that links are distributed from a data holder to a data user. Link distribution is governed by rules, specified by services. This refers to event logic. The figure also shows a chain

of data holders and -users, which can in fact be a network. An enterprise or platform acting as data holder can interface with multiple customs administrations and a customs administration with multiple data holders. There must be agreements on link distribution. Link distribution is the basis for query federation, where downstream query federation depends on implementation by all relevant stakeholders or the first known data holder has the role of data user to all these downstream data holders. The following figure shows such a network, that includes query federation.

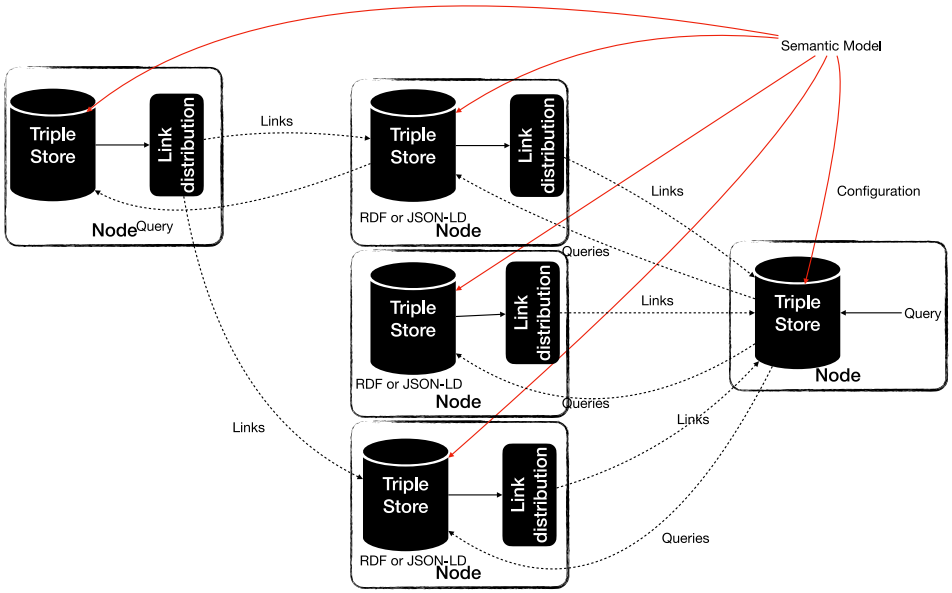


Figure A-4: multiple linked triple stores

The dotted arrows show to which triple stores a link and queries can be distributed (the queries and links can of course also be distributed between the three triple stores at the right side of the figure). Each of these triple stores is implemented by an organisation in its role of customer, service provider, authority, or even physical operator. They all have the same semantic model. Together, they constitute a network where all triple stores have each other's addresses. Queries are only distributed to a triple store if links have been received from that triple store.

A.3 Data distribution – physical processes and event distribution

Where the previous figure shows one triple store, this store is in fact distributed like visualized in the following figure. This requires data distribution mechanisms, specified by event logic.

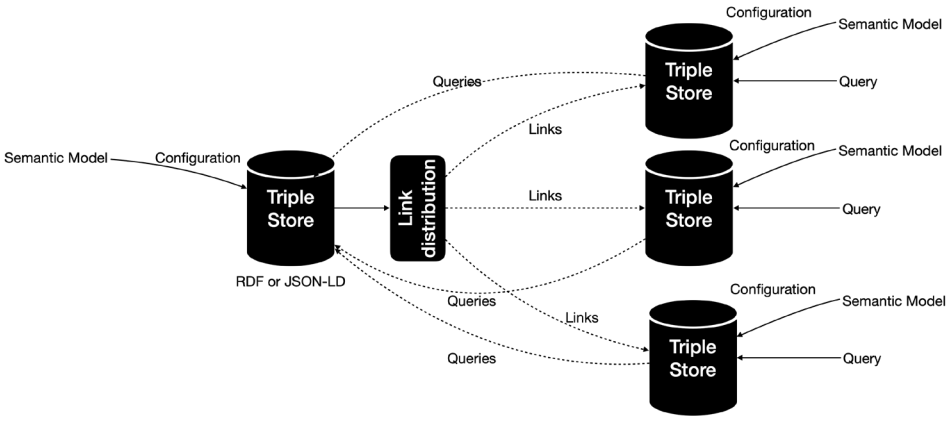


Figure A-5 multiple linked triple stores

The dotted arrows show to which triple stores a link and queries can be distributed (the queries and links can of course also be distributed between the three triple stores at the right side of the figure). Each of these triple stores is implemented by an organisation in its role of customer, service provider, authority, or even physical operator. They all have the same semantic model. Together, they constitute a network where all triple stores have each other's addresses. Queries are only distributed to a triple store if links have been received from that triple store.

Sharing links requires agreements on **link distribution**: what are conditions for sharing links. These will be elaborated later on.

A.4 An example – mapping stakeholder roles and triple stores

This example shows how a customs administration at a country of entry can access data by receiving links from stakeholders in the country of exit and - loading. It considers as example various actors that conceptually implement what we call a node with a triple store.

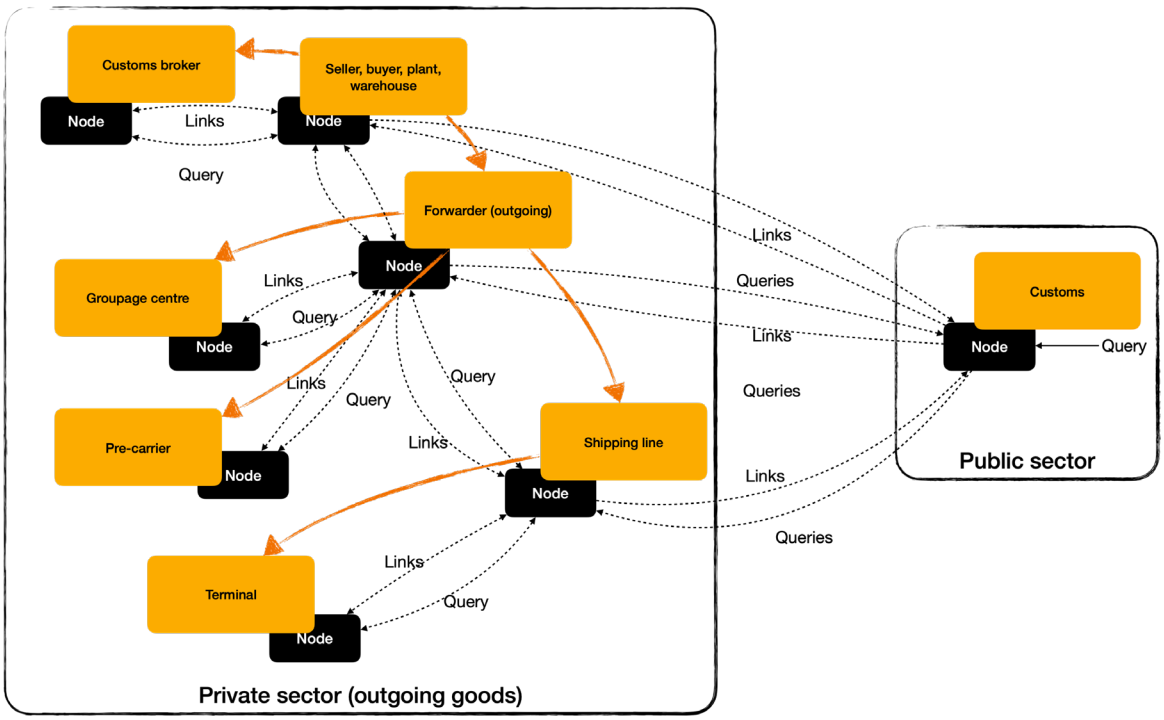


Figure A-6: linked triple stores and stakeholders in supply and logistic chains (outgoing)

The figure shows the following roles (concepts and definitions applied here are given by the semantic model):

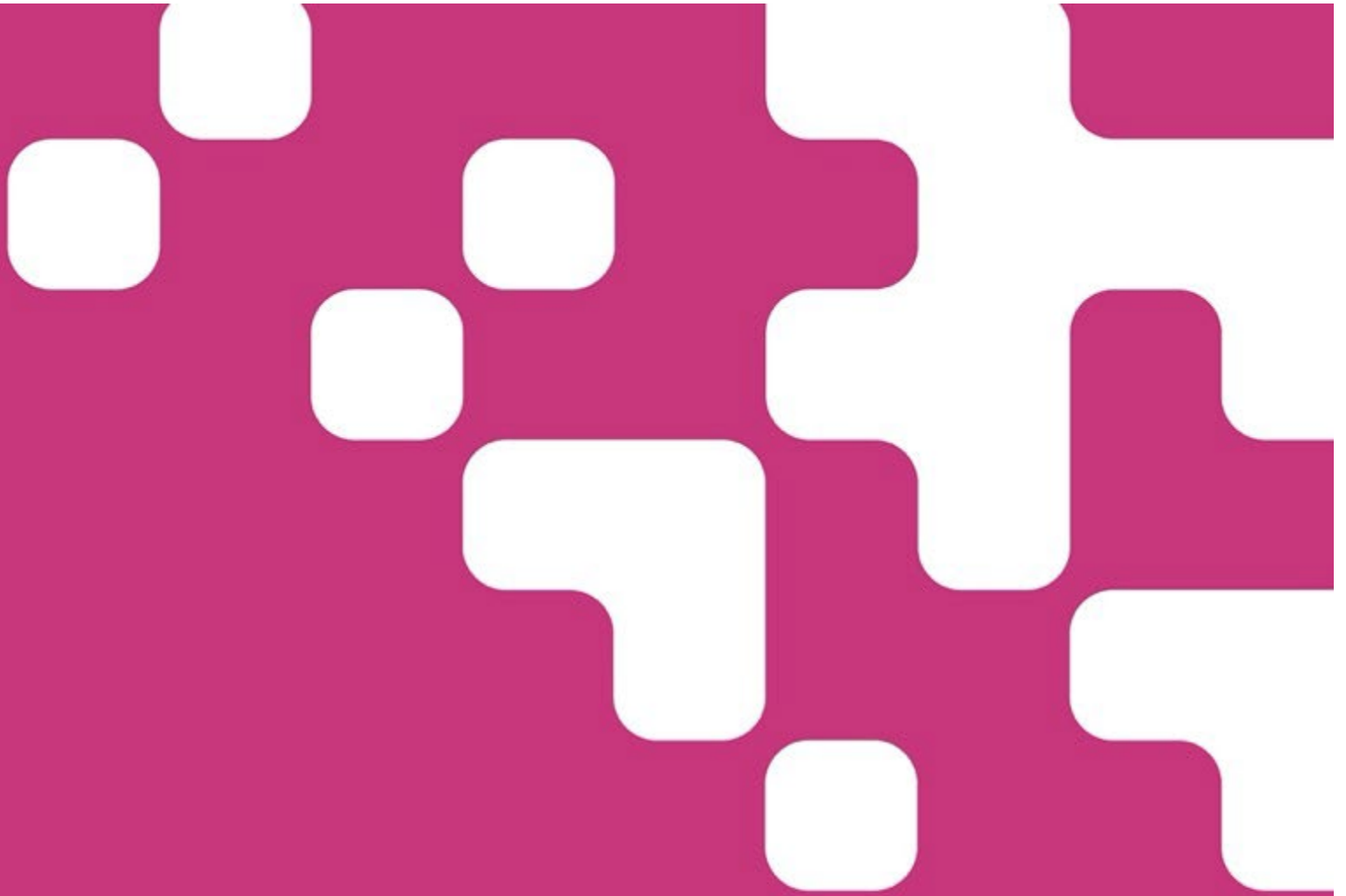
- **Seller/buyer or plant/distribution centre**. This is the origin of the products that are moved from PLA (place of acceptance) to PLD (place of delivery). It can be based on purchasing (seller/buyer relation), Vendor Managed Inventory (from distribution centre to PLD), or stock replenishment (from a manufacturing plant to a distribution warehouse). These can provide product details for export and import declarations and the packing list.
- **Forwarder** (outgoing) – a forwarder manages groupage (outgoing), stripping (incoming), and transportation between a port and the hinterland (especially for LCL (Less than Container Load) transport) and is thus able to provide a stuffing list and status of the transport legs. A forwarder does not know the actual content of the goods, although this forwarder may have more information on the goods than a shipping line based on the goods description provided

by its customer. A forwarder will receive a Bill of Lading (B/L) of a shipping line and share it with the forwarder for incoming cargo.

- **Shipping line** – a shipping line can only provide a load-/discharge list, complemented by transhipment lists, and pre- and on-carriage transport legs for FCL (Full Container Load). A shipping line never knows the actual content of a container, only a description of the container content (STC – Said To Contain), based on international conventions like The Hague-Visby – or Rotterdam Rules. A shipping line is aware of the country of origin and – destination of a container and can provide these types of links to a customs administration, thus providing details of container – and vessel track between ports or PLA/PLD.
  A shipping line will share a B/L with its customer and will inform a so-called notify upon arrival of the cargo in the port of discharge or will transport the cargo, i.e. container, to a place of delivery. The place of delivery can be a stripping centre or the final destination (e.g. distribution centre) of the content of a container.

Other stakeholders active in these types of chains do not directly provide data to a customs administration at entry at a country of exit. They act on behalf of others that contractually can provide links. The following stakeholder roles can be considered:

- **Pre- and on-carriers** – they can provide details of the transport legs between a port and the hinterland, but they are not aware of the country of destination or – origin of a container or the goods they transport. They will never provide links to a customs administration. They are either contracted by a shipping line or a forwarder. It can
- **Groupage – and stripping centres** – they can provide a stuffing list. They are contracted by forwarders and are not aware of destination of a container (groupage centre) or goods (stripping centre).
- **Customs brokers (export (and import))** – these act on behalf of a consignor, consignee, or their agent (forwarder). A customs broker at export has product details for the goods that are to be transported from PLA to PLD. These details are not shared with a forwarder in case the forwarder is their customer. The HS-code of the export declaration is based on the product description in line with export statistics and restrictions of the exporting country. For import, the objective is to have an HS-code that fits VAT purposes. For import and export declarations, a customs broker does not require the link of products to goods, given by the package list.



FEDeRATED
NETWORK OF PLATFORMS

Co-financed by the Connecting Europe Facility of the European Union