MASTERPLAN - TECHNICAL SPECIFICATION
SEMANTICS

Semantics is at the core of the FEDeRATED technical specifications. Through harmonized M2M data interoperability the various parties within the supply chain should be able to seamlessly exchange digital information for multimodal freight transport business transactions as well as compliance to regulations. One must understand each other. To enable all parties to do so, the FEDeRATED semantic model has been developed[1]. The FEDeRATED semantic model is a representation of semantics by a technology, which is semantic web standards. This semantic model is elaborated in this document.

1. The FEDeRATED semantic model

The FEDeRATED semantic model is a multimodal ontology[2] for **data sharing** and its **business transaction** – and **compliance choreography** between any two stakeholders in supply and logistics. To ensure this, the semantic model:

- needs to be expressed by an open standard to be interchangeable and accessible for all.
- has adopted the semantic web standards (e.g. Ontology Web Language (OWL) and Resource Description Framework (RDF)).

The semantic model aligns existing mode – and/or cargo specific ontologies and standards. Communities, industry associations, regulators, and others can align their ontology or standards with the multimodal ontology or apply the multimodal ontology to create what is called in literature a 'lower-level ontology', refining the multimodal ontology to meet their data sharing requirements. In fact, they add constraints to the multimodal ontology by doing the latter.

The semantic model can be considered as a kind of data *translator*, enabling various regular data standards and models to be interoperable. The semantic model aims to support plug and play, modelling of regulatory data requirements, and modelling of business document data sets. It encapsulates **data quality**[3] aspects and creates an open and neutral level playing field by enabling stakeholders to express their **data requirements** and **capabilities**.

2. Functionality of the semantic model

The semantic model consists of two parts, namely a structure for creating alignment between various mode – and/or cargo specific ontologies with Digital Twins and events, and standards and concepts for data sharing (see next figure). Both will be introduced.

[1] For generic information on semantics see the document High Level Architecture, chapter 1

[2] An ontology is a way of showing the properties of a subject area and how they are related, by defining a set of concepts and categories that represent the subject.

[3] Data quality is elaborated in the description of the Service Registry. It consists of aspects like correctness, completeness, relevance, etc.

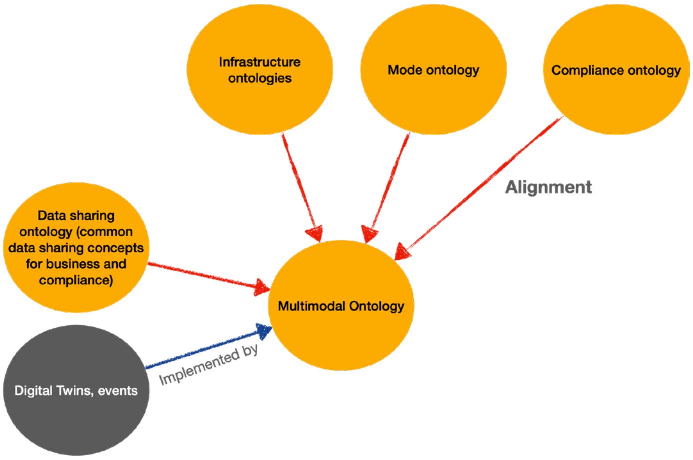

Figure 1 – overview of the ontology components and its structure

3. Digital Twins and events

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** - a taxonomy of real-world objects (container, truck, barge, etc.) and infrastructure, and
- **Event** - the association between at least two Digital Twins in time and space (past, present, and future, where future is 'expected', 'planned/estimated', and 'required').

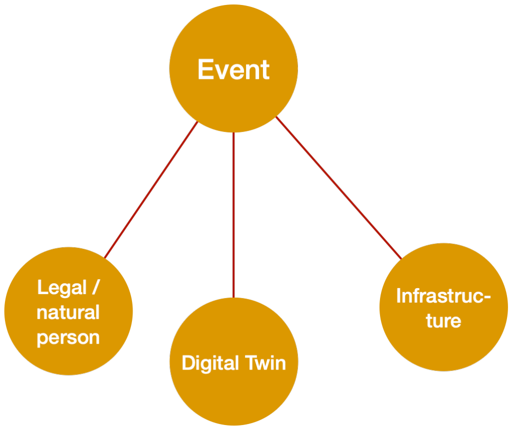

Figure 2 – Digital Twins and events

As figure 2 shows, a legal / natural person can also have an association with a Digital Twin and/or infrastructure. These high-level concepts are detailed by the ontology.

Any role of a legal – (enterprise, authority) or natural (individual) person and a location is assigned when constructing subtypes of 'event' like 'interaction' (see data sharing concepts). Examples are roles like shipper, consignee, carrier, employee, and truck driver for persons and place of acceptance and port of discharge for locations.

The Digital Twins are defined as follows:

Digital Twin	Any real-world physical object
Product	Any commercial object that is bought and sold. It can be discrete (e.g. electronics, cars, and livestock) or bulk (e.g. oil, grain, sand, and LNG).
Goods	(re)packaging of products to facilitate logistics activities. Not all products will or can be packaged (e.g. liquid or solid bulk).
Transport Equipment	Re-usable equipment to facilitate transport of products, goods, or other equipment, for example a wagon or a container
Transport Means	An asset that can move on its own power and that can carry products, goods, equipment, or another transport means.
Infrastructure	Any physical location relevant for logistics
Node/hub/place	Any location where a logistical activity is performed (see also business services)

All Digital Twins except goods can also be considered as **assets**. They are owned and/or used by a legal or natural person. Their use and maintenance can be monitored remotely (condition-based maintenance).

These Digital Twins are a digital representation of real-world objects, associated (in time) by 'event' (figure 3).

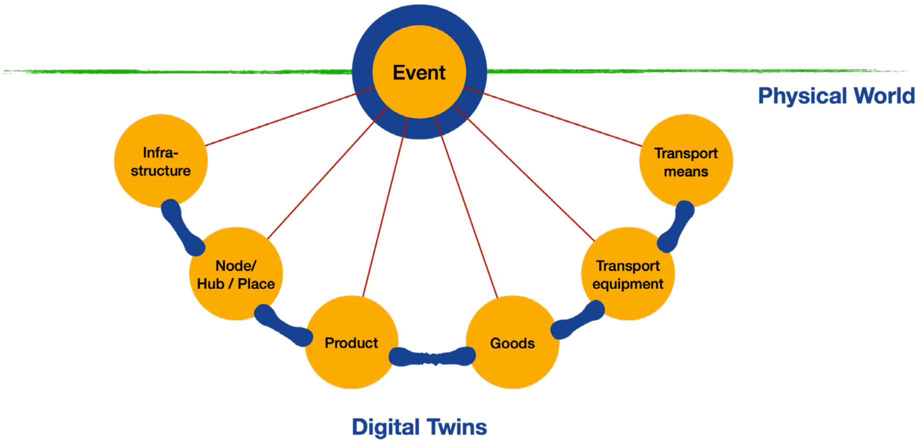

Figure 3 Digital Twins - digital representation of real-world objects, associated (in time) by 'event'

The Digital Twins are further **specialized** via subtypes into a taxonomy[4]. For instance, a transport means is specialized into a vessel, barge, locomotive, and truck. Transport equipment is also specialized into for instance (sea)container, trailer, railway wagon.

Goods are not further specialized; these have a data concept 'type of package' with a set of allowed values (e.g. box) and the 'number of packages'. Goods can be repacked, thus

[4] Taxonomy is the practice and science of classification of things or concepts, including the principles that underlie such classification.

creating, or omitting an association between two goods instances. When goods are packed in multiple containers, a goods item with the number of packages is created for each of these containers.

The association between Digital Twins has additional **constraints** representing data accuracy. A set of constraints results in '**cargo**': all Digital Twins that are carried by another Digital Twin. Cargo is thus at instance level (in time) and is not a separate concept in the model. All 'cargo' constraints are included in the model by specialization of one or more of the subtypes of 'event' by a community. For instance, a trailer can be cargo on a railway wagon, a train is a combination of a locomotive and a sequence of wagons (optionally with a locomotive at the end), etc. Logistics experts must assist in formulating these constraints, or these constraints are constructed when required by a use case.

Many other concepts that are currently common practice in supply and logistics must not be part of the model. For example, the concepts '**consignment**' and '**shipment**' refer to construction of supply and logistics chains (see further). Such types of concepts must be modelled as derived concepts.

Locations are separated from the infrastructure that interconnects them. Infrastructure is further decomposed into different modalities, e.g. rail - and road infrastructure. A **location** has a **function**, i.e. a business activity that can be performed at that location, and a **role** in the process created when specifying states and interaction, see further (e.g. place of acceptance of the goods by a carrier or port of call of a vessel). The function is a data concept of location; the role is part of the association with 'event'.

4. Data sharing ontology[5][6]

The data sharing ontology enables the specification of a choreography[7] for business transactions and compliance. In this context, a choreography is synonym to interaction pattern, e.g. a visibility pattern.

The data sharing ontology enables the specification of any interaction pattern(s) for a business activity. Each interaction pattern consists of states and state transitions ('processes') triggered interactions[8]. State transitions are an association between any two states; a state and an interaction are expressed by constraints to the semantic model. A business document like an eCMR or eB/L is a specialization of a shared, immutable state.

Each **interaction pattern** must be composed of at least one phase with one or more state transitions, one start state, and one end state. In case of multiple phases, the sequencing between the phases must be modelled by an output state of one phase that is the input state of another phase.

---

[5] Not each organization or data sharing community will fully apply the data sharing ontology, but may only apply a part of it. This is supported by the Service Registry according to the migration and adoption strategy that are described in separate notes and are part of the FEDeRATED Master Plan. Furthermore, implementation guides will be provided for applying concepts of the data sharing ontology.

[6] In a blockchain environment, the data sharing ontology can be the basis for developing so-called smart contracts.

[7] Choreography and orchestration languages deal with business processes design and specification. There are two distinct approaches for system representation and management. Choreography describes the system in a top view manner whereas orchestration focuses on single peers description.

[8] According to the formal theory of business process modelling, a process consists of a pre- (input states) and a post-condition (output states), where a firing rule determines the relation between an input – and output state. Modelling these levels of detail adds complexity that is outside scope for this moment.

A **business activity** (or a regulation) must have an interaction pattern. A business activity, like transport or transshipment, specifies constraints to the semantic model that are the basis for formulating states, interactions, and business documents. These latter ones provide additional constraints to the model from the viewpoint of a commercial relation or a regulation.

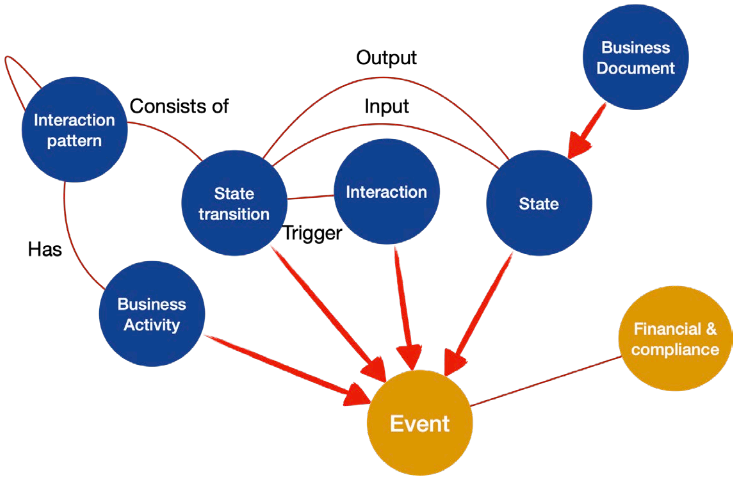

Figure 4 – an ontological representation of data sharing concepts

The concepts of the data sharing ontology are defined as:

1. **Business activity** – a physical activity that is offered as business services. Each of these logistics activities requires a minimal data set.
2. **Interaction pattern** – a logical sequence of interactions given by state transitions . Each pattern has a start - and end state, where a sequencing of patterns has the start state of the first - and an end state of the final pattern of that sequence.
3. **State** – a representation of (access to) data that has been shared.
4. **Interaction** – a logical function in the context of business process collaboration and compliance.
5. **State transition** – the change of state induced by sharing data with an interaction
6. **Business document data sets** – a (subset of) immutable state data agreed amongst customer and service provider. Examples of such data sets are eCMR, eBL, etc.

Since also legal and financial aspects must be supported for business transactions and compliance, the overall high-level semantic model as follows.

For instance, a transport activity has an ordering phase with an order and a plan linking to a contract and resulting in a business document data set and visibility events like shown in the next figure. The dotted lines represent the shared state (e.g. 'order exists'); the arrows the interactions (e.g. transport plan and ETA).

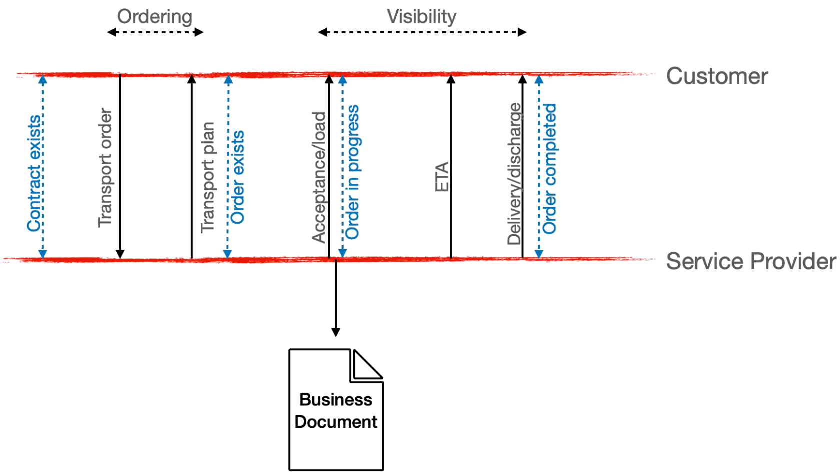

Figure 5 – example of states and interactions

Figure 5 also shows a data set of the state 'order in progress' results in a business document, e.g. an eCMR or an eB/L.

Figure 6 combines the previous ones into the (high-level) multimodal ontology. The financial and compliance parts of this model are further refined. They need to support bookings, for instance.

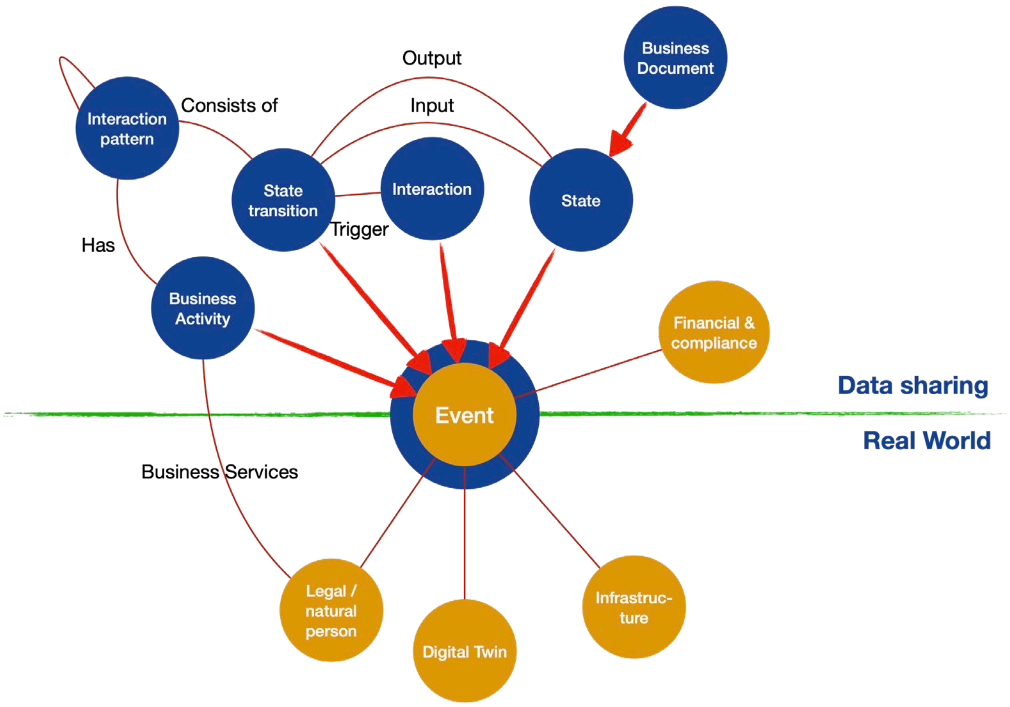

Figure 6 – overall high-level multimodal data sharing ontology

Data quality validation is supported by the Service Registry and the Index.

5. Data sharing in chains[9]

Each organization functions as node in an organizational network with its own outsourcing strategy. Each organization thus manages its chains. As a node in a network, rules for constructing associations between business transactions must be provided to enable data sharing of each node with customers and service providers[10].

Although the semantic model only supports business transactions and compliance between any two stakeholders, it must support additional rules for representing **logistics chain structures**, i.e. sequencing of legs and their relation with customer transactions. These rules enable for instance synchro modality of the various legs by a chain manager and support link evaluation of a subcontractor to access data at its source. This is the capability to relate for instance incoming orders to outgoing and to share events of service providers with customers.

An enterprise that manages various legs, e.g. a forwarder, has a data representation of its logistics chain structures according to the following rules (actual chain composition based on outsourcing strategies is outside scope):

- **Transaction dependency** ('inserted transactions') – the result of a business transaction with a customer depends on one or more business transactions with a service provider. Basically, a first – and last leg in a chain of outsourced business transactions specify the result of a business transaction with a customer.
- **Transaction independency** ('decoupled transaction') – the results of business transactions are independent of each other, although one business transaction initiates another. An example is initiation of a payment transaction during or after completion of a transport transaction.

The hierarchy of dependent business transactions controlling physical activities must be depicted by a transaction tree where the root depicts a customer (e.g. a shipper), a node of the tree a chain manager (e.g. forwarder), and the leaves the parties actually performing a physical activity, e.g a carrier. The root, a node or a leaf in the transaction tree may initiate another transaction tree that operate independent of the originating tree.

Each stakeholder in the tree can derive its customer and service provider business transactions from the events that are shared (see index).

6. Data sharing protocols

Any details of underlying protocols **must not** be implemented by the semantic model. This leads to two additional data sharing requirements:

- **Linked Event Data protocol** – events with links to additional data are shared between a data holder and – user. Data remains at the source. The protocol is applied to all interactions that trigger state transitions.
- **Technical paradigms and syntaxes** – data sharing (events, (SPARQL) queries, and query results) **must** be supported by a technical paradigm with an endpoint supporting for instance openAPIs or messages with a syntax like XSDs or JSON(-LD), and semantic endpoints (SPARQL or RDF). The transformation from semantic

---

9 Besides the implementation of interaction patterns of the data sharing ontology, smart contracts in blockchains can also implement the rules to construct chains.

10 These rules will be applied by use cases that address supply and logistics chains of for instance manufacturers and retailers acting as shippers and consignees.

model to a technical paradigm and syntax must be specified by a clear rule set; this rule set must be implemented by the Service Registry.

The semantic model must contain any relevant semantics of (open) **standards**; it must not contain any technical details of those standards. Instead, a **mapping** to these standards must be supported, allowing for data transformation (see the semantic adapter). The semantic model must also support semantics of any implementation guide of these open standards unless this semantics is too specific. This must be dealt by a transparent change management process.

7. Identifications – UUIDs and external references

Every concept of the model, including 'event', has a **unique identification**: a Universal Unique Identifier or UUID. A UUID is generated by a data source. It is a unique identification generated by an algorithm and is independent of what it identifies. This is an IT generated identifier. A UUID does not directly provide a link to relevant data. A UUID must be provided by a data holder to a data user by sharing an event. A data user can retrieve any data of that UUID from a data holder.

To prevent anyone to scan a UUID in the real-world to access data, a UUID does not contain the URI (Uniform Resource Identifier) where additional data can be accessed. Including a URI and UUID as identifier on packages may be commercial sensitive. This is also an extra security measure[11], combined with link-based access control supported by the index (see the leaflet of the index).

To integrate with existing approaches and to enable searches by humans, **external references** are included as data properties of all relevant concepts. All Digital Twins can have external unique references, either assigned by an internal mechanism (e.g. container numbers of sea containers) or assigned by external bodies (e.g. license plates of trucks). 'Events' can also have additional external references like trip number, flight number, and consignment - or order number. Goods must at least have the UUID for integrating with IT systems.

Any external reference number can be an internal reference of a data holder or -user that they currently want to use in a data sharing context. A data holder or -user may thus apply different external reference numbers that are integrated via a UUID in case of data sharing.

The sender and recipient of any event are specified by the choreography between a customer and service provider. For instance, a service provider (carrier) is the sender of trip information or flight schedules (those are of type 'transport plan') that can have an external reference. A customer just sees a list of events (flights) and books for a particular time; a booking is thus an event that may have an external customer reference.

[11] This way of hiding data is also called 'security by obscurity', which is not proper design for implementing security measures and is also not required for access control. However, commercial sensitivity might be leading and require only a UUID to be printed on for instance a box.