PROPOSAL FOR A COMMON LIVING LAB
IT ARCHITECTURE BOARD

1. The FEDeRATED semantic model

Objective

To develop a common Living Lab based on solutions developed by the FEDeRATED Living Labs according to the architecture and potentially its migration phases.

A proposal is to develop so-called **shared capabilities** for each FEDeRATED LL, whereby they act as a federated network of platforms and data is accessible by authorities. The shared capabilities are **visibility**: estimates and actuals of transport operations.

This document elaborates the setup and interfaces of the common LL:

- Overview of relevant stakeholders
- System setup
- Interface specification

Relevant parts of the architecture will be applied like interaction patterns for visibility, the semantic model, and an initial setup of the Service Registry for each participant in the common LL.

Any applicable assumption for this demonstration will be given in this document.

Stakeholders involved in the common LL

The following stakeholders are involved in the common LL:

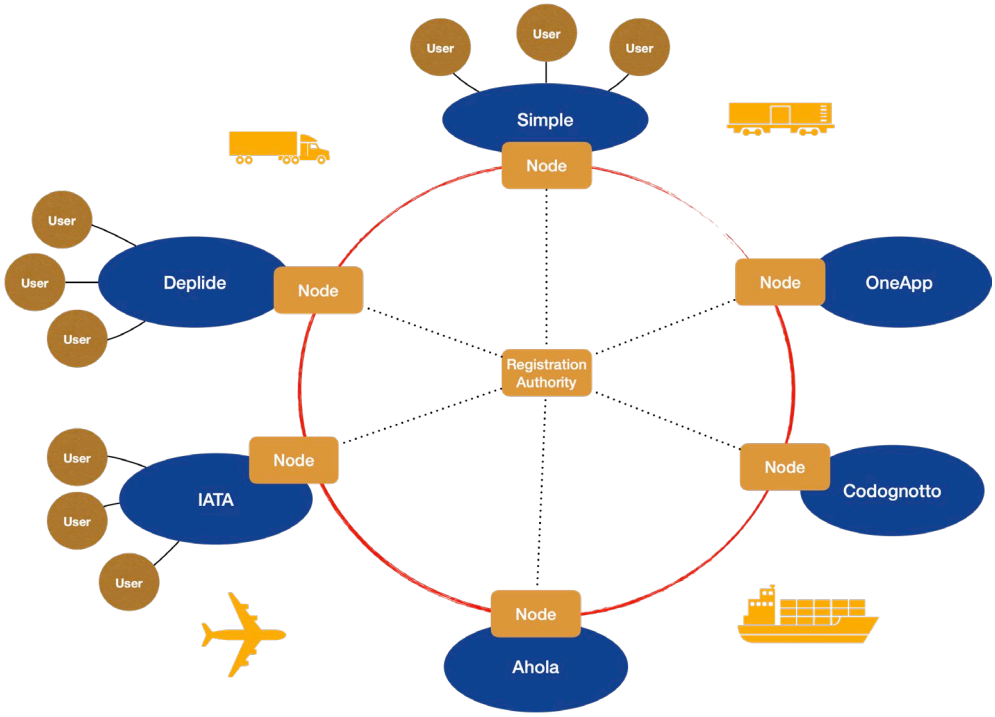
- Italy – Condognotto and TSG
- Spain – Ministry (Simple)
- Finland – Vediafi, Ahola
- Sweden – RISE (Deplide)
- Netherlands – Ministry of I&W (BDI)
- 51Biz – OneApp for accessing data by authorities
- IATA – integrating OneRecord with a hinterland modality.

Setup of the common LL

The setup of the common LL distinguishes between the use cases it supports and the underlying infrastructure. Use cases are driven by enterprises and authorities; these are partially participating in the common LL via (at least) Codognotto and Ahola. There is already a Codognotto use case under development, where Codognotto implements the node prototype.

Each participant in the common pilot will have the same capabilities, namely act as service provider, customer, and authority. Some participants may have limited capabilities, like only supporting an authority.

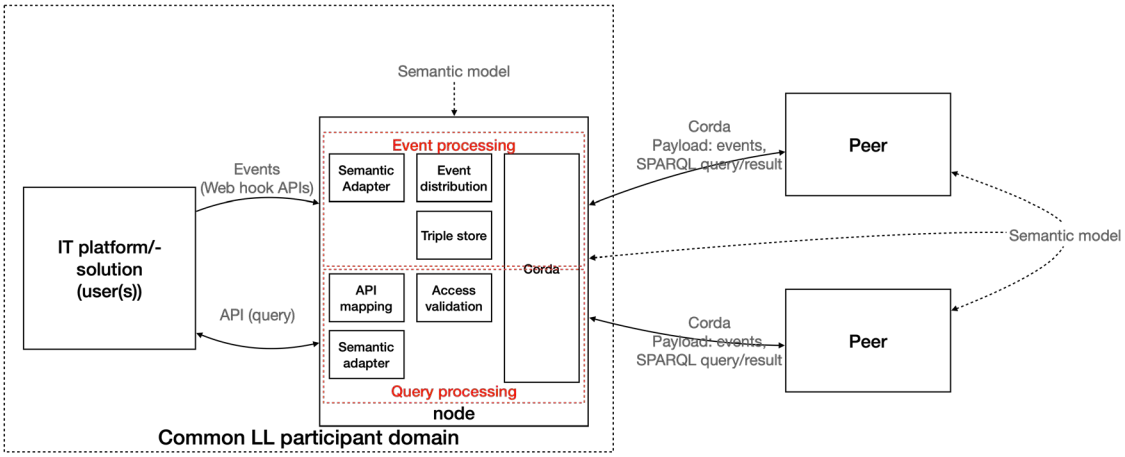
To implement these shared capabilities, a so-called (FEDeRATED) node will be implemented, exposing the capabilities of each stakeholder to others as depicted by the following figure. The FEDeRATED node is based on the BDI node and will be provided by the Dutch Ministry.



This infrastructure of federated platforms[1] must support the visibility pattern from a customer and service provider perspective, including access by an authority. In case a participant in the common LL does not have a user (enterprise or authority) the participant must simulate one or more users, where these users can have a role of customer and service providers. Other participants act as user (Codognotto, Ahola, and OneApp) and yet others may decide to use existing users of their platform to share data in a demonstration setting of the common LL.

The objective of the common LL is to demonstrate one or more (fictive) use cases. Any implementation choices for operational use by stakeholders may change. For instance, each participant may choose to implement 'node' functionality itself.

[1] Other terminology for such an infrastructure is 'Mobility Data Space'. Furthermore, the infrastructure will supported what has been introduced by Dutch Customs Administration and HMRC as 'data pipeline'.



Node functionality

The node that will be provided as Docker/Kubernetes container via github by the Netherlands has the following functionality (development based on the current version 0.2 yet to be done, see previous figure):

- **Internal interfaces** with a platform or IT solution of a participant. There are two types of interfaces:
  - Webhook API for pushing events to a node.
  - REST/openAPI for data retrieval of a query. The query is based on events with links shared between various stakeholders.
- **Interfaces between nodes**. These are based on the current implementation of Corda by the prototype v.02 of the BDI node. Corda provides a registration mechanism (Corda Network Manager) and safe, secure, and reliable data sharing via AMQP and TLS.
- **Data sharing between nodes**. All data is shared a triples (RDF) and SPARQL between nodes over Corda.
- **Event processing**. The capability of a node to receive (JSON) event data, transform it to RDF (semantic adapter), share it with the proper other node(s) (event distribution), and store what has been shared (triple store.
- **Query processing**. The capability of a node to validate that another node has also received a link and is allowed to receive a response to a query. The response will be retrieved via a single REST API from a user's system (API mapping). The semantic adapter will forward the response in RDF to the requesting node/user.

Two components need further configuration to support the visibility pattern, namely the semantic adapter (events, query response) and event distribution.

Event distribution

As the visibility pattern shows, all events are shared by an LSP (Logistics Service Provider, referred to as 'service provider' hereafter) to a customer, whereas some events are also shared with an authority (see Interface specifications).

A service provider role will provide visibility events to a customer role and to a competent authority according to an event distribution algorithm:

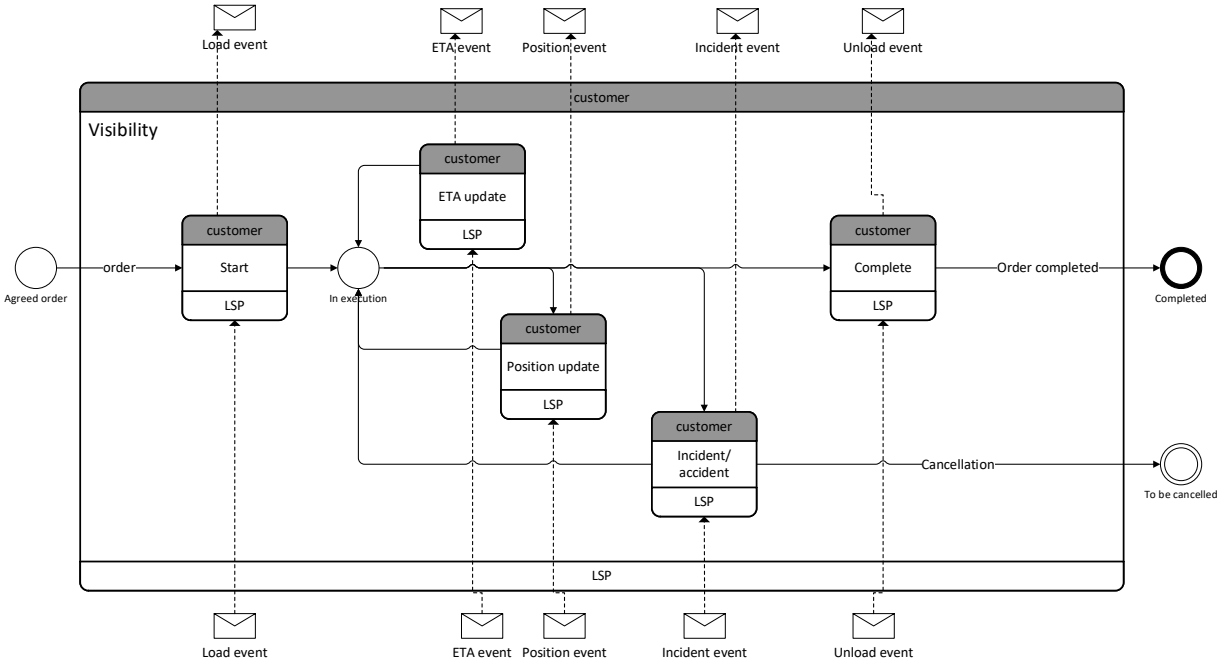
- A customer participating in an order will receive all relevant visibility events for that order from its service provider. These are the events that are formulated by the visibility pattern.
- A competent authority (CA) will receive all visibility events of cargo that passes and is loaded and/or discharged in its competency domain. The following rules are implemented by the event distribution:
    - PLA (Place of Acceptance) is in the territory of a CA → CA will receive a load event.
    - PLD (Place of Delivery) is in the territory of a CA → CA will receive a load (also if PLA is not in its territory) and a unload event.
    - Passing through a territory of a CA: border crossing events for entry and exit of the territory will be shared with the CA.

Assumptions:

1. **Competent Authorities** – these will always receive load/unload events as specified by the event distribution for demonstration purposes, independent of any regulation.
2. **CA territory** – for demonstration purposes, the territory is a country. The country code is part of the UNLOCODE of PLA/PLD.
3. **Customer order data** – this is stored by the participant acting as service provider and contains a customer identification.
4. **Multimodal** – in case a transport leg is outsourced by a service provider, that service provider acts as customer of that leg. It will receive all relevant events, but only pass those to its customer that represent the first (load) and last (unload) relevant to that customer (see the Codognotto example).

Interface specifications for the visibility pattern

The interface specifications are based on the visibility interaction pattern shown in the next figure. It consists of activities by which events can shared between a customer and service provider, where these events can also be shared with an authority. For instance, a service provider submits a load event to its customer, followed by an ETA event. The following events are supported: load event, ETA event, Incident event, and Unload event. Their allowed sequencing is given in the following diagram, where circles represent a state (states: agreed order, in execution, completed, to be cancelled), rectangles represent data sharing processes (processes; start, ETA update, Position update, Incident/accident, complete), and envelopes with an arrow the initiation of an event by one of the roles (the blank role like 'LSP' for 'start' process) and the other the recipient (the grey role like 'customer' for the start process).



The 'agreed order' state data must be initially shared between a customer and service provider to trigger sharing events. This 'agreed order' is the basis of a document data set (like the eCMR) that can be produced at state 'in execution'.

The most basic example of interactions between a customer and LSP are by sharing a load event, followed by an ETA event, and completed with an unload event.

The 'agreed order' state contains the following data set (functional expressed by the semantic model):

Main structure of states	Agreed order	
	Load	Unload
Event (state data)		
UUID (event)	x	
UUID (sender organisation)	x	
event type (value - assigned for the abstract message tree)	x	
milestone		
expected date/time		
estimated date/time		
actual date/time		
External reference (this is the user identification)	x	
Reference type (related to the abstract message tree)	Order number	
Organizations		
Organization Role: consignor, consignee, carrier	x	
UUID Organization	x	
Cargo (at least goods or equipment as a rule; can be multiple	x	
For goods - UUID (goods)	o (details)	
For goods - UUID (goods)	x (totals)	
for equipment - UUID (equipment)	o	
UUID (transport means)	x	
Person - legal person		
UUID	x	
External ID	x	
External ID provider	x	
Name	x	
Infrastructure - Location		
UUID	x	
Location code	x	
Location code type	x	
Address	x	
Postal code	x	
Street name	x	
City name	x	
Country code	x	
Digital Twin - goods		
UUID	x	
External ID	x	
Type of external ID	x	
Type of packages	x	
Number of packages	x	
Gross mass (kg)	x	
Net mass (kg)	x	
Gross volume (m3)	x	
remark		
Digital Twin - equipment		
UUID	x	
Equipment ID	x	
Equipment Type	x	
Equipment Size	x	
Transport Equipment Packed Status	x	
Remark		
Digital Twin - transport means		
UUID	x	
Transport means ID		
Transport means ID provider		
Transport means Nationality		
Transport mode	x	
Event (organization location)		
UUID (event)	x	
UUID (sender organisation)		
event subtype		
milestone	start	
expeted date/time		
estimated date/time		
actual date/time	x	
External reference		
Reference type		
UUID Organization	x	
UUID (location)	x	
Event (visibility data)s		
UUID (event)	x	x
UUID (sender organisation)	x	x
event type (value - assigned for the abstract message tree)	x	x
milestone	start	end
expected date/time	x	x
estimated date/time	x	x
actual date/time		
Locations (at least two)		
Location role: place of acceptance, place of delivery	PLA	PLD
UUID (location)	x	x
Cargo (at least goods or equipment as a rule; can be multiple	x	x
For goods - UUID (goods)	o (details)	o (details)
For goods - UUID (goods)	x (totals)	x (totals)
for equipment - UUID (equipment)	o	o
UUID (transport means)	x	x

This state contains data reflecting a customer order:

- General event data (event (state data) reflecting header data. This refers to cargo (goods or equipment), organizations involved (consignor, carrier, consignee), and a transport modality and/or means.
- Associations are via UUIDs (Universal Unique Identifiers)
- Each concept (organization, location, etc.) has a user interpretable identifier like an equipment id.
- Equipment is generic, in the sense that reflects a trailer, container, or any other type of equipment.
- Actual details of the movement of goods or equipment are given by two visibility events, one with the Place of Acceptance (PLA) and the other with the Place of Delivery (PLD).
- If required, any intermediate location can be included, for instance that of border crossing for cargo going into or moving out of the EU.
- Any queries on individual concepts (like Digital Twin – goods) will only result in those data properties given for these concepts.

Next, the events are specified as follows:

Visibility events		Load	ETA	Incident	Position	Unload
	UUID (event)	✅	✅	✅	✅	✅
	UUID (sender organisation)	✅	✅	✅	✅	✅
	milestone	start	start	start	start	end
	estimated date/time		✅			✅
	actual date/time	✅		✅	✅	
	External reference	✅				
	Reference type	eCMR data set				
	Location					
	Location role: place of acceptance, place of delivery	PLA	PLD	Position	position	PLD
	UUID (location)	✅	✅	✅	✅	✅
	Cargo (at least goods or equipment as a rule; can be multiple)	✅	✅	✅		✅
	For goods - UUID (goods)	✅ (o)	✅ (o)	✅ (o)		✅ (o)
	for equipment - UUID (equipment)	✅ (o)	✅ (o)	✅ (o)		✅ (o)
	UUID (transport means)	✅			✅	
	UUID (equipment-trailer)	✅				
Digital Twin - transport means						
	UUID	✅				
	Transport means ID	✅				
	Transport means ID provider	✅				
	Transport means Nationality	✅				
	Transport mode	✅				
Digital Twin - goods						
	UUID	✅		✅		✅
	Number of packages	✅ (o)		✅ (o)		✅
	remark	✅		✅		
Digital Twin - equipment						
	UUID	✅ (o)		✅		✅
	Equipment ID (authorities only)	✅ (o)				
	Equipment Type (authorities only)	✅ (o)				
	Equipment Size (authorities only)	✅ (o)				
	Remark	✅ (o)		✅		

The load event may have a reference to an eCMR data set (optional), which can be the customer order reference. This reference is not required since:

- For a customer -service provider business relation, this reference is not required. It is based on the existence of a customer order.
- An authority also does not require this reference, since it can search on other criteria like 'transport means ID' (license plate of a truck, vessel code, etc.) and equipment identification (container number, license plate of a trailer, wagon number, etc.).

The **assumption** is that a SPARQL query on a load event UUID results in the complete data set given as 'event (state data)'.

All data sets will be expressed as SHACL constraints to the semantic model and configure the semantic adapter.