Data Platform for the Connection of Cognitive Ports

Title:	Document Version:
D3.4 - Permissioned Blockchain network M18	1.0

Project Number:	Project Acronym:	Project Title:
H2020-871493	DataPorts	A Data Platform for the Cognitive Ports of the Future

Contractual Delivery Date:	Actual Delivery Date:	Deliverable Type*-Security*:
M18 (June 2021)	M18 (June 2021)	O-PU

*Type: P: Prototype; R: Report; D: Demonstrator; O: Other; ORDP: Open Research Data Pilot; E: Ethics.

**Security Class: PU: Public; PP: Restricted to other programme participants (including the Commission); RE: Restricted to a group defined by the consortium (including the Commission); CO: Confidential, only for members of the consortium (including the Commission).

Responsible:	Organisation:	Contributing WP:
Fabiana Fournier	IBM	WP3

Authors (organisation):	
Fabiana Fournier (IBM)	Inna Skarbovsky (IBM)
Miguel Llop (VPF)	Pablo Gimenez (VPF)
Vicente Perales (VPF)	Daniel Garcia Latorre (EVR)
Alejandro Aparicio Blasco (EVR)	Daniel Vilas (EVR)
Konstantinos Glykos (CERTH)	Sofia Terzi (CERTH)
Alexandros Zerzelidis (CERTH)	Vasilis Siopidis (CERTH)

Abstract:

Deliverable 3.4 – Permissioned blockchain network delineates the overall blockchain (BC) implementation activities in the scope of the Dataports project and presents the design of the three blockchain networks that have been devised in the project, namely the Data Governance, Verify Gross Mass, and Container Pick-Up that realize the off-chain and on-chain BC applications. The main outcomes of these activities are: Design of the three blockchain networks; implementation of the IDS broker and clearing house as part of the data governance BC network; placement of the blockchain components in the overall DataPorts platform and their interactions with the platform components; definition of a minimum viable product for each of the networks; set-up of three BC infrastructures for the hosting of the three BC networks; and an incremental plan for implementation. The data governance blockchain solution can be applied to any scenario as a means of defining and enforcing access rules to assets in a transparent, verifiable, trackable, and immutable manner, thus making this a generic component that can be replicated in any domain in which such a requirement fits. Without the loss of generality, the blockchain networks at the ports, although being specific to certain processes, can be replicated to similar processes in other ports as well.

Keywords:

Hyperledger Fabric, blockchain, blockchain network, governance

Revision History

Revision	Date	Description	Author (Organisation)
V0.1	15.05.2021	Sections 1 and 2	Fabiana Fournier (IBM)
V0.2	30.05.2021	Second version, adding content in Section 5	Fabiana Fournier (IBM)
V0.3	02.06.2021	Adding content to Section 4	Fabiana Fournier (IBM)
V0.4	03.06.2021	Update of Section 2.2	Fabiana Fournier (IBM)
V0.5	04.06.2021	Inclusion of Sections 4.3 and 4.4.5	Fabiana Fournier (IBM)
V0.6	06.06.2021	Inclusion of Section 6	Fabiana Fournier (IBM)
V0.7	10.06.2021	Polishing of entire document	Fabiana Fournier (IBM)
V0.8	11.06.2021	Adding Abstract and Conclusions	Fabiana Fournier (IBM)
V0.9	12.06.2021	Version ready for internal review	Fabiana Fournier (IBM)
V0.10	22.06.2021	Version ready for submission	Fabiana Fournier (IBM)
V1.0	25.06.2021	Final version	Santiago Cáceres (ITI)

This project has received funding from the European Union’s Horizon 2020 Research and Innovation Programme under Grant Agreement № 871493.

More information available at https://DataPorts-project.eu

Copyright Statement

INDEX

LIST OF FIGURES

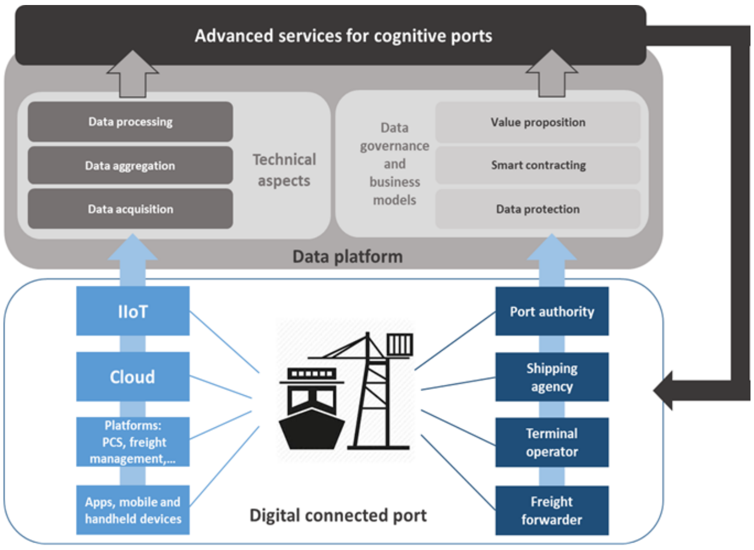
LIST OF TABLES

1 INTRODUCTION

1.1 DATAPORTS PROJECT OVERVIEW

DataPorts is a project funded by the European Commission as part of the H2020 Big Data Value PPP programme, and coordinated by the ITI - Technological Institute of Informatics. DataPorts rely on the participation of 13 partners from five different nationalities. The project involves the design and implementation of a data platform, its deployment in two relevant European seaports connecting to their existing digital infrastructures and addressing specific local constraints. Furthermore, a global use case involving these two ports and other actors and targeting inter-port objectives, and all the actions to foster the adoption of the platform at European level.

Hundreds of different European seaports collaborate with each other, exchanging different digital data from several data sources. However, to achieve efficient collaboration and benefit from AI-based technology, a new integrating environment is needed. To this end, DataPorts project is designing and implementing an Industrial Data Platform.



The DataPorts Platform aim is to connect to the different digital infrastructures currently existing in digital seaports, enabling the interconnection of a wide variety of systems into a tightly integrated ecosystem. In addition, to set the policies for a trusted and reliable data sharing and trading based on data owners' rules and offering a clear value proposition. Finally, to leverage on the data collected to provide advanced Data Analytic services based on which the different actors in the port value chain could develop novel AI and cognitive applications.

DataPorts will allow establish a future Data Space unique for all maritime ports of Europe and contribute to the EC global objective of creating a Common European Data Space.

1.2 DELIVERABLE PURPOSE AND SCOPE

This document provides a comprehensive report on the design of the three blockchain networks identified in the DataPorts project: Data Governance Services, Verified Gross Mass, and Container Pick-Up.

Specifically, the Description of Action (DoA) states the following regarding deliverable D3.4 - Permission Blockchain Network: "This deliverable will implement the permissioned blockchain infrastructure that will define data models, authentication and authorization mechanisms, and that will integrate with the rest of components of the platform". Specifically, D3.4 relates to Task 3.5 – Blockchain implementation led by IBM and carried out by partners IBM, ITI, EVR, and CERTH.

From the early stages of the project, all blockchain activities have been dealt together under the umbrella of "blockchain implementation activities" encompassing all phases of development, from requirements gathering, through design, implementation, and testing of the blockchain applications devised in the project. This enabled a better coordination and harmonization of the different activities carried out. One of the first outcomes of this activity was the identification of a new role for blockchain technology in addition to the one specified in the DoA that is applying blockchain technology to manage the access control of datasets pertaining to users of the DataPorts platform. The idea was the application of blockchain to improve operations in the ports. As a result, a new role for blockchain has been born and has been applied in the context of the Verified Gross Mass use case in the port of Valencia and the Container Pick-Up use case in the port of Thessaloniki.

The purpose of this document is to present the design work that has been made for each of the three blockchain networks in the project. Infrastructures for the three blockchain networks have been put in place and implementation of three minimum viable products for the three networks have been carried out and will be the core of next deliverable (D4.2 Blockchain Based Data Governance Rules M20).

1.3 DELIVERABLE CONTEXT

Its relationship to other documents is as follows:

Primary Preceding documents:

- D2.3 – Blockchain design specification submitted at M9. Provided the reader with an introduction to blockchain technology, comparison among available blockchain platforms, and selection of Hyperledger Fabric as the underlying blockchain platform in the DataPorts project.

Primary Dependant documents:

- D2.4 - Platform architecture and specifications to be submitted at M24. This deliverable will present the different components representing the DataPorts platform and their interrelationships. Therefore, the position of blockchain in the overall platform. Although this report will be submitted at a later stage, we rely on intermediate results of a living document maintained by the technical partners in the project.
- D4.2 – Blockchain based data governance rules to be submitted at M20. This deliverable will complement the design introduced in the report in hand with details on the implementations and demonstrators.

1.4 DOCUMENT STRUCTURE

This deliverable is broken down in the following sections:

- **Section 1** Introduction: summary of the deliverable objective, scope, and structure.
- **Section 2** Blockchain roles in DataPorts architecture: placement of the blockchain services in the overall DataPorts platform architecture and the roles they fulfil.
- **Section 3** Technical objectives: alignment of blockchain technology to the stated project objectives.
- **Section 4** Governance blockchain network: description, requirements, user stories, and architecture for the Governance blockchain Network.
- **Section 5** Data Sharing blockchain network – Verified Gross Mass use case: description, requirements, user stories, and architecture for the VGM use case blockchain application.
- **Section 6** Data Sharing blockchain network – Container Pick-Up use case: description, requirements, user stories, and architecture for the CPU use case blockchain application.
- **Section 7** Conclusions and next steps: actions planned for the following months.

1.5 DOCUMENT DEPENDENCIES

This document is part of an iteration of living deliverables and constitutes the first variant, while the second and last version will be delivered by M30.

2 BLOCKCHAIN ROLES IN DATAPORTS ARCHITECTURE

DataPorts platform implements blockchain technology taking as reference Hyperledger Fabric[1], the most mature permissioned blockchain (open source) available today (for a comparison of permissioned blockchains refer to D2.3[2]. Hyperledger Fabric (simply Fabric) provides distributed ledger and state database.

Blockchain (BC) basic elements and principles are described in D2.3 Blockchain design specification[2], submitted in M9. Therefore, we assume the reader is familiar with BC basic terminology and that concepts such as peers, orderers, certificate authority (CA), channels, chaincodes (smart contracts in Fabric), and private collections, are out of the scope of this document. For more details on Fabric the reader can refer to [1].

This section describes the evolution of the BC component in the overall DataPorts platform architecture and the three BC networks that represent the different roles of BC in the project.

2.1 BLOCKCHAIN EVOLUTION IN DATAPORTS

During the writing of the proposal, as reflected in the DoA, blockchain (BC) technology was intended to provide a trusted environment for the operations in ports where data could be exchanged among partners of the business network. With this in mind, the role of BC was to enable a mechanism to define the access rules for data-by-data providers on the one hand, and to monitor the actual access to the data by data consumers, on the other hand. The envisioned data governance component in the DataPorts platform underpinned by BC technology offers a trusted platform for enabling secure data exchange/sharing between data providers and data consumers of the platform, where the data remains at data owner’s premises and exchanged using peer-to-peer communication mechanisms, and the data access rights are declared and verified using smart contracts.

Following the International Data Spaces (IDS) architecture [2], data owners remain in full control of their data and might change at any time the terms of the data usage. Furthermore, BC was leveraged to act as IDS broker and clearing house components in an IDS compliant architecture. This role of BC is described in detail in Section 4.

Already at the first stages of the project it was evident that some use cases of the ports might require applying BC also in some of the ports’ processes for the secured and trusted sharing of data. To this end, the data should be stored on the ledger (data on-chain) as opposed to data stored off-chain as in the case of the governance policies application mentioned before. Examples of these use cases for the port of Valencia and Thessaloniki are described in detail in Sections 5 and 6 of this document.

Table 1 shows the main differences between the two types of BC networks applied in the project, namely BC for governance rules and BC for data sharing.

	Blockchain for governance rules	Blockchain for data sharing
What for? (Role)	Blockchain manages consent of access to specific document/data	Blockchain records transactions related to shared data and processes
When?	P2P data exchange	Data sharing among participants in the business network

[1] https://www.hyperledger.org/projects/fabric

[2] https://dataports-project.eu/wp-content/uploads/2020/10/DataPorts_D2_3_Blockchain_design_specifications_M09_pu_v1_1.pdf

	Blockchain for governance rules	Blockchain for data sharing
How?	Smart contract decides whether a participant is allowed to access a document based on the invoker's credentials and access rules for the particular dataset	Smart contract records transactions related to shared data and processes of all participants in a business network
Data	Off-chain (owner's premises)	On-chain
Why? (Added value)	Immutability, auditability, provenance, transparency, traceability, trackability, and non-repudiation of **access to information stored in owner's repositories**	Immutability, auditability, provenance, transparency, traceability, trackability, and non-repudiation of **information stored on the chain (all transactions related to this data)**

Table 1 - Comparison between two roles of blockchain networks

2.2 BLOCKCHAIN PLACEMENT IN THE DATAPORTS PLATFORM ARCHITECTURE

A direct outcome of the BC role evolution is the placement of BC services in the overall DataPorts platform high level architecture (Figure 1) and in the DataPorts platform architecture (Figure 2). These figures show the interrelations of the two types of network roles with other services and components in the DataPorts platform and are taken from the current version of D2.4 Platform Architecture and Specifications to be submitted at M24 of the project.

BC for governance rules is an integral part of the DataPorts platform and implements data services offered by the platform. Four main conceptual components in this BC network are: Clearing House, IDS Broker, Identity Management, and the Data Governance Rules implemented as smart contracts (chaincodes in Hyperledger Fabric jargon) that enforce the definition and management of the access rules policies (see Section 4.4.1.).

Within the platform architecture, the Data Access component, and the Semantic Interoperability component work in conjunction with the Data Governance Services component, providing the data and semantic rules to be incorporated into the smart contracts and the metadata written into the ledger itself.

The Data Governance Services aim to register, evaluate, and enforce access rules for the datasets registered in DataPorts and, to this purpose, communication between the different components must be achieved. In this querying and registration process that takes place in the platform, the blockchain is the ultimate source of truth, and every operation regarding data governance ends with a transaction evaluated by the data governance smart contracts, that allows or denies the registration or consumption of the desired datasets. Data Governance Services has a point of access from the connectors of the data owners and data consumers through a Representational State Transfer Application Programming Interface (REST API) that makes use of the Hyperledger Fabric Software Development Kit (SDK) for network access. This BC network is a generic component and part of the overall platform. In fact, it can be applied and replicated in any domain in which data is off-chain and BC serves as a means for defining and enforcing access policies.

In contrast, BC networks for Data Sharing are implementations supporting internal port business processes requiring transparent, verifiable, and trusted data sharing within the port's ecosystem. These implementations are custom and tailored to particular use cases which benefit from BC capabilities. From DataPorts platform point of view, such BC shared data networks can be added as a data source to the platform by the means of semantic interoperability layer the same way as other applicative data sources (Figure 1 and Figure 2). This document describes two of these use cases: one in the port of Valencia (Section 5) and one in the Thessaloniki port (Section 6).

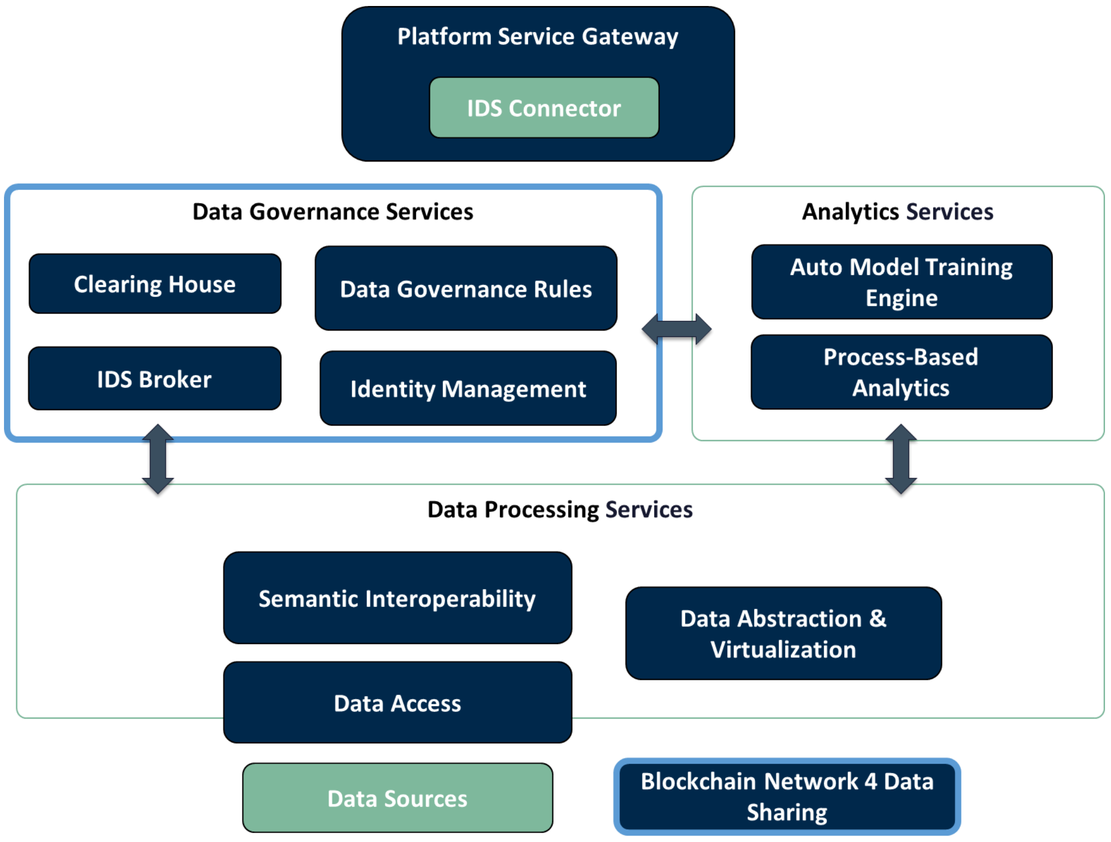


Figure 1 - Blockchain placement in DataPorts high-level overview platform

As can be seen, Figure 2 is a more detailed view of Figure 1. Specifically, with regards to the Data Governance Services, we can see the decomposition of the main logical components as follows (Table 2):

Figure 1	Figure 2
IDS Broker	IDS Broker
Data Governance Rules	Data Governance Rules
	Register Access Rules
	Evaluate Access Rules
Clearing House	Clearing House
	Transaction Log
Identity Management	Identity Management
	Access Control List/Managed Service Provider

Table 2 - Data Governance Services logical components

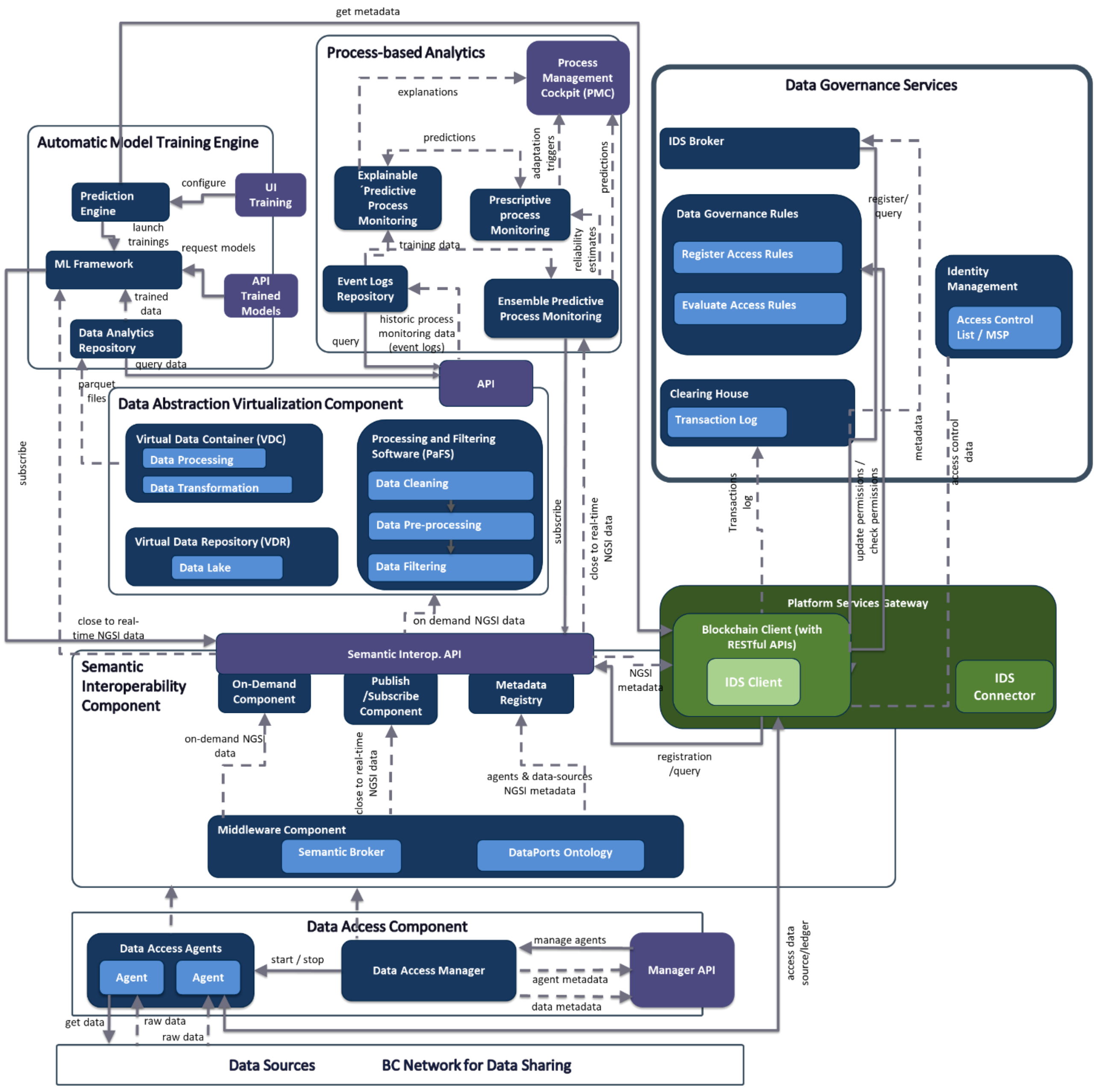


Figure 2 - Blockchain placement in DataPorts platform architecture

3 TECHNICAL OBJECTIVES

DataPorts platform aims to achieve the following technical objectives:

- O1. To address real-life data market use cases in two relevant European seaports, two global use cases including pilot deployment and evaluation of progress against benchmarking-existing deployments KPI's.
- O2. To design and validate next generation set of advanced interoperable data related and AI based services.
- O3. To define an engineering methodology facilitating application of DataPorts architecture and tools, to support cognitive, privacy-aware and secure data sharing processes.
- O4. To define, design and incorporate a novel, scalable, resilient, semantic approach for data sharing that builds upon existing semantic interoperability solutions.
- O5. To define, design and provide a data governance framework and associated tools for the DataPorts architecture.

Deliverable D3.4 relates to both O3 and O5 by providing a secured framework for the data governance and data sharing for ports operations in the project. More specifically, the blockchain ledger ensures:

- Full provenance of each transaction or access to data in the DataPorts platform.
- A non-repudiation process in case of dispute.
- Transparency, trackability, and traceability of transactions.
- Immutable single source of truth for all transactions.

In other words, BC brings the full confidence in the usage and access of data for the stakeholders in the network, hence enhancing the security of data sharing. Section 4 of this document focuses on the data governance framework (O5) while Sections 5 and 6 present the data sharing aspects of port processes (O3 and O5).

In addition, this deliverable belongs to WP3 which is devoted to the development of the core components of DataPorts platform. Specifically, it relates to Objective 3.5 – to provide a permissioned BC network listed below within the entire list of WP3 objectives for completeness.

- O3.1 To identify different data sources to be integrated in the DataPorts data platform, including the mechanisms to store and facilitate data management.
- O3.2 To define ontologies, mechanisms and enablers to provide semantic interoperability with data platforms, IoT devices, robots and other data sources, and develop the semantic-based tools needed to facilitate generation of interfaces for sensing and actuation required in the DataPorts data platform.
- O3.3 To define mechanisms for data management, virtualization including streaming techniques.
- O3.4 To define for the real-time processing, big data analysis and visualization of current and historic context data.
- O3.5 To provide a permissioned blockchain network for exchange and share data across stakeholders in seaports supply chains.
- O3.6 To provide a set data analytics services and AI based algorithms to develop cognitive applications in the context of the involved pilots.

Following sections describe in detail the design of the three BC networks that play the two roles aforementioned: governance rules and data sharing.

4 DATA GOVERNANCE SERVICES

The governance blockchain network enables the handling of data, monitoring its complete lifecycle from the registration of the dataset to its consumption. It increases consistency and confidence of the data registered and improves data security.

The main functionalities of the data governance component consist of data exchange between the participants in a decentralized way through a blockchain network and storing evidence of the transactions carried out in the system, providing it with immutability and being a source of auditability of the operations that have taken place between the data providers and data consumers.

Data governance also establishes sharing rules through smart contracts for each dataset. Smart contracts will enable, revoke, or deny the consumption of datasets based on compliance with the privacy policies defined when the dataset is uploaded. Actions carried out by the participants are stored in the blockchain after the smart contract evaluation is successful, obtaining an auditable and immutable record of the datasets provided and consumed by each organization. The data governance component should also include special access policies, such as requiring special permission from more than one party to access the data.

It is important to note that the data governance services can be applied to any scenario as a means of defining and enforcing access rules to assets in a transparent, verifiable, trackable, and immutable manner, thus making this a generic logical component that can be replicated in any domain in which such a requirement fits. As the data governance services is implemented as a blockchain network, we use *data governance services* and *governance blockchain network* interchangeably throughout this document.

4.1 GOALS

The inclusion of a blockchain network in DataPorts for data governance purposes aims to provide a comprehensive framework in which data ownership and data distribution policies become an integral part of the entire DataPorts platform. It should provide various governance, establishing the rules for data consumption in DataPorts. Furthermore, the Governance blockchain network aims to facilitate a secure exchange of information between participants in the DataPorts platform. Any DataPorts member should be able to access the data governance component and publish a dataset under certain consumption rules. In the same way, organisations that wish to take advantage of a dataset published in DataPorts, can request access to it and rely on data governance policies to be enacted when necessary.

This way of operation ensures that each dataset in the platform is accounted for, and has specific rules for sharing and consumption, also leaving a trail of transactions that will tell an accurate story of the dataset lifecycle since its registration in the platform. The transactional nature of the data governance policies makes their incorporation in a blockchain a seamless operation.

4.2 REQUIREMENTS

Table 3 details the set of requirements that have been identified for proper development of the Governance blockchain network as specified in current version of D2.4 Platform Architecture and Specifications to be submitted at M24.

ID	Description	Category
R2.2	The DataPorts platform must provide a secure interface framework for data exchange between itself and the potential data sources	Non-Functional
R3.11	As a blockchain organization I want to be able to interconnect with other organizations within my network in an agreed upon manner to facilitate trust and transparency	Non-Functional

ID	Description	Category
R3.12	As a blockchain organization I would like to be able to onboard an existing blockchain network and participate as a validating member	Functional
R3.13	As a participating blockchain organization I would like to be able to agree on a consensus algorithm which will provide pilot appropriate transaction validation and ordering functionality for the network	Functional
R3.14	As a participating blockchain organization I would like to be able to securely sign in my network components (peers, CAs) and clients (end-users/application clients) to a blockchain network	Non-Functional
R3.15	As a participating blockchain organization I would like to be able to maintain my own copy of distributed ledger while being assured other organizations on the network have exact same replica of the ledger.	Functional
R3.16	As a participating blockchain organization I would like to be able to endorse transactions by executing agreed upon business logic specified as a binding contract between network organizations	Functional
R3.17	As a participating blockchain organization I would like to be assured of immutability and permanent availability of the shared data	Non-Functional
R3.39	As a data provider, I want to retrieve the history of transactions, so that I am able to track the information flow	Functional
R4.1	As a participating blockchain organization, I would like to be able to develop and run agreed upon business logic in a distributed manner and agree on results	Non-Functional
R4.2	As a participating blockchain organization, I would like to share data in a privacy preserving manner with selected organizations within the business network	Non-Functional
R4.3	As a participating Blockchain organization, I would like to enrol my client applications and end users to access and share data in my blockchain network in a secure manner using appropriate authentication	Non-Functional
R4.5	As a data provider, I would like to have an agreement with data consumer about data manipulation	Non-Functional
R4.6	As a data provider, I would like not only to grant access to data consumer, but also to revoke access	Non-Functional
R4.7	As a participant Blockchain organization, I want to set data governance rules, so that data access is specified	Non-Functional
R4.12	The DataPorts Platform must support the security and governance recommendations from the IDS reference architecture in order to share and transfer data securely	Non-Functional

Table 3 - Governance blockchain network requirements

One of the first tasks in the design of any BC network is the identification of potential players in the network. We denote network participant or a network stakeholder as any entity that has either read or/and write access to the blockchain network (this entity requires to have a properly registered blockchain organizational identity and specified access rights to the blockchain network). We denote as blockchain organization a network participant who shares a copy of the ledger (meaning an organization which has a peer node in the blockchain network) and/or organization that has an orderer node (in most cases it means it also has access to all the ledger data). End users are network participants applications that gain access to the BC network, by invoking functions offered by a blockchain client, which in turn invokes smart contract functions, or the organizational members who are users of such applications.

A network participant (e.g., an application or a user) performs the role of data provider when it provides data to others and writes it on blockchain and performs the role of data consumer when it collects data from others. On the other hand, a data consumer receives event notifications from blockchain and pushes these notifications to the participant's systems (e.g., a platform or an application), as well as it reads data from blockchain on demand.

In the context of the Data governance network, any participant in the DataPorts ecosystem can potentially interact with the blockchain to perform any of these actions: upload metadata of a dataset and define the access rights to it, search for datasets and request access to an existing dataset, revoke access to an uploaded dataset, and view audit log of registration and access of datasets (Figure 3). All interactions are done via the gateway as depicted in Figure 2 and detailed in Section 4.4.1.

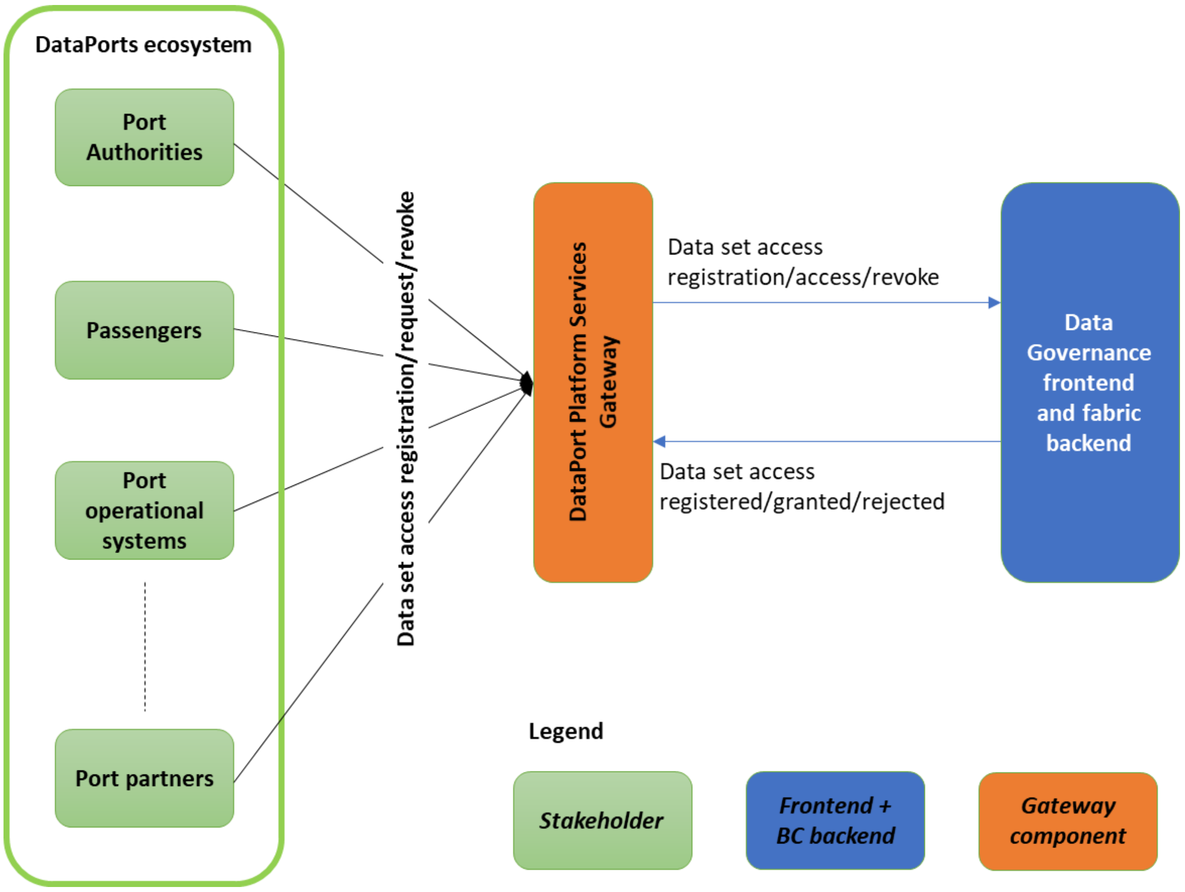

DataPorts ecosystem
Port Authorities
Passengers
Port operational systems
Port partners
Data set access registration/request/revoke
DataPort Platform Services Gateway
Data set access registration/access/revoke
Data set access registered/granted/rejected
Data Governance frontend and fabric backend
Legend
Stakeholder
Frontend + BC backend
Gateway component

Figure 3 - Data providers and data consumers for the Governance blockchain network

The following functionalities need to be supported by a DataPorts ecosystem participant:

- Register metadata of a dataset
- Provide access to a dataset
- Revoke access to a dataset
- Define contract agreement terms and conditions (e.g., expiration date)
- Get notifications of revoked access to a dataset

- View access rights to datasets
- View history of access rights to a dataset
- Search for datasets

4.3 USER STORIES

The **user stories** associated with this use case are the following:

As a *network admin* I want to use DataPorts Governance blockchain business network to:

- Register organizations and users.
- View information for registered organizations and users.

As an *organization admin* I want to use DataPorts Governance blockchain business network to:

- Update my organization's information.
- Register my organization's users.
- View information for my organization and the users of my organization.
- Act sometimes as a data provider and sometimes as a data consumer.

As a *user* I want to use DataPorts Governance blockchain business network to:

- Act sometimes as a data provider and sometimes as a data consumer

As a *data provider* I want to use DataPorts Governance blockchain business network to:

- Upload/Update metadata of my data while unauthorized access is prohibited
- Set data governance rule for my data
    - Access rights (READ/WRITE/SHARE)
    - Contract agreement
    - Terms and condition
    - Legislation
    - Custom access rights
- Get notified when there are access (READ/WRITE/SHARE) requests for my data from data consumers
- Grant/Deny/Revoke access of owned data
- View my datasets
- View the authorized users on my datasets and their access permission

As a *data consumer* I want to use DataPorts Governance blockchain business network to:

- View available metadata of data
- Request access (READ/WRITE/SHARE) to data
- View my permission of provided data
- Be informed about the changes regarding to specific data

In the second phase, the processes about granting access will be extended to organization level and public level. Moreover, more information about transactions history will be provided.

4.4 BLOCKCHAIN ARCHITECTURE

In this section, the main aspects about the blockchain architecture are discussed. The following subsections provide deeper insights on the components that form the architecture of data governance in the DataPorts platform.

4.4.1 Architecture overview

As previously mentioned, the Governance blockchain network is built upon the Hyperledger Fabric that enables corporations to deploy permissioned blockchain networks. It also allows for separation of information between participants in the same network, making it a suitable framework for the blockchain-related components in DataPorts. This distributed ledger is a database that holds a cryptographically linked series of transactions in a way that its forging or counterfeiting is impossible, providing seamless auditing capabilities for all partners to take advantage of. Specifically, in the Governance blockchain network, every relevant organisation that desires to upload and share datasets in the platform must register within it and provide a network peer to be part of the channel of communication between partners.

As part of the DataPorts declared functionalities (refer to F2.6 in D2.4 Platform Architecture and Specifications to be submitted at M24), the DataPorts platform is designed to be aligned with IDS (International Data Spaces) reference model. IDS is a virtual environment, leveraging existing standards and technologies, as well as governance models well-accepted in the data economy, to facilitate secure and standardized data exchange and data linkage in a trusted ecosystem. The premise of IDS is that the data owner retains full sovereignty over his data sources, while utilizing the components defined in the reference architecture to facilitate secure and trusted sharing of data sources with data consumers, supported by total transparency and auditability of the data exchange process [2].

Figure 4 shows the components and the interactions among components of the IDS reference architecture [2]. The two most important components of the IDS reference architecture model are the *broker* and *connector*, which, in fact, realize the decentralized architecture and ensure scalability.

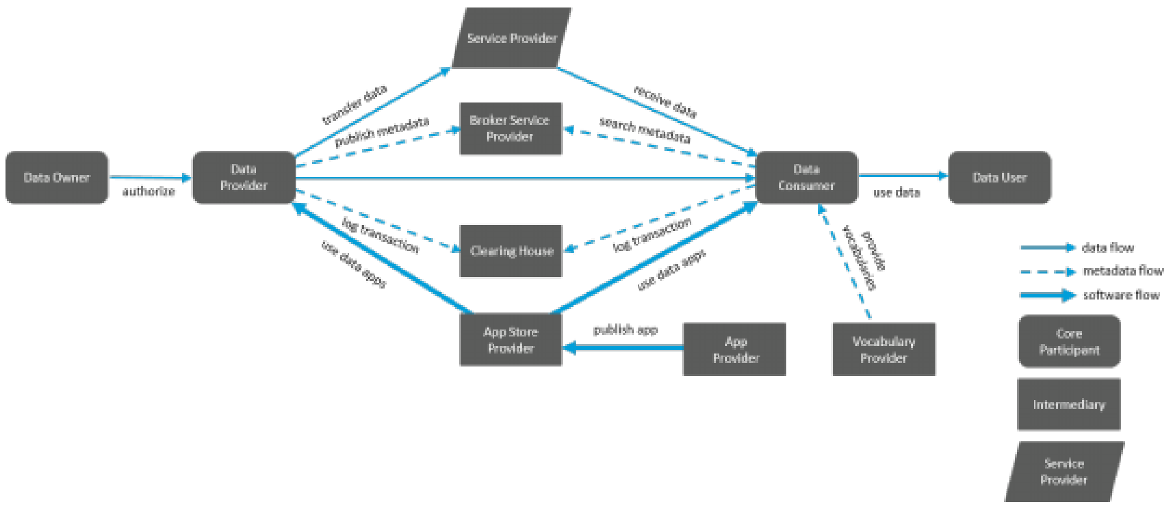


Figure 4 - Roles and interactions in the Industrial Data Space (source [2])

The *Broker Service Provider* is an intermediary that stores and manages information about the data sources available in the International Data Spaces. The activities of the Broker Service Provider mainly focus on receiving and providing metadata. The Broker Service Provider must offer an interface for *Data Providers* to send their metadata. The metadata should be stored in an internal repository for being queried by *Data Consumers* in a structured manner.

The *IDS Clearing House* provides decentralized and auditable traceability of all transactions. It receives logging information about all activities performed during data exchange and confirmation of successful dispatch and reception of data.

In the scope of the DataPorts project we have leveraged blockchain technology and implemented the IDS Broker and the Clearing House as part of the Data Governance Services offered by the DataPorts platform as explained below.

Figure 5 presents the data governance blockchain architecture overview. The users of the data governance use case act as either a data provider or data consumer through the frontend. A Node.js middleware application acts as an intermediary between the frontend and the chaincode installed on the Hyperledger Fabric network. Communication with the chaincode is achieved through the Hyperledger Fabric Client (HFC) subcomponent. The HFC client manages the query/invoke requests to the blockchain network, where we have deployed our data governance chaincode. These smart contracts satisfy the need for application functionality relating to data governance. The application simulates the IDS functionality while not being presently IDS-compatible. Going into a little more detail, we can see that the data provider is able to upload/update the metadata of their data onto the ledger. In addition, they can set/update the data governance rules for each of the data they own. Also, most of the IDS broker fields are present in our metadata fields (Note that the IDS metadata schema is part of deliverable D4.2 Blockchain based data governance rules to be submitted at M20, i.e., August 2021). It is worth noting that the data itself is stored off-chain but the metadata and the rules regarding data access are stored on-chain.

Implementing the broker as a smart contract ensures secure access to immutable data. The chaincode evaluates access based on the granted/revoked access rights. Moreover, at this stage of implementation, the identity management is blockchain based. Finally, using the blockchain offers us a “free”, so to speak, implementation of the clearing house component through the transaction log. In other words, we use the transaction log to simulate the clearing house component.

Broker chaincode allows to register all the metadata pertaining to the dataset a data provider wishes to share, and which is stored at his premises, and provides capabilities of querying this metadata based on various search parameters.

Governance rules chaincodes allow to register the access rules, to define who can access the datasets published in the broker and how, and to evaluate these access rules once a particular data consumer wishes to access the dataset. Upon evaluation of the access rules for a particular dataset and data consumer, the access is either granted or rejected. These chaincodes also allow to manage the lifecycle of the access permissions, allowing to query or revoke the granted access. Additionally, the chaincodes provide user and organization management, allowing to register and query users and organizations (the access rules are defined for a specific dataset for particular users/organizations).

As mentioned above, the clearing house, whose function is to allow provenance and audit on all dataset access requests/granted permissions is implemented using inherent built in blockchain functionality. Blockchain retains history of all transactions executed on the ledger using chain of blocks, to which each new transaction is appended. This history is tamper-proof, immutable, and replicated to all the peers in the network, and therefore can serve as a single verifiable source of truth for audit. For each artifact on the ledger (in our case datasets and granted permissions) Fabric provides us with APIs to check when and by whom the artifact was accessed and modified. This serves as a perfect implementation for the functionality required by the clearing house component.

Identity management is inherent in any permissioned blockchain implementation. Each blockchain organization defines a Certificate Authority (CA) server, which is responsible for issuing crypto material (certificates and public and private keys) for all entities in the network. These include the blockchain nodes (peers and orderers), and the clients of the network (whether these are applications connecting to blockchain with the help of Fabric SDK, or end-users, authenticating, connecting, and executing transactions with the help of these applications). Each transaction on the chain, as well as all additional messages (such as, peers transaction evaluation responses and orderer messages to peers) are signed with the issuing entity crypto-material and validated (by using the public-private key) before being committed to the ledger. This mechanism is used to ensure secure and verifiable trail of updates of the ledger. Additionally, as part of the governance rules identity management, a user registration chaincode is introduced to allow storing organizational and organization user information on the chain, to allow for specification of governance rules and granting of permissions on user/organizational basis for all the shared datasets.

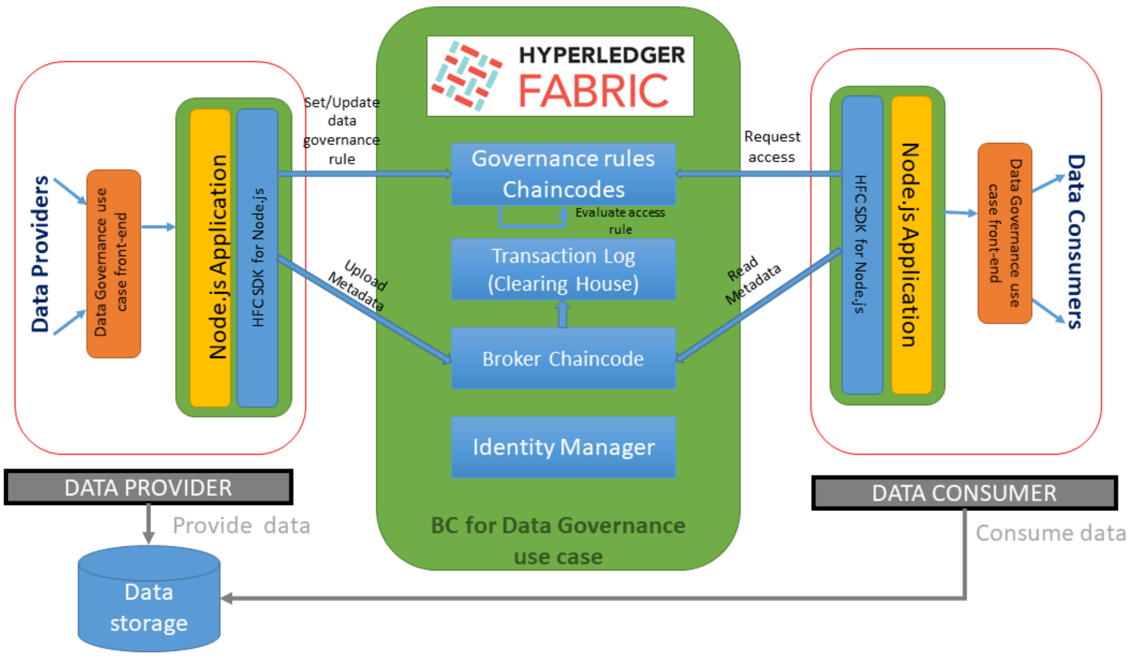


Figure 5 – Governance blockchain high level architecture overview

Although the data flows and interactions among the different chaincodes will be part of D4.2 to be submitted at M20, i.e., August 2021, we introduce below the main flows for the sake of clarity and completeness.

Figure 6 illustrates the flow process when a data producer wishes to upload a new dataset and grant permissions to it. After login, the metadata of the dataset is recorded in the broker for later searching through the data governance chaincode, while specification on how to access it is stored in the data governance chaincode and permissions chaincode (specification includes the users, terms, conditions, and timelines for accessing the particular dataset). All transactions (storing of dataset metadata, specification of governance rules, and permissions) are recorded through the Transaction log component in Fabric for provenance and non-repudiation.

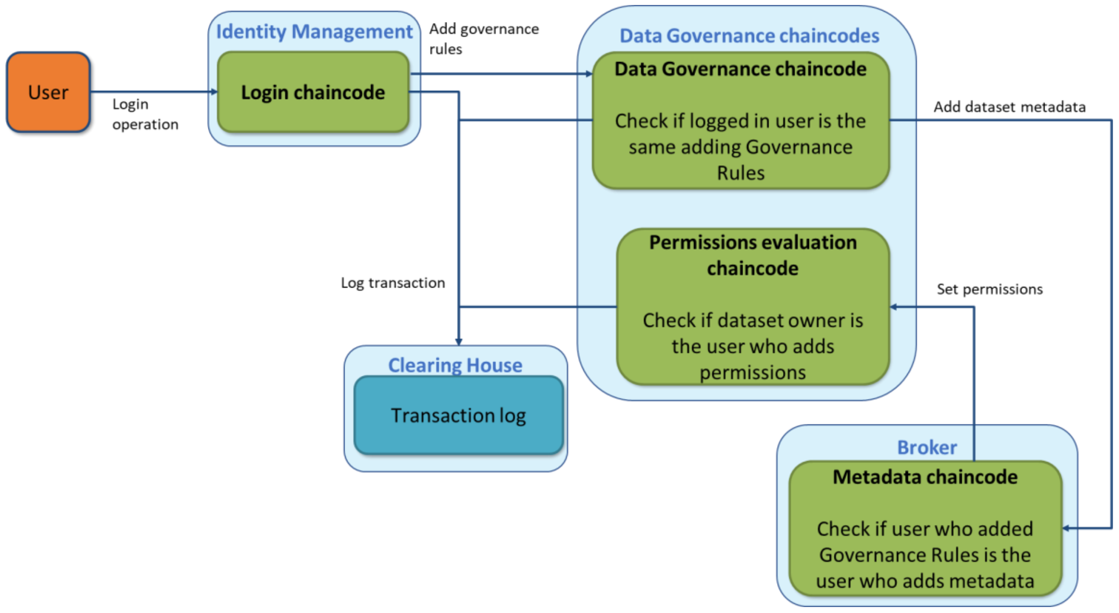

User
Login operation
Identity Management
Login chaincode
Add governance rules
Data Governance chaincodes
Data Governance chaincode
Check if logged in user is the same adding Governance Rules
Add dataset metadata
Permissions evaluation chaincode
Check if dataset owner is the user who adds permissions
Set permissions
Log transaction
Clearing House
Transaction log
Broker
Metadata chaincode
Check if user who added Governance Rules is the user who adds metadata

Figure 6 – Process flow from a data provider point of view

Figure 7 illustrates the flow process when a data consumer wishes to search for a dataset shared through the DataPorts platform.

After login, the user (data consumer) searches for a certain dataset via the broker. Once the specific dataset is found, the data governance chaincode checks the governance rules for the specific dataset and assigns access accordingly through the Permissions evaluation chaincode. Once again, all transactions are recorded through the Transaction log component in Fabric for provenance and non-repudiation.

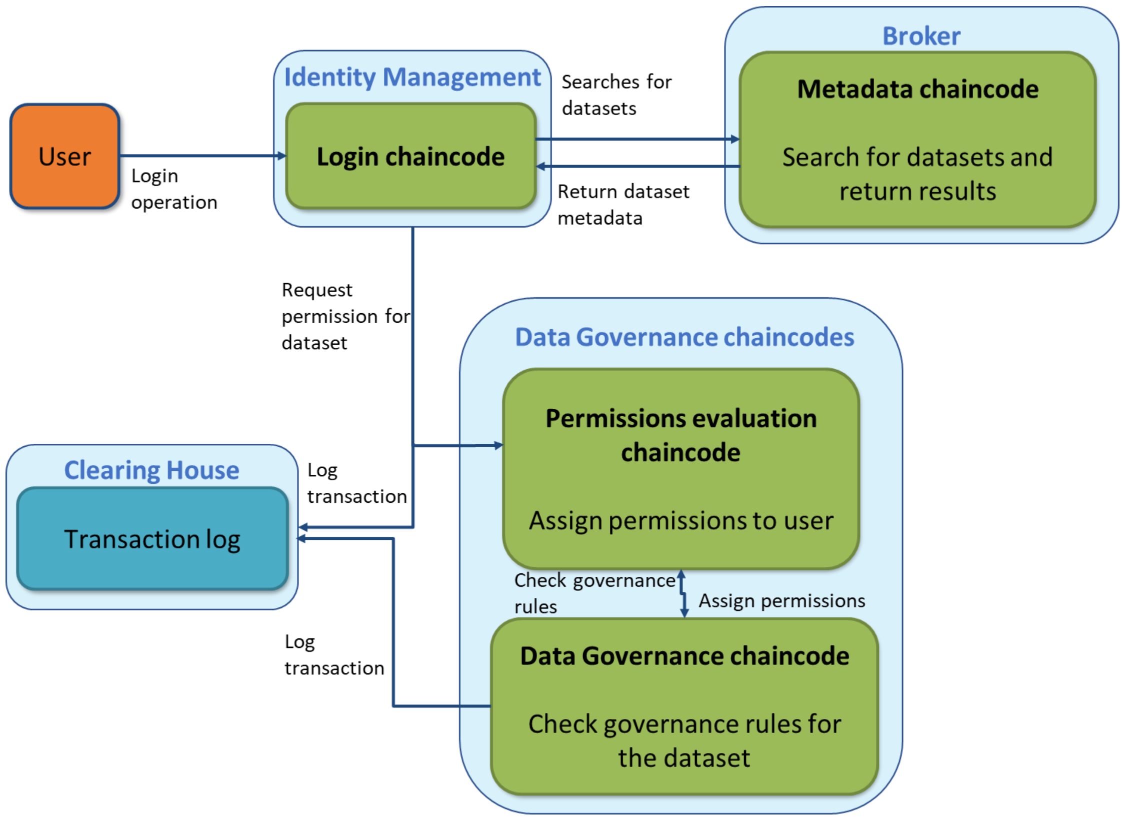


Figure 7 – Process flow from a data consumer point of view

4.4.2 Application overview

In this section, the blockchain application architecture for the governance use case is analysed. Figure 8 depicts the blockchain client application and the functionalities of the Governance use case chaincodes.

- The upper box describes the node.js components in which the server and Hyperledger Fabric Client (HFC) are included to gain access the users through the front-end. The middle box provides description of the chaincodes functions. Each colour refers to functionalities from a different chaincode, while inside the box there is a description of the functionality if it concerns invoke/query request.
- The overview of the Governance blockchain application includes the following components:
  - Hyperledger Fabric v 1.4.8 components
  - Hyperledger Fabric SDK for Node.js
  - Governance chaincodes:
    1. Manage data (Create/Read/Update (CRU))
    2. Manage metadata of data (CRU)
    3. Grant/Revoke/Deny access to data
    4. Register/Read/Update users and organizations
    5. Read transactions history
  - The governance front-end application serves the user interaction with the blockchain components while the intermediate component of the BC client manages the users' requests.

In addition, the BC client registers and enrols the users. The chaincodes provide the functionalities according to the business logic.

- In stage 2, the transactions history chaincode will be enriched with more functionalities about the transaction's history. Moreover, the application will provide organization level and public access to data.

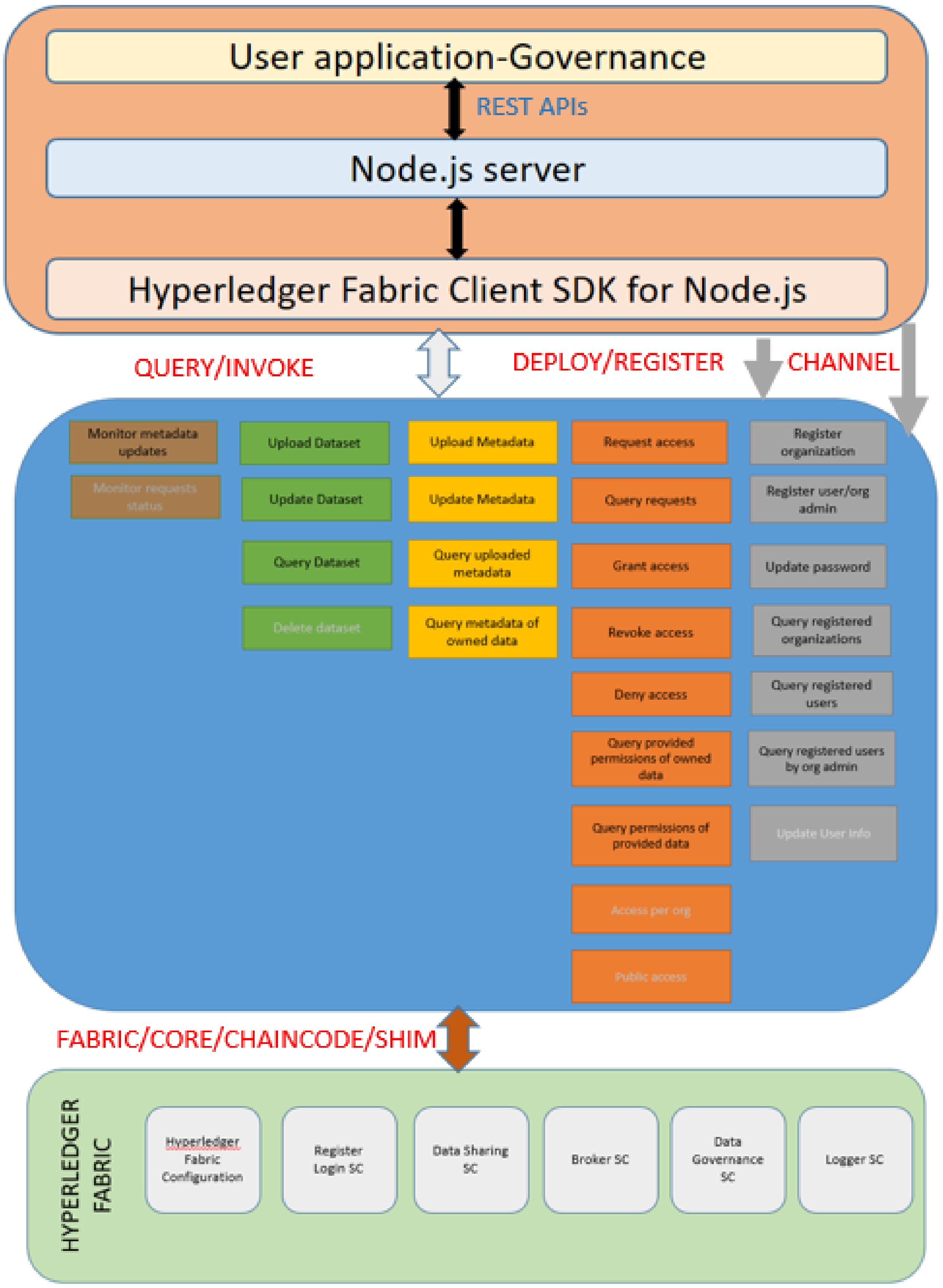


Figure 8 - Governance blockchain application architecture

4.4.3 Component overview

Data governance requirements for the DataPorts platform are included in the Governance blockchain network. To be more precise, these data sharing policies are written into the smart contracts of the network. Chaincodes in Fabric are pieces of computer code that contain the logic to be executed when a certain operation in the network is required, such as the registration of a dataset. Chaincode is installed in every peer that needs to perform these transactions and is executed simultaneously at every peer node that holds a copy of the to-be-updated ledger if the transaction is successful. Network peers evaluate transaction proposals at the same time, against the same set of rules contained in the chaincode, reaching consensus on whether a transaction is correct or not after execution of the chaincode. Once network peers finish evaluating the transaction and verifying its digital signatures, the current network status is updated with the latest changes, notifying all relevant parties of the successful transaction.

The following is a list of all chaincode operations to be performed on an asset of the Governance blockchain network:

- Create, read, update, and delete (CRUD) operations
- Search operations
- Set Terms and Conditions
- Set Data Owner
- Set Data Provider
- Set Data Consumer
- Grant Data Consumer
- Revoke Data Consumer
- Set Contract Agreements
- Set Custom Access Rights
- Get User Identity & Permissions

Figure 9 depicts the component architecture for data governance blockchain solution.

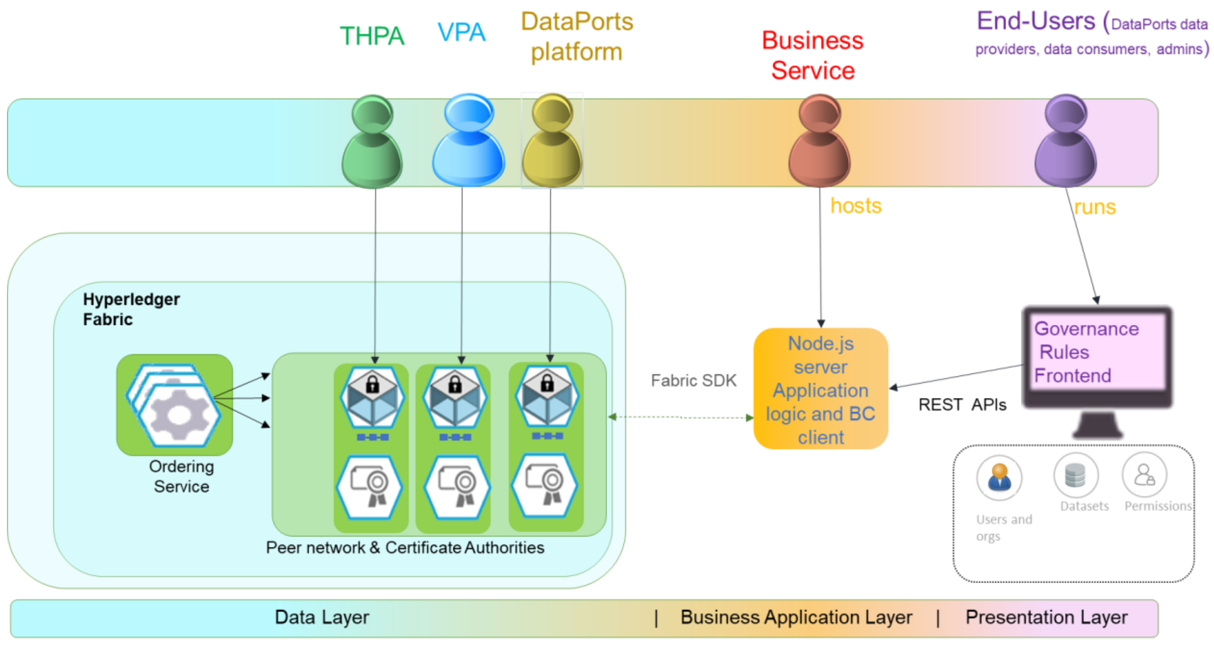

THPA
VPA
DataPorts platform
Business Service
End-Users (DataPorts data providers, data consumers, admins)
hosts
runs
Hyperledger Fabric
Ordering Service
Peer network & Certificate Authorities
Fabric SDK
Node.js server Application logic and BC client
REST APIs
Governance Rules Frontend
Users and orgs
Datasets
Permissions
Data Layer | Business Application Layer | Presentation Layer

Figure 9 – Governance blockchain component architecture

The data layer is composed of all the BC nodes, allowing replication and management of the blockchain ledger, in a secure and authorized manner. This includes the blockchain infrastructure nodes that provide the blockchain participants components and functionalities - the ledger, ordering service, peers for all participating organizations in this blockchain network, CAs, and the smart contract management and execution capabilities.

The business application layer includes the middleware component that act as connectors between the front-end application and the backend data storage layer. This layer is implemented as a Node.js application wrapping Hyperledger Fabric Client (HFC) to forward invocations from the front-end application to the blockchain infrastructure and the chaincodes running on this infrastructure.

The presentation layer includes the front-end application to provide authenticated and authorized front-end capabilities for the different end users. This includes the functionalities detailed in Section 4.4.5 and provides screens for the management of various artifacts, such as users and organizations, datasets, access rules, and current permissions.

4.4.4 Blockchain network

The Governance blockchain network is developed under the Hyperledger Fabric framework, and therefore all its components follow the Hyperledger Fabric architecture model. This network is single channel where all organisations must be registered to perform dataset operations. Ideally, every network participant should be an independent Fabric organisation and therefore have at least a peer node of its own in the network. Since this may not be feasible for every partner, it may be necessary for some “trusted” organisations in DataPorts to issue certificates to network participants that do not have a peer, so they can perform data sharing operations. Figure 10 depicts the proposed architecture for the Governance blockchain network, where new organisations can be added, alongside their certificate authorities and ordering services.

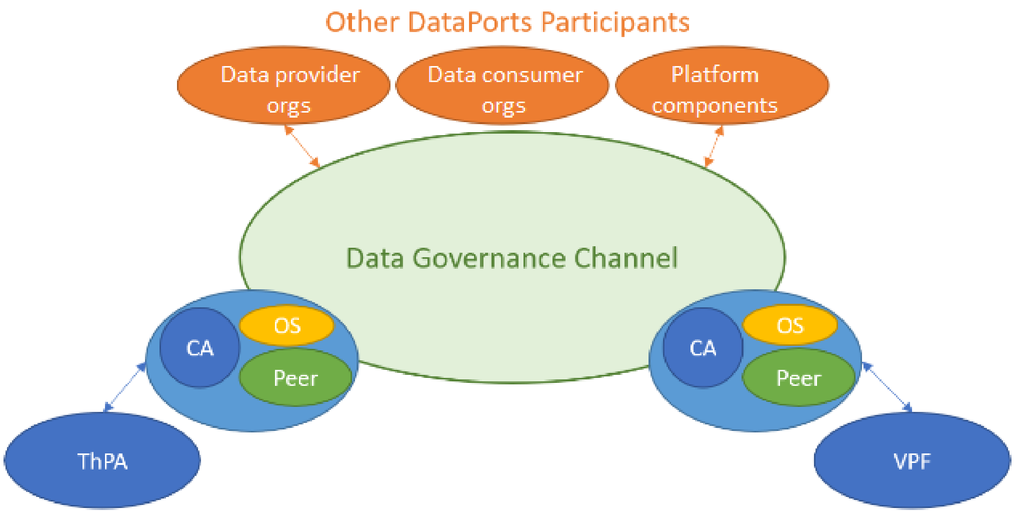


Figure 10 – Governance blockchain network

In this network, both partners Thessaloniki Port Authority (ThPA) and Valencia Port Foundation (VPF) are orderer organizations and peer organizations. Each organization also maintains their own CA and may also provide certificates for other partners in the network that want to participate in the data governance but do not possess nodes in network for practicality reasons. Every peer organization must hold a copy of the ledger and the relevant data governance chaincode.

4.4.5 Front-end

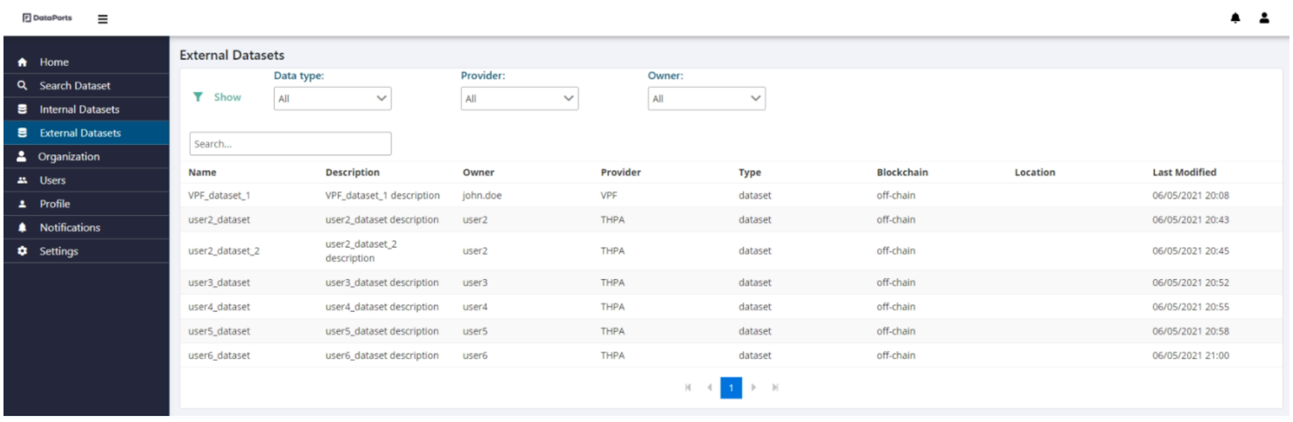
The front-end for the governance rules use case offers a different dashboard view for each user role (user, organization administrator, network administrator). The front-end communicates with the blockchain by calling APIs which are connected with the blockchain. The list below presents the features provided in the front-end:

- Backend (APIs)
    - Programming language: Javascript
    - Web development framework: Express.js
    - Javascript runtime: Node.js
- Front-end
    - Javascript Framework for Single-Page Application (SPA): Angular
    - Libraries for the user interface: PrimeNG
    - Cascading Style Sheets (CSS) and Javascript framework to make the front-end responsive: Bootstrap
- Network administrator:
    - Home (Second stage)
    - Organization & Users: all organizations in a tabular view
        - Add Organization: a new organization can be added
        - Profile (selected organization): organization profile details
        - Organization Users (selected organization): organization users
            - Add User: a new user can be added
            - Profile (selected user): profile of the selected user can be viewed and updated
    - Profile: personal profile can be viewed and updated
    - Notifications (Second stage)
    - Settings (Second stage)
- User (any user that is neither organization administrator nor network administrator):
    - Home (Second stage)
    - Search Dataset: all datasets which belong to others in a tabular view
        - Dataset Metadata: the user can view dataset metadata and can request more permissions for this dataset
    - Internal Datasets: all datasets which belong to this user in a tabular view
        - Upload Dataset: dataset metadata can be uploaded
        - Details (selected dataset): details (e.g., metadata) about the selected dataset
        - Permissions (selected dataset): shows the users who have access permissions to this dataset and the types of permissions they have – the dataset owner can also use this screen to revoke permissions.
        - Content (selected dataset): dataset content (Second stage)
        - History (selected dataset): history changes for the selected dataset
    - External Datasets: all datasets which belong to others for which the user has permissions in a tabular view
        - Details (selected dataset): details (e.g., metadata) about the selected dataset – extra permissions can be requested
        - Content (selected dataset): dataset content (Second stage)
        - History (selected dataset): history changes for the selected dataset
    - Profile: personal profile can be viewed and updated
    - Notifications: notifications regarding datasets permissions requests
    - notification (selected): there are details about the request, it can be accepted or rejected

    - Settings (Second stage)
- Administrator of any organization:
    - Home (Second stage)
    - Search Dataset: all datasets which belong to others in a tabular view
        - Dataset Metadata: the user can view dataset metadata and can request more permissions for this dataset
    - Internal Datasets: all datasets which belong to this user in a tabular view
        - Upload Dataset: dataset metadata can be uploaded
        - Details (selected dataset): details (e.g., metadata) about the selected dataset
        - Permissions (selected dataset): shows the users who have access permissions to this dataset and the types of permissions they have – the dataset owner can also use this screen to revoke permissions.
        - Content (selected dataset): dataset content (Second stage)
        - History (selected dataset): history changes for the selected dataset
    - External Datasets: all datasets which belong to others for which the user has permissions in a tabular view
        - Details (selected dataset): details (metadata, etc) about the selected dataset – extra permissions can be requested
        - Content (selected dataset): dataset content (Second stage)
        - History (selected dataset): history changes for the selected dataset
    - Organization: organization profile can be viewed and updated
    - Users: all users in a tabular view
        - Add User: an organization user can be added
        - Profile (selected user): profile of the selected user can be viewed and updated
    - Profile: personal profile can be viewed and updated
    - Notifications: notifications regarding datasets permissions requests
        - Notification (selected): there are details about the request, it can be accepted or rejected
        - Settings (Second stage)

Figure 11 and Figure 12 are two examples of the different functionality offered by the front-end in the governance rules use case.

Figure 11 shows External Datasets page inside organization administrator dashboard view. This page presents in a tabular view all the datasets of others to which the connected organization administrator has access. As shown in the figure, filters and search button are available. Moreover, the connected organization administrator can select a specific dataset by left clicking on it to view details or to make further actions.

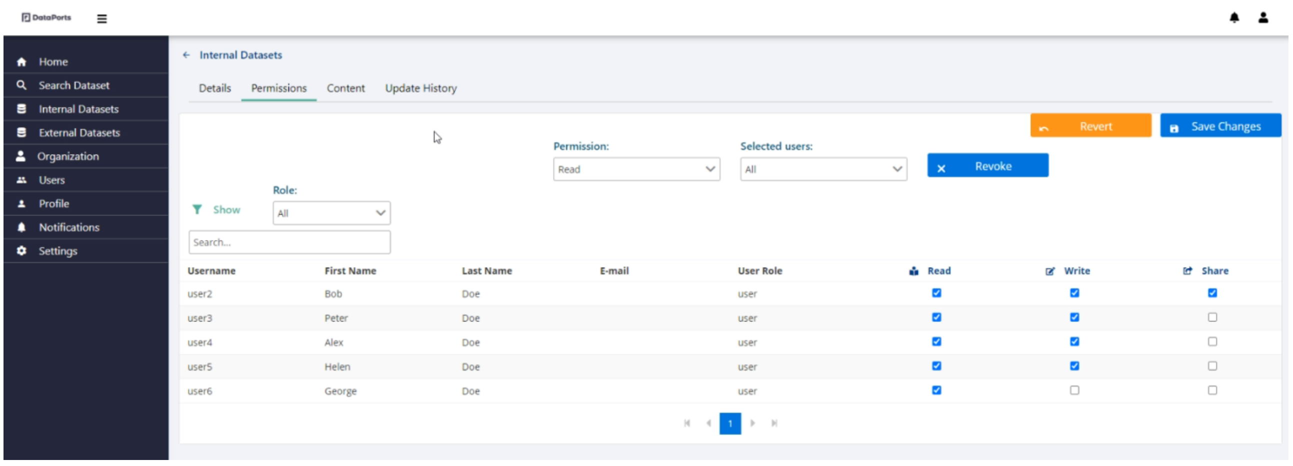


Data type: All; Provider: All; Owner: All

Name	Description	Owner	Provider	Type	Blockchain	Location	Last Modified
VPF_dataset_1	VPF_dataset_1 description	john.doe	VPF	dataset	off-chain		06/05/2021 20:08
user2_dataset	user2_dataset description	user2	THPA	dataset	off-chain		06/05/2021 20:43
user2_dataset_2	user2_dataset_2 description	user2	THPA	dataset	off-chain		06/05/2021 20:45
user3_dataset	user3_dataset description	user3	THPA	dataset	off-chain		06/05/2021 20:52
user4_dataset	user4_dataset description	user4	THPA	dataset	off-chain		06/05/2021 20:55
user5_dataset	user5_dataset description	user5	THPA	dataset	off-chain		06/05/2021 20:58
user6_dataset	user6_dataset description	user6	THPA	dataset	off-chain		06/05/2021 21:00

Figure 11 - External Datasets page (inside organization administrator dashboard)

Figure 12 shows Permissions page (for a selected internal dataset) inside organization administrator dashboard view. This page presents in a tabular view all the users who have access to the selected dataset that belongs to the connected organization administrator. The connected organization administrator can revoke access as well in this page.



Permission: Read; Selected users: All; Role: All

Username	First Name	Last Name	E-mail	User Role	Read	Write	Share
user2	Bob	Doe		user	☑	☑	☑
user3	Peter	Doe		user	☑	☑	☐
user4	Alex	Doe		user	☑	☑	☐
user5	Helen	Doe		user	☑	☑	☐
user6	George	Doe		user	☑	☐	☐

Figure 12 - Permissions page (for a selected internal dataset) inside organization administrator dashboard

4.5 SUMMARY AND NEXT STEPS

The goal of the data governance network is to enable the handling of port data and its sharing across the participants of the DataPorts platform. The application we developed for this network aims at managing the registration and the evaluation of the data access in a secured, transparent, verifiable, and immutable way. The Governance network consists of a single channel whose interconnected members are the ports and their stakeholders. The chaincode installed on the channel satisfies the business processes of the Governance application. More specifically, each organization is running its peers and generates certificates from its CA. It is important to emphasize, that we leveraged blockchain technology to perform the roles of IDS broker (by including most of the IDS broker fields in our metadata) and clearing house in the IDS architecture (simulated by blockchain transaction log).

Section 4 introduces the motivation and role of the data governance network, as well as its placement in the overall DataPorts platform. We present the detailed design of the data governance blockchain, including architecture, application, component, network, and front-end overviews. Implementation is planned to be carried out in phases starting with a Minimum Viable Product (MVP) which is the core of next deliverable (D4.2 Blockchain based data governance rules) to be submitted in two months, i.e., August 2021. Indications have been made (greyed out text) to specific functionality that has been excluded from the MVP and will be implemented at a second stage.

In the overall DataPorts architecture, the BC Governance services should be combined with the analytics and the data processing services, to achieve full integration of the functionalities of the application's different layers.

Next steps for the Governance application will deal with:

- Adding the ability to request and grant/revoke access permissions at the organizational level
- Granting and revoking public access
- Monitoring all transactions relating to access requests
- Defining the off-chain data transfer

In addition, in the next stage, the IDS components will be integrated to the existed architecture. Currently, the Data governance network implementation artifacts are stored in two repositories:

- https://varlab.iti.gr:9443/dataports_871493/dataports_871493.git
- https://umane.everis.com/git/DATAPORTS/dataportsbck.git

5 DATA SHARING BLOCKCHAIN NETWORK - VERIFIED GROSS MASS USE CASE

This section describes the Verified Gross Mass (VGM) blockchain application in the port of Valencia, first use case that has been identified in the port of Valencia in which BC plays the role of sharing data.

5.1 USE CASE DESCRIPTION AND MOTIVATION

In this scenario, the DataPorts blockchain capabilities for on-chain data sharing in a business network would be used for the request and provision of the Verified Gross Mass (VGM). In 2014, the International Maritime Organization amended (MSC.1Circ.1475[3]) the Convention on the Safety of Life at Sea (SOLAS) to require, as a condition of loading a full container on a ship for export, that the container has a verified gross weight. This requirement became mandatory worldwide on July 1st, 2016. The shipper became the responsible for obtaining the VGM of a full container and communicating it to the shipping company. Weight estimation is not allowed.

The regulation prescribes two methods by which the shipper can obtain the verified gross weight of a full container:

- Method 1: once the packaging and sealing of a container has been completed, the shipper can weigh, or arrange for a third party to weigh the full container.
- Method 2: the shipper or, by shipper's order, a third party may weigh all packages and cargo items, including the weight of pallets, stowage wood and other packaging and fastening materials packed in the container, and add the tare weight of the container that appears on the door of the container to the sum of the individual weights of the contents of the container. With respect to both method 1 and 2, the weighing equipment used must meet the precision standards and requirements of the state in which the equipment is being used.

The intent behind the digitalization of the VGM process is to build a solution that will facilitate compliance with the SOLAS regulations on container weighing for the port logistics community. It shall offer users an effective solution to allow containers to arrive at the port with the verified gross weight, reducing last minute incidents or delays at container terminals or the appearance of congestion situations. In addition, it will offer a fast and automated method for the verified gross weight to reach the shipping company and the terminal; and allow the port to be more competitive.

Implementing the VGM use case, including weight requests lifecycle management, certificate management and payment services on top of a blockchain business network is intended to provide more added value than existing solutions (centralized client-server system) by having a verifiable and immutable information of shared data through the entire chain to all concerned business participants serving as a single source of truth and providing transparency and non-repudiation process. In this way, we can be sure that the weight cannot be altered at any point.

5.2 USE CASE REQUIREMENTS

In designing the blockchain solution for the VGM use case, we first identified the organisations that are involved in the use case interactions. These are the *network participants*. We examine this from the perspective of the blockchain network and, therefore, these interactions pertain to reading and writing to/from the common blockchain ledger. The following data providers and data consumers participate in the VGM business network (See Figure 13).

[3] https://www.worldshipping.org/industry-issues/safety/MSC_1-Circ_1475_-_Guidelines_Regarding_The_Verified_Gross_Mass_Of_A_Container_Carrying_Cargo_-Secretariat-.pdf

Shipper company and its representative (if needed) – the *freight forwarder*. The shipper is the entity which owns the goods. The shipper can create a weighting request by themselves or can delegate this task to the freight forwarder who acts on the shipper's behalf.

The *scale operator* is the entity that provides the weighting services and generates VGM certificates. Scale operator can record a weighting request if such was not created by *shipper/freight forwarder* and can also register vehicle information if this was not entered into the system by a *road haulier*. The scale operators in the VGM use case are the port container terminal, and the truck association organization.

Road Haulier is the organization providing the land transportation services for the goods, it also provides vehicle information.

Shipping line and its representative in the port – the *shipping agent*.

In addition to the BC stakeholders the following entities are part of the BC network: conPESO platform, the Port Community System (PCS) of the Valencia port, shipping line, and the port container terminal. The ConPESO platform provides front-end functionality and uses Fabric as the backend for the solution.

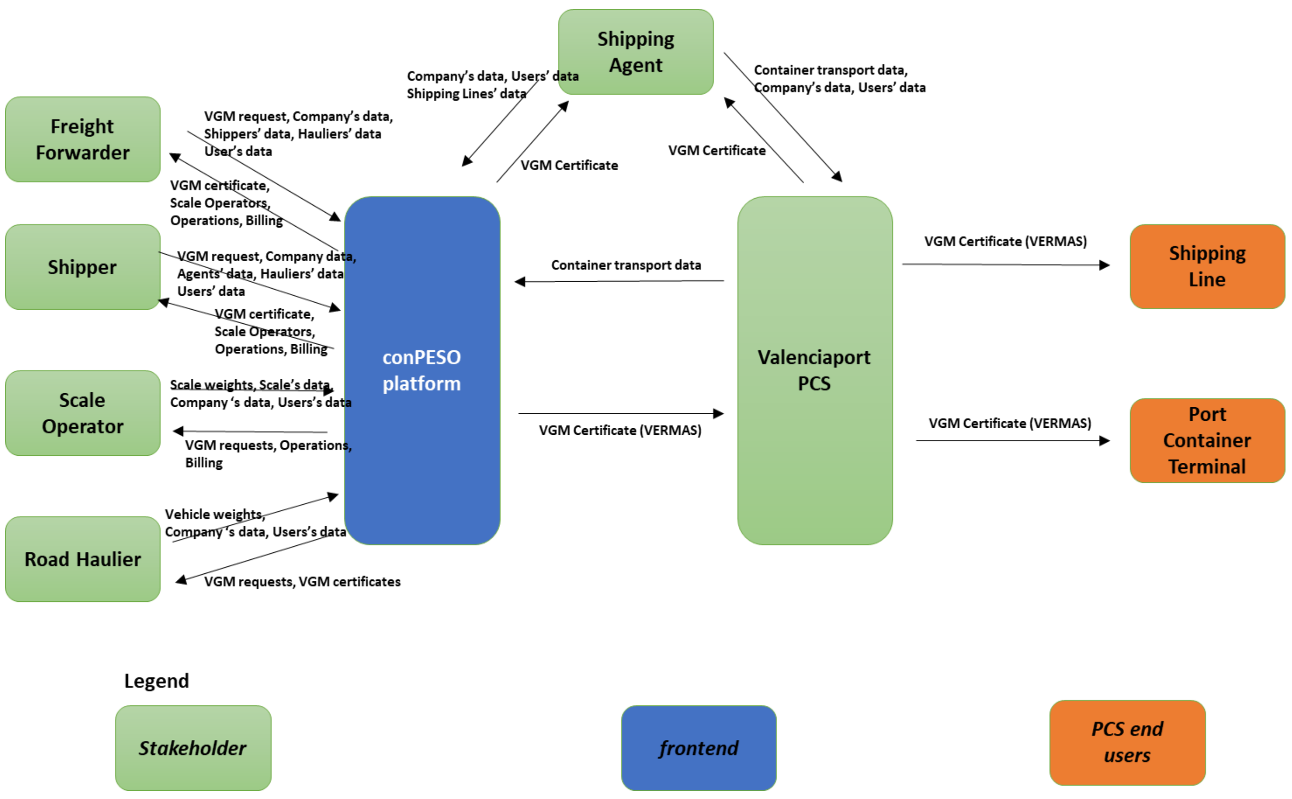


Figure 13 - Data providers and data consumers in the VGM use case

The following functionalities need to be supported for the different stakeholders of the BC network. The greyed-out functionalities are planned to be developed for phase 2 of the VGM implementation.

- Freight forwarder/shipper
    - Creating weight requests
    - Viewing weight requests
    - Get notified on weight request updates
    - Get notified on VGM certificate generation
    - Create/Read their company data
    - Read scale operators data
    - Get history of weight request updates

    - View weighting invoices and operations
        - Generating payments for weighting services

- Scale operators
    - Getting notified on weight request creation
    - View weight requests
    - Create weight request in case one does not exist, and the scale operator allowed to create
    - Update weight requests
    - Registering scales
    - Create/Read company data
    - Generate VGM certificate
    - Get information on vehicle weights
    - Create vehicle weights entries and register the vehicle in their name or in the name of truck operator company
    - Register weighting invoices, viewing operations
    - Receive payments
- Port systems
    - VGM solution provider
        - Register companies participating in VGM use cases and their roles as system admin organization
    - Port Community System
        - Providing container data for the container gate-in operation ("gate in locator")
        - Getting notified on VGM certificate creation
- Road hauliers
    - Register vehicles and their information
    - Get history on vehicle's tare weights updates
    - Create/Read company data
    - Getting notified of Weight Requests
    - Viewing weight requests
- Shipping agents/ shipping line
    - Create company data
    - View VGM certificates

5.2.1 Security and confidentiality requirements

The proposed solution will be using inherent BC features for security, authentication, and authorization. Each organizational user will login using crypto credentials supplied to him during registration and enrolment process by the appropriate BC organization CA server. These crypto materials include X.509 certificate as well as private and public key. The public key is stored at the CA server and is used to verify signature on all transactions invoked by the user, which are signed with their private key. On the client side, all the user crypto-materials are stored in organizational wallet, which in the first phase will be filesystem based, and in the next implementation will be encrypted MongoDB-based wallet.

Some of the weight request and VGM data, such as the scale operator company's price for the weighting operation, is considered business sensitive and should not be shared with the company's competitors, which are the other scale operators on the network. This data should be shared only with "neutral" stakeholders of the network, such as VGM solution provider, PCS system, and the companies which are involved in one or another role in the weight request itself (such as shipper/freight forwarder, road haulier, shipping line).

Scale operators are network stakeholders that hold their own peer and therefore a copy of the ledger. Hence, theoretically, if all the data, including the weight request data, is replicated in the ledger of all peers in equal manner, the scale operator's competitor companies could gain access to this sensitive data.

Therefore, a solution is required where:

- The scale operator company can access their own data,
- The allowed parties, such as the companies participating in the weight request in question, can access and view the scale operator's information, and
- The other scale operators' companies in the network have no copy of the sensitive data in their ledger.

5.3 USER STORIES

While requirements focus on functionality (see previous section), that is, what the solution should do; user stories focus what the person using the solution should be able to do. In this sub-section we focus on the user stories associated with this use:

As a *shipper or its agent*, I want to use the DataPorts blockchain business network for:

- Knowing the scale network, services, and prices available
- Submitting the container weight request, obtaining the VGM certificate and making it available to interested parties.
- Automatically notifying the weighting results to the port and shipping company that transports the container once the weighting has been completed
- Managing the lifecycle of all my submitted and resolved weighting requests/VGM certificates

As a *road haulier/tare weight recorder*, I want to use the DataPorts blockchain business network to register the certified gross mass of my trucks and semi-trailers required for the calculation of the VGM (mainly equipment gross weight and plate identifiers) and to get notified when I have to go to weigh a container.

As a *container weight provider/scale operator*, I want to use DataPorts blockchain business network to check weight container requests from my customers, register the result of the weighing operation and get reimbursed for this service. DataPorts will take care of the generation, distribution, and verification of the VGM certificate using other data like the certified gross mass of the trucks and semi-trailers and guaranteeing the payment.

As a *VGM port service provider / port authority*, I want to be able to certify organizations as stakeholders and participants in the VGM blockchain based solutions.

Second phase of the use case implementation would include extensions to support invoices and payments, and to offer confidentiality for sensitive scale operator's data, such as offered prices for the weight operations

The extended user stories associated with this use case are the following:

- As a freight forwarder, I want to use the DataPorts blockchain business network to ensure the payment of the services related to VGM.
- As a container weight provider, I want to use DataPorts blockchain business network to get reimbursed for the service of weighting operations. I want the ability to submit invoices and get payment for my services.
- As a scale operator, I want to use DataPorts blockchain business network to share my private data with some of the network's stakeholders (freight forwarders, shippers, port authority) while keeping it confidential from competitors (other scale operator companies).

5.4 BLOCKCHAIN ARCHITECTURE

This section describes a high-level overview of the VGM solution architecture (the different infrastructural and software layers comprising the whole blockchain solution), and the component design and integration flows of those components. Subsequent sections delve into more technical information about various aspects of the design.

5.4.1 Architecture overview

Figure 14 shows a high-level overview of the blockchain architecture proposed.

The blockchain comprises three submodules: VGM chaincode, transaction log, and identity management.

The VGM chaincode embodies the applicational functionality of the VGM use case and is developed and deployed as smart contracts, the data to the blockchain will be written and read by the network participants. Data provider and data consumer applications will be the Port Community System of Port of Valencia (ValenciaportPCS), a VGM service provider platform (conPESO) as well as the freight forwarder and scale operators' applications. These applications act sometimes as data providers or as data consumers. The chain in the ledger is a transaction log, structured as hash-linked blocks, where each block contains a sequence of N transactions.

Hyperledger Fabric has a ledger subsystem comprising two components: the world state and the transaction log. Each participant has a copy of the ledger to every Hyperledger Fabric network they belong to. The world state component describes the state of the ledger at a given point in time. It is the database of the ledger. The transaction log component records all transactions which have resulted in the current value of the world state. It is the update history for the world state. The ledger, then, is a combination of the world state database and the transaction log history.

Hyperledger Fabric provides a membership identity service that manages user IDs and authenticates all participants on the network. Access control lists can be used to provide additional layers of permission through authorization of specific network operations.

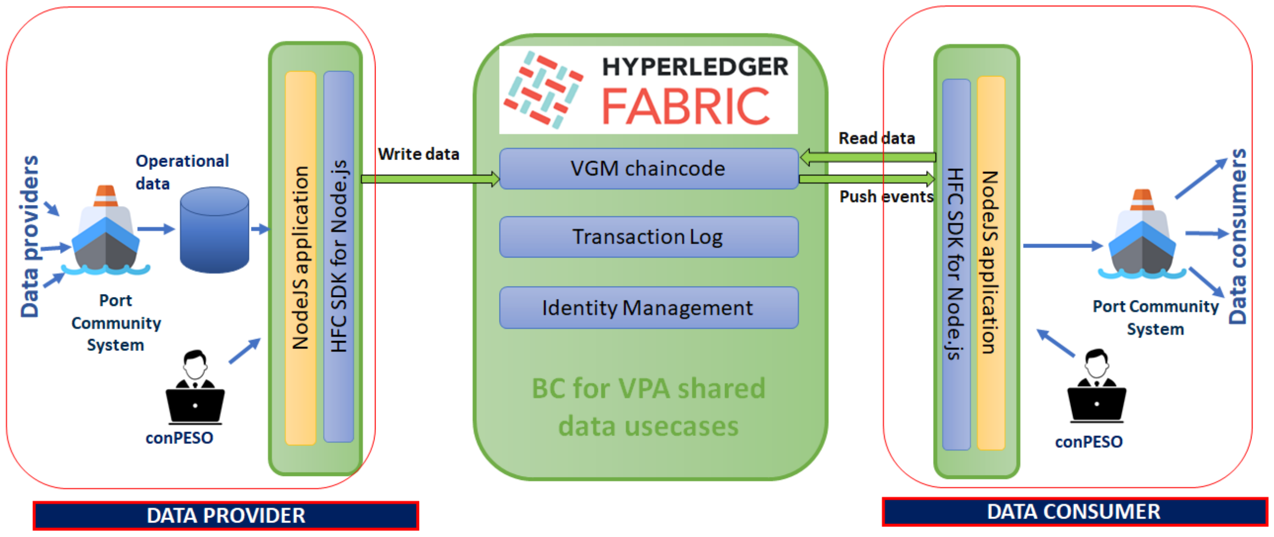


Figure 14 – VGM blockchain high level architecture overview

The adopted incremental approach to implementation allowed us to focus on the business logic of the VGM blockchain application in the initial phase of the implementation, and to push the integration with the platform provided components at second stage as needed and when the platform development roadmap allows so.

5.4.2 Application overview

Figure 15 shows the high level conceptual blockchain application architecture for the VGM use case. Namely, the blockchain client application and VGM specific chaincode components. As aforementioned, business logic for the blockchain part is implemented by smart contracts (chaincodes):

- Smart contract functions are accessed through HFC SDK and the blockchain client application will be used by front-end applications. The VGM solution specific chaincode components are shown as green (query-read functions) and orange (invoke- write functions) boxes in the figure, and frontend apps are shown in the upper layer as blue boxes.
- The overall VGM solution will provide the following components:
  - Hyperledger Fabric v 1.4.8 components
  - VGM chaincodes
    - Lifecycle management of weight requests (Create/Read/Update (CRU))
    - Lifecycle management of company registry (CRU)
    - Reading/writing vehicle information
    - Payments and invoice management (second implementation phase)
  - Hyperledger Fabric SDK for Node.js
  - The BC client application which will be a wrapper around Fabric SDK and provide CRUD operations for various data entities within the use case, as well as user registration and enrolment functionality.
  - Front-end application (conPESO) for user registration and enrolment, and use-case specific front-end functionality for all user case stakeholders.
- Payments, invoices, and private data are not in the scope of VGM Phase 1 and will be added in phase 2.

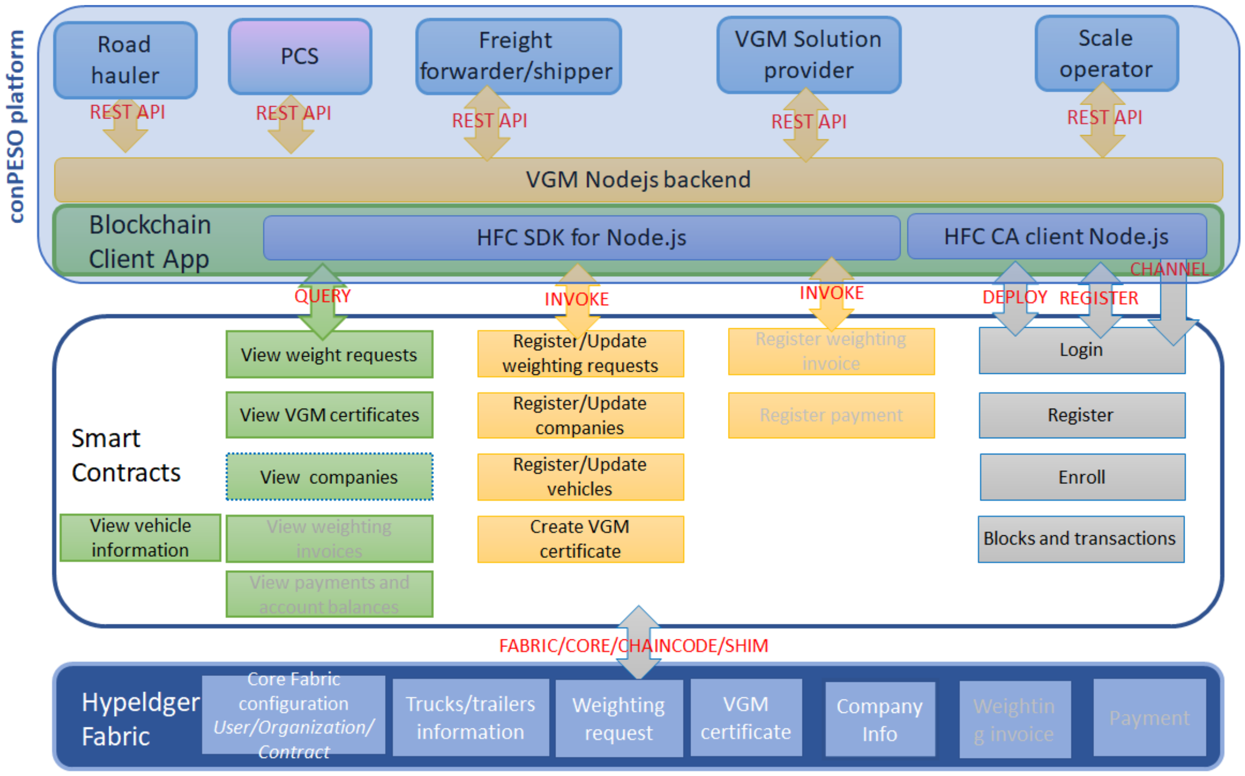


Figure 15 - VGM blockchain application architecture

5.4.3 Component overview

Another view of the solution architecture, where the emphasis is on different physical components of the solution, is provided in Figure 16. We can accommodate the different components in three different functional layers. These include the data, application, and presentation layers.

The data layer is composed of all the components for storing and retrieving data, in a secure and authorized manner. This includes the blockchain infrastructure that provides the blockchain participants components and functionalities - the ledger, ordering service, peers, CAs, and the smart contract management and execution capabilities.

The business application layer which includes the middleware components that act as connectors between the front-end applications and the backend data storage layer. Components in this layer can also process the information provided by the front-end components and retrieved from the data layer by running appropriate business logic on this information. This layer includes the business service consumer that hosts the middle-tier applications which run the business logic not encoded in the chaincode. This layer provides APIs to the presentation layer applications (web/mobile) which act as front-end applications for application of end users. The business service consumer (conPESO) backend will envelop the blockchain Fabric client for invocation of smart contracts and local business logic where applicable. This component uses the Fabric SDK to invoke the chaincode functions to interact with the ledger.

The presentation layer which includes the front-end components to provide authenticated and authorized front-end capabilities for the different end users. This includes dashboard and front-end applications providing end-users with presentation logic on an appropriate device. For example, mobile application or desktop dashboard. There may be multiple end-user applications (often one per organization or user role).

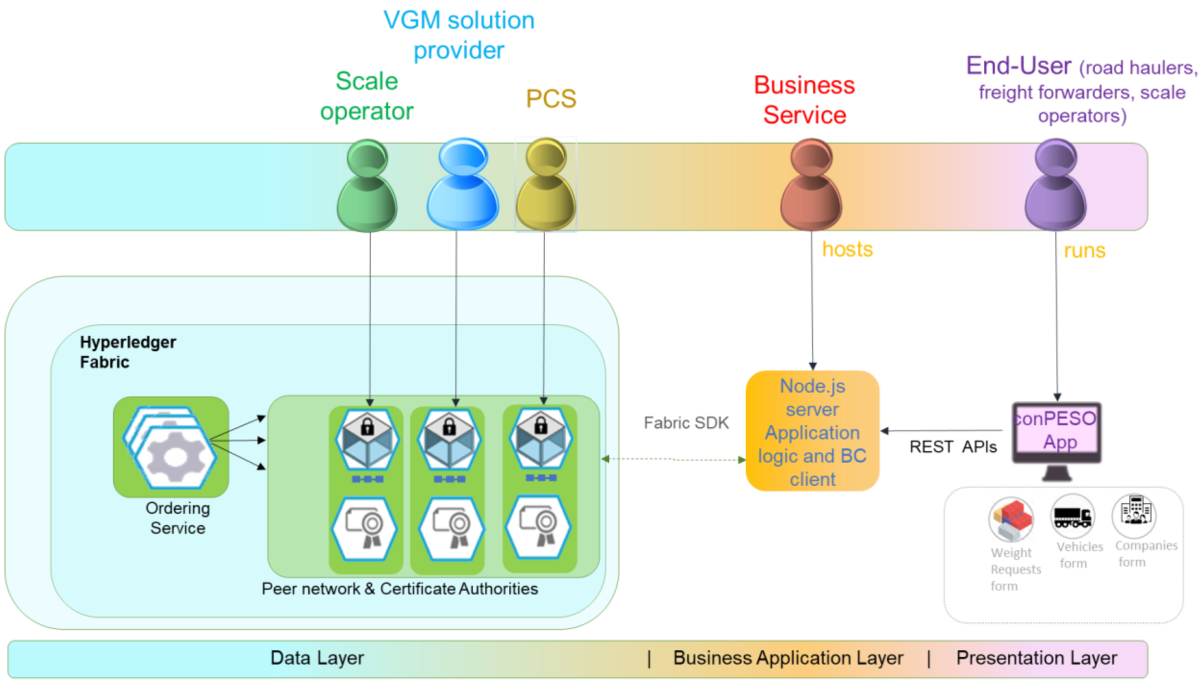

VGM solution provider
Scale operator
PCS
Business Service
End-User (road haulers, freight forwarders, scale operators)
hosts
runs
Hyperledger Fabric
Ordering Service
Peer network & Certificate Authorities
Fabric SDK
Node.js server Application logic and BC client
REST APIs
conPESO App
Weight Requests form
Vehicles form
Companies form
Data Layer | Business Application Layer | Presentation Layer

Figure 16 - VGM blockchain component architecture

5.4.4 Blockchain network

Figure 17 shows the blockchain network structure for the VGM use case. It includes the network participants, the blockchain roles each network participant performs and the functionality it requires at the backend to support its business logic. The blockchain participants in VGM BC network include:

- The VGM solution provider, running the orderer for the network, it is also one of the peer organizations of the network, running a CA and endorsing peer and sharing a copy of a ledger.

- The scale operator org, also one of the peer organizations of the network, running a CA and endorsing peer and sharing a copy of a ledger, but keeping some information of its operation private to itself and "neutral" non-competitive participants of the network, such as PCS and VGM system provider.
- PCS/Terminal representative organization, running a CA and endorsing peer and sharing a copy of a ledger.
- Additional users on the blockchain network (road hauliers, freight forwarders, shippers, and shipping agents) are not presented with a peer (that is, do not have their own copy of the ledger), however they have access to application layer and are able to read/write transactions on the chain via the VGM solution provider peer, according to their access roles and authorization (enrolled users).

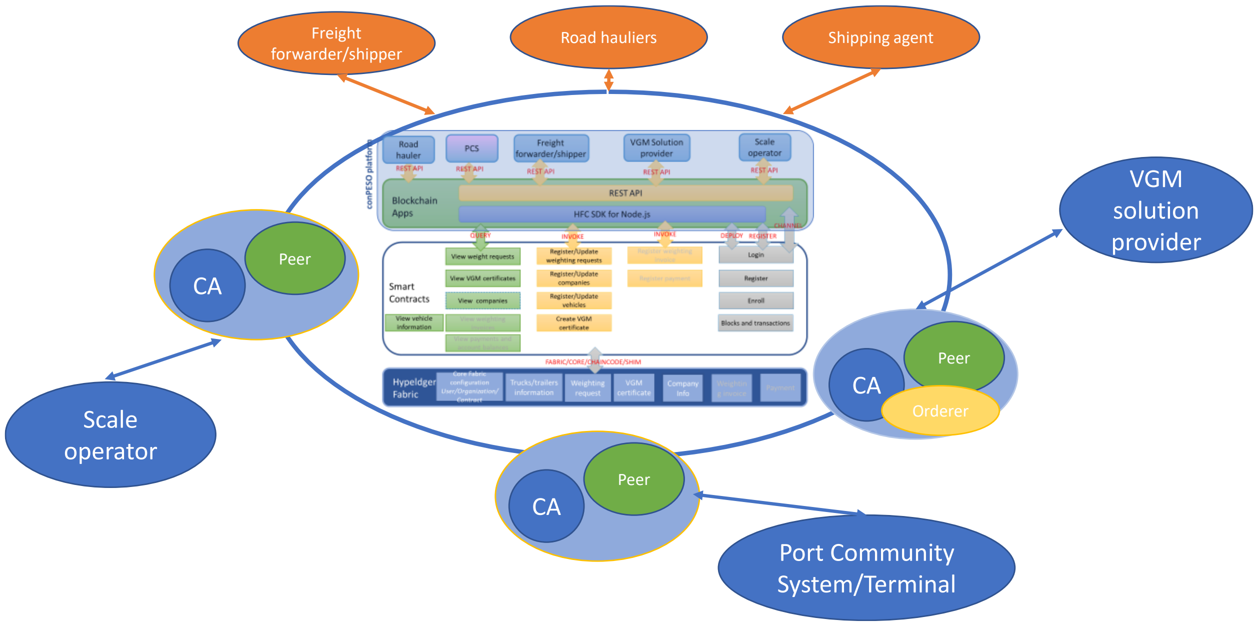


Figure 17 - VGM blockchain network

5.4.5 Addressing confidentiality requirements

As described in Section 5.2.1, some of the weight request and VGM data, such as the scale operator company's price for the weighting operation, is considered business sensitive and should not be shared with the company's competitors, which are the other scale operator's on the network. Two solution options for the problem were evaluated:

- Using different channels for different sub-networks within the VGM network – a channel would include a particular scale operator and the other "neutral" network stakeholders. This could result in several channels equals to the number of scale operators plus one global channel for non-sensitive data.
- Private data collections. These are predefined data collections, usually in JavaScript Object Notation (JSON) format, that enable companies within a Hyperledger Fabric blockchain consortium to keep certain transactional data private, while still allowing for consensus to be reached among all required nodes. The private data collections are not replicated among all peers on the network, but only among the peers which belong to organizations defined in the private collection definition. The hash of the data is stored on the network shared ledger, so the proof of the transaction is accessible to the network, just not the details of the transaction. This private data is also not seen by the orderers of the network.

The selected option was to use the private data collections for the following reasons:

- Channels are more optimal solution for cases where entire transactions and/or ledgers need to be kept confidential within a subset of a network orgs.
- Management and performance overhead of multiple channels.
- The weigh record transaction data need to be shared within the whole network but only part of the record obscured from other scale operator orgs

Figure 18 depicts the proposed architecture for the private data collection solution.

A private collection definition will be created for each of the scale operator organization and the “neutral” peers (VGM solution provider, PCS). Since other “neutral” organizations on the network, the road hauliers and the freight forwarders are not represented as blockchain organizations, their users are affiliated to the VGM solution provider organization, therefore these users will run their queries versus the VGM solution provider peers.

The private data collection will be replicated among the peers participating in the collection definition only. Therefore, this private data will not be replicated to peers not participating in the collection, which are the other scale operator companies of the network.

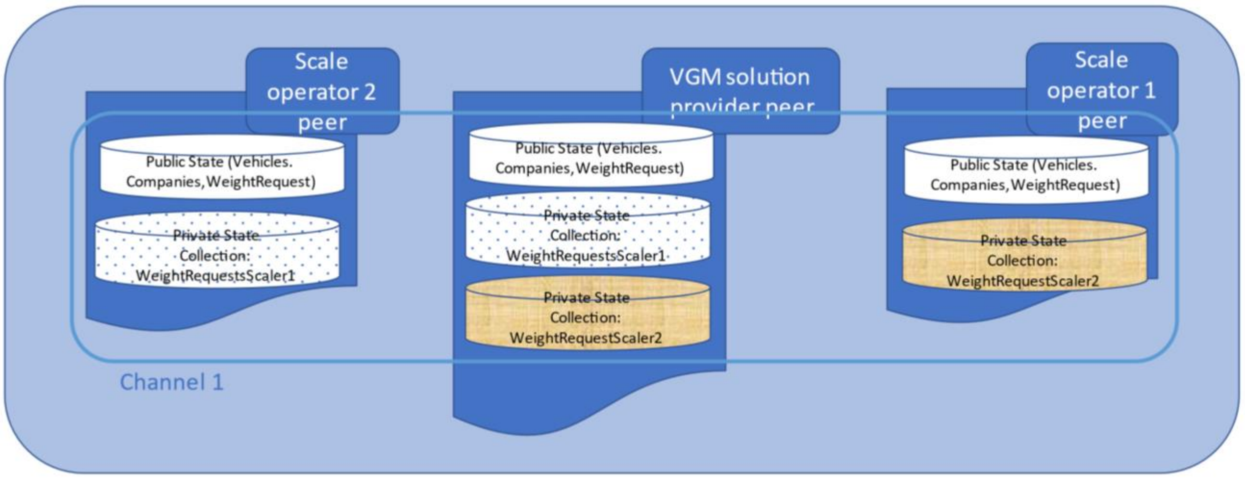
The weight request will have public fields, which will be stored in the “public” state of the ledger and be accessible to all peers, and the sensitive private data which only be accessible to the peers the private data collection will be replicated to.

The following fields of weight request will be “public fields”, written to the ledger of all organizations:

- id
- locator
- status
- updateDate
- creationDate

Since the private data collection will always be replicated to the “neutral” peers, all the neutral companies and users, based on their respective roles, will be able to gain access to this data, and the relevant scale operator company.

Private collections inclusion will happen at phase 2 of the use case implementation.


Scale operator 2 peer
Public State (Vehicles. Companies, WeightRequest)
Private State Collection: WeightRequestsScaler1
VGM solution provider peer
Public State (Vehicles. Companies, WeightRequest)
Private State Collection: WeightRequestsScaler1
Private State Collection: WeightRequestScaler2
Scale operator 1 peer
Public State (Vehicles. Companies, WeightRequest)
Private State Collection: WeightRequestScaler2
Channel 1

Collection WeightRequestScaler1: VGM solution provider org, PCS org, ScaleOperator1 org

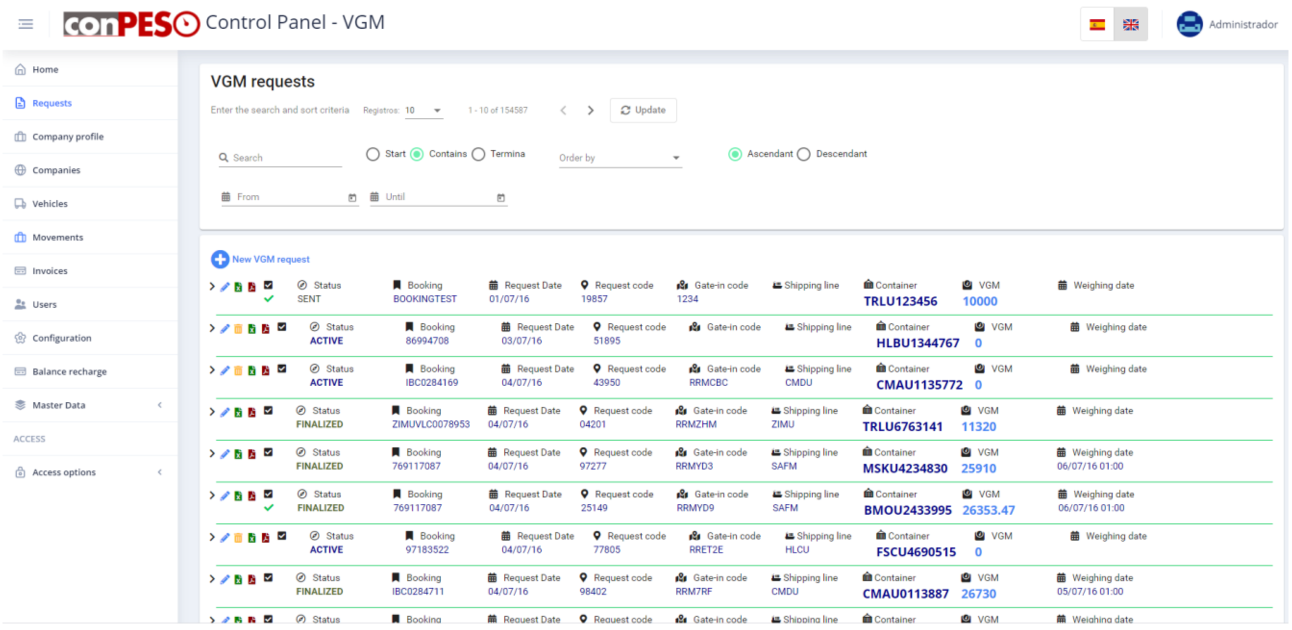
Collection WeightRequestScaler2: VGM solution provider org, PCS org, ScaleOperator2 org

Figure 18 - Private data collection overview architecture

5.4.6 Front-end

The VGM solution developed in the project, offers a web-based front-end application (conPESO) to allow for the solution end users to enter company, vehicle, and weight requests information, read the current state, update requests with weight results, and more. The information entered by the users is written to the blockchain ledger serving as the single source of truth, and at the same time to the mongoDB instances configured as additional backend of the solution. This is done for the purpose of caching and for running more complex filtering queries on the query data returned from blockchain.

Figure 19 shows the entering screen of the conPESO front-end.



VGM requests

Enter the search and sort criteria Registros: 10 1 - 10 of 154587

Search ( ) Start [● Contains] ( ) Termina Order by [● Ascendant] ( ) Descendant

From Until

Status	Booking	Request Date	Request code	Gate-in code	Shipping line	Container	VGM	Weighing date
SENT	BOOKINGTEST	01/07/16	19857	1234		TRLU123456	10000	
ACTIVE	86994708	03/07/16	51895			HLBU1344767	0	
ACTIVE	IBC0284169	04/07/16	43950	RRMCBC	CMDU	CMAU1135772	0	
FINALIZED	ZIMUVLC0078953	04/07/16	04201	RRMZHM	ZIMU	TRLU6763141	11320	
FINALIZED	769117087	04/07/16	97277	RRMYD3	SAFM	MSKU4234830	25910	06/07/16 01:00
FINALIZED	769117087	04/07/16	25149	RRMYD9	SAFM	BMOU2433995	26353.47	06/07/16 01:00
ACTIVE	97183522	04/07/16	77805	RRET2E	HLCU	FSCU4690515	0	
FINALIZED	IBC0284711	04/07/16	98402	RRM7RF	CMDU	CMAU0113887	26730	05/07/16 01:00

Figure 19 - Front-end screen view

The following software components are used in the VGM front-end implementation (Figure 20):

- Backend:
  - Programming language: Typescript
  - Database manager: MongoDB
  - Mailing platform: SendGrid
  - Web development framework: Express.js
  - Javascript runtime: Node.js
  - Javascript Object–Relational Mapping (ORM) for DB accessing: Mongoose
- Front-end:
  - Javascript Framework for SPA: Angular
  - Libraries for the user interface: Angular Material, Nebular, and DevExtreme
  - CSS and Javascript framework to make the FrontEnd responsive: Bootstrap
  - Angular, Bootstrap, and Nebular based template for dashboards: Ngx Admin
  - Vector font and icon repository for using with Bootstrap: Font Awesome

When creating or updating assets using the conPESO front-end, the information is stored on the blockchain ledger and at the same time in the MongoDB instance serving as fast-access cache. The conPESO user interface (UI) provides views into the list of assets of different types by performing complex queries with different filters from MongoDB. When a particular asset is chosen, this asset information is read from the BC ledger since it serves as the single source of truth in the application.

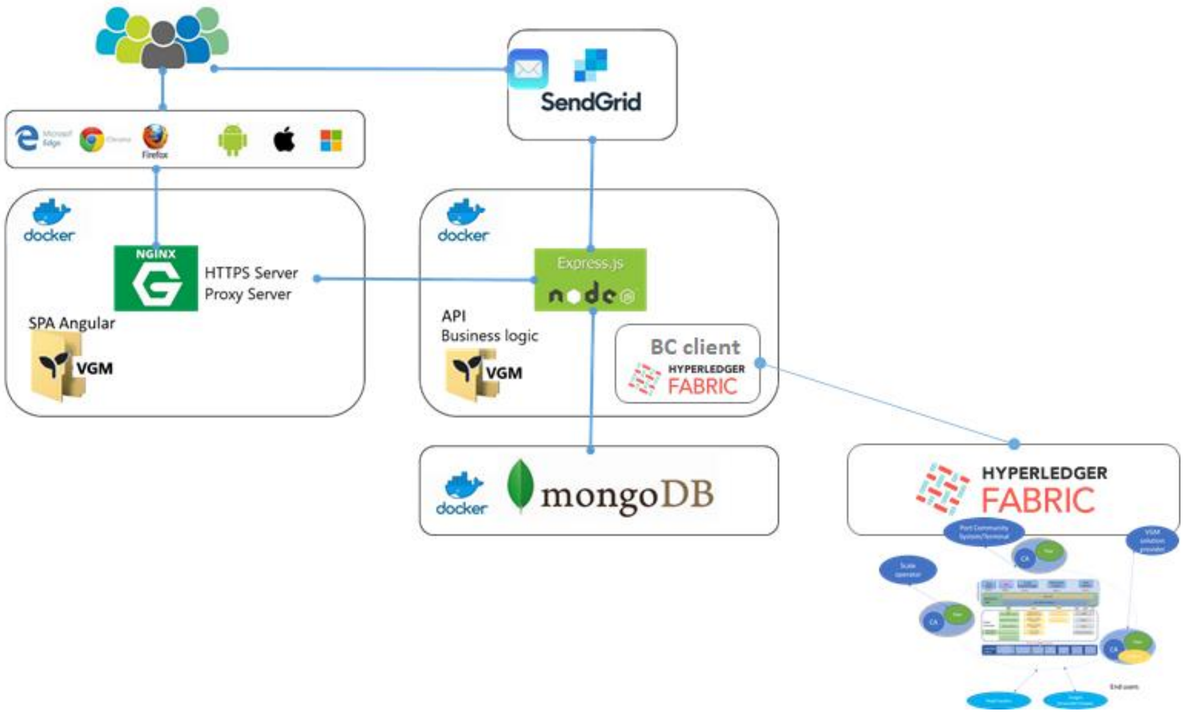


Figure 20 - VGM front-end architecture

Main navigation view of conPESO (left-hand side in Figure 19) offers the following options (See Figure 21 below followed by explanations on the different options):

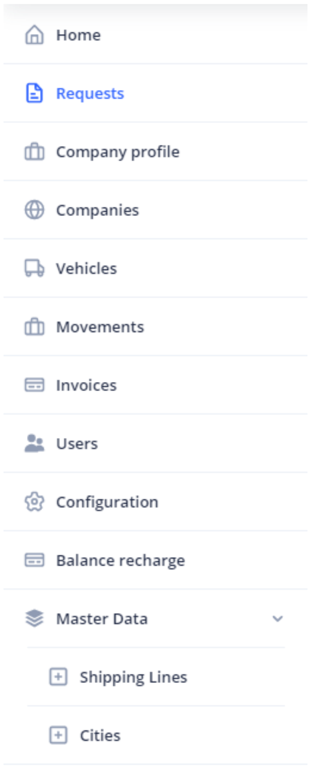


Figure 21 - VGM main navigation view

Requests: Register/modify/query/delete weighing requests. In addition, they can be exported to PDF and Excel

Company profile: See the user's company data and add relationships with other client or supplier companies (usually shipper or haulier).

Companies: Register/modify/query/delete companies (of type carrier, charger, scale, VGM agent, shipping agent, or admin).

Vehicles: Register/modify/query/delete vehicles of a specific transport company. It is also possible to register these vehicles and associate them with the transport company by a second company (usually a scale operator).

Invoices and Movements: Consult the invoices with their associated economic transactions (implementation of payments and invoices on blockchain is planned for phase 2).

Users: Register users, or modify/delete them, in addition to change their password. When a user is registered, we must also indicate their role and what emails type we want them to receive. Depending on the role, the access that the user will have to the application will be limited. When registering, user company affiliation must also be entered to allow more fine-grained access control (Figure 22).

Configuration: Access data and credentials to the PCS system in order to obtain the information of the shipping companies.

Balance recharge: Recharge the companies balance by bankcard.

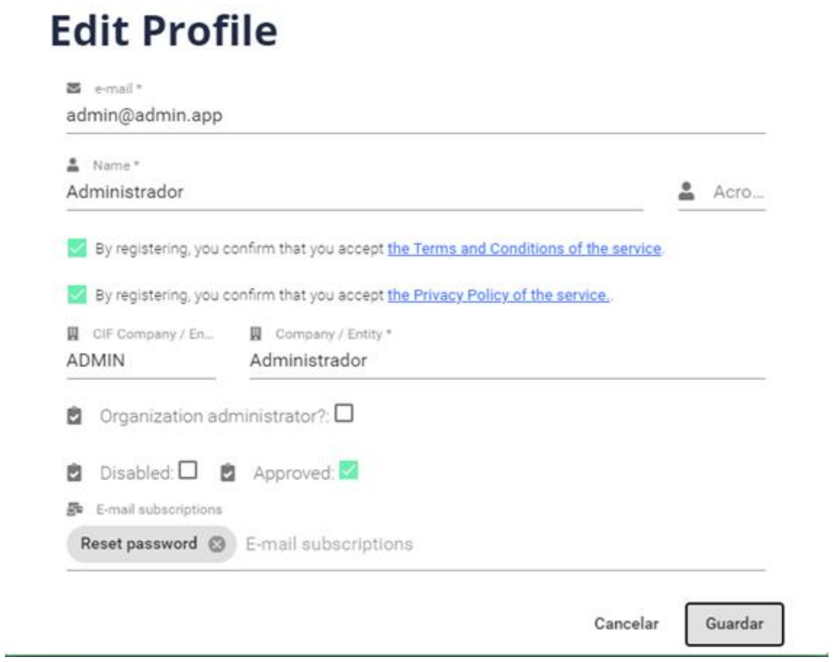
Master Data -> Shipping Lines: Data of the shipping companies extracted from the PCS. It can be edited, and also added from an Excel.

Master Data -> Cities: List of cities. They can be added manually, or from an Excel.

Access options-> Logout: Log out of the application

Access options-> Change password: Change the user's password

The VGM front-end application is available in Spanish and English.



Edit Profile

e-mail *: admin@admin.app

Name *: Administrador

Acro...

✓ By registering, you confirm that you accept the Terms and Conditions of the service

✓ By registering, you confirm that you accept the Privacy Policy of the service.

CIF Company / En...: ADMIN

Company / Entity *: Administrador

Organization administrator?: ☐

Disabled: ☐ Approved: ✓

E-mail subscriptions: Reset password

Figure 22 - User registration and profile creation

5.5 SUMMARY AND NEXT STEPS

An efficient Verified Gross Mass process in the Valencia port is essential to the successful operations of the port as well as necessary to comply with port regulations. The implementation of this use case on top of a blockchain network assures the immutability of the data concerning containers weights and a non-repudiation process, thus adding a competitive edge to the port.

Section 5 introduces the VGM use case blockchain network and its business motivation. We present the detailed design of the CPU blockchain, including architecture, application, component, network, and front-end overviews. Implementation is planned to be carried out in phases starting with a Minimum Viable Product (MVP) which is the core of next deliverable (D4.2 – Blockchain based data) to be submitted in two months, i.e., August 2021. Indications have been made (greyed out text) to specific functionality that has been excluded from the MVP and will be implemented at a second stage.

Next steps mainly include dealing with confidentiality requirements and extending the use case to cover aspects of invoicing and payments.

All VGM implementation artifacts are stored and managed in the project repository.

- Code related to conPESO (the front-end and the backend integrated with BC client) is stored at: https://egitlab.iti.es/dataports-private/vgm
- Blockchain backend, including the chaincodes and the configuration network, is stored at: https://egitlab.iti.es/dataports-private/bc-vgm

6 DATA SHARING BLOCKCHAIN NETWORK – CONTAINER PICK-UP USE CASE

This section describes the Container Pick-Up (CPU) blockchain application in the port of Thessaloniki, the second use case in which BC plays the role of sharing data implemented so far in the project.

6.1 USE CASE DESCRIPTION AND MOTIVATION

As the transport industry becomes more demanding in terms of increased efficiency and lowered costs, sharing and timely provision of accurate information within the network of involved business organisations is more critical than ever. The Port of Thessaloniki (ThPA) has embraced the trend for digitalisation of port activities, where new systems and techniques are added to the capabilities of available systems. Crucial to this effort is improving management of supply chains. More particularly, ThPA is going to make use of the distributed ledger technology (DLT) aspect of the DataPorts platform, to implement a use case that facilitates secure information sharing between ThPA and supply chain-related parties (shipping agents and trucking companies), concerning the business cycle of the electronic request to pick-up a container. We call this the Container Pick-Up (CPU) use case.

Container loading or discharge is a core activity for ThPA. The scope of this use case is to support ThPA ecosystem stakeholders to better organize and manage the pickup and delivery of containers, with emphasis to land gate access. The use case deals with the scenario of discharging a container from the port's premises. The procedure involves a shipping agent, as the beneficial cargo owner who wishes to pick-up a container from the yard, and a trucking company that picks up the container, on the agent's behalf. The type of data exchanged and shared have been chosen to support all relevant operational and decision-making needs. The main piece of data that drives the whole use case is the COREOR (COntainer RElease ORder) message, which is an order to release containers that gives permission for them to be picked up by, or on behalf, of a specified party. It is used in Electronic Data Interchange (EDI) between trading partners involved in administration, commerce, and transport.

Implementation of this use case aims at supporting ThPA in taking proactive and reactive actions regarding the operation of the Gate Control System, improving the operational performance of the port supply chain, increasing visibility of operations for the stakeholders involved, while also improving the environmental burden caused by truck traffic.

6.2 USE CASE REQUIREMENTS

The use case, in short, works as follows: A shipping agent sends a COREOR request through our front-end, specifying, among other data, the trucking company to pick-up the container. An appropriate port employee checks the COREOR eXtensible Markup Language (XML) and relevant info (invoicing, customs clearance); if everything is in order, they issue a permit ID for that COREOR, to be returned to the shipping agent and the trucking company. Using this Permit ID, the trucking company can proceed to make a booking to pick-up the container. The booking data contains, among other, the name of the driver and the truck's license plate number. Upon successful submission, the trucking company receives the Quick Response (QR) code needed to enter the gate.

In designing a blockchain solution for the CPU use case, we first identify the organisations that are involved in the use case interactions. These are the *network participants*. We examine this from the perspective of the blockchain network and, therefore, these interactions pertain to reading and writing to/from the common blockchain ledger. The following participants have been identified for the CPU use case (Figure 23):

- *Port Terminal Operator*: Staff that processes the requests (verifying COREOR XML message, verifying booking, related paperwork)
- *Shipping Agent*: Employee of the company that sends the COREOR request and receives the COREOR validation and Permit ID

- *Trucking company*: Employee of the company specified by the Shipping Agent as eligible to pick-up the container. Upon COREOR submission, they receive a notification containing the Permit ID. They proceed to provide the booking data (e.g., truck license plates) and upon its submission, receive the QR code to be shown at the gate upon arrival.

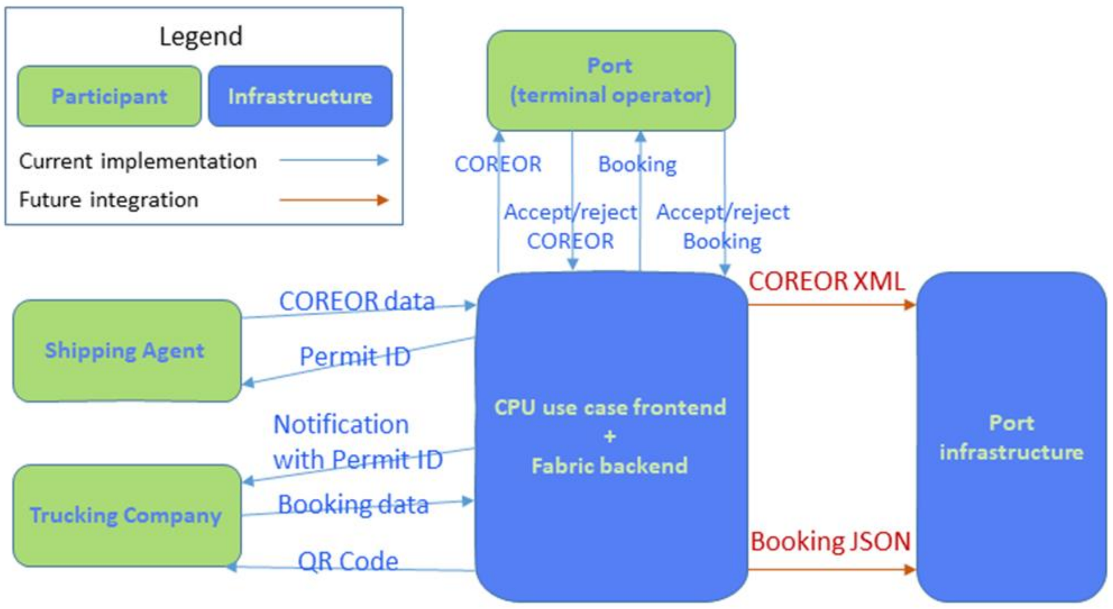


Figure 23 - Data providers and data consumers in the CPU use case

The functionalities that need to be supported are the following:

- Shipping agent
  - Make COREOR request
  - View all created COREORs
- Trucking company
  - Get notification, when COREOR has been issued and references them
  - Make booking request
  - View all relevant COREORs
  - View all relevant bookings
- Port
  - View all COREORs
  - View all bookings
  - Be able to accept/reject COREOR
  - Be able to accept/reject booking

As can be seen in Figure 23, the next stage in the implementation of the use case entails the integration with the port’s computing systems. This refers to the forwarding of the data contained in the COREOR and booking entities in the form of a COREOR XML message and a booking JSON message.

6.3 USER STORIES

While requirements focus on functionality (see previous section), that is, what the solution should do; user stories focus what the person using the solution should be able to do. In this sub-section we focus on the user stories associated with this use:

As a Shipping Agent administrator, I want to:

- Be able to submit a new COREOR, requesting a permit for the release of a container from the port premises
- Be able to receive the permit ID of an accepted COREOR
- View the details of my own COREOR requests

As a Trucking Company administrator, I want to be able to:

- Receive a notification when a COREOR is accepted, to be able to make a new booking
- Make a new container pick-up booking, based on an accepted COREOR request, requesting permission to load and haul away the specified container
- Receive a QR code for an accepted booking, that will allow the driver to enter the port gate
- View the details of my own booking requests

As a Port Authority administrator, I want to be able to:

- View COREOR requests and accept or reject them
- Issue a permit ID for each accepted COREOR
- View the details of all COREOR requests
- View booking requests and accept or reject them
- Issue a QR code for each accepted booking
- View the details of all booking requests

6.4 BLOCKCHAIN ARCHITECTURE

This section describes a high-level overview of the CPU solution architecture (the different infrastructural and software layers comprising the whole blockchain solution), and the component design and integration flows of these components. Subsequent sections delve into more technical information about various aspects of the design.

6.4.1 Architecture overview

The blockchain architecture for the data sharing blockchain network of the CPU use case is depicted in Figure 24. The smart contracts that have been developed and deployed on the Hyperledger Fabric network are satisfying the application functionalities according to the container pick up business processes of the port. At this stage of the implementation, there has not yet been any integration of the blockchain network with any of the port's systems, since the focus has been on completing the main functionality of the use case, while taking care to involve all participants of the container pick-up workflow. In the second stage, the application will be integrated with the relevant port-side platforms, to complete the COREOR/booking workflow.

In a more detailed manner, the participants of the application act as data providers or as data consumers according to the business processes. The applicative functionality of the CPU use case is developed and deployed as smart contracts. Data is stored in the blockchain ledger and each participant is able to write or read data in a permissioned way. The range of roles for the participant organizations are: "port authority", "trucking company", "shipping agent". Each organization contributes to the CPU use case either as a data provider or data consumer.

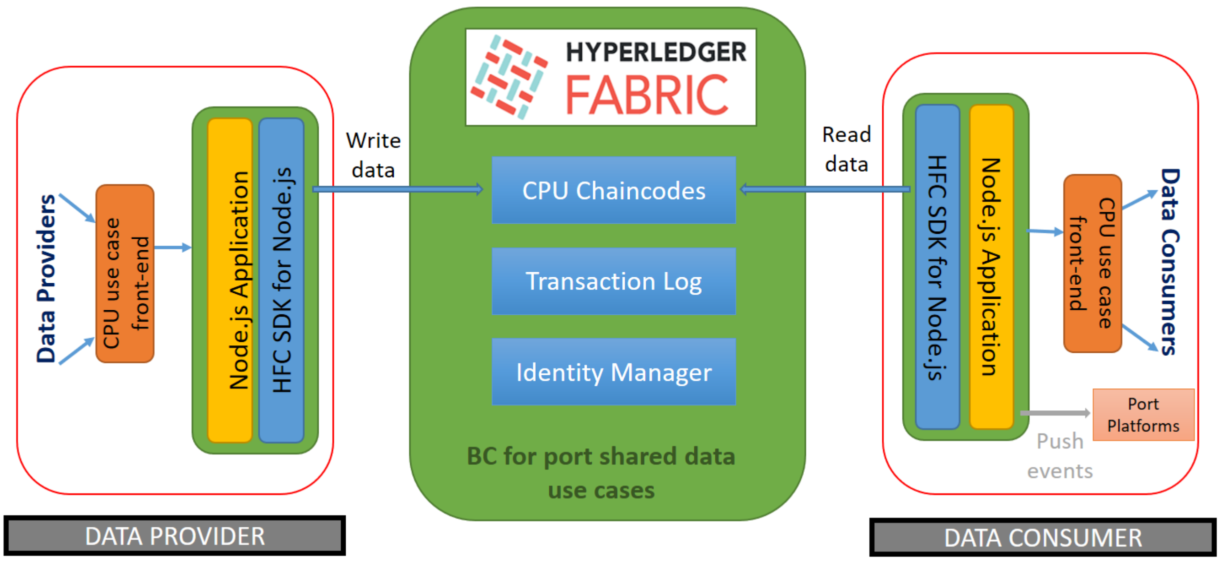


Figure 24 - CPU blockchain high level architecture overview

6.4.2 Application overview

The business logic of our blockchain application has been implemented as a series of smart contracts (chaincode) in Hyperledger Fabric. These smart contracts are accessed by the blockchain client application through use of the HFC SDK. The client application is, in turn, used by our front-end application.

The overall VGM solution will provide the following components:

- Hyperledger Fabric v 1.4.8 components
- CPU chaincodes
  - Lifecycle management of COREOR requests (Create/Read/Accept/Reject)
  - Lifecycle management of booking requests (Create/Read/Accept/Reject)
- Hyperledger Fabric SDK for Node.js
- The blockchain client application, which will be a wrapper for the Fabric SDK and provide read/write operations for the use case's data entities
- Front-end application for use-case specific front-end functionality for all use case stakeholders

Integration with the port's computing platforms will be added in the next stage.

Figure 25 shows the high level conceptual blockchain application architecture for the container pick-up use case. At the top, we can see the CPU use case frontend application, which conceptually is like three distinct applications (shown as blue boxes) in one, each box pertaining to one of the roles involved (shipping agent, trucking company, port operator). The overall frontend application makes use of our blockchain client application through REST calls to a Node.js backend.

In the middle layer, we see the smart contracts, specific to the CPU use case, which the blockchain client application calls. The query/read chaincode components are shown as green boxes, while the invoke/write components are shown yellow. It is worth mentioning that invoking the contract for issuing a COREOR permit amounts to accepting the COREOR, while invoking the contract for confirming a booking amounts to accepting the booking.

Finally, at the bottom we see the Fabric blockchain layer, where the COREOR and booking information is kept.

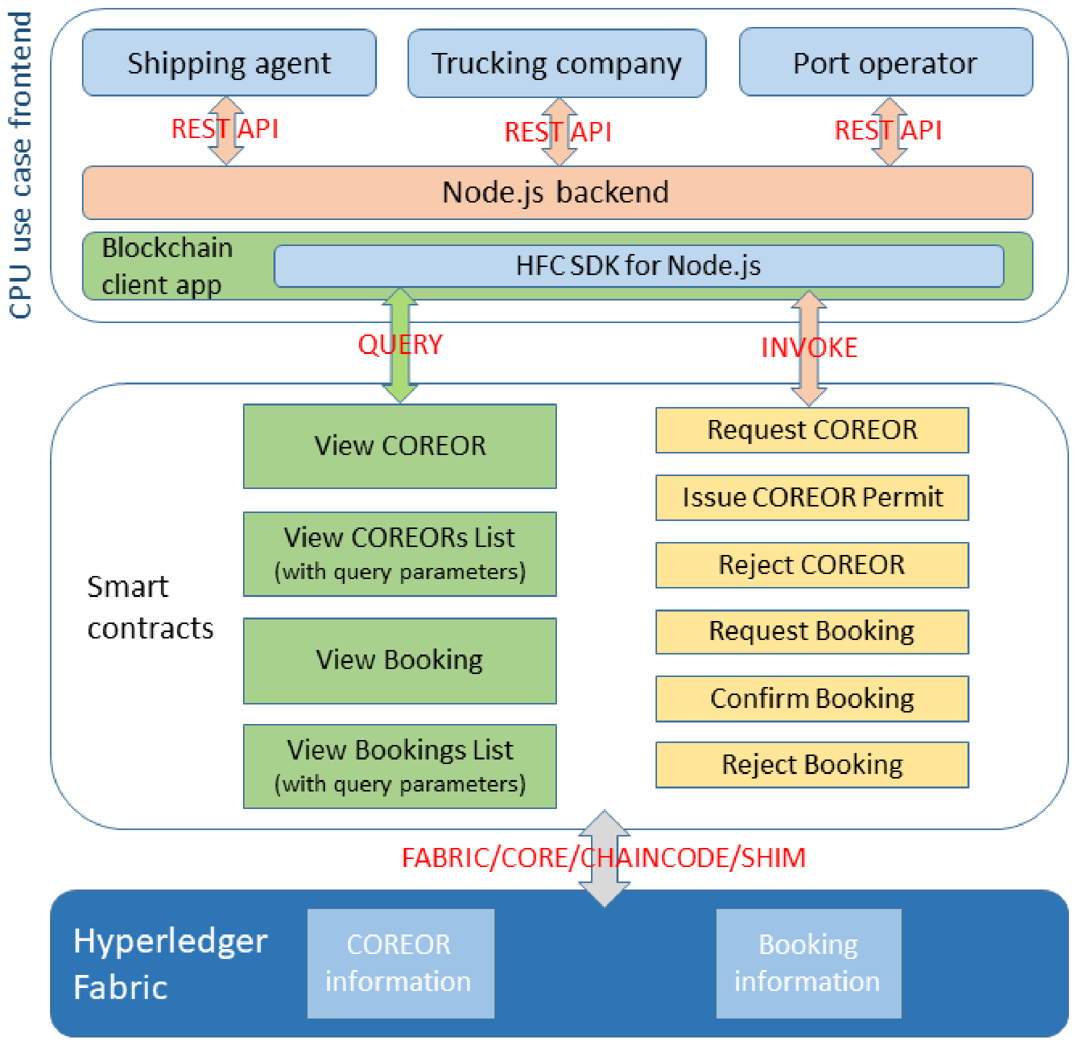


Figure 25 - CPU blockchain application architecture

6.4.3 Component overview

As with Figure 16 for the VGM use case, Figure 26 offers a view of the CPU use case solution architecture where the emphasis is on the different physical components of the solution. We accommodate the different components in three different functional layers: the data, application, and presentation layers.

Firstly, the data layer comprises all the components for storing and retrieving data from the blockchain. This includes the blockchain infrastructure – the CA, ordering service, peer nodes (each containing a copy of the ledger), and the smart contract management and execution capabilities.

Next, the application layer acts as the middleware components that connects the front-end applications and the backend data layer. This layer provides APIs to the presentation layer and implements this API using the Fabric SDK to invoke the chaincode functions and interact with the ledger. Currently, our business logic lies in smart contracts, however, if needed, this layer could also host business logic components that could process information provided either by the presentation or data layers.

Finally, the presentation layer includes the frontend components and provides authenticated and authorized front-end capabilities to different end users. This includes dashboard and front-end applications providing end users with COREOR and container pick-up booking capabilities.

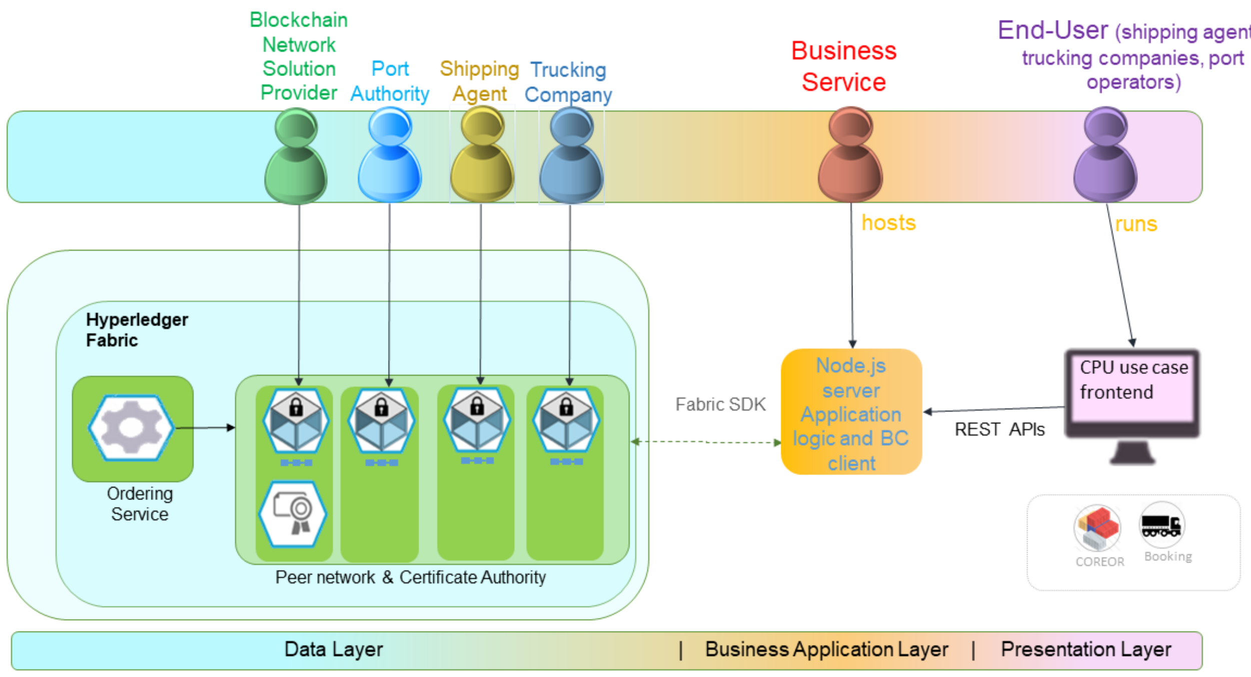

Blockchain Network Solution Provider
Port Authority
Shipping Agent
Trucking Company
Business Service
End-User (shipping agents, trucking companies, port operators)
hosts
runs
Hyperledger Fabric
Ordering Service
Peer network & Certificate Authority
Fabric SDK
Node.js server Application logic and BC client
REST APIs
CPU use case frontend
COREOR
Booking
Data Layer | Business Application Layer | Presentation Layer

Figure 26 – CPU blockchain component architecture

6.4.4 Blockchain network

Figure 27 shows the CPU blockchain network. This is a dynamic blockchain network that initially contains only one organization, the *blockchain network solution provider*. Its network administrator is responsible for registering new organizations on the network. More specifically, the provider's network administrator registers a new organization and that organization's administrator. When it comes to various other users of an organization, they can be registered either by the network administrator or by the administrator of that organization. At blockchain start-up, the blockchain solution provider runs one orderer node, one CA and one peer node. Each new organization joining the blockchain network adds one peer node of its own. More specifically, the CPU blockchain network contains:

- The blockchain network solution provider that is the only organization present at network start-up and is also the only one that is running an orderer node and a CA, in addition to its peer node.
- Each new organization joining the network belongs to one of the following categories: “port authority”, "trucking company" or "shipping agent", and adds a new peer node. Its certificates are generated and managed by the solution provider's CA.

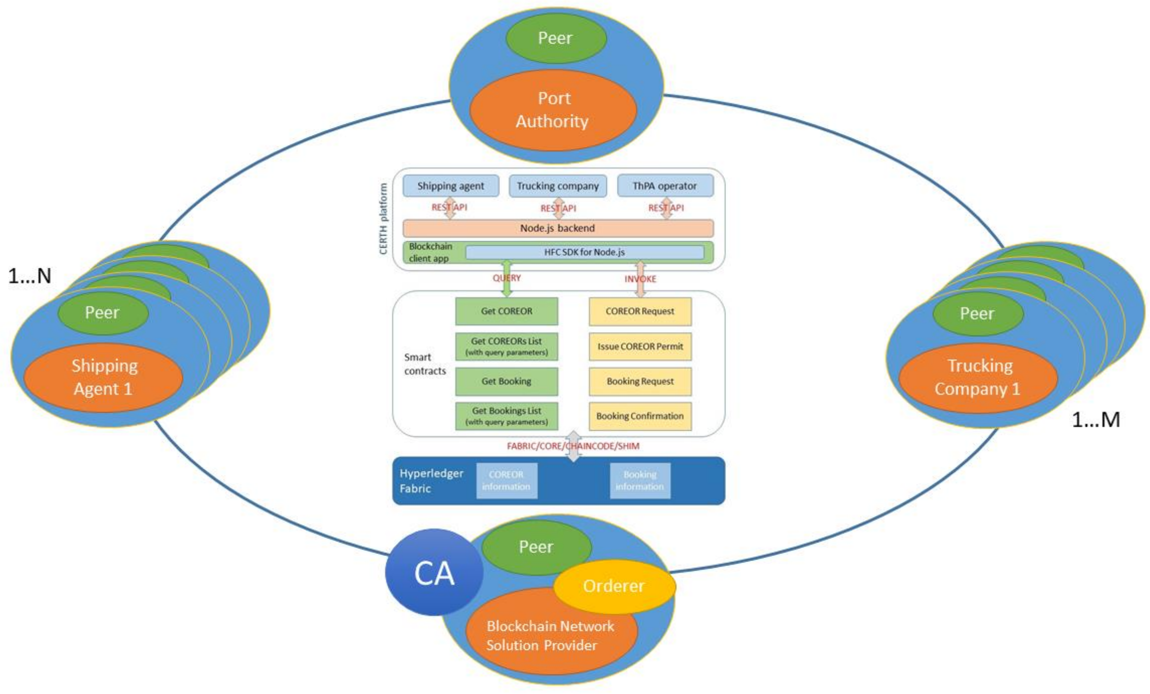


Figure 27 - CPU blockchain network

6.4.5 Front-end

The frontend for the CPU use case offers a different dashboard view which depends on the user role (user, organization administrator, network administrator) and the user organization's role (Port Authority, Shipping Agent, Trucking Company or any other organization role). The front-end communicates with the blockchain by calling APIs which interface with the blockchain.

- Backend (APIs):
  - Programming language: Javascript
  - Web development framework: Express.js
  - Javascript runtime: Node.js
- Front-end:
  - Javascript Framework for SPA: Angular
  - Libraries for the UI: PrimeNG
  - CSS and Javascript framework to make the front-end responsive: Bootstrap
- Network administrator:
  - Home (Second stage)
  - Organization & Users: all organizations in a tabular view
    - Add Organization: a new organization can be added
    - Profile (selected organization): organization profile details
    - Organization Users (selected organization): organization users
      - Add User: a new user can be added
      - Profile (selected user): profile of the selected user can be viewed and updated
  - Profile: personal profile can be viewed and updated

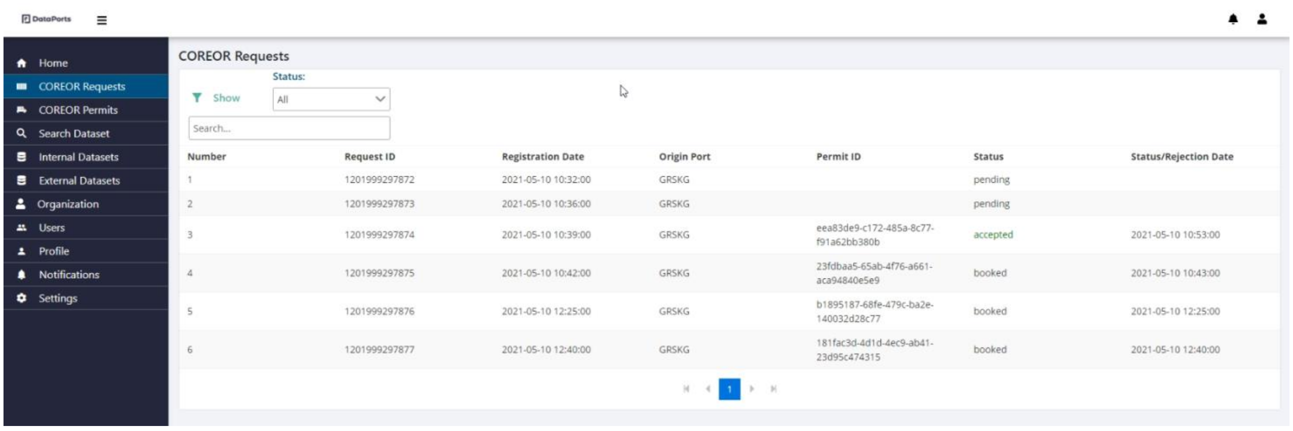
    - Notifications (Second stage)
    - Settings (Second stage)
- User (any user which is neither organization administrator nor network administrator):
    - Home (Second stage)
    - Search Dataset: all datasets which belong to others in a tabular view
        - Dataset Metadata: the user can view dataset metadata and can request more permissions for this dataset
    - Internal Datasets: all datasets which belong to this user in a tabular view
        - Upload File: a file and its metadata can be uploaded
        - Details (selected dataset): details (e.g., metadata) about the selected dataset
        - Permissions (selected dataset): shows the users who have access permissions to this dataset and the types of permissions they have – the dataset owner can also use this screen to revoke permissions.
        - Content (selected dataset): dataset content (Second stage)
        - History (selected dataset): history changes for the selected dataset
    - External Datasets: all datasets which belong to others for which the user has permissions in a tabular view
        - Details (selected dataset): details (e.g., metadata) about the selected dataset – extra permissions can be requested
        - Content (selected dataset): dataset content (Second stage)
        - History (selected dataset): history changes for the selected dataset
    - Profile: personal profile can be viewed and updated
    - Notifications: notifications regarding datasets permissions requests
        - Notification (selected): there are details about the request, it can be accepted or rejected
    - Settings (second stage)
- Administrator of any organization:
    - Home (Second stage)
    - Search Dataset: all datasets which belong to others in a tabular view
        - Dataset Metadata: the user can view dataset metadata and can request more permissions for this dataset
    - Internal Datasets: all datasets which belong to this user in a tabular view
        - Upload File: a file and its metadata can be uploaded
        - Details (selected dataset): details (e.g., metadata) about the selected dataset
        - Permissions (selected dataset): shows the users who have access permissions to this dataset and the types of permissions they have – the dataset owner can also use this screen to revoke permissions.
        - Content (selected dataset): dataset content (Second stage)
        - History (selected dataset): history changes for the selected dataset
    - External Datasets: all datasets which belong to others for which the user has permissions in a tabular view
        - Details (selected dataset): details (e.g., metadata) about the selected dataset – extra permissions can be requested

    - Content (selected dataset): dataset content (Second stage)
    - History (selected dataset): history changes for the selected dataset
  - Organization: organization profile can be viewed and updated
  - Users: all users in a tabular view
    - Add User: an organization user can be added
    - Profile (selected user): profile of the selected user can be viewed and updated
  - Profile: personal profile can be viewed and updated
  - Notifications: notifications regarding datasets permissions requests
    - notification (selected): there are details about the request, it can be accepted or rejected
    - Settings (Second stage)
- Port Authority administrator extra pages:
  - COREOR Requests: all COREOR requests in a tabular view
    - COREOR request (selected): the user views details and can approve or reject it
  - COREOR Permits: all COREOR permits in a tabular view
    - COREOR Permit (selected): the user views details and can approve or reject booking request
    - Data Analytics (Second stage)
- Shipping Agent administrator extra pages:
  - COREOR Requests: all COREOR requests in a tabular view
    - New COREOR Request: the user can make a new request
    - COREOR request (selected): the user views details
  - Data Analytics (Second stage)
- Trucking Company administrator extra pages:
  - Bookings
    - Booking (selected): all bookings in a tabular view
    - New Booking: the user can make a new booking
  - Bookings Notifications
    - Notification (selected): the user views details and can make a new booking with a new Permit ID
  - Data Analytics (Second stage)

Organization Administrator pages are also available for Port Authority, Shipping Agent or Trucking Company administrators.

Figure 28 and Figure 29 show two examples of screenshots in the new offered CPU blockchain based solution.

In Figure 28, the COREOR requests page (inside the dashboard of a Port Authority administrator) is shown. In this page, COREOR requests are presented in a tabular view. The Port Authority administrator can select a request by left clicking on it to view more details.



COREOR Requests

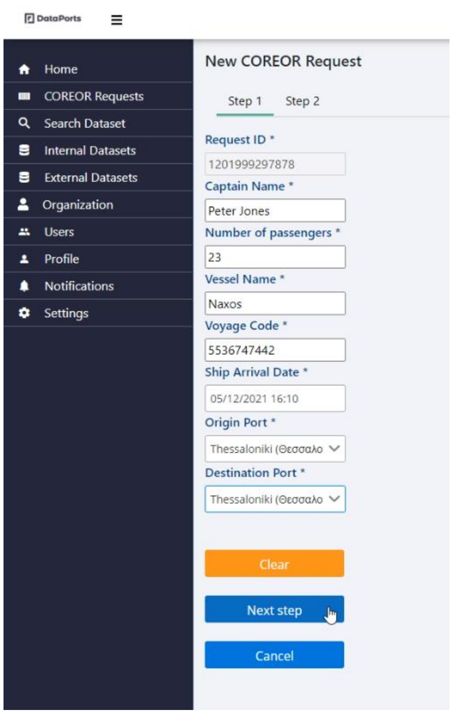
Status: All

Search...

Number	Request ID	Registration Date	Origin Port	Permit ID	Status	Status/Rejection Date
1	1201999297872	2021-05-10 10:32:00	GRSKG		pending	
2	1201999297873	2021-05-10 10:36:00	GRSKG		pending	
3	1201999297874	2021-05-10 10:39:00	GRSKG	eea83de9-c172-485a-8c77-f91a62bb380b	accepted	2021-05-10 10:53:00
4	1201999297875	2021-05-10 10:42:00	GRSKG	23fdbaa5-65ab-4f76-a661-aca94840e5e9	booked	2021-05-10 10:43:00
5	1201999297876	2021-05-10 12:25:00	GRSKG	b1895187-68fe-479c-ba2e-140032d28c77	booked	2021-05-10 12:25:00
6	1201999297877	2021-05-10 12:40:00	GRSKG	181fac3d-4d1d-4ec9-ab41-23d95c474315	booked	2021-05-10 12:40:00

Figure 28 - COREOR requests page (inside Port Authority administrator dashboard)

Figure 29 presents the "New COREOR Request" page (inside the dashboard of a Shipping Agent administrator) in which the logged-in Shipping Agent administrator can make a new COREOR request. As shown in the figure, the COREOR request consists of two steps.



New COREOR Request

Request ID *	1201999297878
Captain Name *	Peter Jones
Number of passengers *	23
Vessel Name *	Naxos
Voyage Code *	5536747442
Ship Arrival Date *	05/12/2021 16:10
Origin Port *	Thessaloniki (Θεσσαλο
Destination Port *	Thessaloniki (Θεσσαλο

Figure 29 - New COREOR Request page (inside Shipping Agent administrator dashboard)

6.5 SUMMARY AND NEXT STEPS

When it comes to the CPU local use case, the aim has been to facilitate the secure, transparent, and verifiable incorporation and usage, by the Port of Thessaloniki, of the COREOR digital message and the corresponding trucking company booking in the process of picking up a container from the port's premises. The overarching goal is to improve the operation of the Gate Control System and the operational performance of the port supply chain. The solution consists of a Fabric channel to which the port and supply chain-related parties participate, each with their own peer node. A separate organization, the Blockchain Network Solution Provider, plays the role of network administrator that provides the network's initial peer node, a CA to produce all identities, and an orderer node. We have implemented all business logic on chaincode running on the peers and exposed through a Node.js API backend to a frontend application that caters to the needs of the users of the different parties.

Section 6 introduces the CPU use case blockchain network and its business motivation. We present the detailed design of the CPU blockchain, including architecture, application, component, network, and front-end overviews. Implementation is planned to be carried out in phases starting with a Minimum Viable Product (MVP) which is the core of next deliverable (D4.2 Blockchain based data governance) to be submitted in two months, i.e., August 2021. Indications have been made (greyed out text) to specific functionality that has been excluded from the MVP and will be implemented at a second stage.

The important next step in the implementation of the use case is the integration with the port's computing systems, more specifically the forwarding of the COREOR and booking data in the form of a COREOR XML and a booking JSON message to keep the port's systems up to date. It should be noted that Data governance rules could be used in the CPU local use case. The users can request or grant permissions for datasets, as is in the Data governance rules use case.

All artifacts of the CPU use case blockchain network implementation are stored at: https://varlab.iti.gr:9443/dataports_871493/dataports_871493.git.

7 CONCLUSIONS AND NEXT STEPS

D3.4 Permissioned blockchain network describes the journey blockchain activities made since the beginning of the project. While in the DoA, BC was assumed to only provide a data governance framework that complies with the IDS reference architecture for data exchange in a secure and reliable way among users of the DataPorts platform, we extended BC in two aspects:

- The implementation of the IDS broker and clearing house components in the IDS architecture.
- The application of blockchain to improve ports operations. We demonstrated this role by building up two BC application for two port use cases: Verified Gross Mass for the Valencia port and the Container Pick-Up for the Thessaloniki port.

DataPorts project uses Hyperledger Fabric as the underlying blockchain technology, the most mature permissioned blockchain (open source) available today for the implementation of all specified blockchain networks. It is important to note that the data governance blockchain solution can be applied to any scenario as a means of defining and enforcing access rules to assets in a transparent, verifiable, trackable, and immutable manner, thus making this a generic component that can be replicated in any domain in which such a requirement fits. Without the loss of generality, the blockchain networks at the ports, although being specific to certain processes, can be replicated to similar processes in other ports as well.

The deliverable in hand delineates the overall blockchain implementation activities in the scope of the DataPorts project and presents the design of the three blockchain networks that have been devised in the project, namely the Data Governance, VGM, and CPU.

The main outcomes of this work are:

- Design of the three blockchain networks.
- Implementation of the IDS broker and clearing house as part of the Data governance BC network
- Placement of the blockchain components in the overall DataPorts platform and their interactions with the platform components.
- Definition of an MVP for each of the networks.
- Set-up of three BC networks infrastructures for the hosting of the three BC networks.
- An incremental plan for the implementation of the three networks.

All our development artifacts are stored and managed in the following GIT repositories with the following URLs:

- https://varlab.iti.gr:9443/dataports_871493/dataports_871493.git
- https://umane.everis.com/git/DATAPORTS/dataportsbck.git
- https://egitlab.iti.es/dataports-private/vgm
- https://egitlab.iti.es/dataports-private/bc-vgm
- https://varlab.iti.gr:9443/dataports_871493/dataports_871493.git

A concrete summary of main outcomes as well as specific future activities for each of the three networks are articulated in the summary of Sections 4, 5, and 6 respectively. More generally, we are finalizing the three MVPs that will be the core of our next deliverable to be submitted in two months from now, i.e., August 2021.

8 REFERENCES AND ACRONYMS

8.1 REFERENCES

[1] N. Gaur, L. Desrosiers, V. Ramakrishna, P. Novotny, SA. Baset, and A. O'Dowd, Hands-On Blockchain with Hyperledger, 2018.

[2] IDS Reference Architecture Model [Online] https://internationaldataspaces.org/wp-content/uploads/IDS-Reference-Architecture-Model-3.0-2019.pdf

8.2 ACRONYMS

Acronym List	
API	Application Programming Interface
BC	Blockchain
CA	Certification Authority
COREOR	COntainer RElease ORder
CPU	Container Pick-Up (use case)
CRUD	Create, Read, Update, Delete
CSS	Cascading Style Sheets
DoA	Description of Action
EDI	Electronic Data Interchange
HFC	Hyperledger Fabric Client
IDS	International Data Spaces
JSON	JavaScript Object Notation
MVP	Minimum Viable Product
MX	Month X of the project
ORM	Object–Relational Mapping
PCS	Port Community System
QR	Quick Response
REST	Representational State Transfer
SDK	Software Development Kit
SOLAS	Safety Of Life At Sea
SPA	Single-Page Application
UI	User Interface
URL	Uniform Resource Locator
VGM	Verified Gross Mass
WP	Work Package
XML	eXtensible Markup Language

Table 4 - Acronyms