Project Acronym:	MARKET4.0
Grant Agreement number:	822064 (H2020-NMBP-PLUG-2018-IA)
Project Full Title:	A Multi-Sided Business Platform for Plug and Produce Industrial Product Service Systems

DELIVERABLE

D1.2 - Reference Architecture and Platform Specifications

Dissemination level	PU - Public
Type of Document	Report
Contractual date of delivery	30/04/2019
Deliverable Leader	INTRASOFT
Status & version	Final – V1.0, 30/04/2019
WP / Task responsible	WP1 – T1.2
Keywords:	Architecture, Micro-services, IDS, app-store, marketplace, apps

Executive Summary

This document aims to document the Reference Architecture of MARKET4.0 platform. The designed is influenced one hand by the IBM, e-commerce for scalable, secure digital retail apps architecture based on IBM cloud and on other hand form D1.1 requirements coming from the platform's stakeholders. A high-level overview is presented in the figure below:

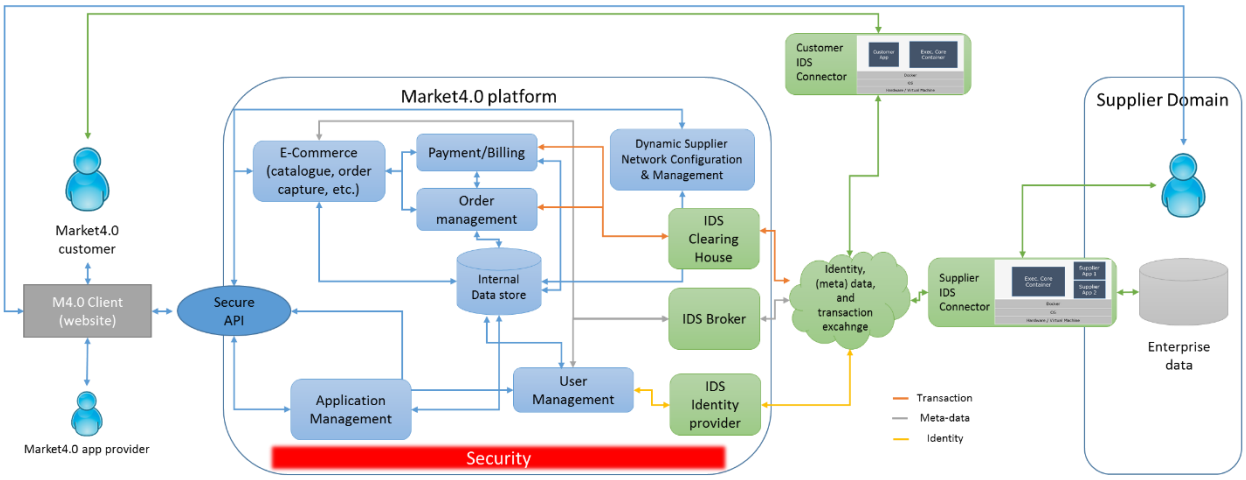

Figure 1 High level overview of MARKET4.0 architecture

The design is based on the IDS ecosystem (in green) for peer-to-peer data exchange and a series of other components realizing the features defined in the requirements. This design considers the following user roles: (1) The customer (looking to buy equipment/services/applications), (2) the supplier (looking to sell equipment/services/applications) to the customer, and (3) the app provider is looking to provide added-value applications to facilitate the interaction between customers and suppliers.

The architecture in this document provides a specification (see section 4) for those functionalities that don't involve peer-to-peer data exchanges and those common components that facilitate peer-to-peer data exchanges (i.e. IDS clearing house, IDS Broker, IDS identity provider, and a common minimal implementation of the connector).

A brief overview of non-IDS components is as follows:

1. The ***M4.0 Client*** facilitate the customer to interact with the functionality of the marketplace.
2. The ***Secure API*** is a secure gateway into the APIs of the various components.
3. ***E-commerce*** acts as an orchestrator of the Payment, Order Management components. It enables features such as product catalogues and user feedback.
4. ***Payment/Billing*** facilitates the initiation of the billing process and provides an overview of the billing status.
5. ***Order Management*** supports order processing, and order visibility.

6. ***User Management*** deals with user management related functionality.
7. ***Application Management*** enables the app providers to manage their applications to the marketplace.
8. ***Dynamic Supplier Network Configuration & Management*** enables the creation and management of Dynamic Supply Networks using automatic matching of potential suppliers.
9. ***Internal Data store*** provides the mechanism to store and retrieve information coming from all other components of the marketplace.
10. ***Security*** is a cross-cutting feature of the architecture grouping together several mechanisms to secure the interactions of the components between them and with the outside world.

Deliverable Leader:	INTRA-LU
Contributors:	ENG, IDSA, OBEO, SEGULA, OPENPLUS, SCC, LMS, TNO, POLIMI, TECNALIA
Reviewers:	Kosmas Alexopoulos (LMS)
Approved by:	Kosmas Alexopoulos (LMS)

Document History			
Version	**Date**	**Contributor(s)**	**Description**
V0.1	06/02/2019	INTRA-LU	Deliverable outline
V0.2	26/03/2019	INTRA-LU	Section 3 draft
V0.3	05/04/2019	INTRA-LU	Section 2 and 3 full draft
V0.4	15/04/2019	INTRA-LU	Section 4 and 5 update
V0.5	23/04/2019	INTRA-LU	Section 4 and 5 update
V0.6	24/04/2019	INTRA-LU	Full draft of section 2-5
V0.7	25/04/2019	INTRA-LU	Full draft
V0.7	25/04/2019	LMS	Review
V0.8	25/04/2019	INTRA-LU	Final
V1.0	30/04/2019	LMS	Editing and final review

Table of Contents

Table of Figures

List of Tables

Definitions, Acronyms and Abbreviations

Acronym	Title
AES	Advanced Encryption Standard
API	Application Programming Interface
B2B	Business to Business
B2C	Business to Customer
DMN	Dynamic Supply Networks
IDS	International Data Spaces
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
M4.0	MARKET4.0
OWASP	Open Web Application Security Project
RA	Reference Architecture
UI	User Interface

1 Introduction

The purpose of this document is to define the reference architecture of the MARKET4.0 platform. The design will have to address requirements set in D1.1 and act as a reference point for the domain specific marketplaces. This document structure is as follows:

1. Section 2 details the high-level overview of the architecture. Before the architecture is introduced the section discusses state-of-the-art architectures including IDS Reference Architecture which acts as an enabler for the MARKET4.0 reference architecture.
2. Section 3 provides details on the specific design of each component of section 3.
3. Section 4 is an overview of how the requirements of D1.1 are addressed.
4. Section 5 provides some conclusions on the work included in this document.

2 Makret4.0 architecture

2.1 Existing reference architectures models

The section provides the details on relevant reference architectures that contribute to the design of MARKET4.0 architecture.

2.1.1 IBM, e-commerce for scalable, secure digital retail apps architecture based on IBM cloud

IBM proposes an architecture for e-commerce solutions[1], based on IBM solutions. Figure 2 provides an overview of the main components and interactions between them.

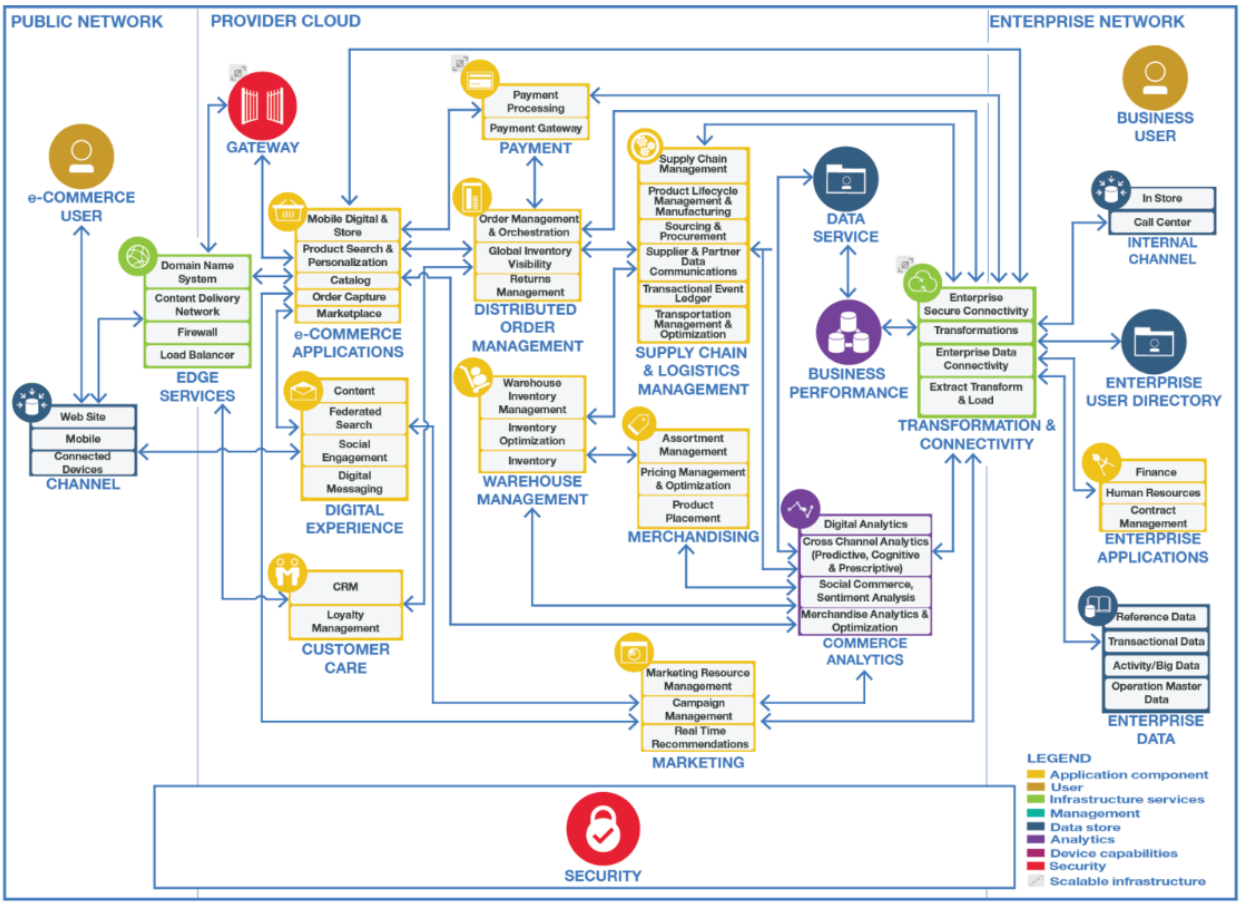

Figure 2 Elements of an e-commerce solution[2]

According to IBM the following user interact with the platform:

[1] https://www.ibm.com/cloud/garage/files/IBM-Advantage-for-e-Commerce.pdf
[2] https://www.ibm.com/cloud/garage/files/IBM-Advantage-for-e-Commerce.pdf

1. **E-commerce user**: a customer who uses various channels to access the commerce solutions on the cloud provider platform or enterprise network.
2. **Business user**: has access to the commerce solutions on the enterprise network.

The following components make up the platform:

1. The **Chanel** provides a seamless shopping experience combining the web, shopping over the telephone, using a mobile device, or all of the above.
2. **Edge services** allow data to flow safely from the Internet into the provider cloud and into the enterprise.
3. **E-commerce applications** enable the management of digital and physical stores, offer cataloguing functionality, and facilitate searching and personalization, order capturing and access to different sellers.
4. **Digital experience** covers the aspects of social engagement and messaging, personalized suggestions, and so on.
5. **The Gateway** allows smart devices to communicate with in-store networks to search or shop and pay.
6. **Payment processing** supports payment transactions using credit cards or electronic fund transfers.
7. **Distributed order management** supports inventory, order processing, and order visibility. It orchestrates the workflow of orders from distribution centres or warehouses, suppliers, and third-party vendors for direct fulfilment and stores.
8. **Supply chain and logistics management** enables systems to plan and manage the products and lifecycle, supply network, and inventory, including replenishments, distribution strategies, partner alliances, and related analytics.
9. **Warehouse management** enables efficient management of warehouse operations.
10. **Merchandising** planning involves marketing the right merchandise or service at the right place, at the right time, in the right quantities, and at the right price with the goal of optimizing margins, gross revenue, or shelf life.
11. **Commerce analytics** enables optimization of the shopper's journey and improves the sales and revenue for the business. Commerce analytics include digital analytics, cross-channel analytics, social commerce and sentiment analysis, and merchandise analytics.
12. **Marketing** involves personalized offers, content, and product presentations that move customers along in their journey from product exploration to purchase decisions to transaction completion. Marketing channels include traditional communication channels, direct mail, email, mobile, and social media.
13. **Data services**: The key to proactive merchandising or faster response to market behaviours is visibility into events and habits combined with data related to the consumer's day-to-day reality.

14. **Business performance** enables describing and understanding the alerts, metrics, and key performance indicators (KPIs) an organization uses to monitor day-to-day commerce activity, keep track of progress against defined goals, and adjust offerings across commerce channels in response to market demand.
15. The **transformation and connectivity** component enable secure connections to enterprise systems with the ability to filter, aggregate, modify, or reformat data as needed.
16. **Internal channel** retailing solutions create an interactive experience whether the customer shops in the store, over the telephone with a customer service representative, or using a web-based call centre.
17. **Enterprise applications** are key data sources for a commerce solution. Enterprise applications use the cloud services and host the legacy applications.
18. **Enterprise data** represents an enterprises master data, reference data and so on.
19. **Enterprise user directory** provides access to the user profiles for both the cloud users and the enterprise users.
20. **Security** supports rigorous security needed at each step in the lifecycle of commerce application—from raw input sources to valuable insights to sharing of data among many users and application components. Security solutions helps detect, address, and prevent security breaches though integrated hardware and software solutions. Security services and products enable identity and access management, protection of data and applications, and actionable security intelligence across cloud and enterprise environments.

2.1.2 International Data Spaces reference architecture

The IDS reference architecture is included here because it is a key component to tackle private data exchange issues such as data sovereignty. Data sovereignty is at the core of the IDS. The IDS specifies techniques, methods and processes to enable trustworthy interaction and data exchange patterns based on the IDS Trusted Connectors. These connectors serve as gateways to sensitive data and provide a controlled environment for any kind of data analytics or processing in the form of isolated containers. Therefore, data owners can restrict the usage of their content but at the same time easily cooperate with other participants of the IDS.

The Industrial Data Space initiative proposes a Reference Architecture Model[3] for this particular capability and related aspects, including requirements for secure and trusted data exchange in business ecosystems. The Reference Architecture consists of five layers and three cross-sectional perspectives. The Business Layer specifies and categorizes the different roles and the main activities and interactions connected with each of these roles. The Functional Layer defines the requirements of the Industrial Data Space. The Process Layer specifies the interactions taking

[3]https://www.fraunhofer.de/content/dam/zv/de/Forschungsfelder/industrial-data-space/IDS_Referenz_Architecture.pdf

place between the different components and provides a dynamic view of the Reference Architecture. The Information Layer defines a conceptual model using Linked Data principles for describing both data and connectors participating in the IDS. The System Layer considers aspects such as integration, configuration, and deployment of these components.

The perspectives Security, Certification, and Governance affect all layers. Security in the context of the IDS contains all aspects related to a protected and trusted data exchange and usage of Data Apps. The Certification Perspective specifies processes to determine the compliance of participants, both organizations and individuals but also software and hardware components, with the IDS requirements. Governance defines the requirements to be met by the business ecosystem to achieve secure and reliable corporate interoperability.

In order to provide a better understating of how the IDS ecosystem operates the **business layer** is further detailed. It consists of the following main roles[4]:

1. The ***Data Owner*** holds all legal rights of, and has complete control over, its data. Usually, a participant acting as a Data Owner automatically assumes the role of the Data Provider as well. However, there may be cases in which the Data Provider is not the Data Owner
2. The ***Data Provider*** makes data available for being exchanged between a Data Owner and a Data Consumer. As already mentioned above, the Data Provider is in most cases identical with the Data Owner, but not necessarily. To submit metadata to a Broker, or exchange data with a Data Consumer, the Data Provider uses software components that are compliant with the Reference Architecture Model of the Industrial Data Space. Providing a Data Consumer with data from a Data Owner is the main activity of the Data Provider. To facilitate a data request from a Data Consumer, the Data Provider should provide a Broker Service Provider (see below) with proper metadata about the data. However, a Broker Service Provider is not necessarily required for a Data Consumer and a Data Provider to establish a connection. Exchanging data with a Data Consumer needs not necessarily be the only activity of the Data Provider. At the end of a data exchange transaction completely or partially executed, for example, the Data Provider may log the details of the successful (or unsuccessful) completion of the transaction at a Clearing House (see below) to facilitate billing or resolve a conflict. Furthermore, the Data Provider can use Data Apps to enrich or transform the data in some way, or to improve its quality.
3. The ***Data Consumer*** receives data from a Data Provider. From a business process modeling perspective, the Data Consumer is the mirror entity of the Data Provider; the activities performed by the Data Consumer are therefore similar to the activities performed by the Data Provider. Before the connection to a Data Provider can be established, the Data Consumer can search for existing data-sets by making an inquiry at

[4] IDSA, IDS Reference Architecture Model, version 3.0, April 2019
https://www.internationaldataspaces.org/publications/reference-architecture-model-3-0/

a Broker Service Provider. The Broker Service Provider then provides the required metadata for the Data Consumer to connect to a Data Provider. Alternatively, the Data Consumer can establish a connection with a Data Provider directly (i.e., without involving a Broker Service Provider). In cases in which the information to connect with the Data Provider is already known to the Data Consumer, the Data Consumer may re-quest the data (and the corresponding metadata) directly from the Data Provider. Like a Data Provider, the Data Consumer may log the details of a successful (or unsuccessful) data exchange transaction at a Clearing House, use Data Apps to enrich, transform, etc. the data received, or use a Service Provider to connect to the International Data Space.

4. The ***Data User*** is the legal entity that has the legal right to use the data of a Data Owner as specified by the usage policy. In most cases, the Data User is identical with the Data Consumer. However, there may be scenarios in which these roles are assumed by different participants.
5. The ***Broker Service Provider*** is an intermediary that stores and manages information about the data sources available in IDS. As the role of the Broker Service Provider is central but non-exclusive, multiple Broker Service Providers may be around at the same time (e.g., for different marketplace domains). An organization offering broker services in IDS may assume other intermediary roles at the same time (e.g., Clearing House or Identity Provider, see below). The activities of the Broker Service Provider mainly focus on receiving and providing metadata. The Broker Service Provider must provide 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. While the core of the metadata model must be specified by the Industrial Data Space (i.e., by the Information Model, see Section 3.4 of the IDS reference architecture), a Broker Service Provider may extend the metadata model to manage additional metadata elements. After the Broker Service Provider has provided the Data Consumer with the metadata about a certain Data Provider, its job is done.
6. The ***Clearing House*** is an intermediary that provides clearing and settlement services for all financial and data exchange transactions. In IDS, clearing activities are separated from broker services, since these activities are technically different from maintaining a metadata repository. The Clearing House logs all activities performed in the course of a data exchange. After a data exchange, or parts of it, has been completed, both the Data Provider and the Data Consumer confirm the data transfer by logging the details of the transaction at the Clearing House. Based on this logging information, the transaction can then be billed. The logging information can also be used to re-solve conflicts (e.g., to clarify whether a data package has been received by the Data Consumer or not). The Clearing House also provides reports on the performed (logged) transactions for billing, conflict resolution, etc.

7. The ***Identity Provider*** should offer a service to create, maintain, manage and validate identity information of and for participants in the Industrial Data Space. This is imperative for secure operation of the Industrial Data Space and to avoid unauthorized access to data.
8. The ***App Store*** provides Data Apps, i.e., applications that can be deployed in the Industrial Data Space to facilitate data processing workflows. Data Apps might be certified by a Certification Body. The App Store is responsible for managing information about Data Apps offered by App Providers. The App Store should provide interfaces for publishing and retrieving Data Apps plus corresponding metadata.
9. ***App Providers*** develop Data Apps to be used in the Industrial Data Space. To be deployable, a Data App has to be com-pliant with the system architecture of IDS. In addition, Data Apps can be certified by a Certification Body in order to increase trust in these applications. Each Data App must be published in the App Store for being accessed and used by Data Consumers and Data Providers. App Providers should describe each Data App using metadata.
10. The ***Vocabulary Provider*** manages and offers vocabularies that can be used to annotate and describe data-sets. In particular, the Vocabulary Provider provides the Information Model of the IDS, which is the basis for the description of data sources. In addition, other domain specific vocabularies can be provided.
11. A ***Software Provider*** provides software for implementing the functionality required by the Industrial Data Space. Unlike Data Apps, software is not provided by the App Store, but delivered over the Software Providers' usual distribution channels, and used on the basis of individual agreements between the Software Provider and the user. This procedure implies that the agreements between Software Providers and Data Consumers, Data Providers, etc. remain outside the scope of the Industrial Data Space.
12. ***Service provider***: If a participant does not deploy the technical infrastructure required for participation in the Industrial Data Space itself, it may transfer the data to be made available in the Indus-trial Data Space to a Service Provider hosting the required infrastructure for other organizations. This role includes also providers offering additional data services (e.g., for data analysis, data integration, data cleansing, or semantic enrichment) to improve the quality of the data exchanged in the Industrial Data Space. From a technical point of view, such a Service Provider can be considered a Data Provider and a Data Consumer at the same time. Unlike the services provided by a Service Provider, Data Apps can be installed in the IT environment of a Data Consumer or Data Provider for implementing additional data processing functionality. To use the functionality of a Data App, the data therefore does not have to be transferred to an external Service Provider.
13. ***Certification Body and the Evaluation Facility*** are in charge of the certification of the participants and the technical core components in the Industrial Data Space.

Figure 3 illustrates how the Business Layer roles interact with each other.

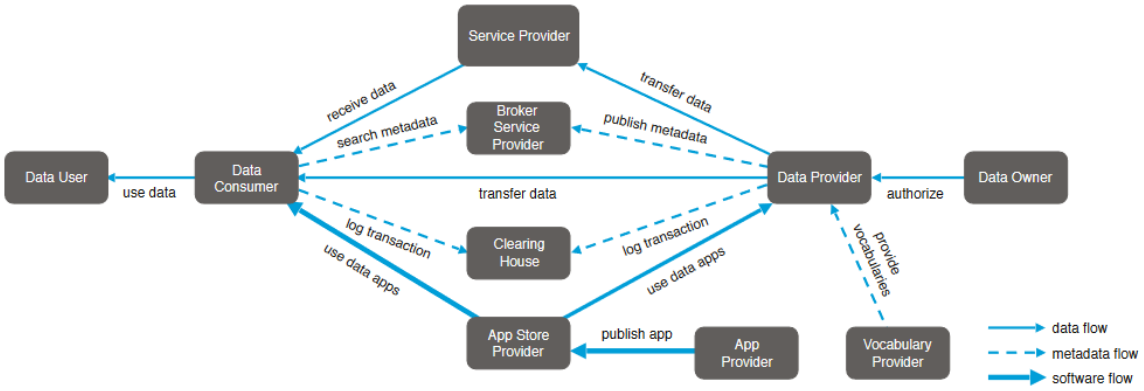

Figure 3 Roles and interactions in IDS

Moreover, in the interaction of the IDS technical components.

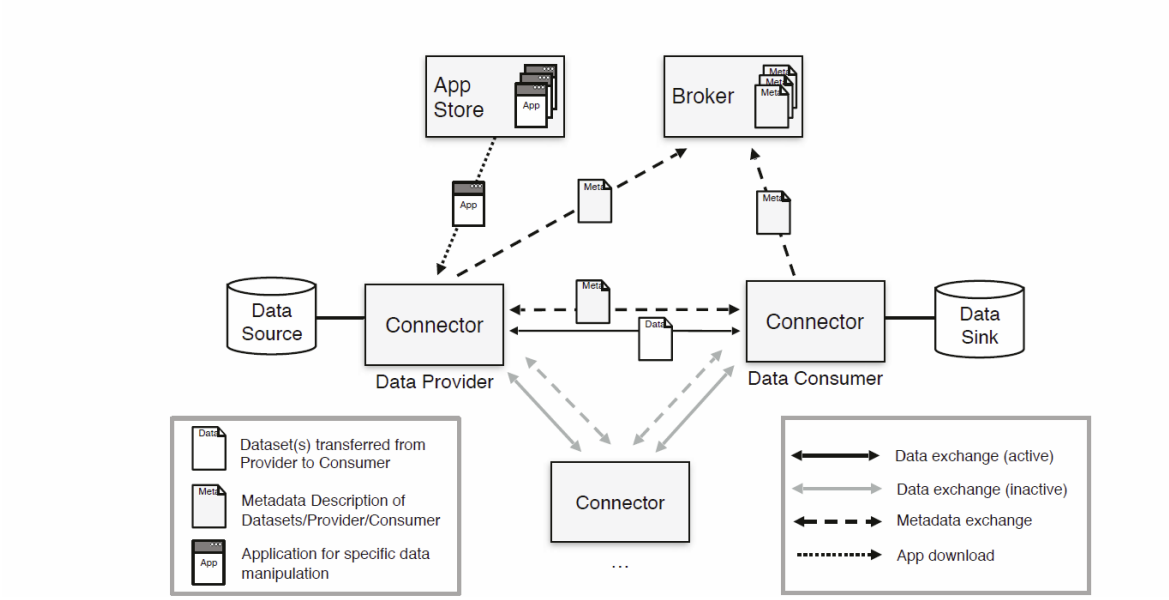

Figure 4: IDS technical components interaction

2.2 High level overview of MARKET4.0 architecture

The section provides a high-level overview of the MARKET4.0 architecture, presents the main high level roles of the architecture, and provides an overview of how the components relate to each other.

2.2.1 High level roles

There're three high level roles with the business layer of theMarket4.0 architecture. The customer, the supplier, and the app provider. A detailed explanation is as follows:

- The **customer** represents a user or organization that is looking to obtain equipment or services from the marketplace or otherwise use the provided functionality.
- The **supplier** represents a user or organization looking to sell equipment or production as a service or production engineering services or applications (software) to the customer. Typically, the supplier will have some publicly accessible data about their offerings and some private information to be shared on a peer-to-peer basis.
- The **app provider** is looking to provide added-value applications to facilitate the interaction between customers and suppliers.

2.2.2 Public and private data exchange

The MARKET4.0 ecosystem foresees two types of data exchanges public and private ones.

1. **Public data exchanges** are handled by the MARKET4.0 platform. This exchange typically involves public supplier data that are stored in MARKET4.0 platform and are sent to the customers. Additionally, meta-data coming from IDS ecosystem are also part of the public data exchanges.
2. **Private data exchanges** are handled on a peer-to-peer basis over the IDS ecosystem. This exchange is facilitated by the IDS connectors.

2.2.3 High level MARKET4.0 component diagram

Figure 5 provides an overview of the MARKET4.0 architecture in terms of major components and interactions. It should be noted that due to the complexity of the IDS data exchanges this drawing contains an abstract representation depicted as Identity, (meta) data, and transaction exchange cloud (see Figure 3 for details). This designed is modular approach based on different services working together.

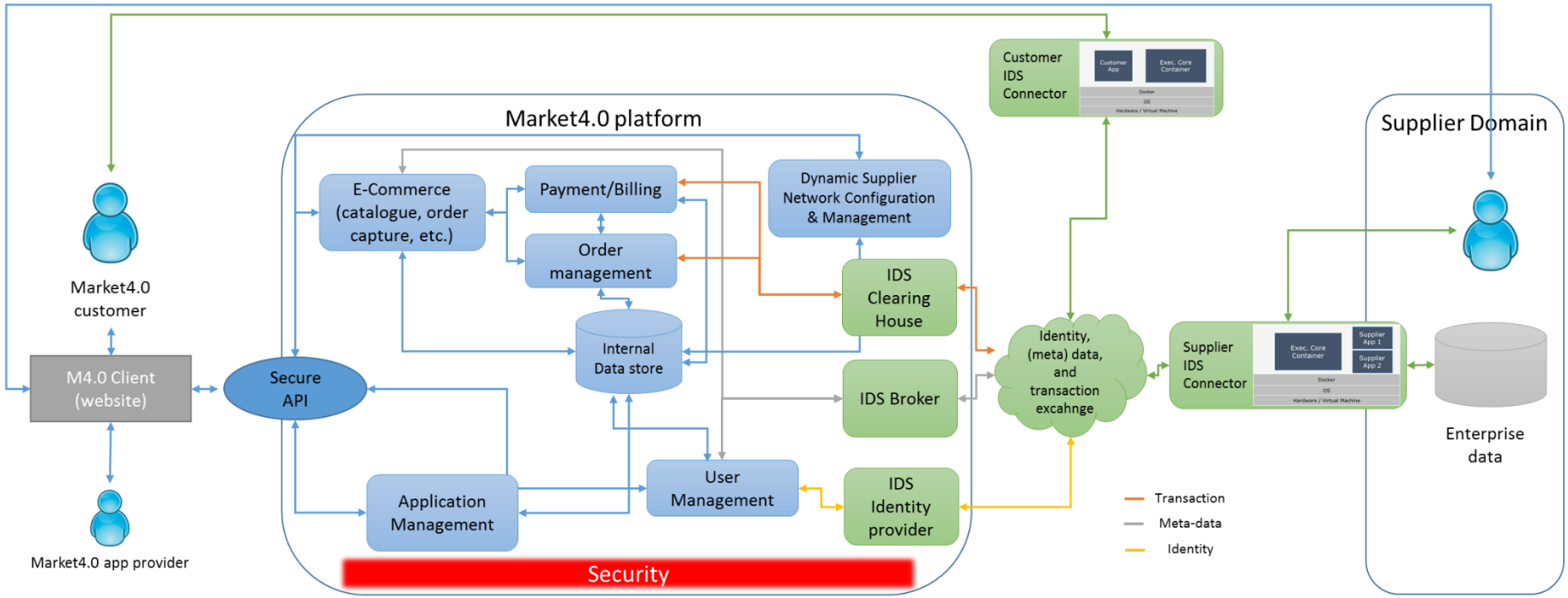

Figure 5 High level overview of MARKET4.0 architecture

The ***M4.0 Client*** facilitate the customer to interact with the functionality of the marketplace. It enables catalogue browsing, initiating orders, accessing customer support, downloading applications and so on.

The ***Secure API*** facilitates communication of the MARKET4.0 platform with the client. Additionally, it facilitates security features such as authentication, and authorization.

E-commerce acts as an orchestrator of the Payment, Order Management components. It enables features such as product catalogues and user feedback, and facilitates interaction with customer requests coming from M4.0 Client through the Secure API. This component also has access to IDS Broker to enrich its responses with IDS meta-data.

Payment/Billing facilitates the initiation of the billing process and provides an overview of the billing status. The component will require a direct connection to IDS clearing house.

Order Management supports order processing, and order visibility and is supported by IDS clearing house.

User Management deals with user registration, controlling access to data, restricting access to legitimate users, and handling the connection with data encryption when required (e.g. for passwords). It also enables synchronization with IDS identity provider and Broker.

Application Management enables the app providers to add their applications to the marketplace.

Dynamic Supplier Network Configuration & Management enables the creation and management of Dynamic Supply Networks using automatic matching of potential suppliers based on the descriptions of data shared via the IDS connectors.

IDS Clearing House provides an implementation of the clearing house as defined in IDS reference architecture (for details see section 2.1.2).

IDS Broker provides an implementation of the Broker service provider as defined in IDS reference architecture (for details see section 2.1.2).

IDS Identity provider provides an implementation of the identity provider as defined in IDS reference architecture (for details see section 2.1.2).

IDS Connector facilitates peer-to-peer data exchange in order to agree on an equipment/app purchase. The connector implementations are in accordance with IDS reference architecture.

Internal Data store provides the mechanism to store and retrieve information coming from all other components of the marketplace.

Security is a cross-cutting feature of the architecture grouping together several mechanisms to secure the interactions of the components between them and with the outside world. Such mechanisms include authentication, authorization, public key infrastructure, and similar.

2.3 Commonalities and differences between IBM RA and MARKET4.0 RA

IBM architecture provides a solution that fits well with both B2C and B2B scenarios and therefore it is expected that there will be significant commonalities and differences with the MARKET4.0 approach (see paragraph 2.2). Table 1 summarizes the commonalties of both architectures:

Table 1 Commonalties between IBM and MARKET4.0 architecture

IBM	Market4.0	Comment
Channel	M4.0 client	The channel and M4.0 client are approximately equivalent. In context of M4.0 the client is mainly considered a website.
E-commerce applications	E-commerce	The MARKET4.0 E-commerce component provides several functionalities such as cataloguing, searching and more.
Payment	Payment/Billing	The actual transfer of funds in the case of MARKET4.0 is outside the scope of the platform. However, the users are able to track the billing status of their orders with the support also of the clearing house mechanisms of IDS.
Distributed order management	Order management	The application for the supply chain management will provide visibility of the supply network status (availability of parts, production capacity and more).
Supply chain and logistics management	Dynamic supplier network	The dynamic supplier network (from IMAGINE project) not only allows user to search for suppliers but also for suppliers to combine their capabilities in order to come up with new offerings. Additionally, it offers functionalities similar to IBM Supply chain and logistics management component.

Apart from commonalties the two architectures have significant differences as well. One key differentiation is how enterprise data is handled. While IBM approach proposes a traditional mechanism to access enterprise data and expose them via the platform to the users, MARKET4.0 utilizes a two-channel approach. One channel goes through the platform and is related to public data and meta-data (from IDS) pertaining to what the seller has to offer. The second channel is a

peer-to-peer connection between the interested parties where private data may be exchanged under the principles of IDS. Thus the MARKET4.0 platform doesn't access/store enterprise data.

Additionally, due to the nature of the MARKET4.0 platform and its requirements components such as the **gateway, warehouse management, merchandising, customer care, and analytics** are not included in the current architecture. However, the design of the architecture allows for such additions if required in the future.

3 Core component specification

This section provides detailed descriptions of the components making up the MARKET4.0 reference architecture.

3.1 IDS connector

According to the IDS reference architecture[5] the connector architecture uses application container management technology to ensure an isolated and secure environment for individual data services. Figure 6 provides an overview of the connector structure for the execution phase of the connector.

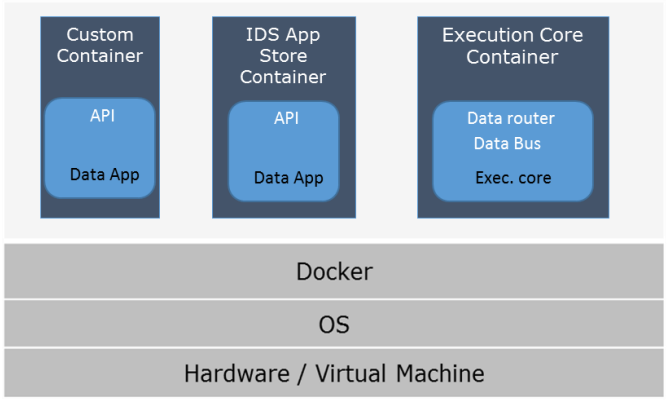

Figure 6 IDS connector structure

The Execution Core Container provides components for interfacing with Data Services and supporting communication (e.g., Data Router or Data Bus to a Connector).

Its Data Router handles communication with Data Services to be invoked according to predefined configuration parameters. In this respect, it is responsible of how data is sent (and received) to (and from) the Data Bus from (and to) Data Services. The Data Router invokes relevant components for the enforcement of usage policies.

Its Data Bus exchanges data with Data Services and Data Bus components of other connectors. It may also store data within a connector. Usually, the Data Bus provides the method to exchange data between Connectors.

The App Store Container represents a certified container downloaded from the App Store, providing a specific Data Service to the Connector.

The Custom Container represents a self-developed Data Service.

[5] IDSA, IDS Reference Architecture Model, version 3.0, April 2019 , https://www.internationaldataspaces.org/publications/reference-architecture-model-3-0/

Each Data App (in either the custom or app store container) defines a public API, which is invoked from the Data Router. This API is formally specified in a meta-description that is imported into the configuration model.

The IDS also supports a configuration phase, which requires the components depicted in Figure 7.

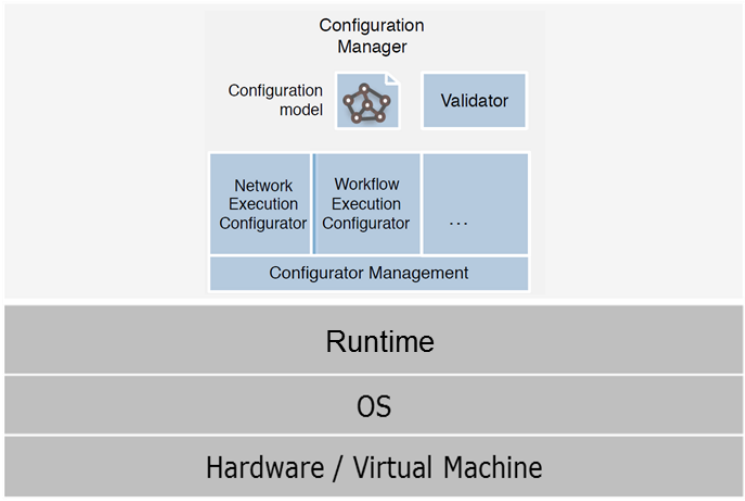

Figure 7 IDS connector components supporting configuration

The Configuration phase of a connector involves the following components:

- The Configuration Manager constitutes the administrative part of a Connector. Its main task is the management and validation of the Configuration Model, followed by deployment of the Connector. Deployment is delegated to a collection of Execution Configurators by the Configurator Management.
- The Configuration Model is an extendable domain model for describing the configuration of a Connector. It consists of technology-independent, inter-connected configuration aspects.
- Configurator Management loads and manages an exchangeable set of Execution Configurators. When a Connector is deployed, the Configurator Management delegates each task to a special Execution Configurator.
- Execution Configurators are exchangeable plug-ins which execute or translate single aspects of the Configuration Model to a specific technology. The procedure of executing a configuration depends on the technology used. Common examples would be the generation of configuration files or the usage of a configuration API.
- The Validator checks if the Configuration Model complies with self-defined rules and with general rules specified by the International Data Spaces, respectively. Violation of rules can be treated as warnings or errors. If such warnings or errors occur, deployment may fail or be rejected.

The details of the configuration model are omitted, but are available in the IDS reference architecture document[6].

It should be noted that execution and configuration components may be developed separately. MARKET4.0 will deliver one such implementation of an IDS connector to support the development of domain specific apps as detailed in D1.3.

Table 2 D1.1 requirements supported by the IDS Connector

ID	Short description	Comment
G-07	Allows a supplier to register a connector.	Enables the configuration step of the registration.
HT-1	The system shall enable the actors to communicate order related data.	Provides the backbone for apps to deliver this functionality.

3.2 IDS clearing house

3.2.1 Summary description

The Clearing House logs all activities performed in the course of a data exchange. After a data exchange, or parts of it, has been completed, both the Data Provider and the Data Consumer confirm the data transfer by logging the details of the transaction at the Clearing House. Based on this logging information, the transaction can then be billed. The logging information can also be used to resolve conflicts (e.g., to clarify whether a data package has been received by the Data Consumer or not). The Clearing House also provides reports on the performed (logged) transactions for billing, conflict resolution, etc. The transactions based on following message types defined in Smart Connected Supplier Network (SCSN) language[7] that makes use of OASIS Universal Business Language (UBL)[8]The following messages are currently constructed in SCSN:

- Order, Order response (i.e. order accept or decline), Order change, and Order status
- Order forecasting
- Request for Quotation, Request for Quotation response
- Invoice
- Dispatch advice (including detailed information on how products are packaged and transported)
- Material certificates/measurement reports attached to purchased goods
- Bill of Material list (used for specifying large sub-assemblies)
- Technical Product Data associated with the requested goods, such as 3D technical drawings

[6] https://www.internationaldataspaces.org/publications/reference-architecture-model-3-0/
[7] https://smartconnected.t4smm.nl
[8] http://docs.oasis-open.org/ubl/UBL-2.2.html

- What-if request and response, which is a "soft" change request, i.e. you can use it to ask your suppliers whether they are able to change the order date of an already purchased item.

3.2.2 Functionality description

The clearing house will have to support the following requirements:

Table 3 D1.1 requirements supported by the IDS Clearing House

ID	Short description	Comment
HT-1	The system shall enable the actors to communicate order related data.	Provides the logging mechanism of the data exchanges.
HT-3	Provides access to accurate and complete information of orders.	Determines the status of an order based on the transactions made.
HT-5	Communication and monitoring of supply chain disruptions	Provides the logging mechanism of disruptions of each order.

In order to realize D1.1 requirements the clearing house supports the following functionality:

1. Log a transaction, add the details to the components internal registry.
2. Find a transaction, searches the registry for a transaction based on the provided details (order id, data provider id, etc.).
3. Find order status, based on recorded transactions defined how far along the order has progressed (the information is relevant to one of the groups in section 3.2.1).

3.2.3 Sequence diagrams

Figure 8 depicts how logging a transaction is realized. First the clearing house API intercepts a request to log a transaction. This request is forwarded to the clearing house core. The core is responsible for generating a new order with the order management component if this transaction represents the creation of a new order, and proceeds to log it in the applications local registry. Since the sequence diagrams for finding a transaction and finding the order status are similar in nature they are omitted.

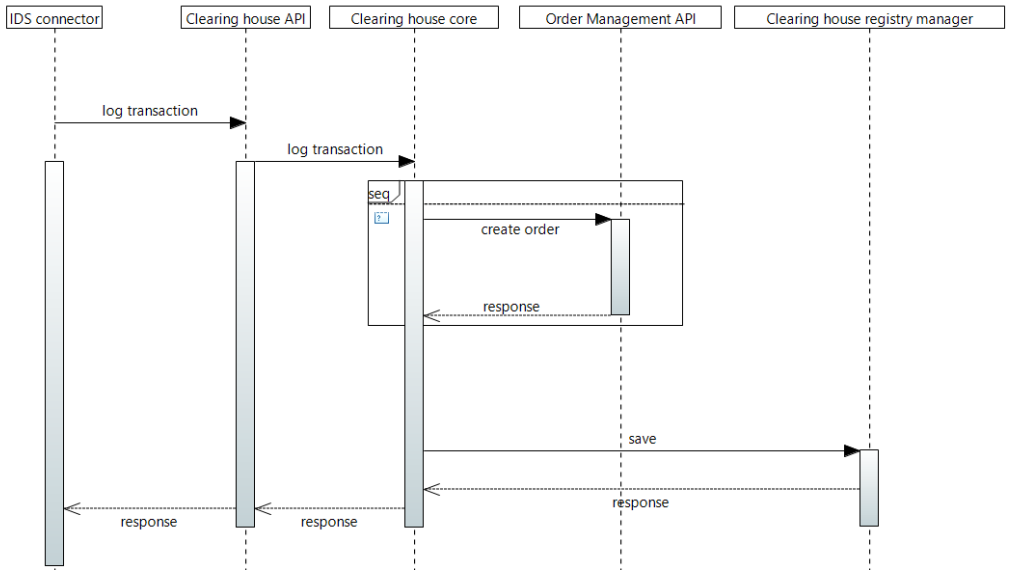

Figure 8 Clearing house transaction logging

3.2.4 API overview

The HTTP POST method is used for all requests to insert or update information, while the HTTP GET method is used to retrieve information. Request and response bodies are encoded using JSON. Table 4 provides an overview:

Table 4 IDS Clearing house API

Endpoint	Input	Output	Description
POST /transaction/log	Order id, transaction category, connector id.	Success or Failure	Logs a transaction.
GET /transaction/find	Order id, or connector id	List of transactions in JSON.	Find transactions.

3.3 IDS Broker

3.3.1 Summary description

The IDS Broker consists of an IDS Connector (as detailed in 3.1), and facilitates the identification of data providers (in the case of MARKET4.0 data providers are considered the platform's supplier users).

3.3.2 Functionality description

The typical functionality to the broker consists of following services:

1. Data provider registration, which adds a new connector to its internal registry.
2. Publication of meta-data associated with a data provider (via their connector).

3. Maintenance, updating the meta-data in the registry and removing connector that have gone offline.
4. Query, allows the data consumer to search for data providers based on a set of criteria.

These functionalities support the following D1.1 requirements:

Table 5 D1.1 requirements supported by the IDS Broker

ID	Short description	Comment
G-07	Allows a supplier to register a connector.	-
HT-1	The system shall enable the actors to communicate order related data.	Supports the workflow for exchanging data by doesn't directly implement it.

3.3.3 Sequence diagrams

Figure 9 illustrates the main collaboration between internal components of the broker to realize the registration functionality.

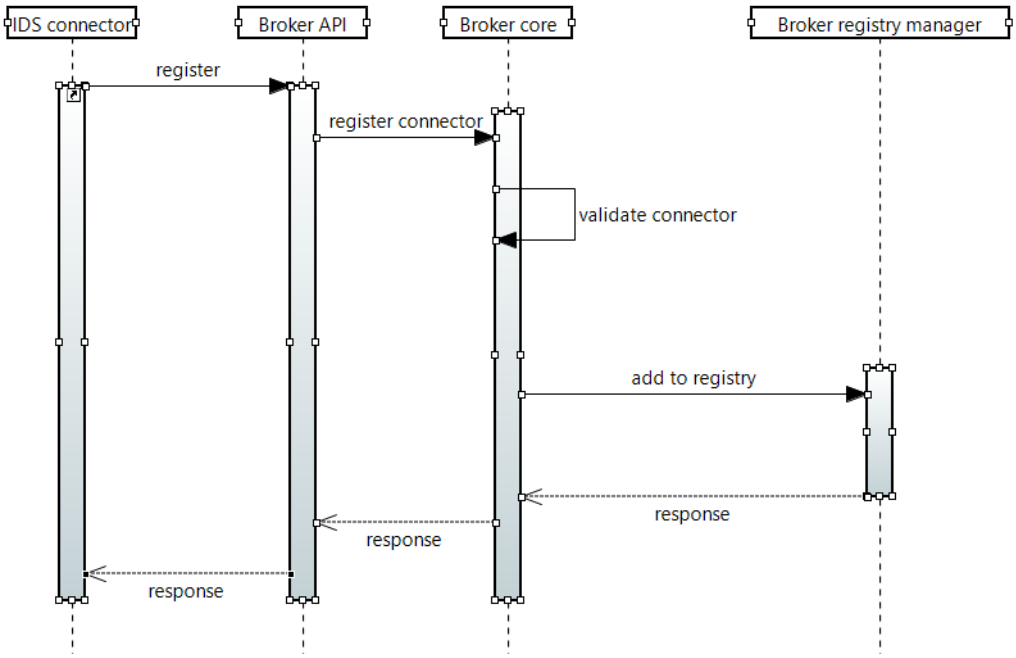

Figure 9 Connector registration sequence diagram

The IDS connector maker a request to the Broker API to register its details. The API forwards the request to the Broker core which in turn validates the associated data and adds them to the internal registry by dispatching a request to the Broker registry manager. As the publication and query functionalities are similar in nature the corresponding diagrams are omitted.

Figure 10 illustrates the sequence diagram representing the maintenance functionality. Periodically the Broker core will issue a request to a connector to access its description. The communication is realized via the Broker API. As soon as the data are available the core checks

the information against is internal registry and updates it accordingly. If the connector is inaccessible the broker will make two additional attempts to access it. If these fail as well the broker will remove the connector from its registry. After a connector has been remover it will have to register again following the workflow in Figure 9.

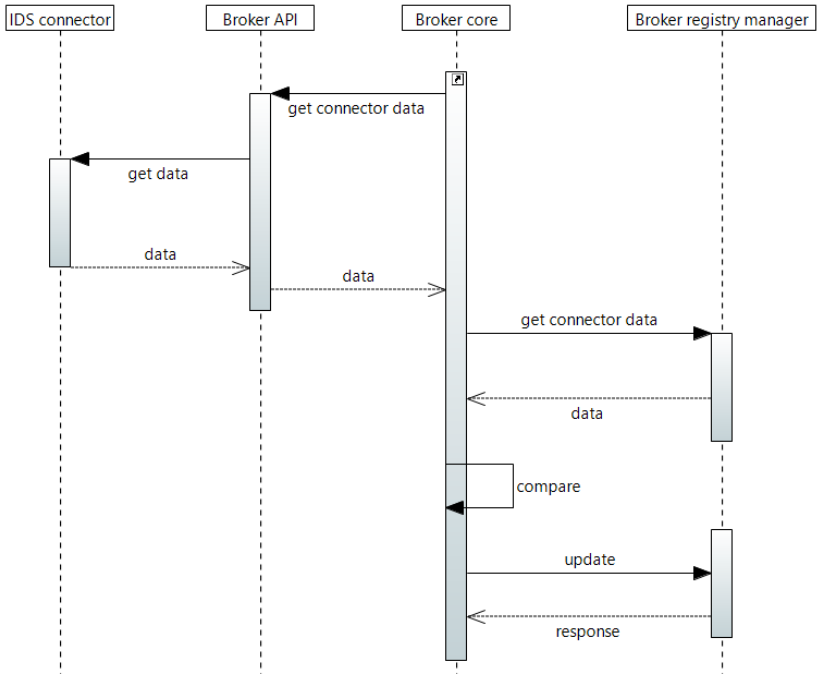

Figure 10 Updating connector details sequence diagram

3.3.4 API overview

The HTTP POST method is used for all requests to insert or update information, while the HTTP GET method is used to retrieve information. Request and response bodies are encoded using JSON. Table 6 provides an overview:

Table 6 IDS Broker API

Endpoint	Input	Output	Description
POST /register/service	ServiceDescritpion element based on IDS documentation.	Success or Failure	Registers an IDS connector with the Broker.
GET /find/service	Service id	ServiceDescritpion element based on IDS documentation in JSON.	Find a connector by id.

GET /query/service	Search criteria e.g. domain, type of data.	ServiceDescritpion element based on IDS documentation in JSON.	Find a connector using a set of criteria.

3.4 IDS identity provider

3.4.1 Summary description

The IDS identity provider addresses the need to be able to make access control related decisions that are based on reliable identities and properties of participants, a concept for Identity and Access Management (IAM) is mandatory.

The following aspects are central for the concept:

- identification (i.e., claiming an identity),
- authentication (i.e., verifying an identity), and
- authorization (i.e., making access decisions based on an identity).

The Certificate Authority issues certificates for all entities. These certificates are used for authentication and encryption between Connectors. An identity may have several attributes, which are linked to that identity. A Dynamic Attribute Provisioning Service is used to provide dynamic, up-to-date attribute information about Participants and Connectors. It should be noted that the IDS identity provider applies only to entities participating in MARKET4.0 private data exchanges.

3.4.2 Functionality description

The typical functionality to the identity provider involves:

1. Issuing access tokens to connectors
2. Verify a certificate

These functionalities support the following D1.1 requirements:

Table 7 D1.1 requirements supported by the IDS clearing house

ID	Short description	Comment
HT-1	The system shall enable the actors to communicate order related data.	Supports the workflow for exchanging data by doesn't directly implement it.

3.4.3 Workflow

The following workflow for accessing a resource (i.e., a supplier's data) using dynamic attributes and access tokens is defined:

1. A Dynamic Attribute Token is requested from the Dynamic Attribute Provisioning Service, presenting the Connector's X.509 certificate.

2. Before accessing a resource, a TLS tunnel is established using the same X.509 certificate. Again, depending on the policy specified, the certificate can be verified at the Certificate Authority.

3. If using several Access Tokens, a token request is performed at a separate Authorization Service in the domain of a use case operator or the domain of the Connector's (or, more specifically, resource's) owner. This step is optional.

4. The resource is requested by handing in either the Dynamic Attribute Token or the Access Token.

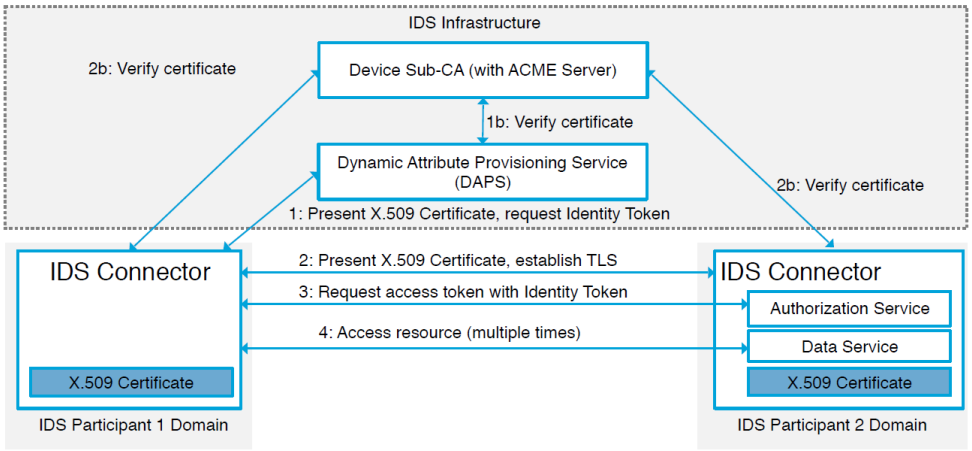

Figure 11 IDS resource access workflow[9]

3.4.4 API overview

GET method is used to retrieve information. Request and response bodies are encoded using JSON. Table 8 provides an overview:

Table 8 IDS identity provider API

Endpoint	Input	Output	Description
GET /certificate/verify	X.509 certificate details	Success or failure.	Verify a certificate.
GET /token/request	ServiceDescritpion element based on IDS documentation.	Token in JSON.	Request a token.

[9] https://www.internationaldataspaces.org/publications/reference-architecture-model-3-0/

3.5 Dynamic Supplier Network Configuration & Management

3.5.1 Summary description

This component is responsible for providing the functionality to support the creation and management of Dynamic Supply Networks (DMN) and the automatic matching of potential suppliers based on the descriptions of data shared via the IDS connectors.

3.5.2 Functionality description

Table 9 presents the requirements of D1.1 supported by this component.

Table 9 D1.1 requirements supported by Dynamic Supplier Network Configuration & Management

ID	Short description	Comment
HT-2	Nesting Ordering processes	Partially addressed. In order to facilitate nesting of orders the platform will provide matchmaking functionality for suppliers. The next step can be negotiated in a peer-to-peer basis.
PL-2	Nested tender bid	HT-2 comment applies in this case as well.

In order to realize these requirements the following functionalities form IMAGINE project will be integrated in the platform:

1. The Production Requirements Composer which allows for defining the required skills in the Dynamic Supplier Network in order to realize a new product.
2. The supplier selection algorithm that matches the requirements to the potential suppliers.

The detailed specification of these features is provided in D3.1.2 “Detailed Design of IMAGINE Platform Version 2” and Deliverable D3.3 “IMAGINE Enlarged Adapters” of IMAGINE project and is therefore not included here.

3.6 Internal Data store

The internal data store offers a solution for persistent storage of marketplace data coming for the various components of the architecture. As of the time of the writing of this document the data stored and exchanged between components can be accommodated in a RBDMS. Examples include user profiles, orders, catalogue information (including binary files).

3.7 Application Management

3.7.1 Summary description

Application Management is responsible for providing the application management features of MARKET4.0 platform for all involved users. This component is only responsible for managing the apps (only IDS-enabled) coming from the app provider role. Other software offerings (e.g. big data applications) offered via the MARKET4.0 catalogue are managed from the E-commerce component.

3.7.2 Functionality description

Table 10 presents the requirements of D1.1 supported by this component.

Table 10 D1.1 requirements supported by application management

ID	Short description
G-10	App store for providing access to MARKET4.0 apps.
PL-3	Use of apps to support buyer-seller interaction

In order to realize these requirements the following functionalities:

1. **Register app** for uploading a new app to the platform. This app will be available to the customers only after it has been approved from the administrators.
2. **Delete app** for removing an uploaded app from the platform.
3. **Update app** for updating the app version or the app description.
4. **Validate app** for marking the app as approved and thus making it visible to the customers. If the app is not approved, it can is removed.
5. **Download app** for accessing the app binary.
6. **List apps** for providing a list of all uploaded apps and filtering them using predefined criteria.
7. **Search apps** for providing search mechanism using keywords.

3.7.3 Sequence diagrams

Figure 12 illustrates the complete scenario of registering an app with covers the Register App functionality and the Validate app functionality.

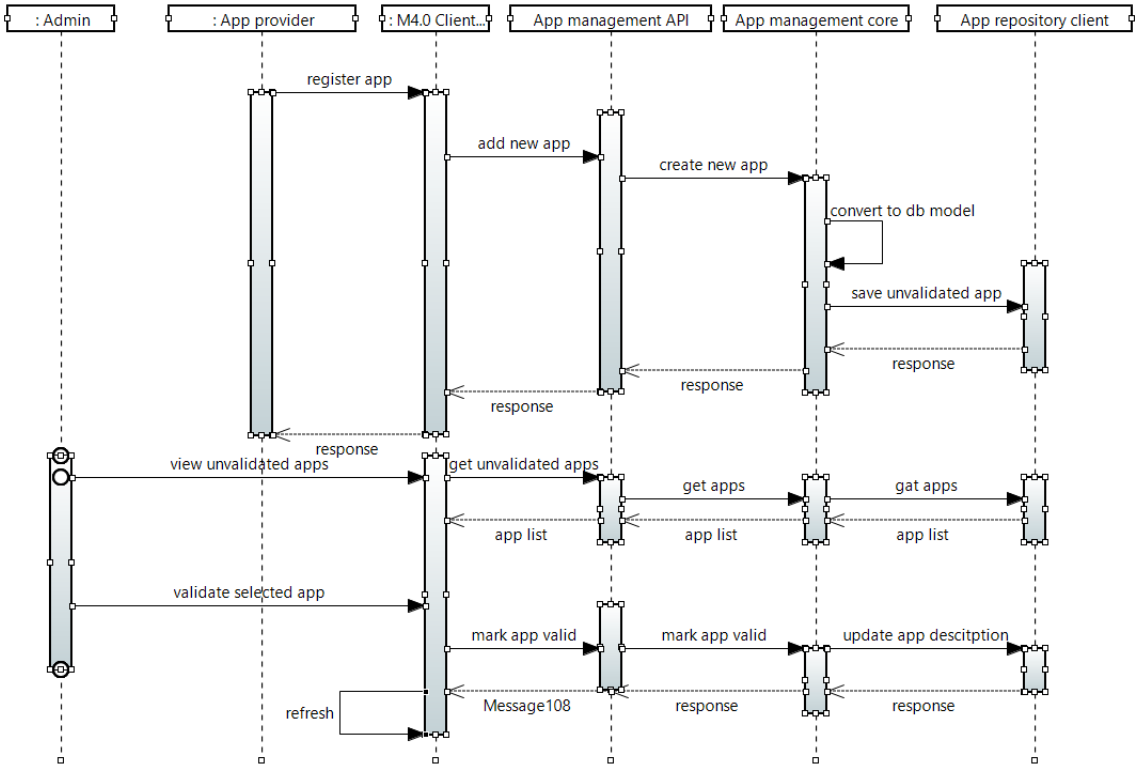

Figure 12 Register app sequence diagram

The interaction starts from the app provider registering a new app via the M4.0 client. This request is propagated via the App management API to the app management core where the details provided are converted to an instance of the database model and the persisted to the app repository via the App repository client. The next phase is initiated from the admin by requesting a list for apps pending validation. This request is propagated to the App management API and then all the way to the App repository client for retrieving the list. As soon as the list is retrieved it is returned to the M4.0 client for visualization. Then the admin selects an app to change its status as validated (the validation process may include local testing of the app and is omitted here) and saves it. This action is propagated all the way to the App repository client which changes the status of the app to the app repository. The response is returned to the M4.0 client which triggers a refresh.

As similar interaction between the aforementioned object instances facilitate all the interactions for deleting, updating, searching and downloading an app further sequence diagrams are omitted.

3.7.4 API overview

The App Management API is a RESTful API. The HTTP POST method is used for all requests to insert or update information, while the HTTP GET method is used to retrieve information. The

DELETE method is used to delete entities. Request and response bodies are encoded using JSON. Table 11 provides an overview.

Table 11 Application Management API

Endpoint	Input	Output	Description
POST /app/register	App details in JSON and binary.	Success or failure.	Register an app.
DELETE /app/delete	App id.	Success or failure.	Delete an app.
POST /app/update	App details in JSON and/or binary.	Success or failure.	Update app.
POST /app/validate	App id.	Success or failure.	Change status to validated.
GET /app/download	App id.	App binary file	Download the app to the local computer.
Get /app/list	Optional browse criteria (e.g. domain, functionality)	List of apps in JSON.	List the apps using criteria.
Get /app/search	Keyword	List of apps in JSON.	Search apps using a keyword.

3.8 User Management

3.8.1 Summary description

The user management components deal with user creation/deletion/update, controlling access to data (login/logout), restricting access to legitimate users (data access rights), and handling data encryption when required (e.g. passwords).

In order to enable data access rights, the users have to be divided in groups. One user may belong to one or more groups. For the purposes of MARKET4.0 the following groups are identified:

1. Administrator
2. Equipment Supplier
3. App provider
4. Customer

The following access rights are applicable:

Table 12 Access right per user group.

Group	Access Rights
Administrator	Access to all functionalities.
Equipment Supplier	Register, update profile, view/download apps, view catalogue, manage catalogue (own listing), register connector, and delete account.

App provider	Register, update profile, view/download apps, view catalogue, manage apps (own listing), and delete account.
Customer	Register, update profile, view/download apps, view catalogue, and delete account.

3.8.2 Functionality description

Table 13 presents the requirements of D1.1 supported by this component.

Table 13 D1.1 requirements supported by user management

ID	Short description
G-01	Allows the user to create an account that enables access to the platform functionality.
G-04	Allows the user to modify their profile information.
G-05	Allows the user to delete their account.
G-07	Allows a supplier to register a connector.

In order to realize these requirements the following functionalities:

1. **Create user** is responsible for creating a user entity and storing it in the internal data store of the platform. It is supported by the data encryption service for encrypting the passwords.
2. **Update user** is responsible for updating the internal data store records with new information on existing users. An authenticated user is a prerequisite.
3. **Retrieve user details** retrieves the user information from the internal data store for a given user id.
4. **Delete user** removes a user form the internal data store. An authenticated user is a prerequisite. If the supplied user id is not found in the internal data store an error is produced.
5. **Synchronization with IDS** is responsible for associating IDS connectors with users and synchronizing the identity records of the two systems.

3.8.3 Sequence diagrams

In order to provide an overview of the entities that will be developed in order to realize the functionalities in 3.8.2 a sequence diagram is illustrated in Figure 13 for the functionality entitled “Create user”. As all functionalities are similar in nature other sequence diagrams are omitted.

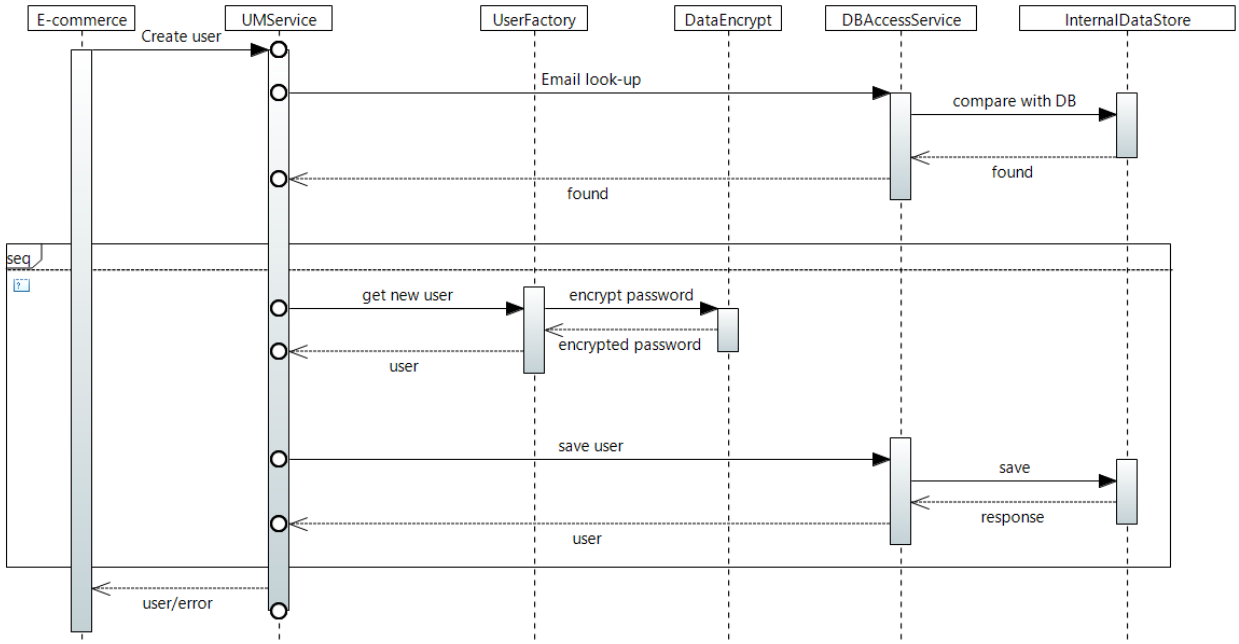

Figure 13 Create user sequence diagram

Figure 13 presents the user creation sequence diagram. Interaction is initiated by the E-Commerce service request for the creation of a new user. The UM (User Management) Service intercepts the request and checks if the user email is already in the database. If it isn't it requests a new user object to be created from the UserFactory. The UserFactory copies the data associated with the request into a new user object and encrypts the password using the DataEncrypt service. The user object is then returned to the UMService which initiates the process to save the user to the Internal Data Store. Finally, the persisted user is returned to the E-Commerce service. If the user email already exists in the database an error is returned. The update user use case is similar and therefore it is omitted.

3.8.4 API overview

The User Management API is a RESTful API. The HTTP POST method is used for all requests to insert or update information, while the HTTP GET method is used to retrieve information. The DELETE method is used to delete entities. Request and response bodies are encoded using JSON. Table 14 provides an overview.

Table 14 IDS identity provider API

Endpoint	Input	Output	Description
POST /user/create	User details in JSON	Success or failure.	Create a user.
POST /user/update	User id and details in JSON	Success or failure.	Update a user.
GET /user/find	User id	User details in JSON	Find a user by id.
DELETE /user/delete	User id	Success or failure.	Delete a user.

POST /user/connector	IDS connector id, user id	Success or failure.	Associate a user and a connector.
DELETE /user/connector	IDS connector id, user id	Success or failure.	Remove a connector from a user.

3.9 Order management

3.9.1 Summary description

Order management components facilitates the initialization of orders for the customers and provides the order status. As the greatest part of the order process is handled on peer-to-peer basis using IDS applications (covered in D1.3) the functionality of this component is only supplemental and address the part of order management that has to be realized by the MARKET4.0 platform.

3.9.2 Functionalities

The MARKET4.0 platform in terms of order management has to address the requirements in Table 15.

Table 15 D1.1 requirements supported by order management

ID	Short description
HT-1	Ordering items. The system shall be able to support the ordering process of items by enabling the actors to communicate order information
HT-3	Order status information. The system shall support the live tracking of order statuses over time for nested orders.
PL - 1	The system is able to handle tender bids.

These requirements correspond to the following functionalities:

1. **Identify an app** to realize peer-to-peer communication for an order.
2. **List past orders** of each user.
3. **View status** of an order.
4. **Create order,** which adds an order in the platform internal data store.

It should be noted that the listing of past orders and viewing the status of an order relies on the transaction data recorded by the IDS clearing house.

3.9.3 Sequence diagrams

Figure 14 illustrates the sequence diagram for identifying an app. This functionality boils down to a simple query in the app repository via an appropriate client. Creating an order follows a similar approach.

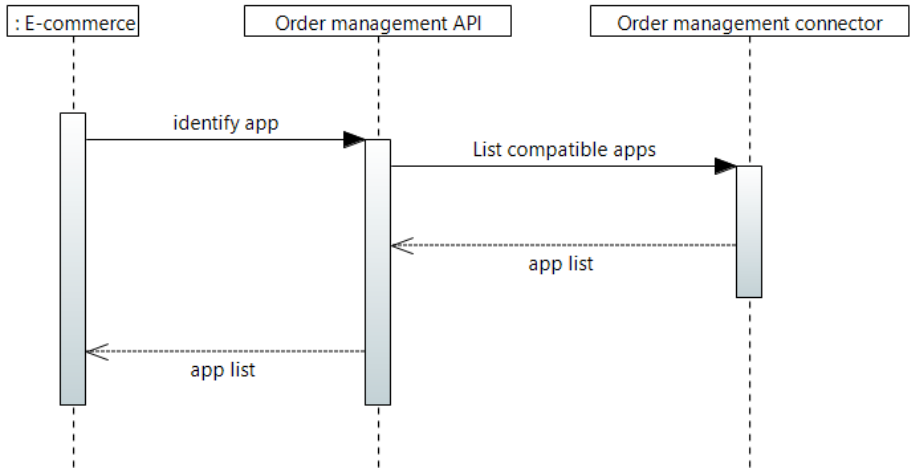

Figure 14 Identify app for starting an order

Figure 15 illustrates the sequence diagram for getting the order status. First a list of orders is requested to the Order management API and in turn it's forwarded to the Order management core. The core requests a transaction list for the Order management connector which is responsible for communication with the IDS clearing house. As soon as the list is retrieved, it is processed and returned. Next a request for the status of an order is made. Similarly the transactions are retrieved processed and the status is returned.

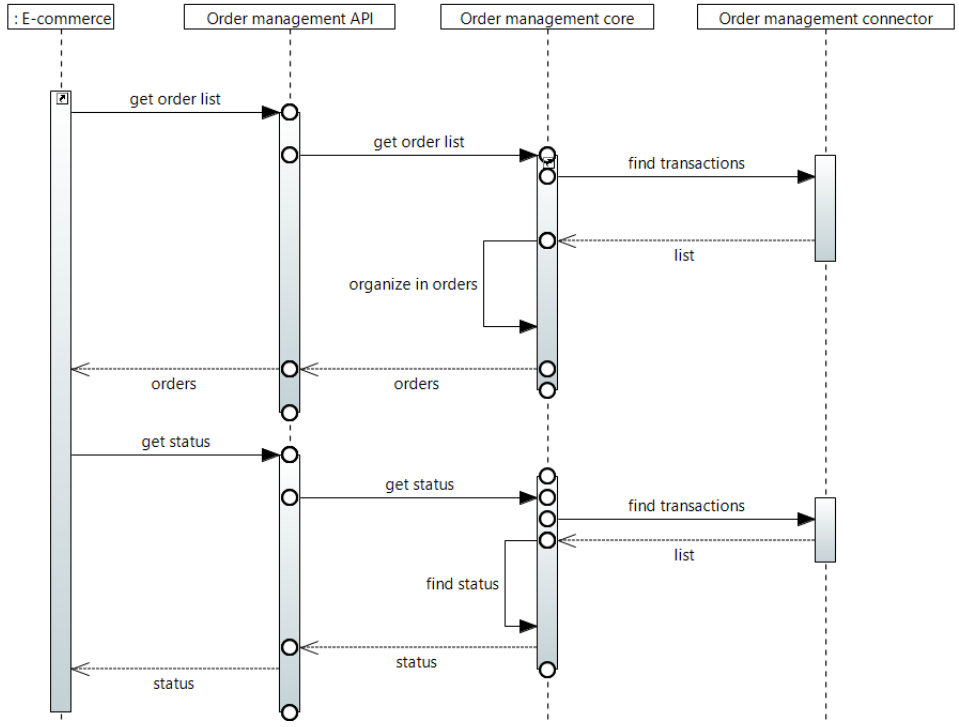

Figure 15 Order status sequence diagram

3.9.4 API overview

The order management API uses the HTTP GET method to retrieve information and HTTP POST to create new orders. Request and response bodies are encoded using JSON. Table 16 provides an overview.

Table 16 Order management API

Endpoint	Input	Output	Description
GET /order/apps	Offering id.	App list in JSON.	Find app for ordering an offering listed in the catalogue.
GET /order/list	User id	Order list in JSON.	Get user's orders.
GET /order/status	Order id	Order status in JSON.	Get order status.
POST /order/create	Transaction id.	Order id.	Create a new order.

3.10 Payment/Billing

The functionality of this app is to facilitate buying apps from app providers. For this reason this component provides a payment gateway. A payment gateway is a merchant service provided by an e-commerce application that authorizes credit card or direct payments processing. There are several existing payment gateways such as PayPal, Authorize.Net, SecurePay.com, and so on. Therefore, one of the existing solutions will be selected and integrated with the platform.

As this is a part of the MARKET4.0 app-store it facilitates requirement G-10.

3.11 E-Commerce

3.11.1 Summary description

This component provides the functionalities for product search (find products effectively) and cataloguing, which provides a consistent view of items offered by the suppliers. Moreover, it enables the creation of shopping carts, provides payment information, and the conversion of shopping cart to an order. Order capture also allows orders started and monitored. Finally, this component will provide the services so that suppliers are able to manage (add/remove/update) their offerings.

3.11.2 Functionalities

Table 17 presents the relevant requirements from D1.1.

Table 17 D1.1 requirements supported by e-commerce

ID	Short description
G-06	The platform will allow the user (with role supplier) to manage their production equipment, service or application offerings.
G-08	Search the catalogue for production equipment, services and apps using keywords and filters.

G-09	Browsing and navigating through the MARKET4.0 on-line catalogue
G-11	Anonymized feedback service for rating suppliers

These requirements correspond to the following functionalities:

1. **Add a new offering**, which add elements (products).
2. **Delete an offering**, which permanently removes them.
3. **View the list of offerings**, lists the offerings of supplier.
4. **Edit an offering**, changes the details of an offering.
5. **View an offerings**, lists the offerings.
6. **Search,** keyword-based search (supports all users).
7. **Browse catalogue,** lists all offerings but can be also filtered to provide a more specific results.
8. **Feedback,** supports customer comments for each supplier. The comments are not publicly tied to the customer.
9. **Shopping cart,** add/removes offerings to shopping cart
10. **Order creation**, by delegating to the order management.

It should be noted that the shopping cart feature facilitates buying apps from the app-store (Application management). Ordering products of the catalogue is subject to peer-to-peer interaction.

3.11.3 Sequence diagrams

Figure 16 illustrates the sequence diagram for adding an offering. The details are forwarded to the E-Commerce connector for saving in the database. Functionalities 2-5 and 9 have a similar approach.

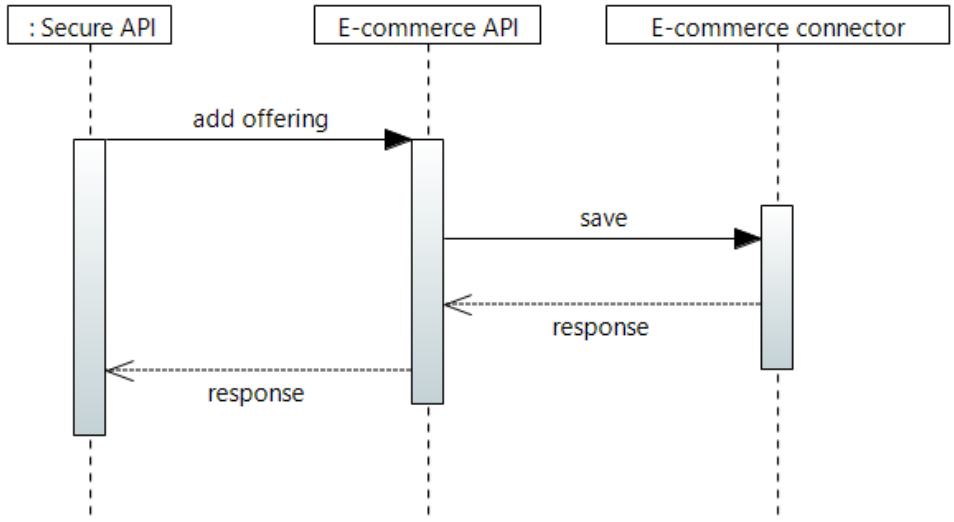

Figure 16 Adding an offering

Figure 17 illustrates the process for anonymized feedback. The request is forwarded to the E-Commerce core where the supplier is loaded form the internal data store from the E-Commerce connector. Next the feedback details are inserted in a new feedback element associated with the given supplier. Finally, the feedback is persisted.

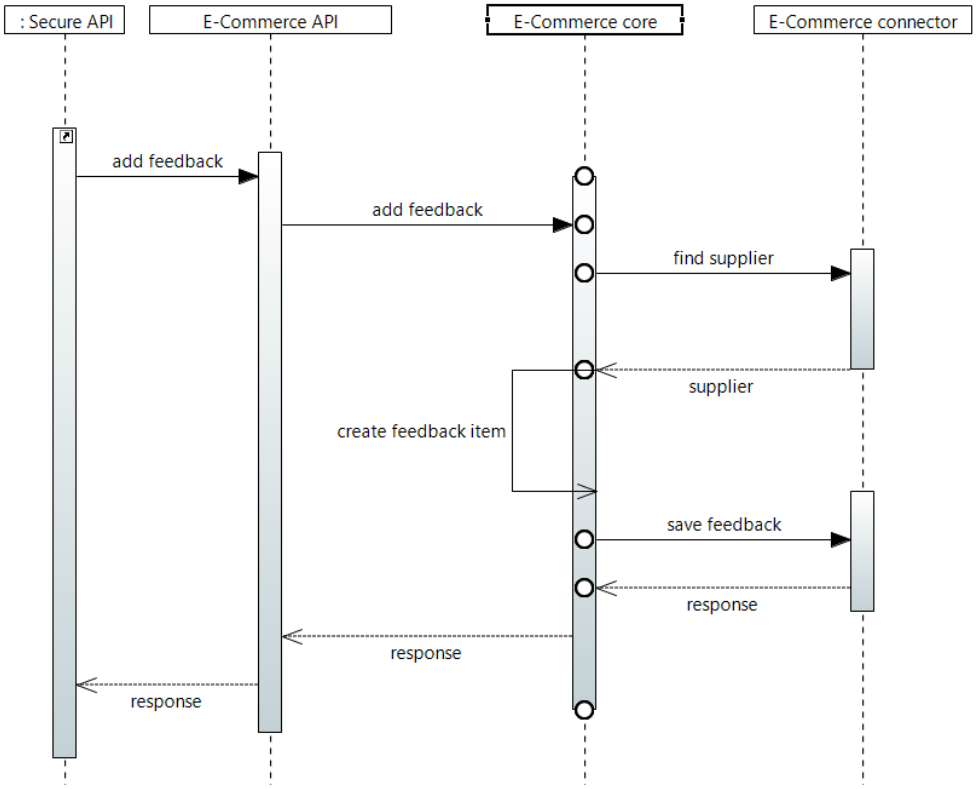

Figure 17 Adding feedback

Figure 18 illustrates the ordering process. When the process is initialized an order is created by forwarding the request Order management API. Next the user is forwarded to the payment interface. The user sends their payment details to the payment gateway, which process the payment. Once the payment is complete the E-Commerce API is notified to update the status of the order. Finally, confirmation is sent to the user.

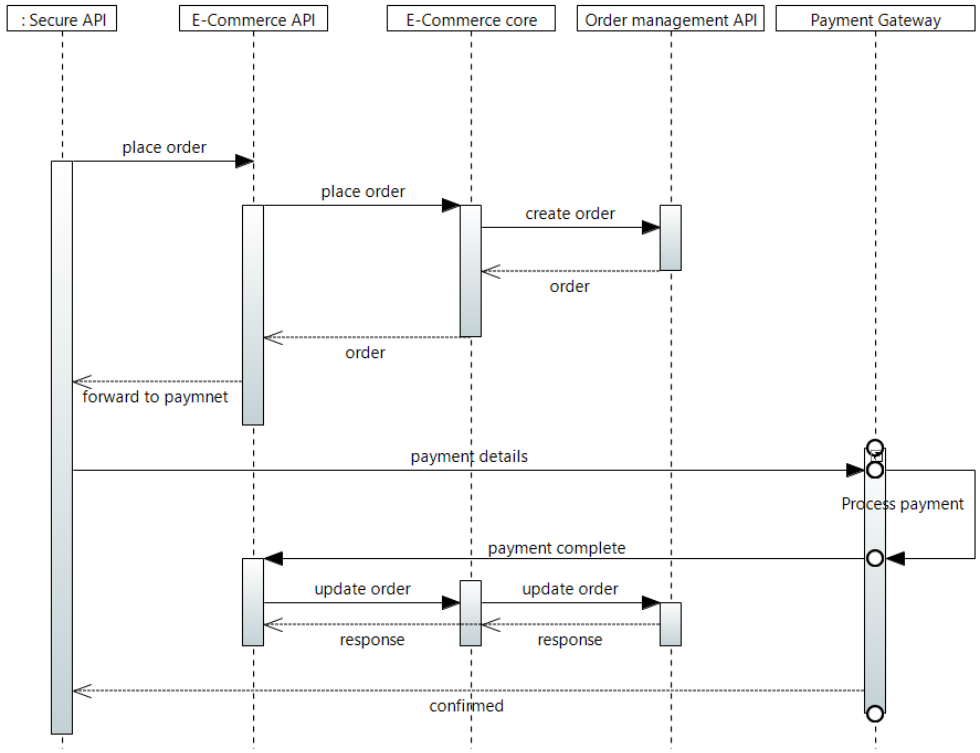

Figure 18 Order placement

3.11.4 API overview

The E-commerce API uses the HTTP GET method to retrieve information and HTTP POST to add or update data in the system. Request and response bodies are encoded using JSON. Table 18 provides an overview.

Table 18 E-commerce API

Endpoint	Input	Output	Description
POST /offering/add	Offering description in JSON.	Success or failure.	Add an offering.
POST /offering/update	Offering description in JSON.	Success or failure.	Update an offering.
DELETE / offering /remove	Offering id	Success or failure.	Delete an offering.
GET /offering/	Offering id	JSON offering data	View an offering
GET /offerings/supplier	Supplier id	JSON offerings data	View all supplier offerings
GET /offerings/browse	Browse criteria in JSON optional	JSON offerings data	View all offerings

GET /offerings/search	Keyword	JSON offerings data	View all offerings related to a keyword
POST /supplier/feedback	Supplier id, rating, description in JSON.	Success or failure.	Add feedback to supplier.
GET /shoppingcart/	-	JSON offerings data	Get items in shopping cart
POST /shoppingcart/remove	Offering id	Success or failure	Remove items from shopping cart
POST /shoppingcart/add	Offering id	Success or failure	Add items from shopping cart
POST /shoppingcart/order	-	Forward to order management.	Make an order.

3.12 Secure API

The Secure API acts as an orchestrator of all the APIs offered by E-commerce, User Management, Application Management, and Dynamic Supplier Network Configuration & Management components. Furthermore, it makes sure that authorization and authentication mechanisms of the security layer are applied for all incoming user requests.

3.13 M4.0 Client

3.13.1 Summary description

The M4.0 Client is a website facilitating user interaction with the platform. As such it provides a series of User Interfaces that deliver information to its users and facilitate data input. The M4.0 Client has to realize the use cases presented in D1.1 section 8. For this purpose, the client will have to communicate with the Secure API which orchestrates the APIs exposed by the rest of the components in section 4.

3.13.2 Design

The client may be implemented using a web application framework. Such frameworks utilize controllers for managing the view. Services are used to request data from the secure API and update the model. The model is bound to the user view (HTML page) that is also updated accordingly. Different controller may be used for the different parts of the HTML page allowing a modular development approach.

The design in Figure 19 is proposed for the main view of the M4.0 Client. It consists of 5 distinct areas.

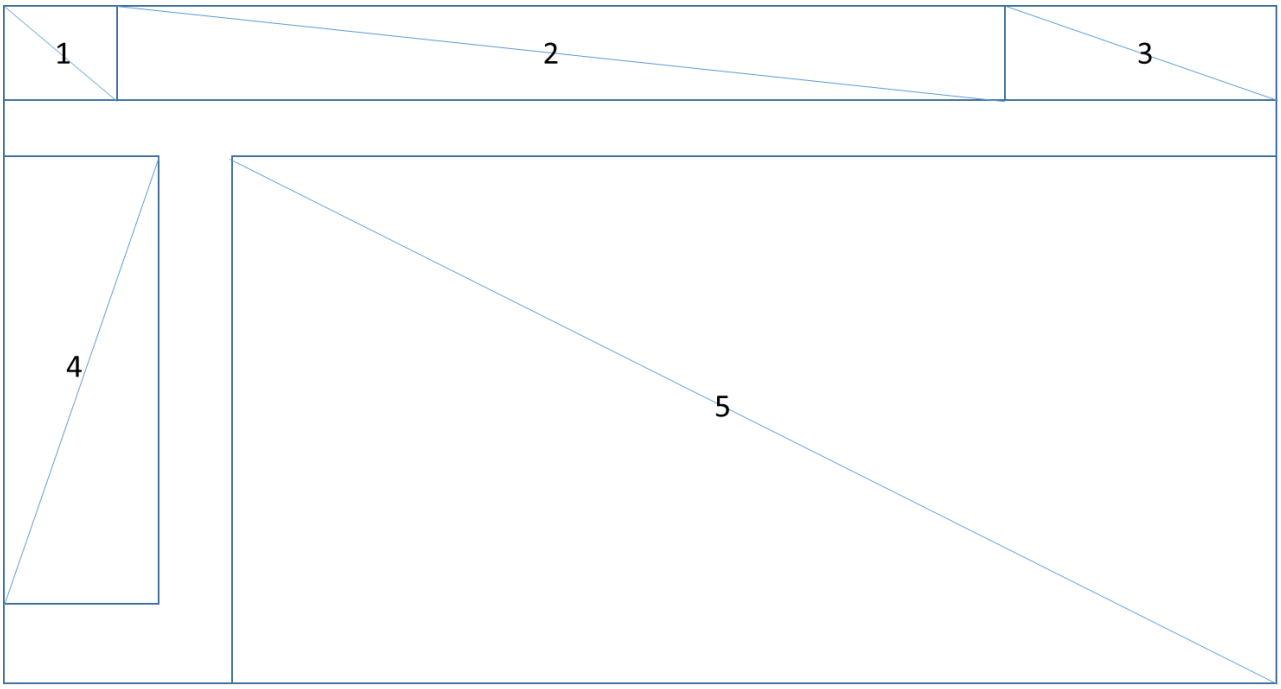

Figure 19 Wireframe design of the M4.0 Client

Area 1 is for logo placement. It may contain other useful information but at the time of the writing no further info is foreseen.

Area 2 holds the main search functionality and the main menu and is will be split into two lines. One for the menu and one for the search fields.

Area 3 holds the user profile functionality.

Area 4 is reserved for a secondary menu when applicable (e.g. when the user is browsing the catalogue or the app store). When the secondary menu is not present the element is removed and area 5 centred.

Area 5 is the main content holder. Here products or apps are visualized based on the menu selections of the user.

Utilizing the web application framework approach different controllers will be developed for controlling the 4 main areas that provide functionality to the user. These controllers will be complemented by a set of services for accessing the secure API.

3.14Security

This cross-cutting element groups together different security aspects. The following mechanisms are supported:

- **Identity and access management**: Identify and monitor who needs access to sensitive data and systems in the platform. It will enable access control mechanisms for users and services. The mechanism in this category will leverage JSON Web Tokens standard.
- **Data encryption**: Secures the data interchange between components to achieve confidentiality and integrity by following standards such as data security standards. The Advanced Encryption Standard (AES) will be used for the purposed of data encryption.
- **Application security**: Lists common attack types for Internet-facing applications. Prevents illegal access into the system through Denial of Service and checks for cross-site scripting such as OWASP to protect the application before deploying into a production environment.
- **Data privacy**: Describes the approach to handle and store sensitive personally identifiable information. It enables customers to retain data sovereignty and residency to adhere to relevant regulatory requirements (for the data within the platform, it doesn't cover IDS).

4 Mapping to specific pilot domain requirements

This section provides a mapping between components and requirements from D1.1. The internal data store, M4.0 Client, and Secure API are omitted as they indirectly support all requirements realized by E-commerce, order management, application management, user management, order management, and payment/billing components.

Table 19 Components vs D1.1 requirements

Component name	Requirement IDs
E-commerce	G-06: The platform will allow the user (with role supplier) to manage their production equipment, service or application offerings. G-08: Search the catalogue for production equipment, services and apps using keywords and filters. G-09: Browsing and navigating through the MARKET4.0 on-line catalogue
Payment/Billing	G-10: App store for providing access to MARKET4.0 apps.
Order Management	HT-1: Ordering items. The system shall be able to support the ordering process of items by enabling the actors to communicate order information HT-3: Order status information. The system shall support the live tracking of order statuses over time for nested orders. PL-1: The system is able to handle tender bids.
User Management	G-01: Allows the user to create an account that enables access to the platform functionality. G-04: Allows the user to modify their profile information. G-05: Allows the user to delete their account. G-07: Allows a supplier to register a connector.
Application Management	G-10: App store for providing access to MARKET4.0 apps. PL-3: Use of apps to support buyer-seller interaction
Dynamic Supplier Network Configuration & Management	HT-2: Nesting Ordering processes PL-2: Nested tender bid
IDS Clearing House	HT-1: The system shall enable the actors to communicate order related data. HT-3: Provides access to accurate and complete information of orders. HT-5: Communication and monitoring of supply chain disruptions
IDS Broker	G-07: Allows a supplier to register a connector. HT-1: The system shall enable the actors to communicate order related data.

IDS Identity provider	HT-1: The system shall enable the actors to communicate order related data.
IDS Connector	G-07: Allows a supplier to register a connector. HT-1: The system shall enable the actors to communicate order related data.
Security	G-02: Allows the user to log in. G-03: Allows the user to log out.

5 Conclusions

This document has presented the Reference Architecture of MARKET4.0 platform. The design is influenced on one hand from the IBM, e-commerce for scalable, secure digital retail apps architecture based on IBM cloud and on the other hand of the requirements coming from the needs of the stakeholder and presented in D1.1. The design uses service-oriented approach to facilitate integration of new components to further extend the original functionality of the platform. The major functionalities of each component have been defined and a first approach of their API is presented. This specification will pave the way for the development of the platform. This architecture design can be further adapted in terms of specific apps to realize the domain specific needs.

As the architecture may evolve as the development process progresses any updates made to the architecture or the specific components APIs will be documented in upcoming deliverables of WP2 and WP3.