Ecosystem for COllaborative Manufacturing PrOceSses – Intra- and Interfactory Integration and AutomaTION
(Grant Agreement No 723145)

D7.9 Lab scale Use Case Deployment with Lessons Learnt II

Date: 2019-01-07

Version 1.0

Published by the COMPOSITION Consortium

Dissemination Level: Public

Co-funded by the European Union's Horizon 2020 Framework Programme for Research and Innovation under Grant Agreement No 723145

Document control page

Document file: D7.9 Lab Scale Use Case Deployment with Lessons Learnt II_V1.0.docx
Document version: 1.0
Document owner: TNI-UCC

Work package: WP7 – Integration of Internal and External Elements
Task: T7.4 – Components Integration
Deliverable type: R

Document status: ☒ Approved by the document owner for internal review
☒ Approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Peter Haigh (TNI-UCC) Matteo Pardi (NXW) Vasiliki Charisi (ATL) Alexandros Nizamis (CERTH), Nikolaos Vakakis (CERTH) Mathias Axling (CNET) Paolo Vergori (ISMB)	2018-12-07	Updates throughout document Updates throughout document CERTH components updates CNET, updates to System diagram Import from D7.8 and analysis, Update to sections 3,4,5, Contribution to chapter 7
0.2	Peter Haigh (TNI-UCC)	2018-12-19	Updated after reviewers' comments. Two items remaining to update. Add Reputation and Trust and add field codes.
0.3	Peter Haigh (TNI-UCC)	2018-12-19	Updated and tidied up Figures 2 and 3. Added reputation and trust details to tables 1 and 3
0.4	Peter Haigh (TNI-UCC), Helene Udsen (IN-JET)	2018-12-20	Updates to comments, fixed codes, added Reputation and trust to section 6 References updated
1.0	Peter Haigh (TNI-UCC)	2019-01-07	Remaining reviewer comments addressed Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Dimitris Gkortzis (ELDIA)	2018-12-12	Approved
Helene Udsen (IN-JET)	2018-12-14	Approved with numerous comments and suggestions. Many issues (consistency, clarity, formatting, language) need addressing

1 Executive Summary

This deliverable outlines the architecture defined for integration activities, a brief overview of the components developed as well as any lessons learnt during the initial integration activities in a lab scale environment. It also includes an integration plan for the inter-factory and intra-factory use cases and a set of recommendations, conclusions and next steps. The first docker server used in the project was deployed by ISMB for lab scale use by the COMPOSITION components. The server was used in the first phase of the project. When real and live data started to be used, the server could not accommodate the project's needs. A new production server was deployed by FIT, with more resources and able to accommodate the data traffic, calls and requests. All COMPOSITION components now use the production server for deployment and dockerisation.

A reference architecture has been successfully developed that is now being used in individual use cases. To date the following components have been developed based on this architecture, where possible leveraging from commercially available software modules and open source widely used protocols.

- Authentication
- Big Data Analytics
- BlockChain Connector
- Commissioning System
- Deep Learning Toolkit
- Intrafactory Interoperability Layer
- Manufacturing Big Data Storage
- Manufacturing Decision Support System
- Market Event Broker
- Matchmaker
- Real Time Multi-Protocol Event Broker
- Reputation and Trust Model
- Requester Agent
- SIEM (Security Information and Event Management)
- Simulation and forecasting tool
- Supplier Agent

The development and dockerisation state of the components have been outlined, along with action plans for use in inter- and intra-factory use cases.

At a high level there are two communication mechanisms in the COMPOSITION system: message-based communication over MQTT or AMQP using the Event Broker[1], and request-response REST HTTP interfaces (D2.4 The COMPOSITION Architecture Specification II).

With a few exceptions[2], the components are loosely coupled and integrated in the system by conforming to a common communication infrastructure and data schemas. There are dependencies on data generated by other components, but for most of the data, the dependency is not direct as this data is distributed through the event broker, so the direct connection is to this component.

[1] Real-time Event Broker, Message Broker

[2] E.g. Big Data Analytics and Deep Learning networks are tightly integrated.

2 Abbreviations and Acronyms

Acronym	Meaning
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
CMMS	Computerized Maintenance Management System
DLT	Deep Learning Toolkit
DSS	Decision Support System
FST	Finite State Machines
HTTP	Hypertext Transfer Protocol
KPI	Key Performance Indicator
MQTT	Message Queuing Telemetry Transport
OSSIM	Open Source Security Information Management
SFT	Simulation and Forecasting Toolkit

3 Introduction

3.1 Purpose, context and scope of this deliverable

This deliverable follows on from D7.8 Lab scale use and deployment with lessons learnt I and describes the technology components and efforts of COMPOSITION partners for lab scale use case deployment. These deployments are critical preparation steps for the technology components and testing their inter-operability and suitability for the various industry use cases. This includes a definition of the architecture devised and an overview of the components developed so far. All of the use cases can be described as either 'inter-factory' and 'intra-factory' and an integration plan is outlined with latest status update at the time of writing. The new production server deployed by FIT accommodates data traffic, calls and requests. The server is used throughout the project for dockerisation and deployment

It also captures lessons learnt, conclusions and recommendations.

3.2 Content and structure of this deliverable

This document is organised as follows. Section 4 describes the architecture and their components at a system level. Section 5 gives a summary of technology components in development and their status. Section 6 describes each technology component in detail. Section 7 maps these components to specific use cases and shows how the JIRA tool is used in the project. Section 8 gives an overview of lessons learnt whilst sections 9 and 10 discusses the intra and inter factory plans. The conclusions and next steps are reviewed in section 11.

4 Technology Component Architecture

The diagram in Figure 1 below describes the COMPOSITION system from a business architecture functional view. The Agents are the intra-factory interface to the inter-factory (Marketplace) responsible for interacting and sharing data with other actors in the Marketplace. Agents and user interfaces (HMI) depend on tools for modelling, simulation and analysis of factory data and processes. Business functionality, e.g. manufacturing analysis, is implemented by components providing generic functionality for complex event processing (CEP) and neural networks (deep learning). These in turn use the communication functionality, the intra-factory interoperability and shop floor connectivity functional packages. The inter-factory packages provide the necessary infrastructure for the agents to find other agents, negotiate and enter agreements. The components are described in section 6.

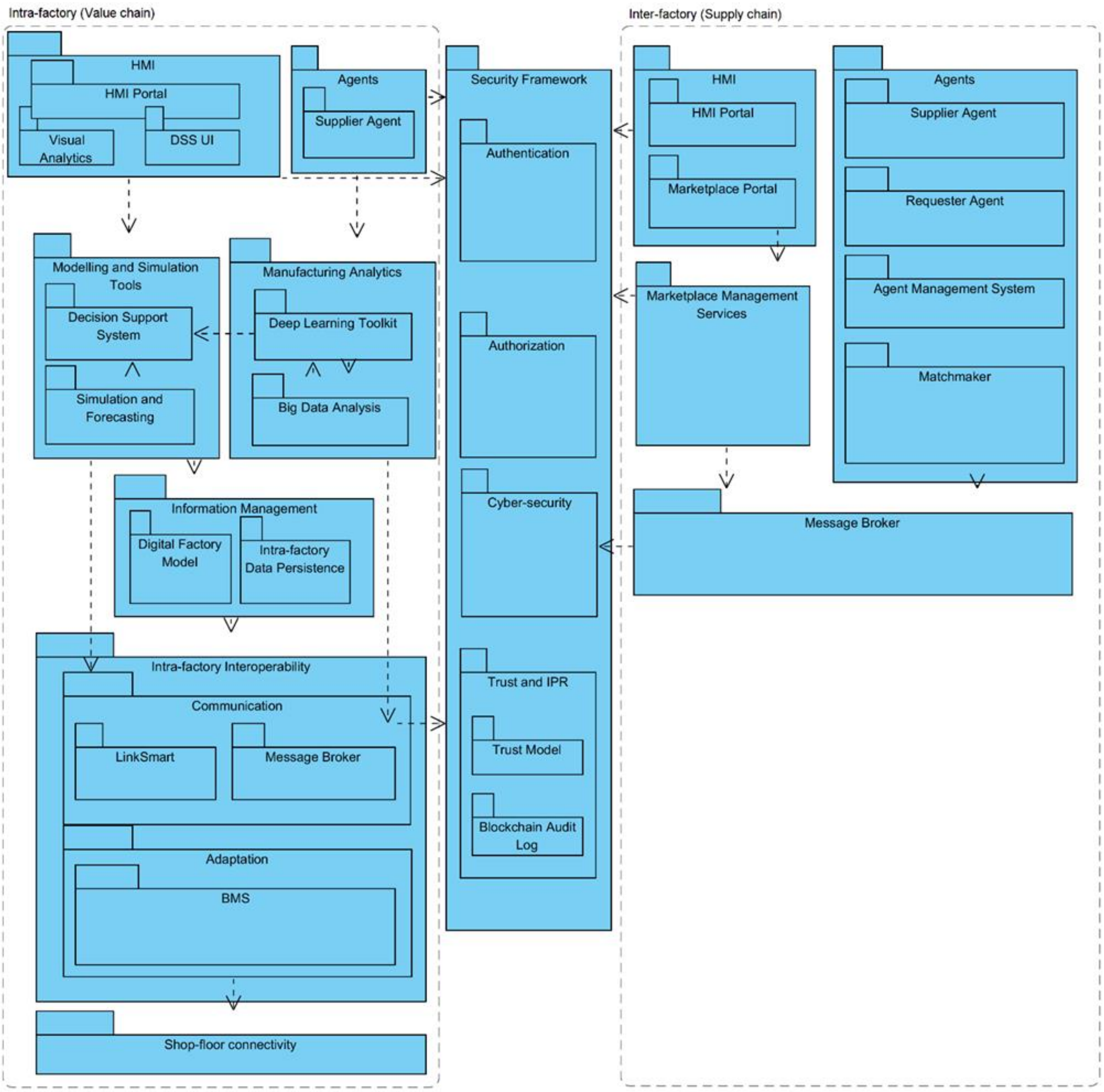

Figure 1: High-level functional view of COMPOSITION architecture.

There are two communication mechanisms in the COMPOSITION system: message-based communication over MQTT or AMQP using the Event Broker[3], and request-response REST HTTP interfaces (D2.4 The COMPOSITION Architecture Specification II). With a few exceptions[4], the components are loosely coupled and integrated in the system by conforming to a common communication infrastructure and data schemas. There are dependencies on data generated by other components, but for most of the data, the dependency is not direct as this data is distributed through the event broker, so the direct connection is to this component.

This architecture is then used to draw out more detailed flow path for specific use cases. Here are 2 examples:

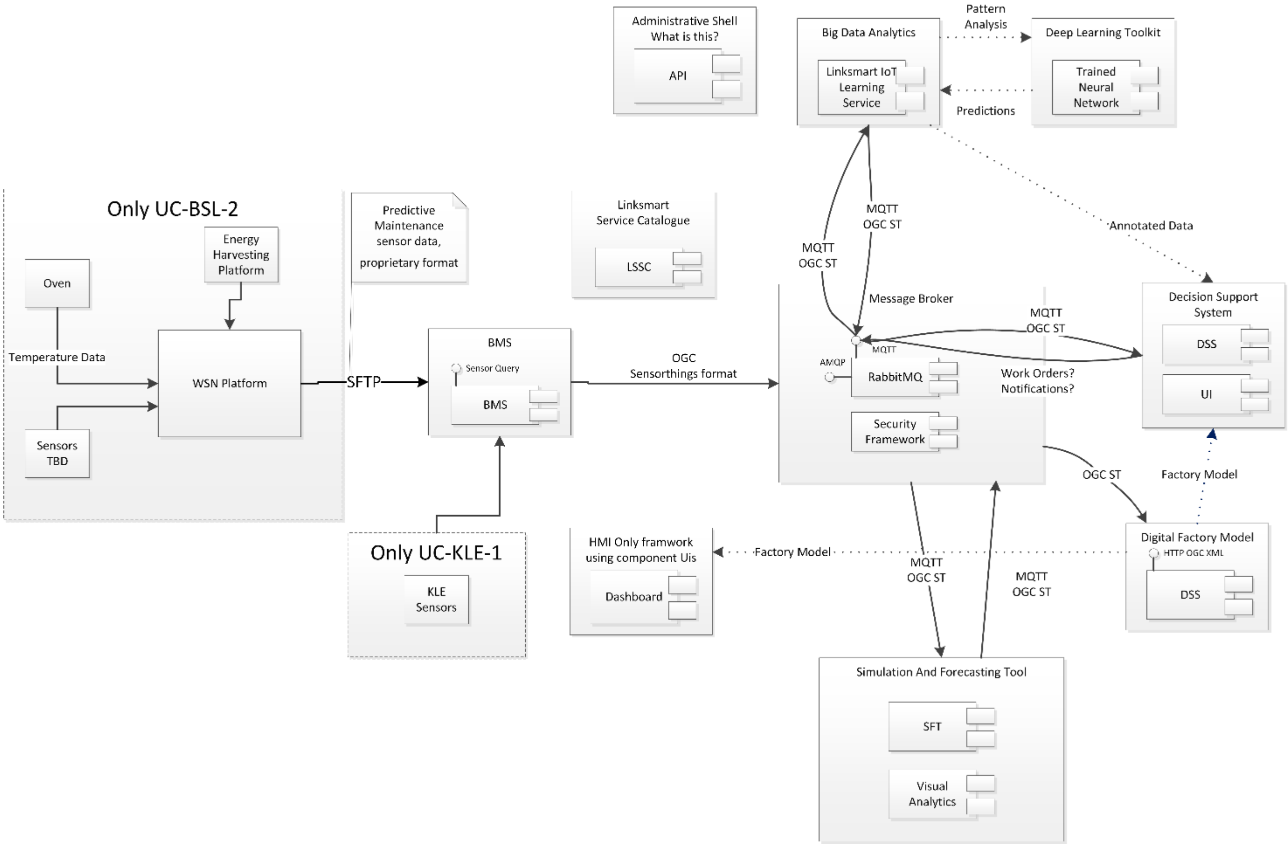

Figure 2: UC-BSL-2 and UC-KLE-1 use case flow path example

[3] Real-time Event Broker, Message Broker
[4] E.g. Big Data Analytics and Deep Learning networks are tightly integrated.

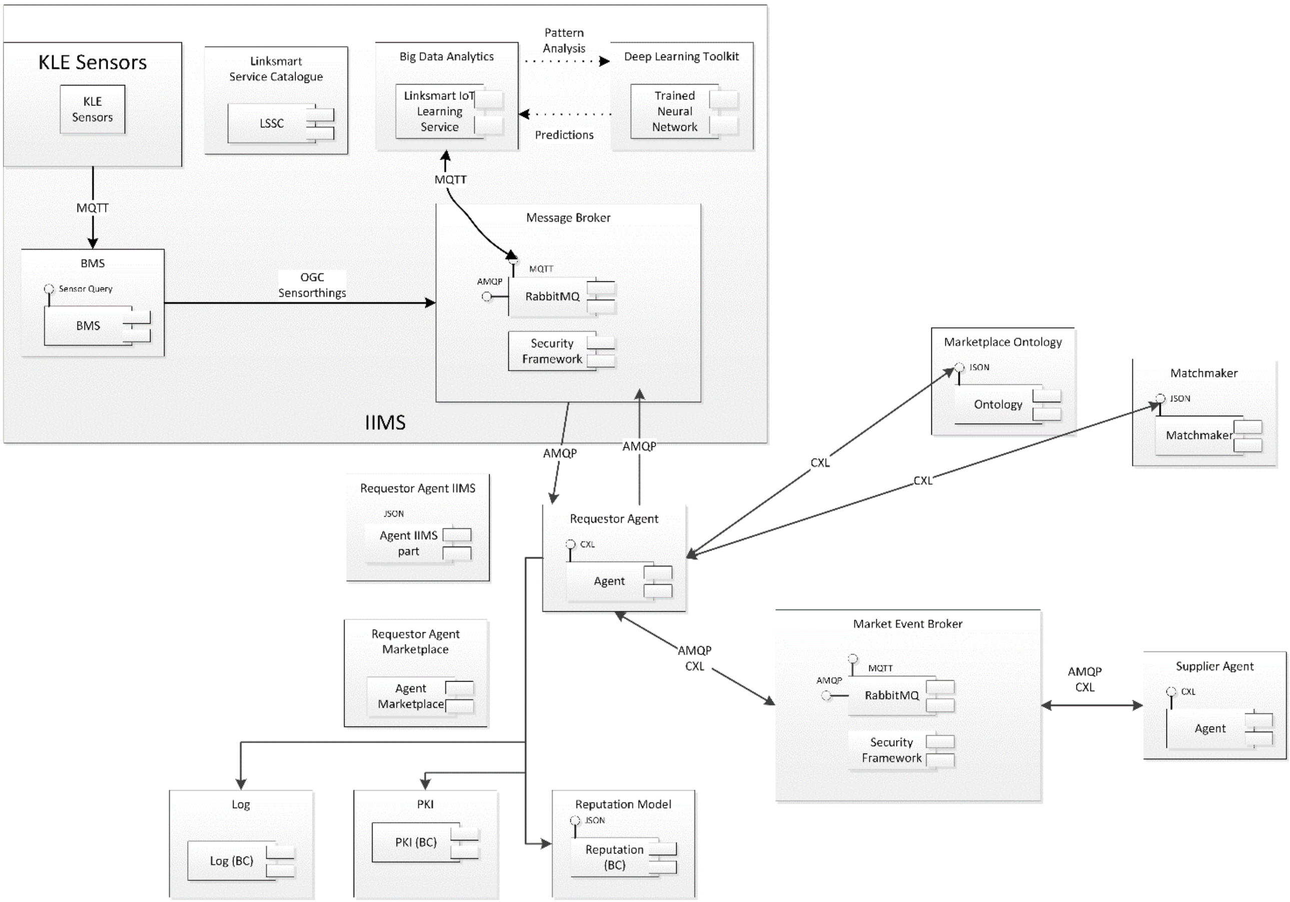

Figure 3: UC-KLE-4 use case flow path examples

5 Summary of Technology Components in Development

The following table lists the technology components developed in COMPOSITION, Further details can be found in section 6.

Technology component	Owner	Dockerised?
Access Control	ATOS	No, but underway
Agent Marketplace	ISMB	No
Authentication	ATOS	Yes
Big Data Analytics	FIT	Yes
BlockChain Connector	CNET	Yes
Commissioning System	CNET	No
Deep Learning Toolkit	ISMB	Very first draft version dockerised
Intrafactory Interoperability Layer	ISMB	Not a component itself, but the baseline of sub-components for inter-operability is dockerised
Manufacturing Big Data Storage	NXW	No
Manufacturing Decision Support System	ATL	Underway
Market Event Broker	CNET	Yes
MatchMaker	CERTH	Yes
Real Time Multi-Protocol Event Broker	CNET	Yes
Requester Agent	ISMB	No
Reputation and Trust Model	ATOS	No
SIEM	ATOS	No
Simulation and Forecasting Tool	CERTH	Yes
Supplier Agent	ISMB	No

Table 1: Technology components developed

6 Detailed Description of Technology Components

The following is a detailed description of the technology components developed, each marked with the lead COMPOSITION partner.

6.1 Access Control [ATOS]

Access Control[5] is provided by EPICA, an Authorization component based on XACML v3.0[6] that provides an attribute-based access control mechanism. It provides the means to define the security policies used to protect resources, and any request to access a protected resource will first be evaluated against these policies where after the evaluation result will be enforced depending on the outcome.

The following Figure 4 presents an overview on the architecture of EPICA

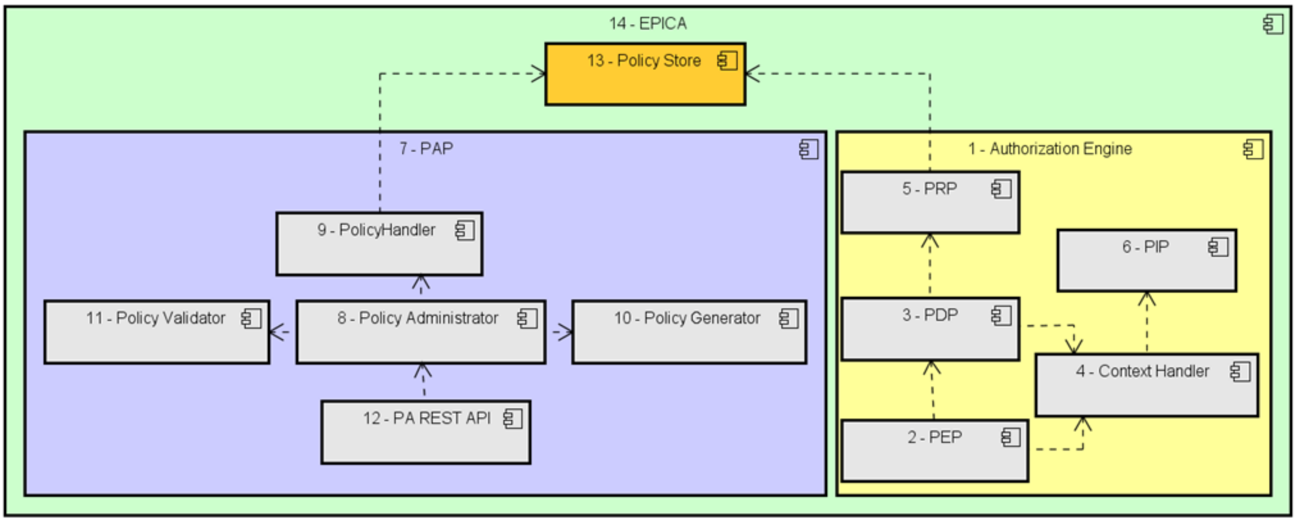

Figure 4: EPICA Architecture

To evaluate the access to a resource EPICA needs a token, with the user information and the resource to be accessed. The token, in this case, is the one obtained from the COMPOSITION Authentication component when a user is authenticated.

The following Figure 5 offers an overview of the possible interactions between the Access Control component and other components that need authorization mechanisms within COMPOSITION.

[5] D4.2 Design of Security Framework II
[6] https://www.oasis-open.org/committees/xacml/

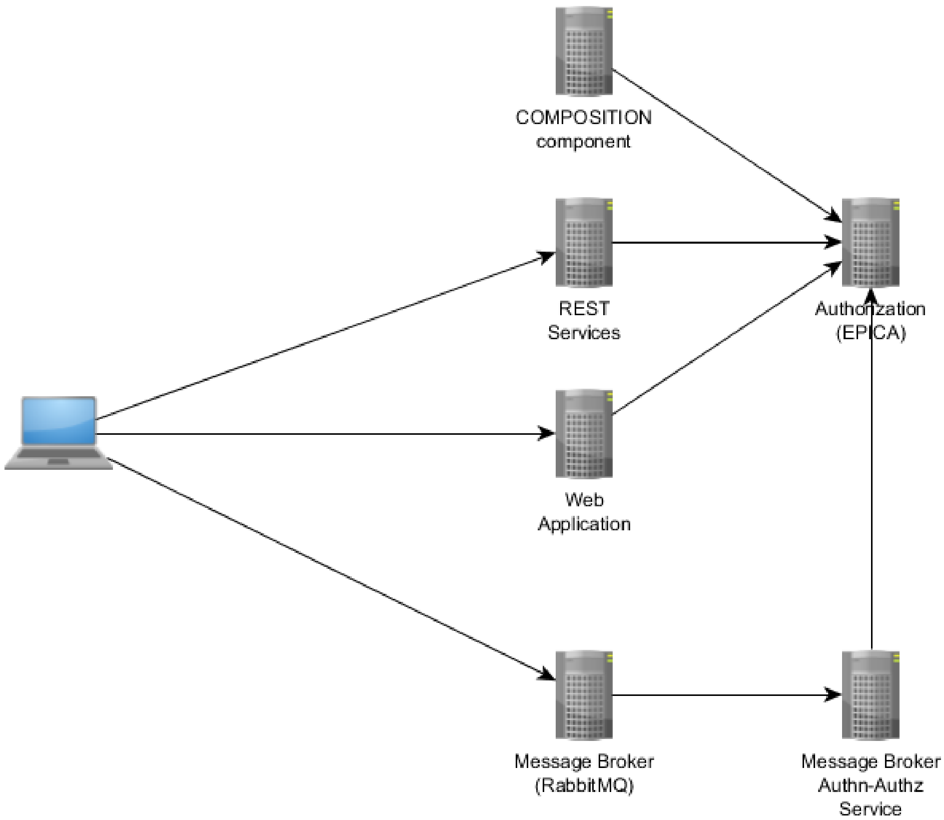

Figure 5: Interaction between Authorization and other components

More information on the Authorization component can be found in D4.2 Design of Security Framework II

The component is currently being dockerised. Once deployed, the rules for accessing resources need to be defined depending on the needs and will be on-demand as other components are being deployed.

6.2 Agent Marketplace [ISMB]

The COMPOSITION Agent Marketplace is the container of the COMPOSITION agents. Agents within the Marketplace implements market-specific services (such as the whitepages or the Matchmaker), or they can act on behalf of industry stakeholders participating in the Marketplace. Required communication infrastructure is provided by a suitable message broker (namely the Marketplace Event Broker), which provides message delivery services to all other components through a well-known, publish-subscribe, interaction paradigm. (See D2.3) A well-defined set of messages in JSON format exists that define the different communications between interacting agents. A full detailed overview will be included in D6.4 COMPOSITION Marketplace II, which is due in M34.

6.3 Authentication [ATOS]

The COMPOSITION Authentication[7] component is responsible for providing authentication mechanisms for users, applications, services and devices; it is based on the Keycloak[8] open-source system.

From the available standard authentication protocols in Keycloak, the COMPOSITION Authentication component makes use of the Open ID Connect protocol (OIDC) which is based on OAuth 2.0[9], but unlike this, it is an authentication and authorization protocol.

One of the major features of Keycloak is the possibility to customize the Authentication Service through the Service Provider Interface (SPI) framework which offers the possibility to implement custom providers or override built-in ones. This feature is used to provide authentication to the COMPOSITION Message Broker

[7] D4.2 Design of Security Framework II
[8] http://www.keycloak.org/
[9] https://oauth.net/2/

components, overriding built-in authentication mechanisms. With this approach there is only one centralized point for authentication and user management.

Figure 6 offers an overview of the possible interactions between the Authentication component and other components that may need authentication mechanisms within COMPOSITION.

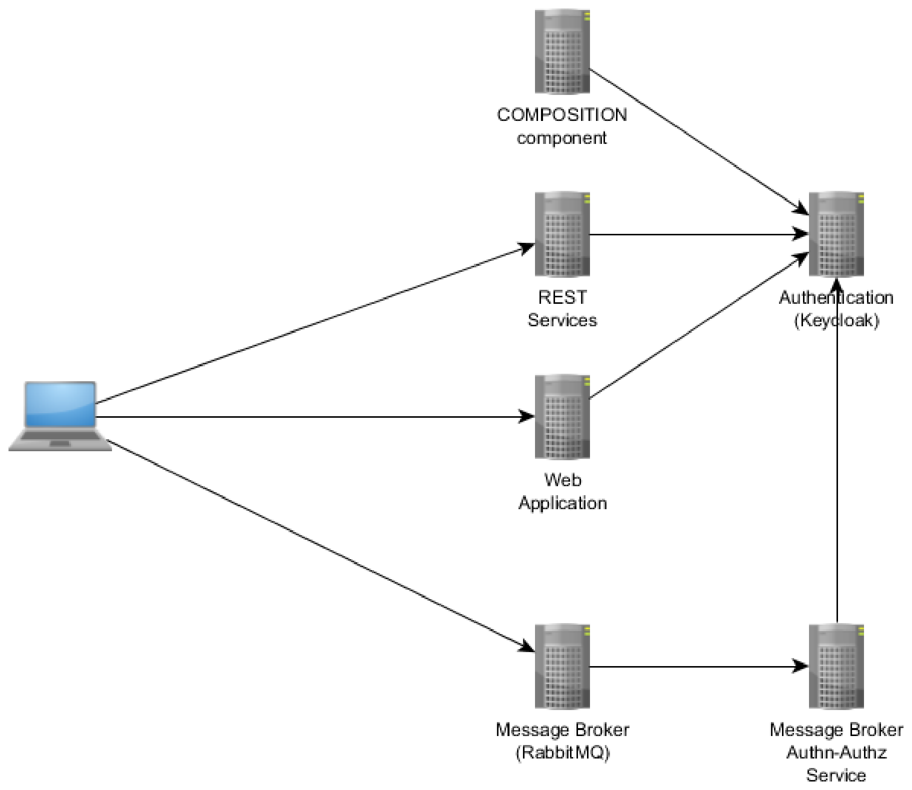

Figure 6: Possible interaction between Authentication and other components needing authentication

The component is currently dockerised and deployed although configuration is still needed and will be done on demand depending on the services, applications or devices to be secured, as well as the users' needs

More information on the Authentication component can be found in D4.2 Security Framework II.

6.4 Big Data Analytics [FIT]

For the analysis of the incoming data, we will use a tailored version of the LinkSmart® IoT Learning Agent. The LinkSmart® IoT Learning Agent was developed for all kinds of store-less data processing, from simple data annotation or aggregation to complex data machine learning techniques. The agent is ideal for intelligent on-demand data management or analysis in IoT environments, from edge computing to cloud computing.

The LinkSmart® IoT Learning Agent provides Complex-Event Processing as a service and Real-time Machine Learning Orchestration as a service. The agent provides three APIs, the Stream Mining API (Statement API), the Learning API (CEML API) and the IO API. The Statement and CEML (see below) APIs are CRUD (Create, Read, Update, Delete) and JSON based, while the IO are write-only (for Input) or read-only (for Output). The APIs are implemented as HTTPs RESTful and MQTT.

The Complex-Event Machine Learning (CEML) framework is a methodology that allows to predefine the learning phases of the phenomena to learn. In this manner, the learning process can be unattended, and orchestrated, distributed and remotely managed in all kinds of industrial environments. More information about the LinkSmart® Learning agent can be found in the deliverable D5.1 Big Data Mining and Analytics Tools I.

6.5 BlockChain Connector [CNET]

COMPOSITION will use blockchain technology to provide a log of transactions that will ensure the integrity and non-repudiation of messages, e.g., agent negotiation and contracts in the marketplace or material/shipment tracking in the factory. The blockchain connector is the component that interfaces COMPOSITION, implements the COMPOSITION specific functionality and interacts with a blockchain implementation. Multichain, an open source product implementing the Bitcoin protocol with some additions, has been selected as the blockchain implementation. Multichain is relatively easy to configure and deploy, not dependent on an underlying "currency" to function, very versatile and free to use as deployed by COMPOSITION. The blockchain connector can work off a specific exchange in the Rabbit MQ event broker using the shovel as described in D6.1 Real-Time Event Broker I, where all events sent to this exchange are automatically forwarded to the regular exchange, while also being processed by the blockchain connector. The exact integration with the broker is still in development. The means of verification against the chain is also in development.

A blockchain adapter will also be developed to store and retrieve the public keys needed by the subscribers to verify the signature of messages received on the message queue. Each participant in COMPOSITION will need to deploy a blockchain node to have access to the public key and thus be able to verify the signature of messages.

A proof-of-concept (connector? adapter?) has been developed and deployed on the Docker test server for review. A similar test was performed at an industry hackathon in another Docker environment. Docker images for the blockchain need a parameterized configuration for deployment.

6.6 Commissioning System [CNET]

During technical scenario sessions, the need was identified for a component to configure sensor setup, e.g. PLC register mappings and hardware and software identifiers, when deploying the COMPOSITION IIMS, re-configuring or installing new sensors. The information about the deployed equipment will have to be distributed to the involved components from the Digital Factory Model instance for the factory. The component responsible for coordinating this was named the commissioning system. The responsibilities of this component are currently managed by manual configuration and static setup.

6.7 Deep Learning Toolkit [ISMB]

The Deep Learning Toolkit is the COMPOSITION component in charge of providing predictions based on the analysed data concerning a fixed future timeframe.

The Core of the Deep Learning Toolkit dwells in its Artificial Neural Networks that are specifically designed for fitting data that are fed to the component. Due to the nature of machine learning and consequently deep learning data science, the Deep Learning Toolkit will not be able to have a single implementation to be reused among different scenarios. Instead, one instance of the component will be specifically designed for each use case, that necessarily will envisage different data types. Hence, each and every one of its Artificial Neural Networks is specifically designed to provide one and only one prevision correlated to a specific use case.

Despite having different implementations, the Deep Learning Toolkit has a common background, which is composed of four different and completely separated instances. Every instance has one expected output that is inputted to the next instance in a waterfall schema, in an asynchronous manner. In the following, the four phases are described:

- The first phase is called Data Pre-processing and happens after the initial Data Harvesting phase. It is in charge of loading and formatting data coming from the shop floor and is a totally offline phase. This module is designed under the assumptions that data types are predefined and operatively consolidated. Therefore, it is not expected to receive batches from the shop floor with missing features. It is required that the number of input features and the batch size is predetermined in the design phase. What this module does, is to consolidate incoming data in order to transform human readable information in datasets that are allowed as input to Artificial Neural Networks. This submodule is usually applied to historical data that are retrieved from an existing information infrastructure at the end user's premises.
- The second phase is called Training and happens offline, too. This step uses between 60% and 80% of the data from the output of the previous phase. This part of the dataset is fed to the Artificial Neural Network in order to train the network itself, alongside a list of hyperparameters. It is beyond the scope

of this deliverable to list and explain the hyperparameters and the reasoning behind them. For details, see D5.3 Continuous deep learning toolkit for real time adaptation I and D5.4 and Continuous deep learning toolkit for real time adaptation II.

- The third phase is called Validation. This is the last offline phase and it is a necessary step required by every Artificial Neural Network. The Validation submodule uses the remaining portion of the dataset left over from the Training phase and provides metrics for evaluating the reliability of the networks. As for phase two, used metrics and outcomes related to different validation constraints are going to be discussed in detail in D5.3 and D5.4. This phase ends with the designer of the Artificial Neural Network manually evaluating the validation phase metrics. If they satisfy the required accuracy, the output is then deployed in the next phase.
- The fourth phase is the online step, and it is what brings the system live in the factory. This operational phase is called Continuous Learning. Among the four instances, this is the only one that requires to be deployed, usually at the shop floor level at least in case of intra-factory activities. In cases where inter-factory interoperability is required, the component will follow the same deployment as the Marketplace Agent.
  This phase expects as input the same batches as the Validation phase which were exactly identical to the ones used to feed the Artificial Neural Network in the training phase. The main difference is that instead of being a historical collection, this time these batches are the result of an aggregation of live data, measured from the very same sensors, It is fundamental to note that batch size may vary from the one used in the two previous steps, while the number of features remains the same.
  The output of this phase provides the latest prevision on the data that will vary from the originally trained Artificial Neural Network, deployed only when enough data is provided to the network itself. So, it should not be expected in a predictive maintenance scenario which might have decades of sensor data collected and thousands of documented breakdowns, that a few dozen breakdown events could affect the previsions. In this phase, live data collected in batches are acquired and processed automatically by the network that provides feedback in an asynchronous manner.

Each of the described phases is implemented in a standalone component that is self-contained and operates indistinctively and independently from the others. This choice has been done on purpose during the design phase, in order to allow to operate and leverage on different parameters without affecting directly all phases at once. In fact, each of the phases depend on the input of its predecessor and produces the output for its successor, but the parameters involved in the process are independent and differ from one another in each of them.

Two other scenarios are the retraining phase and the dispatching phase. The retraining phase is a command process that is issued by an internal evaluation process, in which the whole Training phase is done on the deployed network transforming tensors and neurons of the Artificial Neural Network. It is worth mentioning that this phase is not systemic or scheduled and depends only on the live batches provided. The latter is a set of common operations that envisage actions such as authentication, authorization, publishing and subscribing for performing dispatching operations. In the latest and final implementation, the DLT is self-contained in a standalone Docker component and connected through a private network to the Learning Agent. This allows the two to communicate in a safe manner where less restricted operations are therefore required. These two phases will also be detailed in D5.3 and D5.4.

6.8 Intra-factory Interoperability Layer [ISMB]

The COMPOSITION ecosystem requires a common ground for exchanging data at shop floor level. The intra-factory interoperability layer carries out the task by providing an infrastructure using the publisher/subscriber paradigm. The infrastructure leverages on a broker- based protocol for handling communications among the intra-factory components. Many broker-based systems have been tested in heterogeneous scenarios, with the final choice being a MQTT protocol supporting software, namely RabbitMQ.

In the intra-factory scenario there is the need to create a hierarchy that defines the topics structure for the event broker. The COMPOSITION project sets a background common to all components that will be identified by:

- a topic root that will use the “COMPOSITION” tag as identifier;
- the discriminative dichotomy identifier of the intra or inter-factory scenario;
- the component name that is in charge of generating the data;

- the scope of the data produced.

Every component that needs to exchange information within the COMPOSITION intra-factory communication layer will be required to use the event broker, registering a scope-based topic. COMPOSITION intra-factory components will leverage on this interconnection scalability and capability, and most importantly without the burden of securing yet another communication channel that would not benefit from the enhancements of the security framework that mediates access token renewals and credentials retaining.

A key component in the intra-factory interoperability layer is the Building Management System (BMS). It provides a model for interconnecting the COMPOSITION ecosystem, acting as a translation layer within the shop floor. In fact, it provides connectivity from sensors to the COMPOSITION components. Within this component, information is pervasively collected from any connected systems in order to support the management operators in making decisions and to take direct control of automation tasks. Seamlessly inter-connecting them altogether, the BMS enables the connection with all the major automation standards (such as BACnet, Konnex, Modbus, etc.), thus acting as a bridge between the cyber-physical systems (sensors, gateways, etc.) and the other IIMS components.

6.9 Manufacturing Big Data Storage [NXW]

Manufacturing in assembly lines consists of hundreds, thousands or millions of small discrete steps aligned in a production process. For each of those steps, automatized production processes (or production lines) produce small bits of data in form of events. The events possess valuable information, but the data in the events are usually meaningless unless they are contextualized, either by other events, sensor data or process context. To extract the maximum value from the data, they must be processed in real time and on demand. This reduces latency, provides reactivity and context and prevents having to archive unnecessary data. On the other hand, some information must be stored somewhere for retrieval when necessary, as a historical trace of what has been collected during the process lifetime.

This complex event processing and storage service is provided by the Building Management System (BMS). In the first place, the BMS provides a set of tools for collecting, annotating, filtering or aggregating the real-time data incoming from the production facilities. These tools make it possible to build applications on top of real-time data. Secondly, the BMS provides a storage for information that needs to be kept during the whole machine lifetime. These raw measurements can also be enhanced by providing additional metadata to be attached to them, in case it should become necessary.

All the information contained in the storage service are described using the OGC Sensor Things format.

6.10 Manufacturing Decision Support System [ATL]

The manufacturing decision support system facilitates a number of engineering practices supporting the manufacturing processes (see D3.8 Manufacturing Decision Support System I). The system addresses the need for decision-making support for the senior managers in manufacturing environments, including all elements of the manufacturing process. The DSS Rule Engine is based on Finite State Machines (FST) which create the implemented rules. Each state machine leads deterministically from state to state, using transitions and variables. The tuple of states, transitions and parameters is able to define the rule engine. Enhanced functionality is added when the deterministic tuples use probabilities and training algorithms for the rule engine and the deterministic values become non-deterministic, which express multiple probabilistic outcomes. DSS also collaborates with the Deep Learning Toolkit for maintenance prediction and simulation purposes. The DLT predictions are incoming data for the DSS rule engine and create rules based on the different model probabilities of failure by the DLT. DSS and SFT (Simulation and Forecasting Toolkit) cooperate the same way as DSS and DLT. The SFT sends probabilities of three types of failures (mechanical, electrical and hydraulic) to the DSS. Also, SFT sends the probability for normal operation. DSS uses the probabilities in the rule engine for rule creation.

The DSS is integrated into the IIMS system and elaborates MQTT and HTTP to exchange information with other subsystems, in order to provide context for better decision support and process visualisation. The DSS communicates with the message broker using the topics for the necessary data. Data include live data from sensors on the shop floor, as well as historical data from various sources on the shop floor, such as integrated machine sensors and CMMS data. The COMPOSITION project includes two kinds of users: physical users or persons and applications. The DSS is a COMPOSITION component and works as an application user for the rest of the system. It is authenticated by the BMS and gains access and subscription rights to the message broker, for both MQTT and HTTP. After authentication, the system is able to receive the messages from the broker, with the necessary data.

The DSS uses the collected data, for the decision-making process in the rule engine. The visualisation element of the DSS is fully exploited in the KPI DSS sub-component. Knowledge hidden in the data can be retrieved through the KPI component and visualised for the users. The combination of historical and live data gives the opportunity to visualise the current state of the system, especially for maintenance purposes and to create visual tools and graphs for performance indicators. In a manufacturing environment, the maintenance indicators are very important for reducing downtime of equipment. The KPIs indicate the trends of mean times needed for repair (MTTR, Mean Time To Repair) or between failures (MTBF, Mean Time Between Failures). Maintenance and technical managers see these pieces of information and are thus advised of the state on the shop floor. They also receive predictions of potential breakdowns. For visualisation purposes, it uses the models from the simulation toolkit expressed in BPMN format. Both rule engine and KPI sub-component require data – persistence, as well as steady data acquisition. The data format is based on the Digital Factory Model, deployed for the COMPOSITION project. The data acquisition is guaranteed by the use of MQTT and HTTP protocols, which are, at the moment, the most reliable protocols for data transfer through the internet.

User authentication, authorization and access control is implemented through the integration of the security framework. The user is authenticated by the framework and then seamlessly transferred to the different components. Single sign-on processes have been implemented, and all COMPOSITION HMIs, including the DSS HMI, are integrated and offer a full and unified solution. Web components are used in the HMI integration and unification process, presenting the user with the same experience for all COMPOSITION components. The DSS uses the microservice paradigm, packaged as a docker image for testing and production purposes. Moreover, it provides a mobile app for in-place uses.

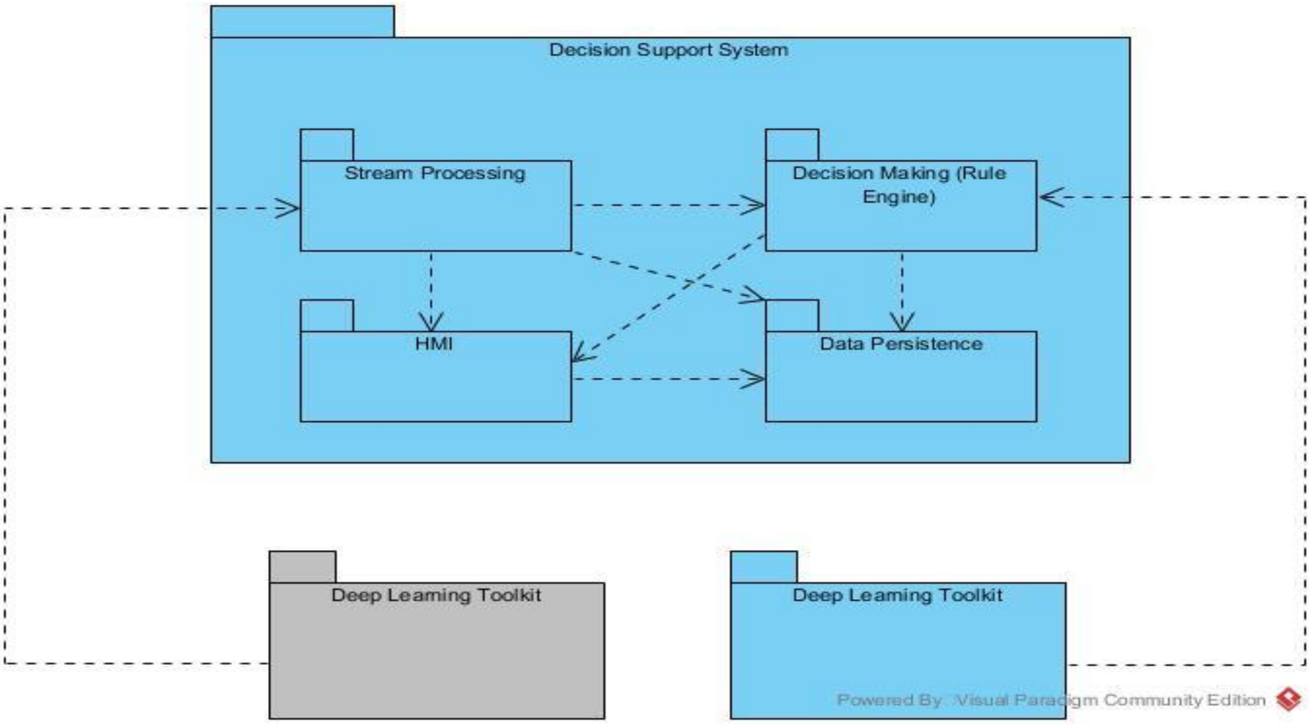

Figure 7: DSS architecture and component diagram

6.11 Market Event Broker [CNET]

The market event broker is the instance of the message broker used in the COMPOSITION Marketplace (see deliverable D6.1). It interacts with most components and is the hub through which marketplace agents communicate. However, this is through the standard AMQP protocol and needs no special configuration or development of the broker itself. The broker is tightly integrated with the security framework, which provides identity and access management for all brokers in the COMPOSITION system (federated or clustered). Scalability configuration tests (cluster, federation setup in Docker) have not yet been performed. The REST tunnel is in proof of concept phase. The intra-factory instance has been deployed and integrated with the security framework, see section 5.15.

6.12 Matchmaker [CERTH]

The Matchmaker is one of the core components of the COMPOSITION Marketplace. This component aims to match requester and supplier agents participating in the Marketplace based on different selection criteria. Furthermore, the Matchmaker component is used by agents in order to match requests and offers between the agents. The Matchmaker's functionality is exclusively depending on Collaborative Manufacturing Services Ontology. The Matchmaker infers new knowledge by applying rules on the knowledge stored in the Ontology.

Figure 8 presents the interactions of Matchmaker component. Internally, the Rule-based Matchmaker sub-component interacts with the Marketplace Ontology store. The Rule-based Matchmaker applies rules to Ontology and infers new knowledge in order to perform matching. Externally, the Matchmaker component interacts with the Marketplace's agents. The Matchmaker offers matching services to agents. Moreover, it offers services related to Ontology manipulation.

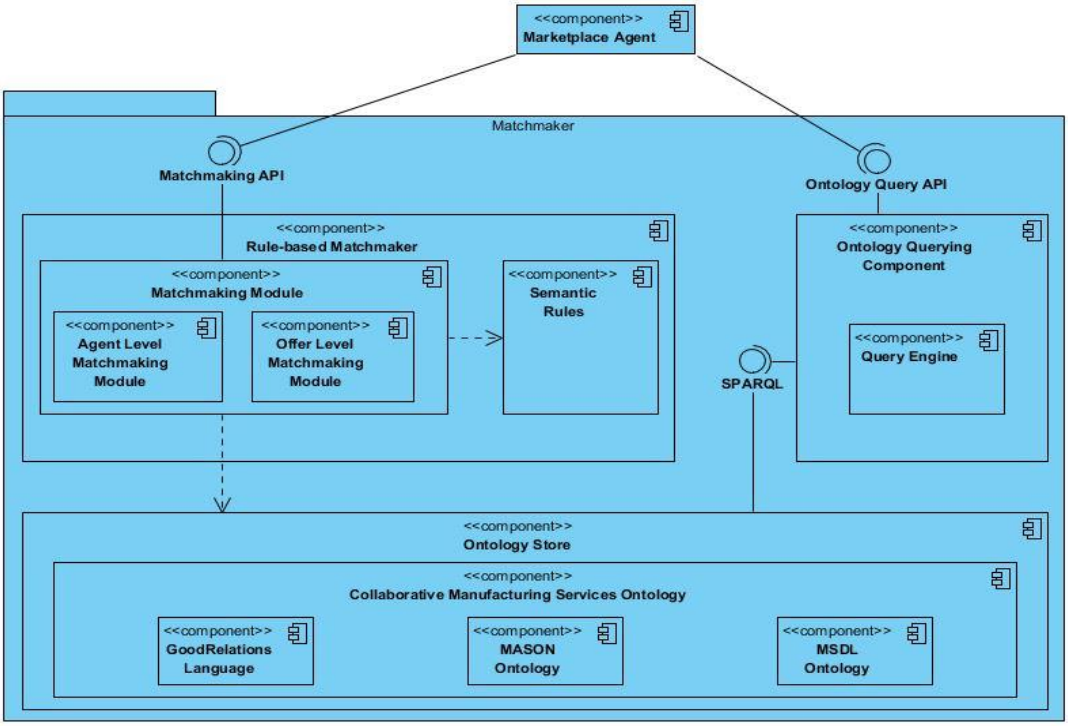

Figure 8: Matchmaker architecture

Development status

The current version of the Ruled-based Matchmaker has been developed in Java and it is offered through RESTful web services. The Matchmaker can infer new knowledge by applying semantic rules in the knowledge stored into the ontology. The set of rules (Jena rules) supports matchmaking between services (requested & offered). The rules also support offer and request matching based on price, quantity, delivery time and company ranking.

In the next releases, the set of semantic rules will be extended to support more complex decision criteria based on real data.

Furthermore, a version of the Marketplace Ontology has been developed in OWL language, covering both manufacturing and e-commerce domains. Moreover, an Ontology Query API has been implemented. The API offers CRUD operations to Marketplace Agents.

Deployment

A Docker image has been created for the current version of the Matchmaker. The image of the Matchmaker represents the complete Semantic Framework of COMPOSITION as this image contains the Matchmaker, the Ontology Query API and the Ontology Store. It is deployed as a Docker container on the COMPOSITION Inter-factory production server.

6.13 Real-Time Multi-Protocol Event Broker [CNET]

The real time multi-protocol event broker is the instance of the message broker used in the COMPOSITION factory IIMS (see D6.1). As the hub for all message-based communication, it interacts with most components in the IIMS. However, this is through the standard MQTT protocol and needs no special configuration or development. The broker is tightly integrated with the security framework, which provides identity and access management for all brokers in COMPOSITION system (federated or clustered). Scalability configuration tests (cluster, federation setup in Docker) have not yet been performed. The REST tunnel is in proof of concept phase. The main component, RabbitMQ, is deployed on the test server Docker host. The integration with security framework and blockchain connector is partly complete and deployed at the test server.

6.14 Requester Agent [ISMB]

The Requester Agent is the agent exploited by a factory to request the execution of an existing supply chain or to initiate a new supply chain. Due to the dynamics of exchanges pursued in COMPOSITION, there is no actual distinction between the two processes, i.e., for any supply need a new chain is formed and a new execution of the chain is triggered. The Requester agent may act according to several negotiation protocols, which can possibly be supported by only a subset of the agents active on a specific marketplace instance. (See D2.4)

6.15 Reputation and Trust Model

In the COMPOSITION Reputation Model, the basic idea is to follow the selected reference model[10], in order to infer the basic requirements that should be satisfied, depending on the specific context of the project. Each agent of the marketplace must be able to provide a rating related to each single transaction, when they act as the requestor (trustor): these ratings could be integer values within a predefined interval for measured attributes (e.g., **trust**, **reputation**, quality of service provided, seller reliability, **critical level of the refined information**) processed through computation engines (e.g., summation, average, fuzzy, belief, continuous/discrete).

[10] S. Vavilis, M. Petković and N. Zannone, “A reference model for reputation systems,” Decision Support Systems, vol. 61, pp. 147-154, 2014.

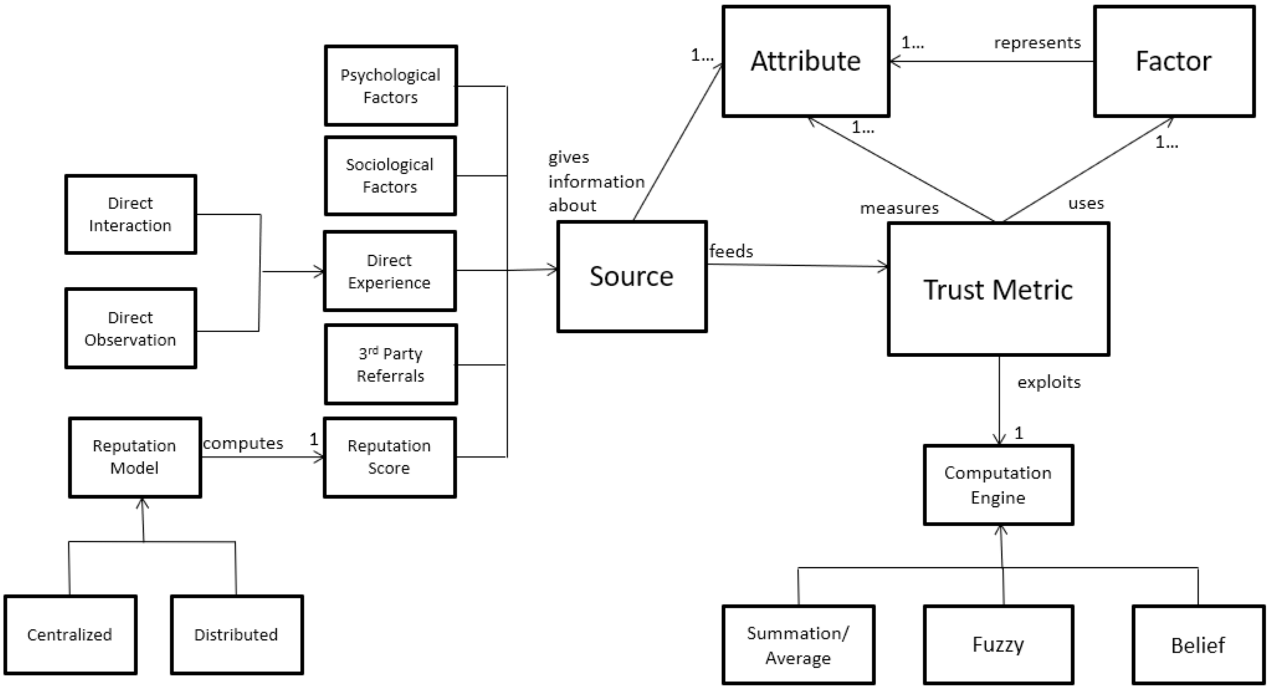

Figure 9: Reputation and Trust Model

- Dynamic and continuous trust assessment for all involved entities exploiting both internal and external knowledge
- Speed-up decision making phase
- Reputation models fit very well in open, or semi-open, environment
  - new member could join
  - actual member could leave, and re-join after leaving
- Reputation values and 3rd party referrals could be stored into the blockchain [2] (in case of distributed model)
  - easy to track possible fake and malicious behaviours
- Computed trust values can take into account specific information specifically related to COMPOSITION
  - Critical level of the information received
  - Quality of the provided information
  - Context effectiveness in the related service
  - Time constraints

COMPOSITION is relying on blockchain technologies as the central component of its log-oriented architecture. This technology will be used for implementing a secure, trusted and automated information exchange related to supply chain data. Considering the distributed nature of blockchain and, more in general, of the COMPOSITION infrastructure, it makes sense to rely on a distributed Reputation Model: each agent will compute his own reputation values and will be in charge to provide these values to the other entities.

The Reputation and Trust Model is further described in D4.2.

6.16 XL-SIEM [ATOS]

XL-SIEM[11] (Cross-Layer SIEM) provides the capabilities of a Security Information and Event Management (SIEM) solution with the advantage of being able of handling large volumes of data and raise security alerts from a business perspective thanks to the analysis and event processing in a Storm cluster. The main XL-SIEM functionalities can be summarized in the following points:

- Real-time collection and analysis of security events.
- Prioritization, filtering and normalization of the data gathered from different sources.
- Consolidation and correlation of the security events to carry out a risk assessment and generation of alarms and reports.

The architecture of XL-SIEM is presented in Figure 10:

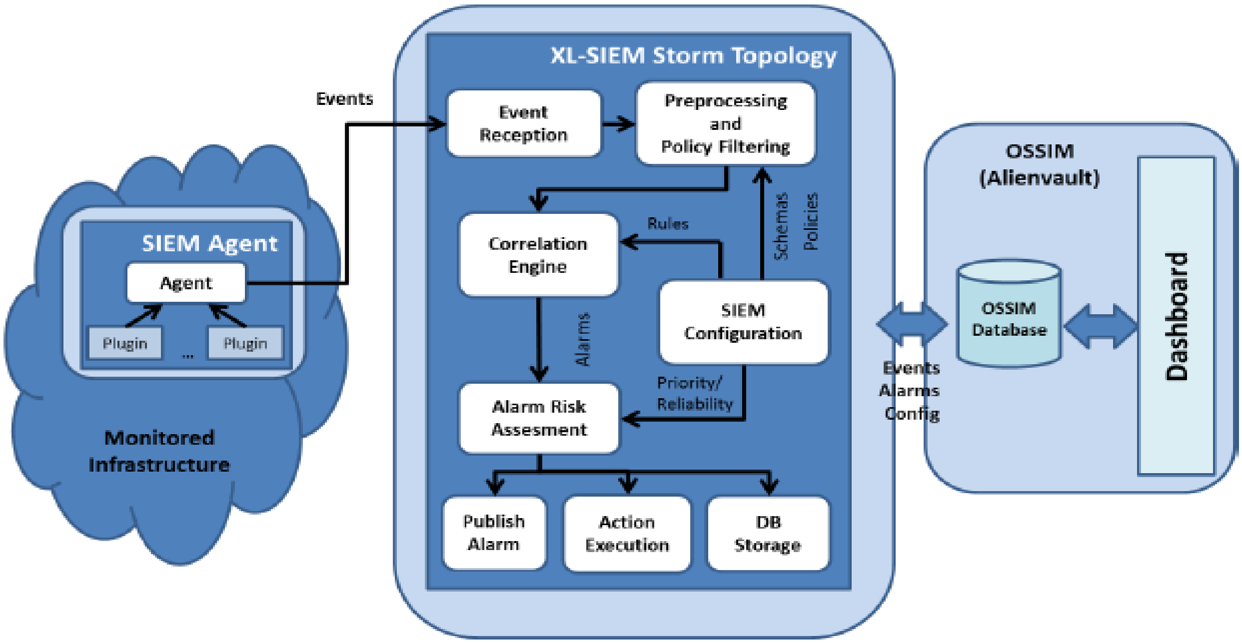

Figure 10: XL-SIEM architecture

The SIEM Agents are responsible for data collection and are deployed within the monitored infrastructure. In case of any event occurrence, they are sent to the XL-SIEM core where they are processed and correlated. The OSSIM is responsible for storing gathered events and eventual alarms that were produced during the correlation process. The OSSIM also has visualisation capabilities enabling data inspection.

More information on the XL-SIEM component can be found in D4.2 Design of Security Framework II.

The XL-SIEM is deployed on ATOS premises and is under continuous improvement.

6.17 Simulation and Forecasting Tool [CERTH]

The Simulation and Forecasting Tool component is part of the high-level platform of COMPOSITION, the Integrated Information Management System (IIMS). The main purpose of the Simulation and forecasting too is to simulate process models and provide forecasts of events whose actual outcomes have not yet been observed.

Its main interactions are LinkSmart middleware, Digital Factory Model, Data persistence storage, Decision Support System (DSS) and Visual Analytics tool.

Development Status

[11] D4.1 Security Framework I Section 4.4

Analytic algorithms of pilot site data related to BSL, KLE and ELDIA use cases has been implemented. Algorithms related to Descriptive statistics, linear regression analysis, Markov models, Genetic algorithms for optimization and Correlation heatmaps have been developed in Python programming language and applied to related data and use cases.

Deployment

A Docker image for the Simulation and forecasting tool algorithms has been created and deployed on the COMPOSITION Intra-factory production server. Algorithm dockerisation is in progress.

6.18 Supplier Agent [ISMB]

The Supplier Agent is the counterpart of the Requester agent in the COMPOSITION marketplace. It is usually adopted by actual suppliers to respond to supply requests coming from other stakeholders in the marketplace. Factories transforming goods typically employ at least one Requester agent to get raw materials, and one supplier agent to sell intermediate products to other factories. (See D2.4 The COMPOSITION Architecture Specification II).

7 Mapping to Use Cases

In Confluence, the COMPOSITION partners have created a mapping from the Components defined in the JIRA repository to the use cases, which is monitored and tracked regularly. The mapping is created based on a set of tiers/priorities as determined in consortium discussions. The entire set of entries is too lengthy for inclusion here, but an example from one of the use cases is shown in the table below

Confluence Spaces People Create •••

Pages /... / Task 2.3 COMPOSITION Architecture

Implementation of Use Cases through Composition Components

Created by Marc Jentsch, last modified by Alexandros Nizamis on Nov 14, 2017

Tier 1

UC-BSL-2 Predictive Maintenance

Component	Responsible Partner	Expected Input from other Components (Source/Input)	Description of the Functionality in the UC	Output	Development Status	Next Release Steps	Expected Time for First Deployment
Deep Learning Toolkit	ISMB	• NXW: oven live data connected to IIMS • TNI-UCC: data from all sensors deployed in the shop-floor • FIT: pattern analytics on all data	The component will provide the latest prevision on next possible breakdowns, based on live data analysis on a trained artificial neural network with historical data.	Report in the form of JSON data, ready to be redistributed to the designated container.	Already adapted the low-fi demo structure from KLE-6 to design an ad-hoc ANN that fits the UC. Currently assessing provided historical data, but waiting for full data availability.	Trained ANN with real historical data and synthetic live, until existing machinery will be connected to the IIMS providing live reading from sensors.	> M16
Wireless sensing network (WSN) platform	TNI-UCC	(Expecting the oven built in sensors to separately supply temperature data to DSS.)	Adding sensors on or near the blowers to gather extra info and determine if fans are operating correctly. Investigating use of power consumption, speed, magnetic, acoustic/ultrasonic and vibrational. These are feed to the DSS (deep learning toolkit?)	A to D conversion done on WSN board and data sent to gateway/edge device/LAN. Architecture to achieve this is under review.	Looking at SoA in acoustic and vibrational sensors for suitability. At system level determine what can be done to aggregate sensor readings (e.g. 1 current clamp to measure combined output of 3 fans) Also need to figure out how data is received and fed to LAN/DSS repository	Select sensors, devise WSN architecture	Aro. M15-16 in TNI-UCC labs and BSL white space
Energy harvesting (EH)platform	TNI-UCC	Battery or DC power source may also be available	Determine if we can extend battery life of the WSN platform using ambient energies (or ultimately eliminate the need for battery replacement)	Power source	It will depend on the WSN infrastructure, sensors use, duty cycle etc. These all govern power consumption	Develop simulation model of WSN load Develop prototype energy harvesting solution	Simulation model aro. M18 EH proto around M24
Big Data Analytics (LinkSmart® IoT Learning Service)	FIT	• Live Data Coming From the Shop-floor as MQTT messages using OGC SensorThings Standard Payload (NXW) • A endpoint to request predictions (ISMB)	The component can apply on-demand stream processing algorithms and redistributed to the to other REST or MQTT endpoints. Additionally, provides (near-)real-time learning orchestration and management capabilities. Also, it can be used to learn using existing implemented algorithms in it.	• Processed data (aggregated, annotated, etc.) • Pre-processed labeled data ready to be train • Real-time alert generation • Predictions using trained models	• Ready to be integrated. • Ready to be use. • 100% OGC Sensor Things 1.0: part 1 complaint • Improving: documentation and continuous integration • Waiting for any change request for any partner.	Current release version is the 1.5.1. The FIT-UCC-Data team is studding the oven data.	>M15
Human-Machine-Interfaces	FIT	• (at least) Predicted data (~ 3 weeks of likelyhood to break down, time frame might need to be adjustable) [Deep Learning Toolkit] • Live and historical (adapatable time frame) sensor data • Name of data source (machine + part)	Visualization about past and future development of parts that need to be maintained	• Visualization (no data outputs)	• First interaction models and wireframes available	Refined interface wireframes	>M16
Digital Factory Model	CERTH	• Static data about actors, assets, equipment etc from BSL (many of them are available at D2.1 or BSCW) • BPMN diagram from FIT • Dynamic sensor data from shop-floor	DFM component will keep in common format information about static and dynamic data related to UC. Other components will be able to get/post data from/to DFM by using the DFM API	• A common format for data description (DFM Schema) • Static and dynamic data in XML format based on DFM Schema (DFM API)	• DFM Schema is available. It is based in well-known standards such as B2MML, gbXML and BPMN for static data representation. Dynamic data representation is based in OGC Observations and Measurements standard • A first version of DFM API is available to consortium. Other components will be able to get or set data to a DFM 'live' instance by using HTTP requests. Data will be stored in a MongoDB in a format compatible with DFM Schema	Replacement of OGC XML format with OGC JSON format, for the description of sensors data	N/A

Table 2: Example of mapping components in JIRA to use cases

8 Findings and Lessons Learnt

Based on discussions amongst partners, the following is a list of lessons learnt through M27 (November 2018). The first six Lessons were already reported in D7.8.

1. The use of tools such as Confluence assisted us in defining and adjusting priorities, architectures and methodologies
2. In general, it is difficult to get clear insights about market segmentation and potential since the inter and inter-factory collaboration field/ topic is relatively new and strong competition is currently forming. However, an initial indicative competitor analysis has been provided in D9.10 Exploitation planning framework and first draft of exploitation plans.
3. Ever changing use-cases and use-case prioritization made it very hard to structure and focus on the most relevant aspects but it was a worthwhile process and it tested the modularity and re-configurability of the technology components
4. As there was an intense focus on delivering high impact use cases, the ones selected are quite “high-class” and thereby complex. Ideally, they should be should be as simple as possible initially with a minimal viable product-thing remit. However, this is a trade-off versus showing something that has high impact and relevance to the COMPOSITION industry partners.
5. It was found that when consortium partners got together much progress was made by virtue of having everyone together physically and acting in a dynamic environment. In future projects it is recommend to do more of this where possible and as early as possible.
6. Interaction with the other FoF-11 project has been proven very valuable, as it boosts knowledge sharing and it brings together people that are in essence trying to address digital automation for collaborative manufacturing and logistics.
7. Data analytics correlation needs to be addressed beforehand. Low data correlation is a risk that needs to be assessed in the CA/GA and also in the DoA, so there would be a common understanding among partners on how to react and mitigate.
8. Continuous integration needs to have tighter milestones and be representative of the status of each component at a given time. Freezing versions is also a must in order to avoid single components refactoring gone wrong to be responsible for slowing down the ecosystem development.
9. Coordination among tasks that use similar cutting-edge technologies for different scopes (e.g. artificial neural networks) needs to have a common background to commonly validate state of the art analysis and iterating among different solutions in a shared vision, avoiding overlaps and re-dos.
10. The information view in architecture deliverables should follow standard IEEE 1471 and also provide detailed information of the virtualized environment. In this way every component can check if it fits its incremental requirements.
11. The real benefits of adopting an existing standard should be evaluated in an impact/benefits model for maximizing project impact and minimizing development overhead

The following tables summarise the technology components in development with assigned owner and overview of its use of and compatibility with off the shelf (OTS) software.

Table 3: Technology component owner, SW used and compatibility

Technology component	Owner	OTS SW used	Compatible with
Access Control	ATOS	EPICA	XACML v3.0
Agent Marketplace	ISMB	RESTful web Services RabbitMQ	AMQP protocol
Authentication	ATOS	Open ID Connect protocol (OIDC) which is based on OAuth 2.0	Keycloak OS
Big Data Analytics	FIT	LinkSmart® IoT Learning Agent	MQTT & OGC SensorThings
BlockChain Connector	CNET	Rabbit MQ broker	Bitcoin protocol
Commissioning System	CNET	RESTful Web Services	PLC register mappings, OGC SensorThings
Deep Learning Toolkit	ISMB	Supporting libraries such as Numpy. Frameworks such as TensorFlow.	MQTT, Tensorflow, Keras and Pyro adapters.

Technology component	Owner	OTS SW used	Compatible with
		Implementation of Pyro interfaces	
Intrafactory Interoperability Layer	NXW	(NXW) BMS	BacNET, Konnex, MODBUST, MQTT, etc. OGC SensorThings
Manufacturing Big Data Storage	NXW	BMS	MQTT, REST, OGC SensorThings
Manufacturing Decision Support System	ATL	Rabbit MQ MQTT Restful web services	AMQP Protocol MQTT Protocol PMML BPMN
Market Event Broker	CNET		AMQP Protocol
MatchMaker	CERTH	Jena API RESTful web services	OWL Ontology, HTTP protocol
Real Time Multi Protocol Event Broker	CNET	REST tunnel Rabbit MQ	MQTT protocol
Requester Agent	ISMB	RESTful web Services RabbitMQ	AMQP protocol
SIEM	ATOS	XL-SIEM OSSIM database	
Simulation and Forecasting Tool	CERTH	No. Only editors such as ATOM	
Supplier Agent	ISMB	RESTful web Services RabbitMQ	AMQP protocol

9 Intra-factory Action Plan

The following is an action plan that was devised and captured on JIRA based on consortium meetings in order to co-ordinate and monitor progress to develop and integrate the various technology components.

Table 4: Intra-factory action plan

Title	Description	Deadline	Responsible	Status
Docker deployment Wiki	A page/guide describing where and how to deploy the docker components, ATOS, ISMB	15.10.2017	ISMB, ATOS	Done
Security setup	Security Framework auth setup description (RabbitMQ, not REST)	9.10.2017	ATOS	Supervised by FIT and tested by ISMB
Deploy LinkSmart Service Catalogue on docker lab server at ISMB	Get access to Portainer and deploy. Provide example of registration message.	15.10.2017	FIT	Already dockerised
Deploy Big Data Analytics (LS Learning Service) on Docker lab server	Provide examples on wiki.	15.10.2017	FIT	Already dockerised
Simulation and Forecasting deployment on Docker lab server	Simulation and Forecasting deployment on Docker lab server	M28	CERTH	Most of SFT algorithms are dockerised
Visual analytics deployment on Docker lab server	Visual analytics deployment on Docker lab server	M29 – 30	CERTH	In progress
Digital Factory Model deployment on Docker lab server	Interface descriptions published for other partners (No JSON format, XML at this date). JSON conversion at later date (end November?)	done	CERTH	Already dockerised
Deep learning Toolkit deployment on Docker lab server	The first of the expected three ANNs models is already deployed and up for testing. Two more are expected to follow.	M28	ISMB	Under testing by FIT, and BLS + ALT.
Predictive Maintenance sensors	Install acoustic and power monitoring sensors in BSL	mid Nov M15	TNI	N/A
Asset tracking sensors	Do initial lab experiments in Tyndall and BSL whitespace with UWB WMU & GPS	mid Nov M15	TNI	N/A
BSL Sensors connected to COMPOSITION installation	legacy equipment connected to the composition cloud via local PC	end of M20	TNI, NXW, BSL	Connected to BMS through SFTP
Interoperability Layer	T5.5 ensure interoperability of components	Done	CNET, ISMB	Testing is necessary after latest component updated, Expected full end-to-end dry run at M29.
Fill level sensors, vibration at KLE	Vibration and acceleration sensors first prototype (custom sensors)	done	CERTH	Sensors are deployed at KLE and ELDIA plants
Authorization service (to be deployed at docker lab)	Make changes to integrate with keycloak and create ACL:s (XACML)	mid Nov M15	ATOS	To be done

Title	Description	Deadline	Responsible	Status
Decision Support System deployment on Docker lab server	DSS for process performance visualisation	end Oct M14	ATL	Done
Decision Support System deployment on Docker lab server	DSS for predictive maintenance	mid Nov M15	ATL	Done
Decision Support System deployment on Docker lab server	KPI mechanism	Mid Jul M23	ATL	Done
Decision Support System deployment on Docker lab server	Data persistence through the COMPOSITION components and data acquisition for DSS purposes. Data readiness and availability	Mid Sept M25	ATL	Done
Decision Support System mobile application	Notification service for mobile devices, and backend	mid Nov M27	ATL	Done
Decision Support System mobile application	Packaging the whole DSS component for dockerisation and applying HMI integration with the rest for the COMPOSITION components	Mid Jan M29	ATL, CERTH, CNET, FIT, NWX	To be done.
BMS	integrate with LS and output OGC ST	end Dec M16	NXW	Done
BMS sensor integration	Actual sensors	Jan 2018 M17	NXW	Done
Verify information flow in system and test	Components can listen to sensor data and messages	M29	CNET ISMB	Expected full end-to-end dry run at M29
Verify API calls between components	Component can call other component APIs	M29	CNET ISMB	Expected full end-to-end dry run at M29

10 Inter-factory Action Plan

A similar inter-factory plan was also developed.

Table 5: Inter-factory action plan

Title	Description	Deadline	Responsible	Status
Semantic Matchmaking (+Ontology) deployment on Docker lab server	Swagger definition of API	done	CERTH	Dockerised
Set up another docker host for the inter-factory	Should be on separate host. Configuration management (networks,1or2nginx, etc.)	end Oct M14	ISMB, ATOS	Done
Requestor agent I	Not double faced connectivity at this stage	end Dec M16	ISMB	In progress
Requestor agent II	Connects to both intra and inter installation	TBD	ISMB	In progress
Supplier agent		end M16	ISMB	In progress
Marketplace Broker	Broker (cloned from intra)	end Oct M14	ISMB, ATOS	In progress
Security Framework	Security framework (cloned from intra) only authentication initially	end Oct M14	ATOS	In progress
Blockchain	To be used by PKI, Log, RM, decide on how to use streams et c. "Configurable setup"	M20	CNET, ATOS	In progress
Reputation Model		M22	ATOS	To be done
PKI		M22	ATOS	To be done
Log	What transactions, messages do we log & how to configure. (Blockchain log)	M20	CNET, ATOS	Ongoing
Communication Agents-Matchmaker	Agent communication with matchmaker finalized and working in installation	done	ISMB, CERTH	done
Interoperability verification	Ensure interoperability of components	M22	CNET, ATOS, ISMB	Ongoing. Continuous verification of interoperability will start when all standalone Docker images deployed will have started integration, relying on the Intra-factory Event based broker
KLE sensors installed & sending data	Linked with intra-installation task	done	CERTH	Sensors are deployed at KLE and ELDIA plants
Marketplace Management Services	API and backend for marketplace management services	M27	NXW	In progress
Marketplace Management Portal	UI for marketplace management		CNET	

11 Conclusions, Recommendations and Next Steps

In COMPOSITION a reference architecture has been successfully developed that is now being used in individual use cases.

To date the following components have been developed in COMPOSITION based on this architecture, where possible leveraging from commercially available SW modules and open source widely used protocols

Technology component	Functionality
Access Control	Provides the means to define the security policies used to protect resources, for afterwards any request to access a protected resource will first be evaluated against these policies and the evaluation result will be enforced depending on the outcome.
Agent Marketplace	Container of the COMPOSITION agents. Agents within the Marketplace may implement market-specific services (such as the whitepages or the Matchmaker), or they can act on behalf of industry stakeholders participating in the Marketplace
Authentication	Provides authentication mechanisms for users, applications, services and devices
Big Data Analytics	Provides datastream processing and online learning orchestration through the CEML framework.
BlockChain Connector	Provides a log of transactions that will ensure the integrity and non-repudiation of messages, e.g. agent negotiation and contracts in the marketplace or material/shipment tracking in the factory
Commissioning System	A component to configure sensor setup, e.g. PLC register mappings and hardware and software identifiers, when deploying the COMPOSITION IIMS, re-configuring or installing new sensors
Deep Learning Toolkit	Provides predictions based on the analysed data concerning a fixed future timeframe.
Intrafactory Interoperability Layer	Provides a common ground for exchanging data at shop floor level providing an infrastructure using the publisher/subscriber paradigm.
Manufacturing Big Data Storage	[Whilst most data should where possible be handled in real time]…. Some valuable information must be stored somewhere, to be retrieved when necessary, as an historical trace of what has been collected during the process lifetime
Manufacturing Decision Support System	The manufacturing decision support system brings support to a number of engineering practices. It visualises results from prediction and simulation and provides context knowledge to the final users
Market Event Broker	This is the instance of the message broker used in the COMPOSITION Marketplace. It interacts with most components and is the hub through which marketplace agents communicate.
Matchmaker	Aims to match requester and supplier agents participating in the Marketplace based on different selection criteria. Used by agents in order to match requests and offers between the agents.
Real Time Multi-Protocol Event Broker	This is the instance of the message broker used in the COMPOSITION factory IIMS. As the hub for all message-based communication, it interacts with most components in the IIMS.
Requester Agent	The agent exploited by a factory to request the execution of an existing supply chain or to initiate a new supply chain.
Reputation and trust model	Provides dynamic and continuous trust assessment for all involved entities in the marketplace exploiting both internal and external knowledge. By using adoption of the blockchain for reputation distribution; all the agents will have a global view of every interaction related to each agent of the marketplace in a secure, trusted and automated manner.
SIEM	Provides the capabilities of a Security Information and Event Management (SIEM) solution with the advantage of being able of handling large volumes of data and raise security alerts from a business perspective thanks to the analysis and event processing in a Storm cluster.
Simulation and forecasting tool	To offer analytics services and provide forecasts of events whose actual outcomes have not yet been observed.
Supplier Agent	The counterpart of the Requester. It is usually adopted by actual suppliers to respond to supply requests coming from other stakeholders in the marketplace.

At a high level there are two communication mechanisms in the COMPOSITION system: message-based communication over MQTT or AMQP using the Event Broker[12], and request-response REST HTTP interfaces (D2.4 The COMPOSITION Architecture Specification II)

With a few exceptions[13], the components are loosely coupled and integrated in the system by conforming to a common communication infrastructure and data schemas. There are dependencies on data generated by other components, but for most of the data, the dependency is not direct as this data is distributed through the event broker, so the direct connection is to this component.

Agents and user interfaces (HMI) depend on tools for modelling, simulation and analysis of factory data and processes. Business functionality, e.g. manufacturing analysis, is implemented by components providing generic functionality for complex event processing (CEP) and neural networks (deep learning). These in turn use the communication functionality, the intra-factory interoperability and shop -floor connectivity functional packages. The inter-factory packages provide the necessary infrastructure for the agents to find other agents, negotiate and enter agreements.

Next steps are to

(i) Further develop and dockerise all the components following the inter and intra-factory plans.

(ii) Integrate the components into use cases and demonstrate their viability and effectiveness

(iii) Work on the architecture as we progressively experiment with its suitability for various use cases.

[12] Real-time Event Broker, Message Broker

[13] E.g. Big Data Analytics and Deep Learning networks are tightly integrated.

12 List of Figures and Tables

12.1 Figures

12.2 Tables