RAASCEMAN

Resilient and Adaptive Supply Chains for Capability-based Manufacturing as a Service Networks

Grant Agreement No. 101138782

Deliverable 1.3

Software & information architecture

Funded by
the European Union

Project title	**RAASCEMAN - Resilient and Adaptive Supply Chains for Capability-based Manufacturing as a Service Networks**
Grant Agreement number	101138782
Funding scheme	Call: HORIZON-CL4-2023-TWIN-TRANSITION-01 Topic: HORIZON-CL4-2023-TWIN-TRANSITION-01-07
Project duration	1 September 2024 – 31 August 2027 (36 months)
Project coordinator	DFKI – Deutsches Forschungszentrum für Künstliche Intelligenz GmbH
Deliverable number	**D1.3**
Title of the deliverable	**Software & information architecture**
WP contributing to the deliverable	**WP1**
Deliverable type	R
Dissemination level	PU
Due submission date	**30 June 2025**
Actual submission date	
Partner(s)/Author(s)	**INTRA, DFKI, FM, CEA, LMS, CTU, RPTU**
Internal reviewers	**DFKI**
Final approval	

Disclaimer

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Health and Digital Executive Agency (HADEA). Neither the European Union nor the granting authority can be held responsible for them.

History of changes		
When	**Who**	**Comments**
13/05/2025	INTRA	Outline
27/05/2025	INTRA, DFKI, FM	Contributions to section 4
03/06/2025	CEA, DFKI, FM, LMS, CTU, RPTU	Contributions to sections 1, 2 and 3
17/06/2025	CEA, DFKI, FM, LMS, CTU, RPTU	Finalization of contributions to section 4
24/06/2025	INTRA	Finalization of contributions to section 4 and 5
28/06/2025	DFKI	Internal review
31/06/2025	CEA, DFKI, FM, LMS, CTU, RPTU, INTRA	Improvements after the internal review
8/07/2025	DFKI, INTRA	Finalization of the internal review process

Confidentiality	
Does this report contain **confidential** information?	Yes ☐ No ☑
Is the report **restricted** to a specific group?	Yes ☐ No ☑ *If yes, please precise the list of authorized recipients*:

Table of Contents

Executive Summary

This deliverable presents the RAASCEMAN's system architecture, while also highlighting the participating technological building blocks. In order to achieve this, the system's requirements are identified and converted into technical specifications. These specifications are then mapped to technological building blocks. The system's architecture includes these technological blocks along with the interfaces between them. Additionally, for reinforcing the architecture design process, a thorough analysis of state-of-the-art technologies in different aspects including industrial reference architectures, interoperability and security will be taken into consideration. The goal of this approach is to bring together all the identified components in an efficient and scalable way.

1 Introduction

This deliverable presents the software and information architecture of the RAASCEMAN project in order to support adaptive capability-based Manufacturing as a Service (MaaS) networks. The methodology used for the architecture was to combine pilot requirements analysis results with "State of the Art" analysis and last but not least the components/methodologies that have been identified as outputs of the various RAASCEMAN technical tasks from the "Detailed Work Description/Description of Work" (DoW) of the RAASCEMAN project.

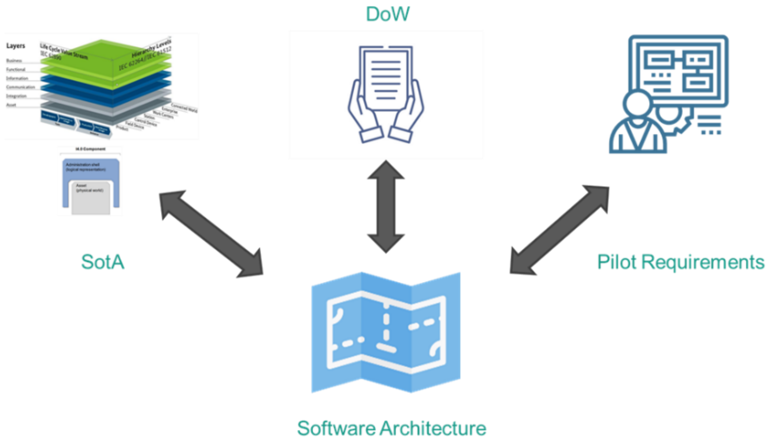


Figure 1: Software architecture methodology

For the technical requirements, a structured technical requirement analysis is performed for each pilot to identify the critical functional and non-functional requirements. This analysis is crucial for ensuring alignments between business goals and technical implementation. Each of these requirements have been associated with a technological building block from project's work plan and each block is linked with at least one functional layer. In this way, the architectural functional layer groups are derived, including business logic, infrastructure, integration, information, security, presentation and persistence.

State-of-the-art technologies are explored based on the afore identified functional layers and most specifically the infrastructure, integration, information, and security layers. Regarding the infrastructure layer, a generic architecture state-of-the-art is performed to examine architectures like RAMI and IIRA, while for the integration layer data exchange technologies are explored such as OWL and RDF. For the information layer analysis, the focus was towards standardized protocols and data formats, like HTTP and JSON. Finally, concerning security, intra- and cross- company security mechanisms and approaches are evaluated, such as authentication mechanisms and data spaces.

Building upon this methodology, a conceptual architecture is proposed, defining the main participating components and their interaction across the system.

2 Use Cases & Requirements Analysis

2.1 Introduction and Methodology

Requirements analysis is a critical activity in system engineering, serving as a bridge between stakeholder needs and the overall system architecture. It ensures that the system's functionalities are clearly defined, and traceable throughout the development lifecycle. In complex, distributed, and domain-specific environments such as industrial IoT and digital manufacturing, requirement analysis ensures alignment between business goals and technical implementation.

The process followed for technical requirement analysis, as depicted below, was based on a structured, traceable methodology grounded in software engineering best practices. The technical requirements presented in D1.1 was the basis, then they were linked to relevant use cases, as recommended in IEEE 830 and ISO/IEC/IEEE 29148 standards for software requirements specification [1][2]. In a later step, these requirements were mapped to architectural components, associating system responsibilities to specific elements within the architecture [3]. Next, a component-centric view was created based on Requirements Traceability Matrices (RTMs) [4][5] and finally were categorized into functional architecture layers, including Infrastructure, Integration, Business Logic, Information, Security, and Persistence layers. The identification of core functional components will help identify the fields that would require a state-of-the-art analysis but also allow a preliminary conceptual design of the overall architecture.

2.2 Technical Requirements Analysis

The table below shows the first steps on the requirement analysis phase depicting the mapping of the requirements to the use cases but also the architecture component that fulfills each requirement.

Table 1 Requirements to Use Cases Mapping

ID	Requirement	Use Case ID	Architecture Components
REQ1.1	The service, capability and skill modelling shall be able to represent all necessary information to exchange the offers and quotations between a manufacturing service provider and a requester.	5	"Service, capability and skill models"

REQ1.1.1	The service, capability and skill modelling shall provide all manufacturing services and information from a manufacturing service provider/requester and their machine, production lines and to manufacture the requested part.	5	"AAS Infrastructure", "Service, capability and skill models"
REQ1.1.1.1	The service, capability and skill modelling shall include standard specification and submodels for the AAS/digital representation of the machine, production lines and requested product.	5	"AAS Infrastructure", "Service, capability and skill models"
REQ1.1.1.2	The service, capability and skill modelling shall include a standard dictionary like ECLASS and IEC 61360 to have a common understanding of the provided/requested services and capabilities.	5	"Service, capability and skill models"
REQ1.1.1.3	The service, capability and skill modelling shall be editable by the manufacturing service provider/requester to adapt/add value to the services they provide/request.	5	"Service, capability and skill models"
REQ1.1.1.4	The service, capability and skill modelling shall be extensible by a consortium to add new submodels required to exchange information between two participants.	5	"Service, capability and skill models"
REQ1.1.1.5	The service, capability and skill modelling can have GUI to visualize the noted	5	"Service, capability and skill models", "User Interface"

	machines, production lines, requested parts, services and capabilities from the user of the MaaS.		
REQ1.1.1.6	The service, capability and skill modelling shall have a secure connection via the data from the machines and production lines to the MaaS platform to provide an update on the production of the requested part.	5	"Service, capability and skill models", "AAS Infrastructure/Shopfloor Connection"
REQ1.1.1.7	The service, capability and skill modelling shall use a common language, such as the I4.0 language, which describes the vocabulary, message structure and interaction protocols.	5	"AAS Infrastructure", "Service, capability and skill models", "MaaS Platform/I4.0 Language"
REQ1.1.1.8	The service, capability and skill modelling can provide the AAS/Digital Representation locally in each factory.	5	"AAS Infrastructure", "Service, capability and skill models"
REQ1.1.1.9	The service, capability and skill modelling shall provide different access levels to connect and edit the AAS/Digital representation.	5	"AAS Infrastructure", "Service, capability and skill models"
REQ1.2	The PDT shall allow the service requester/provider participating in the MaaS network to describe information related to their product, so that the product can be used easily over its full lifecycle.	5	"Product Digital Twin Models"
REQ1.2.1	The PDT shall include the capability, service, and skill (CSS) model of the product that encompasses the	5	"AAS Infrastructure", "Service, capability and skill models", "Product Digital Twin Models"

	standardized AAS models and submodels.		
REQ1.2.2 & REQ1.2.3	The PDT shall include editable models to store information about skills and values related to relevant features like process duration, cost and carbon footprint, to name a few, based on the need of the product.	3	"Product Digital Twin Models"
REQ1.2.4	The PDT shall have a GUI or editable models to specify products based on standards such as ECLASS ID.	3	"Product Digital Twin Models", "User Interface"
REQ1.2.5	The PDT shall provide mechanisms to provide an aggregated view of different information such as BoM, BoP, quality control, to name a few.	3	"Product Digital Twin Models"
REQ1.2.6	The PDT shall include editable models to store information about the different steps used in the manufacturing of a product, in order to create a holistic digital product passport (DPP).	3	"Product Digital Twin Models"
REQ1.2.7	The PDT shall include editable models to aggregate information about the different digital twins (such as system and process models) to track the entire lifecycle of a product.	3	"Product Digital Twin Models"
REQ2.1	Support the creation and manipulation of AASs (API). To ensure real-time communication between field level devices (machines, sensors etc.)	1, 2, 3, 4, 5	"AAS Infrastructure/Communication Gateway"

	and their digital representatives (AAS); standardized communication protocols should be supported.		
REQ2.1.1	Support standardized protocols (OPCUA, MQTT, REST etc.) for linking shopfloor data with corresponding data models (AAS).	1, 2, 3, 4, 5	"AAS Infrastructure/IIoT Infrastructure"
REQ2.1.2	Provision of historical data (IoT data). The system should support information storage and retrieval for historical purposes (i.e. process optimization etc.)	1, 2, 3, 4, 5	"AAS Infrastructure/IIoT Infrastructure"
REQ2.2	Provision of the ability to define data sharing policy. The system should support standardized interfaces to ensure secure data exchange between different systems/companies.	1, 2, 3, 4, 5	"MaaS Platform/Data Sovereignty"
REQ2.2.1	Provision of standardized interfaces for sharing and consuming information. Supply Chain Level Support Tools for impact prediction of disruptive events. For manufacturers, the main objective is to be able to analyse and assess potential risks and costs for their industry.	1, 2, 3, 4, 5	"MaaS Platform/Communication Gateway"
REQ3.1	Impact Prediction Tool must be able to assess the risk of different supply chain disturbances and predict associated impact to the business performance.	1	"Impact Prediction Tool"

REQ3.1.1	Impact prediction tool must be adaptable to different business cases and integrate information regarding its current status.	1	"Impact Prediction Tool"
REQ3.1.1.1	The software provides connectivity with industrial data dynamically and identify the latest status and events during runtime.	1	"Impact Prediction Tool", "AAS Infrastructure/IIoT Infrastructure"
REQ3.1.2	Impact prediction tool must calculate and display the likelihood of specific events and an estimation of its impact in the company's KPIs.	1	"Impact Prediction Tool"
REQ3.1.2.1	There should be a specific list of event types upon which the model must be able to be re-trained based on a company's historical data.	1	"Impact Prediction Tool/Event Handling Module"
REQ3.1.2.2	The tool must provide the ability to select among different events and display the probability of the event happening upon a specific horizon, along with the cost/benefit for the company.	1	"Impact Prediction Tool/Event Handling Module"
REQ3.2	The decision support tool shall be able to present decision support information in terms of manufacturing goals.	1	"Decision Support Tool"
REQNF3.2.1	The decision support tool must provide feedback upon requests within 10 seconds.	1	"Decision Support Tool"
REQ3.2.1	The decision support tool shall have access to historical and state data for its analysis.	1	"Decision Support Tool", "AAS Infrastructure/Communication Gateway", "MaaS

			Platform/Communication Gateway"
REQ3.2.1.1	The decision support tool shall incorporate historical data from the MaaS network for its analysis.	1	"Decision Support Tool", "MaaS Platform/History"
REQ3.2.1.2	The decision support tool shall be able to use data describing the current state of the MaaS network.	1	"Decision Support Tool", "MaaS Platform/Communication Gateway"
REQ3.2.1.3	The decision support tool shall be able to use data describing the current state of the internal production system.	1	"Decision Support Tool", "AAS Infrastructure/Communication Gateway"
REQ3.2.1.4	The decision support tool shall be able to use product digital twin data with embedded manufacturing goal metrics.	1	"Decision Support Tool", "Product Digital Twin Models"
REQ3.2.2	The decision support tool shall be able to use the impact prediction tool with what-if scenarios.	1	"Decision Support Tool"
REQ3.2.2.1	The decision support tool shall be able to send a scenario in the input format of the impact prediction tool.	1	"Decision Support Tool"
REQ3.2.2.2	The decision support tool shall be integrated with the output format of the impact prediction tool.	1	"Decision Support Tool", "Impact Prediction Tool"
REQ3.2.3	The decision support tool shall visualize a trade-off between finding a new supplier and changing production.	1	"Decision Support Tool"
REQ3.2.3.1	The decision support tool shall include a visualization of the temporary make-or-buy analysis with a	1	"Decision Support Tool/User Interface"

	comparison of the manufacturing goals.		
REQ3.2.3.2	The decision support tool shall include an explicit visualization of uncertainty on every estimated manufacturing goal metric.	1	"Decision Support Tool/User Interface"
REQ3.2.4	The decision support tool shall include a user interface to edit manufacturing goal metrics and events.	1	"Decision Support Tool/User Interface"
REQ3.2.4.1	The editor shall allow users to define uncertainty for a metric in terms of a probability distribution.	1	"Decision Support Tool/User Interface"
REQ3.2.4.2	The editor shall allow users to define uncertainty for an event in terms of a probability.	1	"Decision Support Tool/User Interface"
REQ4.1.1.1	During the manufacturing service onboarding to the MaaS, the audit tool shall validate if the service corresponds to the manufacturer's production capabilities.	3	"Trustworthiness Audit Tool"
REQ4.1.1.2	The audit tool shall automatically validate if the offered service can be provided by the manufacturing service provider.	3	"Trustworthiness Audit Tool"
REQ4.1.1.3	The audit tool shall provide a performance score about the potential manufacturing service.	3	"Trustworthiness Audit Tool"
REQ4.1.2.1	The recommendation engine shall generate the supply chain alternatives for the requested manufacturing service.	3	"Recommendation Engine"

REQ4.1.2.2	The recommendation engine shall be able to automatically negotiate with the potential manufacturing service providers.	3	"Recommendation Engine", "MaaS Platform/Communication Gateway"
REQ4.1.2.3	The recommendation engine shall rank the potential manufacturing service providers based on their production capabilities and the requester's goals.	3	"Recommendation Engine", "Trustworthiness Audit Tool"
REQ4.1.2.4	The recommendation engine shall provide all relevant information about the offered service, such as CO2 footprint, environmental and health impact.	3	"Recommendation Engine", "Product Digital Twin Models"
REQ4.1.2.5	The recommendation engine shall provide the information about the manufacturing services in compliance with the common information model.	3	"Recommendation Engine", "Service, capability and skill models"
REQ5.1	The capability matching tool shall employ a semantic framework that integrates with the broader RAASCEMAN information model, supporting standards such as Asset Administration Shells (AAS) and OPC-UA as well as the use of standard dictionaries for consistent capability descriptions.	2	"AAS Infrastructure", "Capability Matching Engine"
REQ5.1.1	The capability matching tool shall represent all relevant service, capability, and skill data of resources in a GraphDB-based	2	"Capability Matching Engine", "Service, capability and skill models"

	structure. It shall enable real-time querying based on capability requirements specified by manufacturing service requesters.		
REQ5.1.2	The capability matching tool shall support manufacturing systems (MES, ERP) to reflect changes in machine states, tool availability, and personnel scheduling.	2	"Capability Matching Engine"
REQ5.1.3	The tool shall integrate with dynamic planning and scheduling systems from sections 5.3.2 and 5.3.3. It will support the task planning and execution tools by exporting capability-matching results for operational readiness.	2	"Capability Matching Engine", "Dynamic Planning & Scheduling"
REQ5.1.4	The user interface shall accept input in natural language, processed by an LLM for compatibility and ease of use. It will provide actionable recommendations for resource-task matching and highlight resource unavailability.	2	"Capability Matching Engine"
REQ5.2	The dynamic planning and scheduling tool shall be triggered if unforeseen or planned events occur to adapt the current production plan.	4	"Dynamic Planning & Scheduling"
REQ5.2.1	The dynamic planning and scheduling tool shall have an interface to communicate to intra- and inter-factory components to provide meaningful production plans.	4	"Dynamic Planning & Scheduling", "AAS Infrastructure/Communication Gateway", "MaaS Platform/Communication Gateway"

REQ5.2.1.1	The dynamic planning and scheduling tool shall provide different executable plans that can be easily applied in the production procedures.	4	"Dynamic Planning & Scheduling"
REQ5.2.1.2	The dynamic planning and scheduling tool shall notify connected services about the current progress of the planning and scheduling procedure and already available results.	4	"Dynamic Planning & Scheduling"
REQ5.2.1.3	The dynamic planning and scheduling tool shall be able to provide adapted production plans in real-time such that it can be used dynamically.	4	"Dynamic Planning & Scheduling"
REQ5.3	The dynamic execution of tasks on the shopfloor shall be able to react to unforeseen events and change the production equipment on the shopfloor in a short time to execute a new process.	4	"Dynamic Execution Engine"
REQ5.3.1	The dynamic execution of tasks on the shopfloor shall have a software interface to change the production with a small number of parameters.	4	"Dynamic Execution Engine"
REQ5.3.1.1	The dynamic execution of tasks on the shopfloor shall have a parameterizable software interface to prepare the execution of production changes for production lines and machines.	4	"Dynamic Execution Engine"
REQ5.3.1.2	The dynamic execution of tasks on the shopfloor shall be able to trigger the	4	"Dynamic Execution Engine"

	execution of production by the manufacturing service provider.		
REQ5.3.1.3	The dynamic execution of tasks on the shopfloor shall provide information on the duration and scope of production for the manufacturing service provider.	4	"Dynamic Execution Engine"
REQ5.3.1.4	The dynamic execution of tasks on the shopfloor shall notify the manufacturing service provider that the production is ready to produce the new order.	4	"Dynamic Execution Engine"

For the “Requirements Traceability Matrices” the table below shows a component-centric view and also maps the components to functional layers.

Table 2 Component-centric RTMs

Main Component	**Subcomponent**	**Fulfilled Requirements**	**Functional Layer**
AAS Infrastructure		['REQ1.1.1', 'REQ1.1.1.1', 'REQ1.1.1.7', 'REQ1.1.1.8', 'REQ1.1.1.9', 'REQ1.2.1', 'REQ5.1']	Infrastructure Layer
	Communication Gateway	['REQ2.1', 'REQ3.2.1', 'REQ3.2.1.3', 'REQ5.2.1']	Integration Layer
	IIoT Infrastructure	['REQ2.1.1', 'REQ2.1.2', 'REQ3.1.1.1']	Infrastructure Layer
	Shopfloor Connection	['REQ1.1.1.6']	Infrastructure Layer
Capability Matching Engine		['REQ5.1', 'REQ5.1.1', 'REQ5.1.2', 'REQ5.1.3', 'REQ5.1.4']	Business Logic Layer

Decision Support Tool		['REQ3.2', 'REQNF3.2.1', 'REQ3.2.1', 'REQ3.2.1.1', 'REQ3.2.1.2', 'REQ3.2.1.3', 'REQ3.2.1.4', 'REQ3.2.2', 'REQ3.2.2.1', 'REQ3.2.2.2', 'REQ3.2.3']	Business Logic Layer
	User Interface	['REQ3.2.3.1', 'REQ3.2.3.2', 'REQ3.2.4', 'REQ3.2.4.1', 'REQ3.2.4.2']	Business Logic Layer
Dynamic Planning & Scheduling		['REQ5.1.3', 'REQ5.2', 'REQ5.2.1', 'REQ5.2.1.1', 'REQ5.2.1.2', 'REQ5.2.1.3']	Business Logic Layer
Impact Prediction Tool		['REQ3.1', 'REQ3.1.1', 'REQ3.1.1.1', 'REQ3.1.2', 'REQ3.2.2.2']	Business Logic Layer
	Event Handling Module	['REQ3.1.2.1', 'REQ3.1.2.2']	Business Logic Layer
MaaS Platform			Infrastructure Layer
	Communication Gateway	['REQ2.2.1', 'REQ3.2.1', 'REQ3.2.1.2', 'REQ4.1.2.2', 'REQ5.2.1']	Integration Layer
	Data Sovereignty	['REQ2.2']	Security Layer
	History	['REQ3.2.1.1']	Persistence Layer
Product Digital Twin Models	I4.0 Language	['REQ1.1.1.7']	Information Layer
		['REQ1.2', 'REQ1.2.1', 'REQ1.2.2', 'REQ1.2.3', 'REQ1.2.4',	Information Layer

		'REQ1.2.5', 'REQ1.2.6', 'REQ1.2.7', 'REQ3.2.1.4', 'REQ4.1.2.4']	
Recommendation Engine		['REQ4.1.2.1', 'REQ4.1.2.2', 'REQ4.1.2.3', 'REQ4.1.2.4', 'REQ4.1.2.5']	Business Logic Layer
Service, capability and skill models		['REQ1.1', 'REQ1.1.1', 'REQ1.1.1.1', 'REQ1.1.1.2', 'REQ1.1.1.3', 'REQ1.1.1.4', 'REQ1.1.1.5', 'REQ1.1.1.6', 'REQ1.1.1.7', 'REQ1.1.1.8', 'REQ1.1.1.9', 'REQ1.2.1', 'REQ4.1.2.5', 'REQ5.1.1']	Information Layer
Dynamic Execution Engine		['REQ5.3', 'REQ5.3.1', 'REQ5.3.1.1', 'REQ5.3.1.2', 'REQ5.3.1.3', 'REQ5.3.1.4']	Business Logic Layer
Trustworthiness Audit Tool		['REQ4.1.1.1', 'REQ4.1.1.2', 'REQ4.1.1.3', 'REQ4.1.2.3']	Business Logic Layer
User Interface		['REQ1.1.1.5', 'REQ1.2.4']	Presentation Layer

3 SotA Analysis & Technology Assessment

From Table 2 we can derive the following functional layer groups (main components):

- **Business Logic Layer** — 8 components
- **Infrastructure Layer** — 4 components
- **Integration Layer** — 2 components
- **Information Layer** — 3 components
- **Security Layer** — 1 component
- **Presentation Layer** — 2 components
- **Persistence Layer** — 1 component

Out of these, the infrastructure layer which compose the biggest part of the architecture would require a State-of-the-art analysis, along with the security aspects. The Integration and Information Layers would also need SotA with focus on Interoperability (Semantic & Technical). Business layer components are functional specific that require their own approach and the same applies for presentation and persistence layers, consequently no SotA is needed at this stage. On each component development tasks, SotA might be provided depending on the nature of the implementation.

3.1 Generic Architectures (RAMI, IIoT, IIRA)

The Fourth Industrial Revolution led to the implementation of IoT technologies into manufacturing systems. To address this development while also maintaining interoperability, several architectural models have emerged, each of which refers to different aspects of industrial IoT systems. Some of the most prominent architectures in the context of Industry 4.0 are explored below, including RAMI4.0, IIRA and other proposed IIoT frameworks.

3.1.1 RAMI

RAMI 4.0 (Reference Architectural Model Industry 4.0) constitutes a service-oriented framework that establishes hierarchical levels in order to classify the digitalization of industrial elements [8]. It is based on the Smart Grid Architecture Model (SGAM) and extends it, aiming to be aligned with the needs of Industry 4.0 , serving as a foundational reference architecture for this kind of systems [7]. This approach reinforces the decomposition of complex interconnections into simpler and easier to handle components [6].

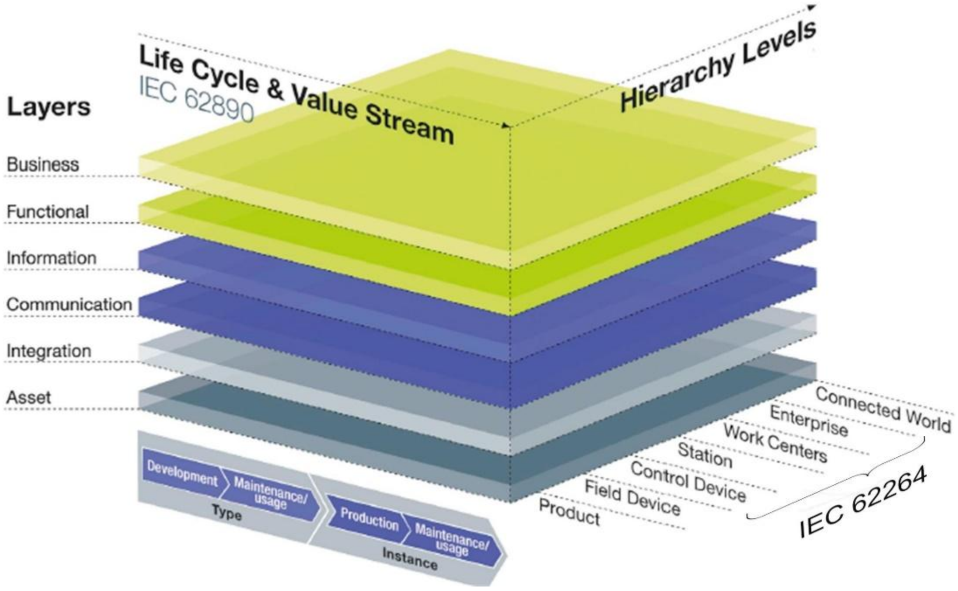


Figure 2: RAMI 4.0 reference architecture [7]

The levels that define RAMI 4.0 are:

- Hierarchy Levels

  The right horizontal axis of RAMI 4.0 is structured upon four hierarchy levels (Enterprise, Work Centers, Station, Control Device) defined by IEC 62264, a global standard for enterprise IT and control systems. The purpose of these levels is the description of various functional aspects of manufacturing facilities [6]. Extending this approach, RAMI 4.0 introduces three additional layers in order to support the concept of a smart factory. Specifically, the 'Field Device' layer addresses the intelligent control of machines or systems, the 'Product' layer refers to the standardization of the manufactured product and the 'Connected World' layer is responsible for the cross-company collaboration [7].

- Life Cycle & Value Stream

  The left horizontal axis demonstrates the life cycle of entities like products and facilities [7], complying with the IEC 62890 standard for life cycle management. Apart from this, a differentiation is established between "types" and "instances". A "type" serves as a template of an entity, which transitions into an "instance" during the production phase [6].

- Layers

The six layers along the vertical axis aim to depict the breakdown of a machine into its individual attributes [6]. Each layer defines a different aspect, like data structures and communication behavior [7].

3.1.2 Industrial Internet Reference Architecture

Industrial Internet Reference Architecture is a reference architecture developed by IIC intending to support IIoT systems. Its objective is to reinforce industrial interoperability and establishment of standards and technologies [10]. Its architecture is divided into three tiers: the Edge Tier, the Platform Tier and the Enterprise Tier. Within the Edge Tier, the various devices and sensors participating in the system are connected to the Edge Gateway through wired or wireless networks forming the Proximity Network. The role of the Edge Gateway is to manage these devices and pass their data to the Platform Tier through the Access Network. On the Platform Tier the data is processed and analyzed in order to be sent to the Enterprise Tier. On the Enterprise Tier, the user monitors the operations and provides the appropriate commands. These commands are transferred to the Platform Tier and then to the Edge Tier in order to trigger the appropriate actions [11].

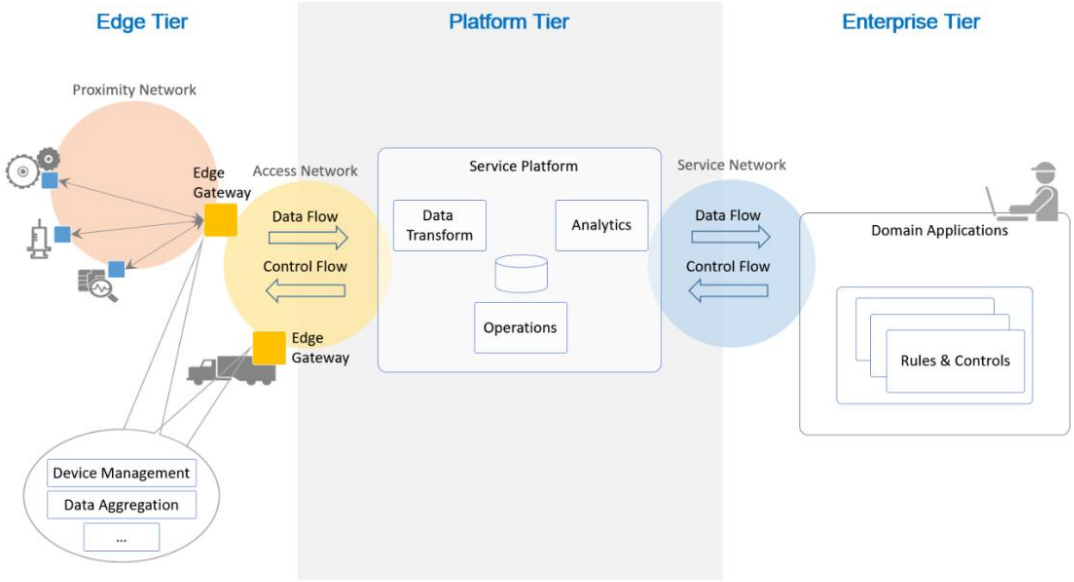


Figure 3: IIRA architecture [11]

IIRA defines four viewpoints to address the key aspects of an IIoT system

- Business
  The “Business” viewpoint defines the purpose of implementing an IIoT system by identifying the stakeholders and mapping them with the system’s capabilities [10] [12].

- Usage
  The “Usage” viewpoint focuses on the utilization of the system components in order to fulfill the defined capabilities [10] [12].

- Functional
  The “Functional” viewpoint outlines the system’s structure through the identification of its key components and the way they interact with each other and with external systems. It is further divided into five domains [10] [12].

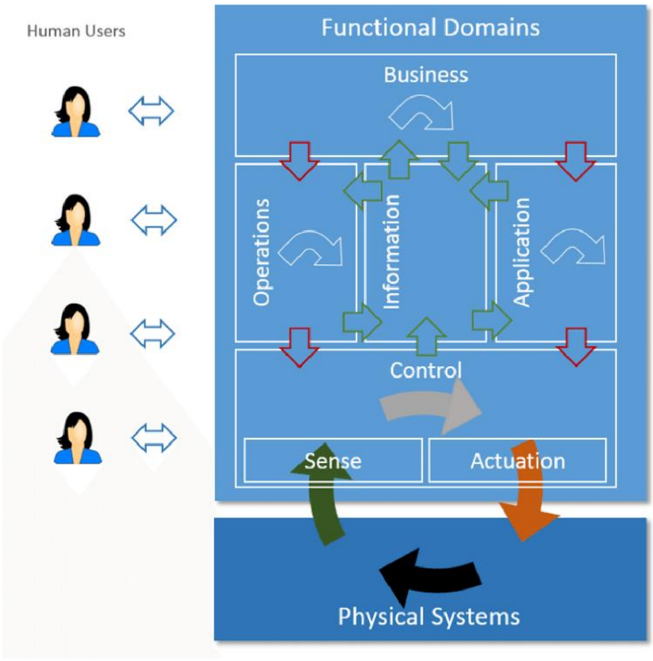


Figure 4 : IIRA domains [11]

  - Control domain
    The “Control” domain provides sensing and actuation functionalities, accomplishing the control of an industrial system. Additionally, it supports communication-related functions that facilitate data exchange between components using various technologies, like APIs [11] [13].

  - Operation domain
    The “Operation” domain is closely related to the “Control” domain. Specifically, it supports provisioning and deployment functions that enable remote access and lifecycle management of the asset [11] [13].

  - Information domain
    The "Information" domain is responsible for transformation, modeling and processing of data from system components. In this way, it enhances optimization based on informed decision-making [11] [13].

  - Application domain
    The "Application" domain includes functions for management and monitoring through application logic and rules. It also enables interaction with relevant information through the use of APIs and user interfaces [11] [13].

  - Business domain
    The "Business" domain describes various functionalities associated with business activities and processes. These functionalities include ERP, MES and Payments [11] [13].

- Implementation
  The "Implementation" viewpoint plays a vital role in identifying the necessary technologies for implementing functional components and defining the appropriate communication frameworks of IIoT systems [10][12].

3.1.3 Industrial Internet of Things

While a plethora of IoT definitions exist, those associated with industrial use emphasize the integration of smart technologies into conventional objects in order to function as IoT devices [16]. In this context [17] provides a relevant definition for IoT:

"The IoT represents a scenario in which every object or 'thing' is embedded with a sensor and is capable of automatically communicating its state with other objects and automated systems within the environment. Each object represents a node in a virtual network, continuously transmitting a large volume of data about itself and its surroundings..."Based on these considerations a preliminary definition of IIoT could be:

- The application of IoT technologies within industrial environments [16].

Regardless of the indisputable significance of IIoT in Industry 4.0, a unified architecture has not been established. However, there have been various attempts to develop IIoT architectures. Two of these approaches are presented below.

In [14], an IIoT architecture is proposed based on the integration of various layers and components including physical components, communication methods, data aggregation, data storage and analysis, and user interface.

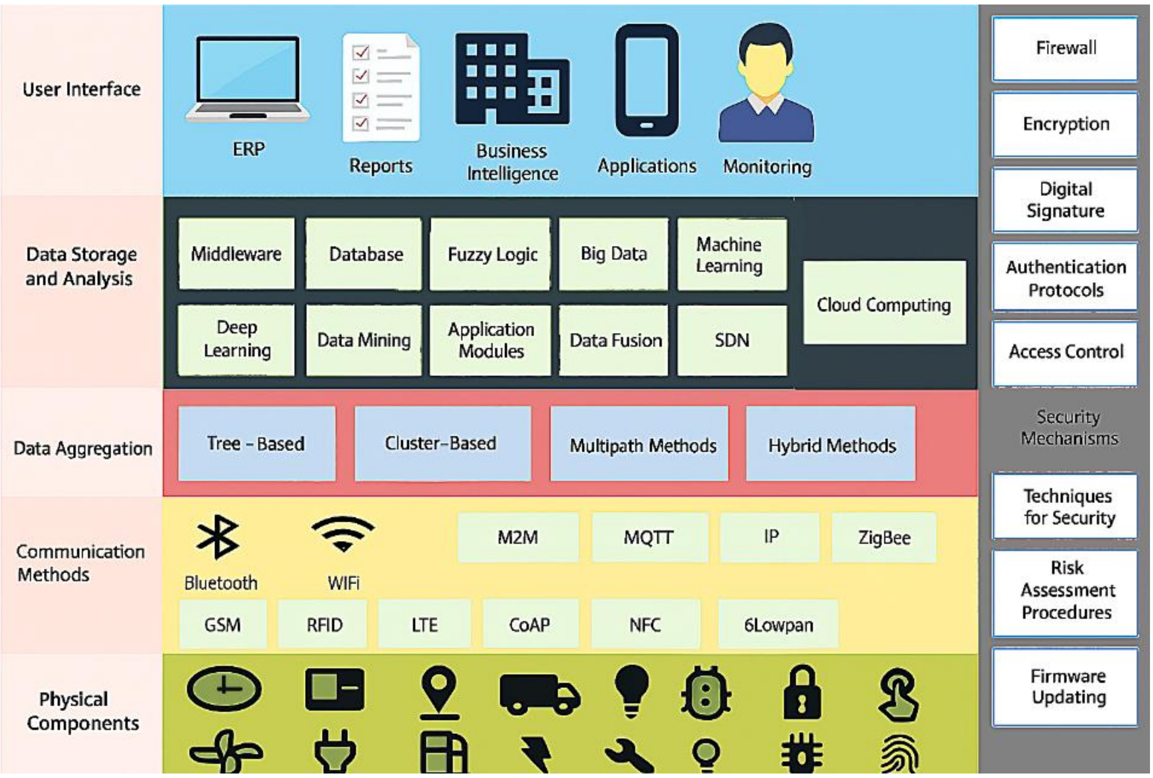


Figure 5: Proposed IIoT architecture [14]

Physical components

Physical components represent all the physical entities within the manufacturing system, such as machines, sensors and actuators.

Communication Methods

Wireless protocols play a pivotal role in data exchange procedures between network elements and must meet several requirements, like low energy consumption and high throughput.

Data aggregation methods

Data aggregation is correlated with the collection of various different packets and the generation of a single output packet. In this way, the network's lifespan is extended and the reduction of energy use is achieved [18]. The key data aggregation types include centralized, in-network, tree-based and cluster-based approaches [19].

Data Storage and analysis

The secure software and hardware infrastructure establishes the cloud as the primary data storage component used in IIoT systems, offering reliability and scalability. Data stored in the cloud is available for further processing using AI, data mining algorithms and machine learning algorithms. This process ensures the minimization of resource consumption, the enhancement of quality of service and the implementation of automated support.

User Interface

The interfaces' compatibility should be extended to support various applications and hardware platforms. Through the use of these interfaces, IIoT reinforces industrial networks by enabling remote control of the system.

Security Mechanisms

Security plays a vital role in manufacturing systems due to the sensitive nature of industrial data. Addressing this issue, several security mechanisms are implemented through various methods, such as protocols, encryption techniques and firewalls.

In [15], the proposed architecture is centered around edge computing across various IIoT cases, aiming to minimize network traffic and decision-making delay. This architecture is based on three primary layers: the Device layer, the Edge layer and the Cloud Application layer.

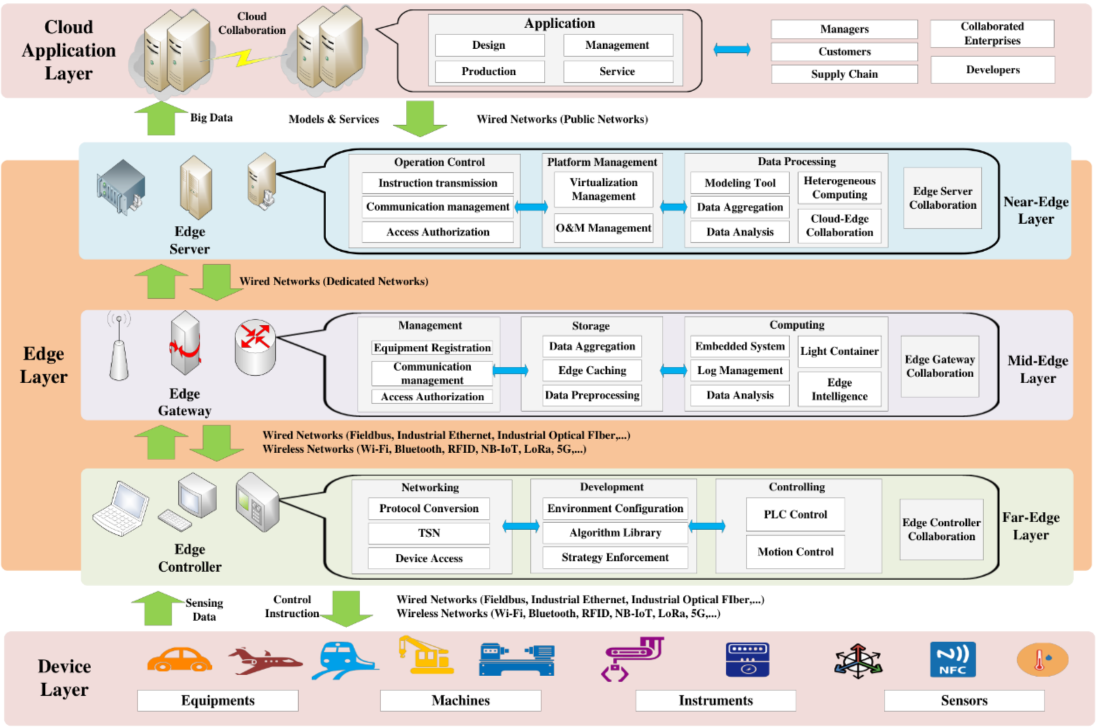


Figure 6: Proposed IIoT architecture [15]

Device Layer

The Device layer involves all the physical assets of the manufacturing system like sensors, machines, vehicles and robots. These assets gather parameter-related information via the use of sensors and transmit it to the Edge layer. For this process, wired communication technologies, such as Fieldbus and Industrial Optical Fiber, and wireless networks, including Wi-Fi and Bluetooth, are used.

Edge Layer

The Edge Layer focuses on receiving and processing data from the Edge Layer. It supports time-sensitive functions such as edge security and privacy protection, data analytics, process optimization and real-time control.

Cloud Application Layer

The Cloud Application layer obtains valuable insights from large-scale data regarding resource distribution. It receives data from the Edge layer through public networks and provides models and microservices as feedback to the Edge layer for further execution.

3.2 Security (Intra & Cross Company)

3.2.1 Intra-company security

3.2.1.1 Authentication & Authorization

- Authorization refers to the procedure of deciding the level of access a user or a device has to certain resources. For instance, it determines whether an entity is permitted to read or modify data, execute programs and control actuators [21].
- Authentication specifies the verification of an entity and is a necessary step before authorization [21].

3.2.1.2 Authentication & Authentication mechanisms

3.2.1.2.1 ABAC

The Attribute-Based Access Control (ABAC) leverages attributes related to user, subject and environment in order to generate an access token. This token is evaluated against several access policies stored locally or accessed remotely. The ABAC model provides flexibility regarding the creation of access tokens as they are formed from a wide variety of attributes, unlike the RBAC model described in the next section, which is limited by a fixed set of predefined roles and token types, making it ideal for heterogeneous systems [23].

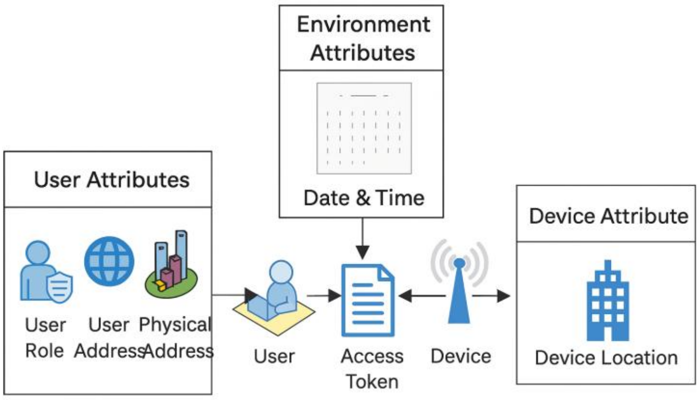


Figure 7: ABAC model [23]

Although ABAC offers a reliable access control mechanism, its design and implementation cause significant challenges [23].

- ABAC performs intensive computational processes excluding, in this way, resource-constrained devices.
- While the combination of a wide range of attributes for the generation of the access token reinforces its implementation to heterogenous systems, it also creates conflicts among access policies.
- The access token contains sensitive information about the user and the subject. Therefore, this data should be protected otherwise the trust of the model might be undermined [22].

3.2.1.2.2 RBAC

The Role-Based Access Control model (RBAC) defines access control rules based on user responsibilities, privileges and administrative functions, while abstracting the user's underlying tasks [22]. In RBAC, the access decisions are based on the user's role instead of its identity. As presented in the figure below, every user is associated with at least one role and every role is correlated with a set of operations that determines access permissions. Within this approach the system security administrator is responsible for assigning these roles and their corresponding permissions, while it is possible to reassign them without the need to revoke the user's access entirely. The accessor in RBAC model, similar to the CapBAC model described later, generates a token that encapsulates access-related data. This token is evaluated from the targeted device regarding its alignment with policies defined by the administrator in order to determine whether access should be granted or not [23].

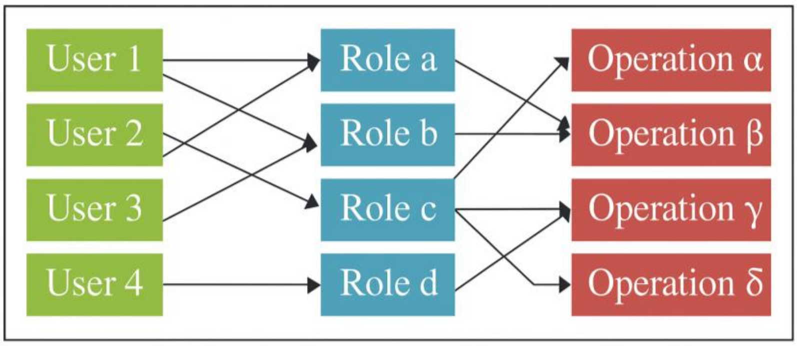


Figure 8: RBAC model [23]

Despite the significant advantages of the RBAC model, its implementation to general-purpose systems reveals several challenges [23].

- Establishing a concise set of roles for a wide range of heterogenous devices causes role proliferation across multiple domains [22].
- Within large-scale environments it becomes challenging to map roles to specific operations.
- The existence of a centralized administrator in systems consisting of various subsystems is impractical.

3.2.1.2.3 OAuth

OAuth is an authorization protocol designed to enable secure access delegation. It supports access to server-hosted resources on behalf of a resource owner, without requesting the owner to share his credentials. This framework allows third-party applications to gain limited access to HTTP services either through the approval of the resource owner, or by acting on their own behalf. The resource owner delegates distinct access rights, while maintaining granular control over the access of his private resources. The resource server hosts the protected assets and responds to requests authenticated by access tokens. These tokens are populated by an authorization server after the resource owner's identity and consent have been successfully verified [20]

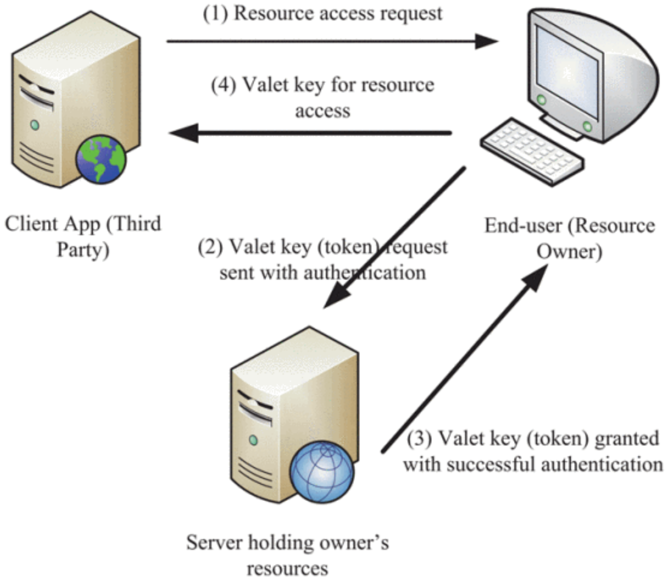

(1) Resource access request
(4) Valet key for resource access
Client App (Third Party)
End-user (Resource Owner)
(2) Valet key (token) request sent with authentication
(3) Valet key (token) granted with successful authentication
Server holding owner's resources

Figure 9: OAuth protocol [29]

3.2.1.2.4 CapBAC

Capability-Based Access Control (Cap-BAC) is based on the Access Control Matrix (ACM) model but distinguishes itself from ACM by adopting a row-oriented approach, where each subject is correlated with one or more object-rights pairs referred to as capabilities. This list of capabilities is managed by the accessor rather than the resource. The accessed resource does not contain any information about access policy; instead, it enforces access control by validating the access list presented by the accessor. Specifically, the accessor generates a token that encapsulates access rights-related information and submits it to the targeted resource. The device then evaluates the access request based on the access control policy and the data provided by the token [23].

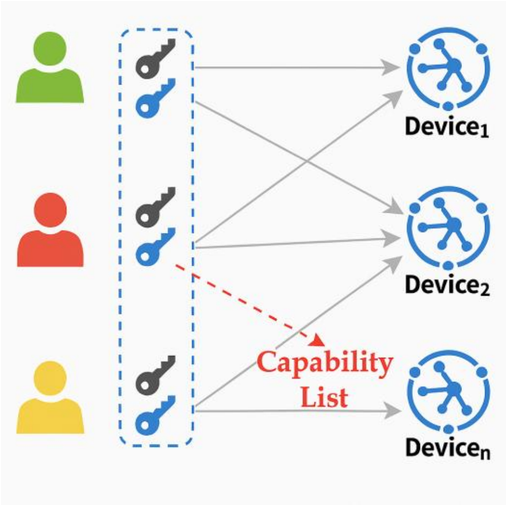


Figure 10: capBAC model [28]

3.2.1.2.5 orBAC

orBAC extends the RBAC model by introducing an additional "organization" dimension [22]. This model intends to address the complexity of security policies through the integration of two abstraction layers: the abstract layer and the concrete layer. The basis of orBAC is the establishment of relationships between roles, activities and views to subjects, actions and objects respectively. Unlike other approaches that rely on two binary relations, the first links roles to organizations and the second connects subjects to roles. The orBAC model employs a ternary relation aiming to correlate subject role directly to an organization [27].

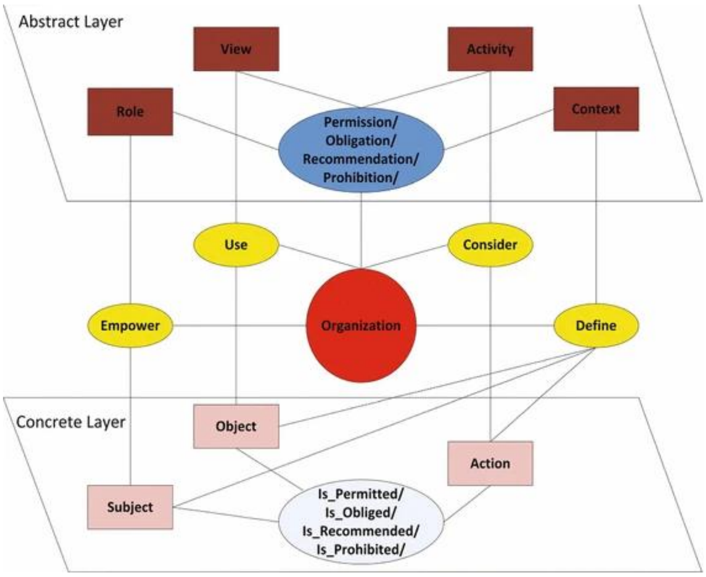

Abstract Layer
View
Activity
Role
Context
Permission/ Obligation/ Recommendation/ Prohibition/
Use
Consider
Empower
Organization
Define
Concrete Layer
Object
Action
Subject
Is_Permitted/ Is_Obliged/ Is_Recommended/ Is_Prohibited/

Figure 11: orBAC model

- The Abstract Layer enables the definition of security policies based on abstract entities, independent of their implementation within each organization.
- The Concrete Layer is responsible for assigning privileges to subjects based on the subject's role, the requested action, the targeted object and the context.

3.2.1.2.6 UCON

The UCON model offers a dynamic solution regarding access control, allowing permissions to be revoked and usage to be terminated. This approach is ideal for distributed and heterogeneous systems. Within the policy model layer, several conditions are defined based on features of objects and subjects, system attributes and required user actions [22].

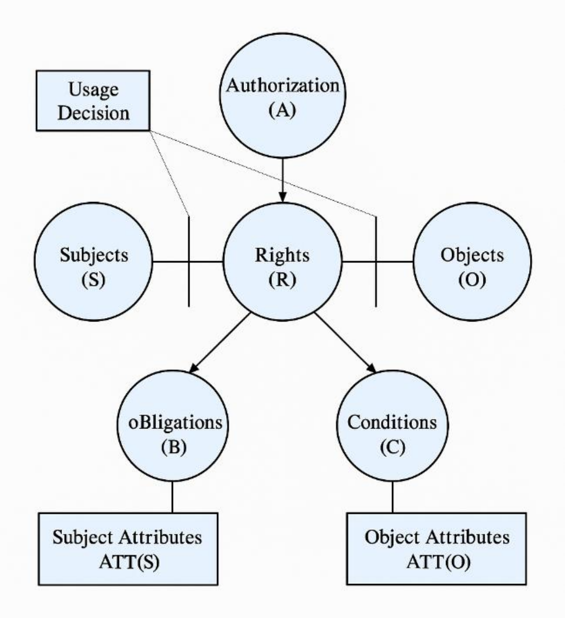


Figure 12: UCON model [26]

3.2.1.2.7 ReBAC

Relationship-Based Access Control (ReBAC) introduces the concept of a binary relationship manager, differentiating itself from the traditional approach based on identity, role or attribute unary predicates. Instead, ReBAC establishes a relationship between the accessor and the asset. As shown in the figure below, access is granted only if the accessor has a friendship connection with the owner of the asset and the action is aligned with the appropriate access policy. Current applications of ReBAC primary found in the domain of social media applications, however it is believed to expand into more general-purpose access control mechanisms in the future [23].

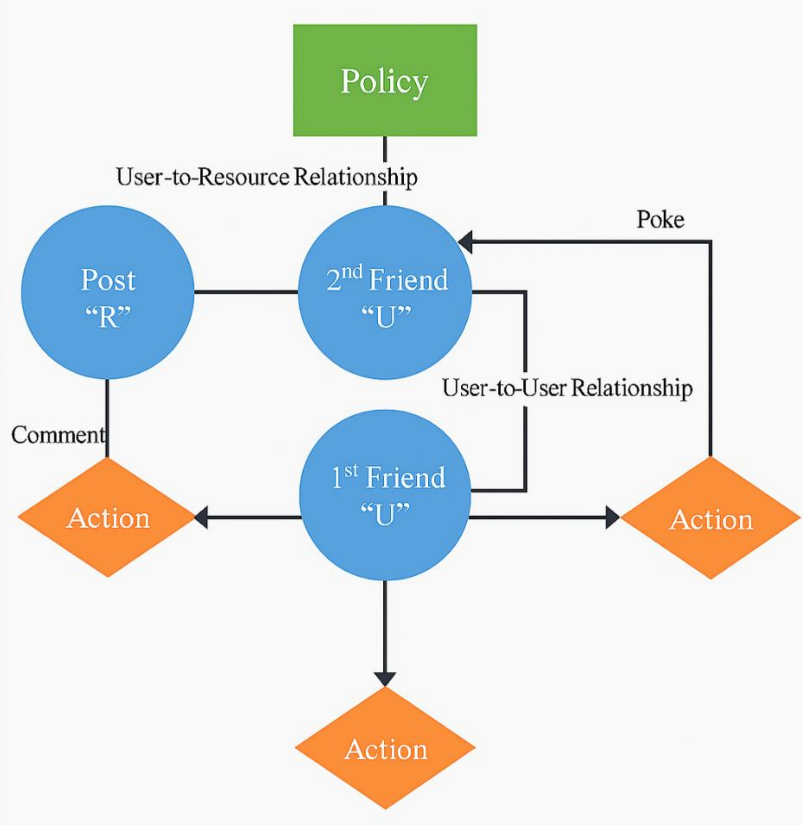


Figure 13: ReBAC model

3.2.2 Cross-company security

3.2.2.1 Federated Identity Management

Federated Identity Management is an approach that supports collaboration between entities such as identity providers and service providers on identity, policies and technologies related issues. It allows users to access shared resources seamlessly. The users are managed by the identity provider, which acts as an authoritative source of identity.

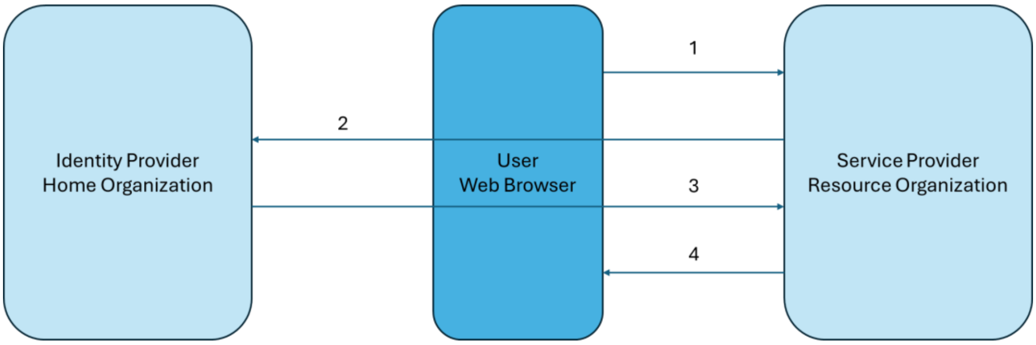


Figure 14: Federated Identity Management [48]

Specifically, the identity provider populates authentication tokens to service providers and then they provide their services to the requestor [48].

3.2.2.1.1 Identity Federation Architectures

3.2.2.1.1.1 SAML

SAML is an XML-based infrastructure intended to describe and share information between organizations regarding security. This approach aims to provide a vendor-independent solution for achieving Single Sign-On and identity federation functionalities across different domains. SAML consists of the following building blocks: Assertions, Protocols, Bindings, Profiles, Metadata and Authentication Context [49].

- "Assertions" defines the format of the exchanged security-related information between the SAML participants. The information that is encompassed in this block is related to the subject of assertion, the validation of the assertion and the statements about the subject.

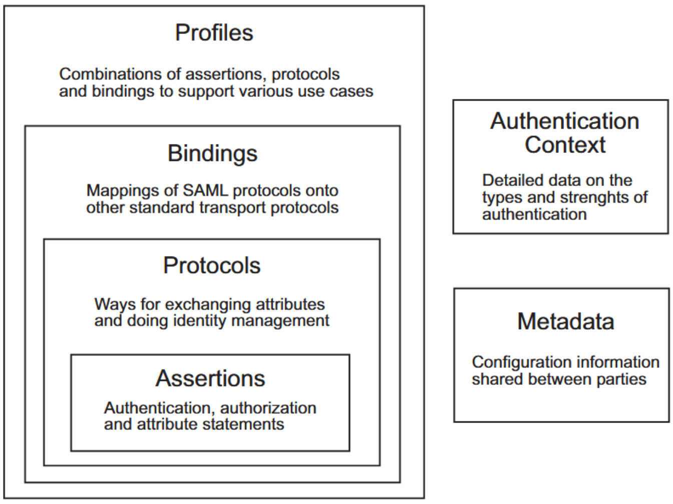

Profiles
Combinations of assertions, protocols and bindings to support various use cases
Bindings
Mappings of SAML protocols onto other standard transport protocols
Protocols
Ways for exchanging attributes and doing identity management
Assertions
Authentication, authorization and attribute statements
Authentication Context
Detailed data on the types and strenghts of authentication
Metadata
Configuration information shared between parties

Figure 15: SAML components [49]

- "Protocols" establishes the mechanisms for exchanging assertions and other necessary information to carry out the operations enabled by the SAML framework.
- "Bindings" outlines the way that the SAML protocols are applied over different transport protocols.
- "Profiles" is responsible for appropriately linking assertions, protocols and bindings depending on a specific usage scenario.
- "Metadata" is correlated with the sharing of configuration information between the participants.
- Authentication Context is related to detailed information of the authentication process of a subject, including its method and its strength.

3.2.2.1.1.2 Liberty Alliance Framework

Liberty Alliance is a consortium focusing on open standards for federated identity management and identity-based web services. In this context, the Liberty Alliance Framework is based on standardized technologies, including XML, SOAP and SAML. Its architecture involves three key components: the Identity Federation Framework, the Identity Web Services Framework and the Identity Interface Specifications [49].

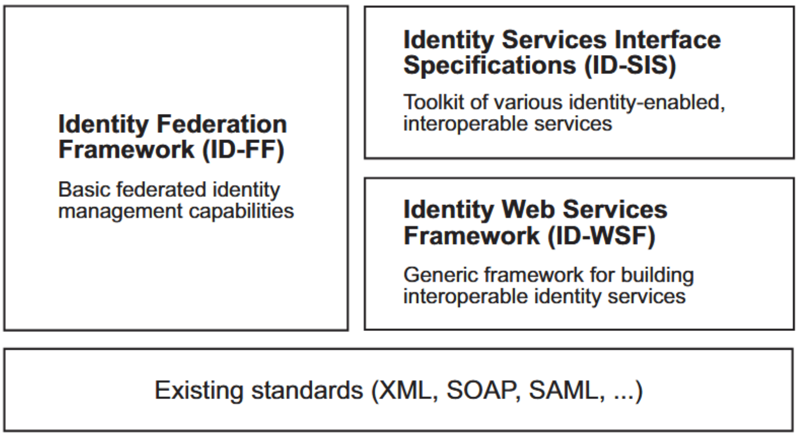

Identity Federation Framework (ID-FF)
Basic federated identity management capabilities
Identity Services Interface Specifications (ID-SIS)
Toolkit of various identity-enabled, interoperable services
Identity Web Services Framework (ID-WSF)
Generic framework for building interoperable identity services
Existing standards (XML, SOAP, SAML, ...)

Figure 16: Liberty Alliance Framework components [49]

- Identity federation framework: This component provides identity federation services between the participants that form the so-called circle of trust. The circle of trust consists of entities that adopt Liberty-aligned technologies and bound by mutual trust established through operational agreements.
- Identity web service framework: This component is built upon the capabilities of the identity federation framework component in order to create, use and consume identity services. These services are designed to retrieve or update identity-related information.
- Identity services interface specification: This component utilizes the identity federation framework component and the identity web service components to establish practical identity-enabled web services.

3.2.2.2 Data spaces

A data space is a technology that emerged to tackle the challenge of collaborative data ecosystems in both enterprises and academic organizations. They enable data sharing and exchange among shareholders effortlessly, providing significant opportunities particularly when integrated with data analytics tools, including enhanced decision-making and insights extraction. In its core, data spaces constitute a collection of heterogenous data sources, enabling efficient access, management and analysis. This concept highlights the need for a shared information environment rather than individual data points [50]. Unlike the traditional centralized data platforms where the control is distributed among a few entities, data spaces pave the way for data sovereignty and interoperability [51][52].

3.2.2.2.1 Data space technology initiatives

Data spaces are closely aligned with the European data strategy, which has led to the development of Common European Data Spaces. Its purpose is the reassurance of secure and reliable data exchange across the EU. The application of these data spaces contributes to a wide range of significant sectors including energy, agriculture and health [53]. The leading initiatives regarding data spaces are Gaia-X and the International Data Spaces Association (IDSA), both playing a pivotal role in establishing architectures, standards and governance models. The collaboration and alignment between these initiatives have caused international interest in the field. Alongside them, open-source ecosystems such as FIWARE provide critical components to the data space infrastructure, including standardized data models [52]. However, the vast increase of data spaces initiatives generates a continuously more complex and challenging to track technical landscape [54].

3.2.2.2.1.1 IDSA

The International Data Space Association (IDSA) constitutes a non-profit organization aiming to provide global standards for secure and sovereign data sharing. This approach targets to establish trustworthy ecosystems, where between its participants, data/service providers and consumers, there will be comprehensive and mutually accepted policies over data sharing. In this context, IDSA introduced the International Data Space Reference Architecture Model (IDS-RAM), which serves as the base for developing data spaces [55]. The IDS-RAM presented below highlights the interaction between its components and clarifies the necessary functionalities to create a secure network. Specifically, it is organized into three primary dimensions: security, certification and governance that are structured across five architectural layers: business, functional, informational, process and system.

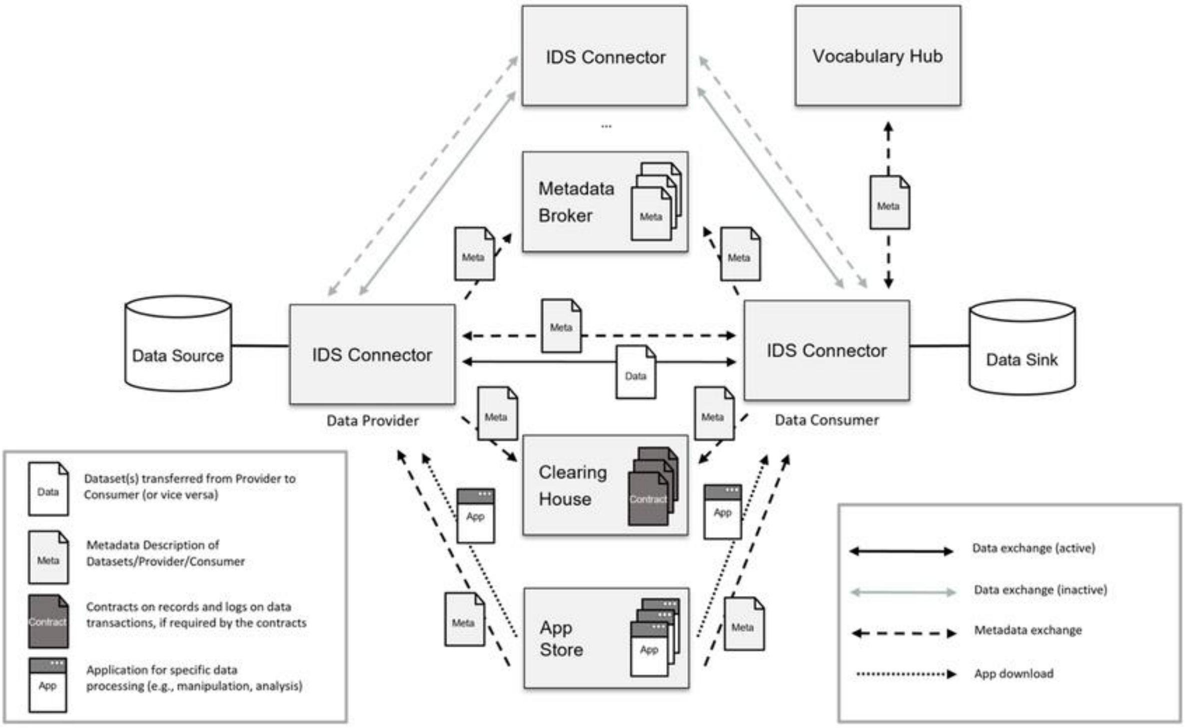

IDS Connector
...
Vocabulary Hub
Metadata Broker
Meta
Data Source
IDS Connector
Data Provider
IDS Connector
Data Consumer
Data Sink
Data
Clearing House
Contract
App
App Store
Data
Dataset(s) transferred from Provider to Consumer (or vice versa)
Meta
Metadata Description of Datasets/Provider/Consumer
Contract
Contracts on records and logs on data transactions, if required by the contracts
App
Application for specific data processing (e.g., manipulation, analysis)
Data exchange (active)
Data exchange (inactive)
Metadata exchange
App download

Figure 17: IDSA reference architecture [56]

The cornerstone of this approach is the IDS Connector component, which is responsible for facilitating data and metadata exchange. Additionally, third-party applications can be downloaded from the App Store component and executed directly within the IDS connector environment. Following a successful negotiation procedure, handled by the Clearing House component, data can be securely shared between the provider and the consumer [56]. The main focus of IDSA lies on the assurance of data sovereignty and the retainment of complete control over the way that data are accessed and used, paving the way for the rise of innovative cross-organizational services [50].

3.2.2.2.1.2 EDC

Eclipse Dataspace Components (EDC) is a modular framework for data spaces implementation, aligned with the International Data Spaces standards. It includes several significant components such as Connector, Federated Catalog, Identity Hub and Registration Service. The functionalities of these components are accessible through reusable standard APIs, enhancing customized integrations in this manner. Similar to the IDS-RAM the Connector component constitutes the key element of this approach. It facilitates secure and policy-compliant inter-organizational data exchange, encompassing data querying, sharing and policy enforcement attributes. EDC-based data spaces offer the possibility of transactions even between participants with diverse levels of trust, including market competitors [50]. In order to ensure compatibility with both Gaia-X and IDSA standards, EDC is designed in alignment with the Gaia-X architectural principles, while it also supports IDS-based messages and policy mechanisms. Except from the support of IDSA compliant components, like Metadata Broker and Dynamic Attribute Provisioning Service, it also explores decentralized solutions such as identity management via Decentralized Identifiers and Federated Catalogs. Federated Catalogs facilitate publishing and detecting contract offers while the Identity Hub provides the identity information of the participants and the Registration Service acts as a registry of the participants [54].

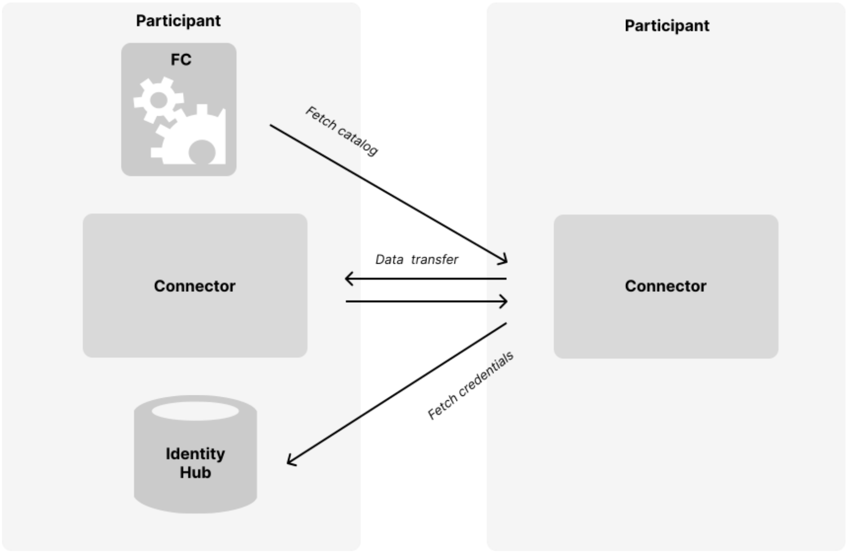


Figure 18: The Dataspace Context [59]

3.2.2.2.1.3 FIWARE

FIWARE is an open-source initiative providing a sustainable and interoperable software ecosystem, which supports the development of smart solutions and data spaces. Its application extends to a wide variety of digital transformation sectors. To achieve this, it provides a set of configurable software components (building blocks) that can be adapted to the requirements of each application [50]. The FIWARE reference architecture enables the integration with other platforms, facilitating the form of smart systems [57].

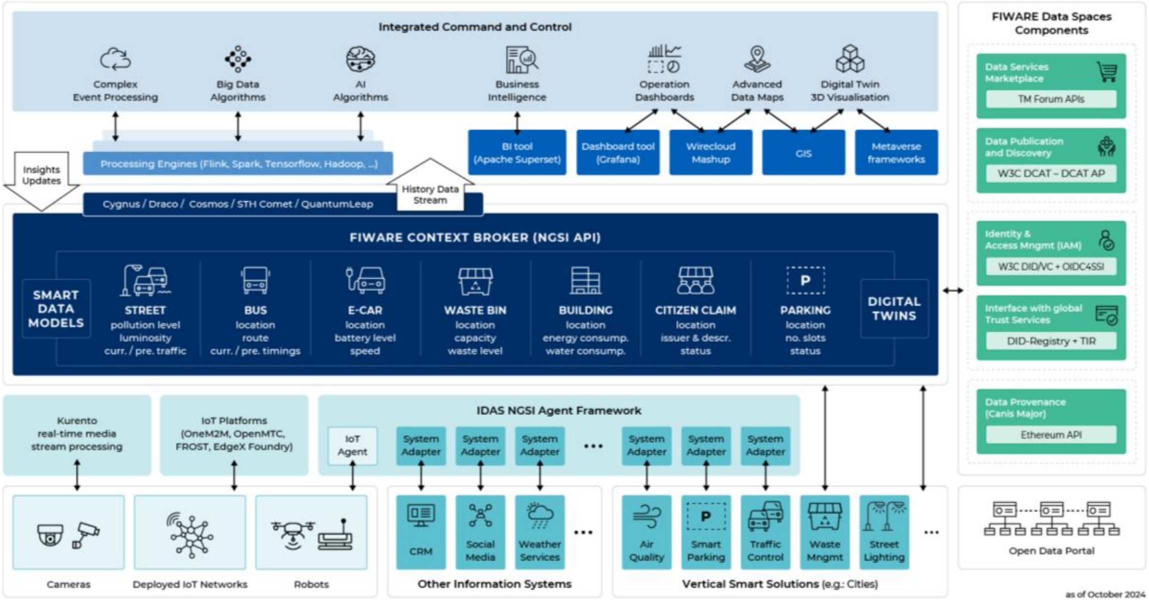

Integrated Command and Control
Complex Event Processing
Big Data Algorithms
AI Algorithms
Business Intelligence
Operation Dashboards
Advanced Data Maps
Digital Twin 3D Visualisation
Processing Engines (Flink, Spark, Tensorflow, Hadoop, ...)
BI tool (Apache Superset)
Dashboard tool (Grafana)
Wirecloud Mashup
GIS
Metaverse frameworks
Insights Updates
History Data Stream
Cygnus / Draco / Cosmos / STH Comet / QuantumLeap
FIWARE CONTEXT BROKER (NGSI API)
SMART DATA MODELS
STREET pollution level luminosity curr. / pre. traffic
BUS location route curr. / pre. timings
E-CAR location battery level speed
WASTE BIN location capacity waste level
BUILDING location energy consump. water consump.
CITIZEN CLAIM location issuer & descr. status
PARKING location no. slots status
P
DIGITAL TWINS
FIWARE Data Spaces Components
Data Services Marketplace
TM Forum APIs
Data Publication and Discovery
W3C DCAT – DCAT AP
Identity & Access Mngmt (IAM)
W3C DID/VC + OIDC4SSI
Interface with global Trust Services
DID-Registry + TIR
Data Provenance (Canis Major)
Ethereum API
Kurento real-time media stream processing
IoT Platforms (OneM2M, OpenMTC, FROST, EdgeX Foundry)
IDAS NGSI Agent Framework
IoT Agent
System Adapter
System Adapter
System Adapter
...
System Adapter
System Adapter
System Adapter
Cameras
Deployed IoT Networks
Robots
CRM
Social Media
Weather Services
...
Other Information Systems
Air Quality
Smart Parking
Traffic Control
Waste Mngmt
Street Lighting
...
Vertical Smart Solutions (e.g.: Cities)
Open Data Portal
as of October 2024

Figure 19: FIWARE reference architecture [56]

The core of its design is the context broker which facilitates seamless communication among smart applications through an interoperable data space. The content broker utilizes the NGSI API, a standardized interface for accessing and modifying data. Moreover, in order to ensure scalability and interoperability, FIWARE is equipped with an IDS connector intending to be aligned with the International Data Space Reference Architecture [56].

3.2.2.2.1.4 Gaia-X

Gaia-X is a European initiative that aims to establish an open, transparent and secure digital ecosystem focusing on ensuring secure data sharing and providing reliable data services. To accomplish this, common policies are adopted across data spaces and built on top of already existing cloud infrastructure [55]. Regarding its architectural approach, Gaia-X relies on decentralization and federation, enabling the coexistence of different platforms in a harmonious manner, by following a shared set of patterns. Gaia-X is not operating as a single cloud provider, but it envisions a federated ecosystem with a plethora of cloud and data services. This interoperable and sovereign data ecosystem is based on agreed-upon policies and specifications [57].

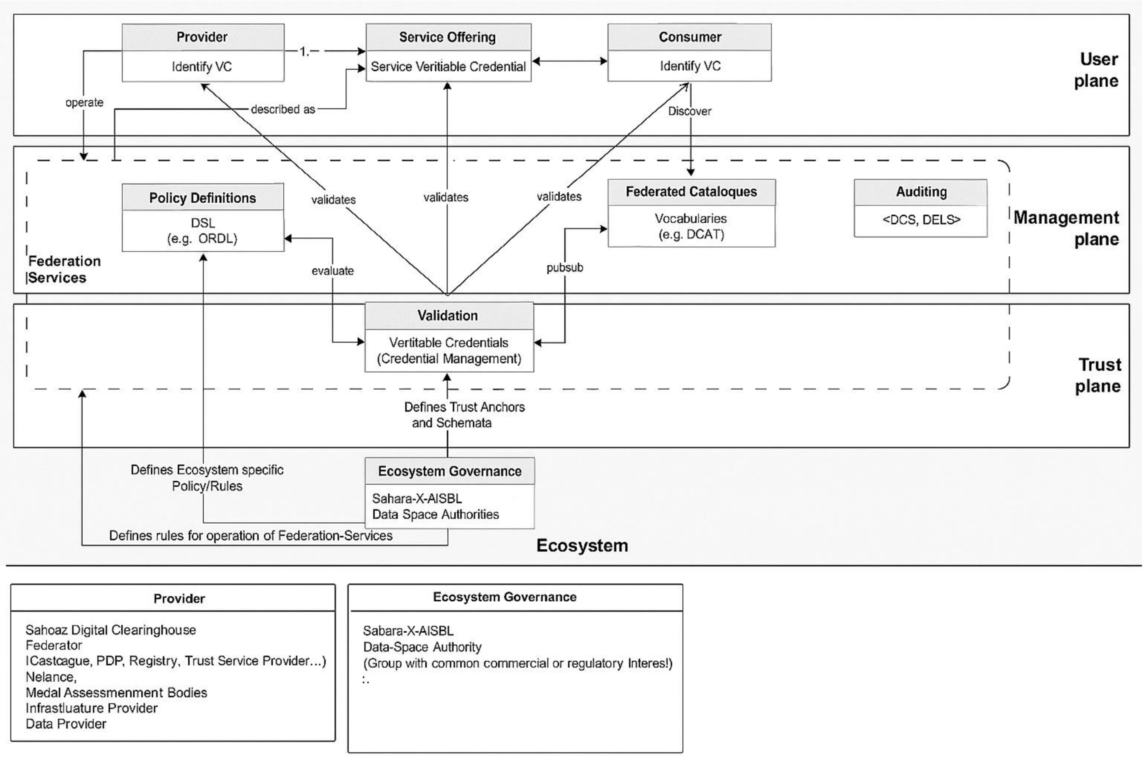


Figure 20: Gaia-X reference architecture [56]

The Gaia-X reference architecture is divided into three conceptual layers: the user plane, the management plane and the trust plane. Within this architecture several federated services support the secure information sharing between the participants. The trust between these interactions is established through verifiable credentials, which are located within the participant's self-description along with usage policies that define constraints of data usage [56].

3.3 Interoperability

Interoperability is a term aiming to describe the ability of diverse entities to communicate and exchange information in a meaningful way with one another. In this context, interoperability extends to various levels, such as communication protocols, business models, hardware and software composition and policies. Therefore, in order to achieve a comprehensive approach, interoperability is examined through three distinct dimensions: semantic interoperability, syntactic interoperability and technical interoperability [35].

3.3.1 Interoperability challenges

Ensuring interoperability constitutes a challenging task. Some of the primary issues encountered are [43]:

- **Data Inconsistency**: The growing complexity of systems generates a large amount of heterogenous data, leading to data inconsistencies. These inconsistencies require additional resources in order to process unstructured data. In contrast, the existence of structured data enables query operations to enhance effective analysis and filtering.
- **Scalability**: Scalability issues arise from the combination of data from different sources with information provided from legacy systems. Addressing this challenge, it is crucial to update and modify these systems intending to align them with new technological requirements.
- **Accommodate Scope of Data**: The development and support of new domains for handling large amounts of data introduces another interoperability challenge. This issue generates the need for high-performance computing environments and advanced data storage solutions.

3.3.2 Semantic Interoperability

In the context of industrial environments, semantics are used to describe the link between signifiers. Semantic interoperability ensures that the meaning of the exchanged information remains intact and that it is comprehensible from the devices that take part. This constitutes a significant aspect of such systems due to the inability of the devices to comprehend vague and ambiguous data. In order to achieve this standardization, vocabulary needs to be established todefine and translate information reliably. However, within industrial systems, data is structured in formats, such as CSV, JSON and XML, that are often not semantically aligned. Except from that, the various devices across the system use different models and languages and gateway devices lack the ability to unify them into a shared framework. [35].

3.3.2.1 Data exchange technologies

For these reasons, the development of data exchange technologies supporting semantic interoperability is crucial.

3.3.2.1.1 Web Ontology Language (OWL)

Web Ontology Language (OWL) is a well-known standard established by the web consortium (W3C) regarding knowledge representation on the Semantic Web. Specifically, it focuses on structuring information about real-world entities and their relationships. OWL serves as a powerful and flexible tool that can be utilized across various sectors, such as healthcare and automotive industries. It is based on Description Logics, a well-established class of logic systems, providing OWL a reliable semantic foundation. One of the primary advantages of Web Ontology Language is the support of reasoning services, which include techniques for processing background information [36].

3.3.2.1.2 Resource Definition Framework (RDF)

The Resource Description Framework (RDF) is a framework intended to describe resources in a way that facilitates exchange and reuse of structured metadata. Specifically, it supports the representation of any identifiable entity in data, involving virtual entities such as webpages and websites, concrete entities such as people and places and abstract entities like categories and relationships between entities [38].

RDF is based on XML, applying distinct constraints in order to ensure the provision of accurate semantic expressions. These expressions are intended to be comprehensible by humans as well as machines, assuring the semantic interoperability of the system. The metadata vocabularies and semantic definitions are developed by an information community, which defines the purpose and the structure of the approach, are exchanged and reused through the RDF infrastructure [37].

3.3.2.1.3 OPCUA Modelling and Communication Framework

OPC UA (Open Platform Communications Unified Architecture) is a standard for communication and information modeling. It is considered one of the pillars of Industry 4.0 and ensures interoperability at the machine level. The primary benefits of OPC UA include the ability to define semantics in domain-specific companion specifications, high IT security, and vendor-independent interoperability. OPC UA is fundamentally an information-centric data model that establishes basic rules for how data is exposed to any application or device that wishes to consume it. It provides a comprehensive framework for information modeling, enabling the structured representation of industrial data and processes. The core of an OPC UA information model is built upon objects, which can encapsulate variables and methods, and establish references to other objects. Clients can perform read and write operations on these variables and invoke methods that are then executed by the server. The fundamental unit of data within the OPC UA address space is a node, which is uniquely identified by a Node ID that includes a namespace URI. Nodes represent pieces of information about various components, such as sensors and actuators. The OPC UA address space itself provides a standardized and hierarchical way for OPC UA servers to represent these objects to OPC UA clients. Nodes possess attributes (the actual data value) and references to other nodes within their own address space, facilitating the creation of complex data structures and relationships.

A cornerstone of OPC UA's ability to achieve semantic interoperability lies in its Companion Specifications. These specifications formalize industry-specific data models, allowing OPC UA to serve as a standard transport method for this specialized data. By providing an agreed-upon, standardized data model for collecting similar information in consistent formats within a particular industry, Companion Specifications ensure that different devices and systems from various vendors can understand and exchange data in a semantically consistent way. This directly addresses the challenge of data interpretation across disparate systems and vendors. The OPC Foundation[1], in collaboration with numerous partners, plays a crucial role in jointly creating these standardized information models. This collaborative effort positions the OPC Foundation as a central, neutral platform for industrial interoperability, akin to a "United Nations of automation," vital for achieving widespread, cross-vendor semantic understanding. These specifications not only define the object-oriented information of a system but also integrate IT security by defining access rights.

3.3.2.1.4 Asset Administration Shell - Concept Description (AAS-CD)

Asset Administration Shell (AAS) is a standardized digital representation of any physical or intellectual asset. It provides the complete digital description of the asset and its functionalities, which can be retrieved via API.

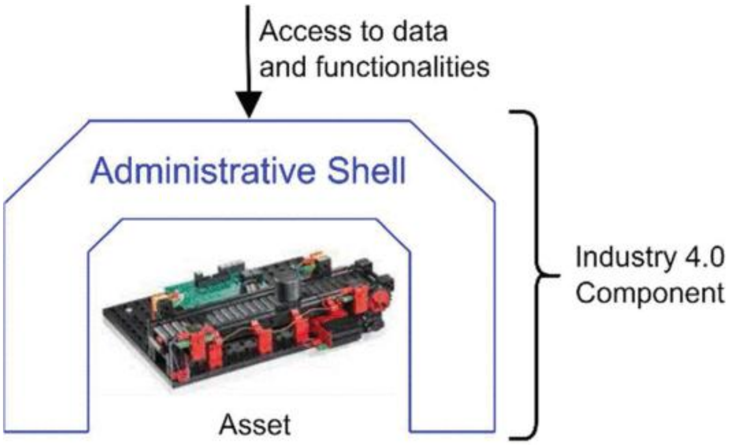


Figure 21: Industry 4.0 Component [44]

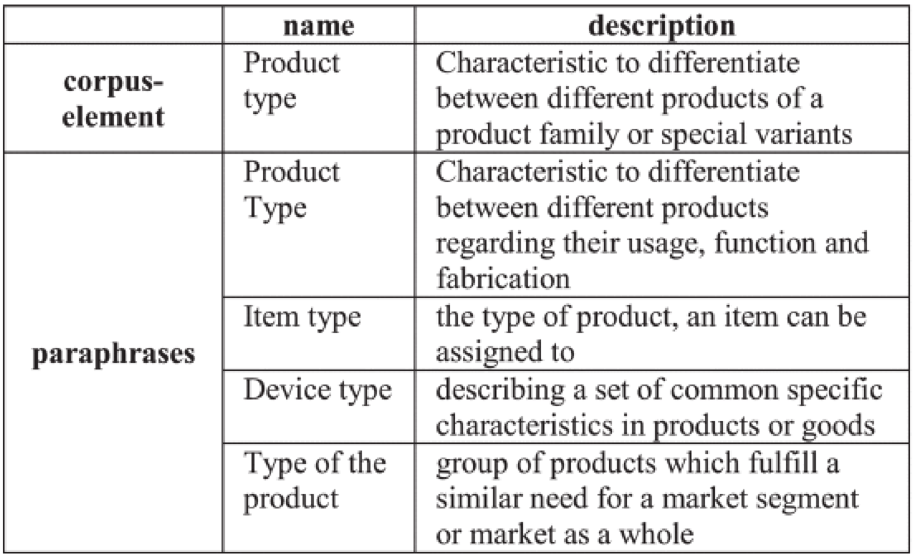
In the context of Asset Administration Shell (AAS), semantic interoperability is achieved using Semantic IDs and Concept Descriptions that enhance the AAS information with external references [44].

Concept Descriptions were established in order to describe operational values within the AAS, such as pressure, temperature, speed and humidity and can be enriched with IEC61360 content, assuring a standardized semantic representation of an AAS element. AAS supports various types of identifiers ensuring unique identification of an element within the Asset Administration Shell. An important identifier is the semantic id, which in the case of a submodel it links it with semantic definitions or concept descriptions. These semantic definitions can be defined externally and be referenced via globally recognized semantic IDs. A key example is ECLASS, which provides a set of globally unique

[1] https://opcfoundation.org/

identifiers, known as International Registration Data Identifier (IRDI) [46] and, in this way, offers standardized Concept Descriptions for industrial product classification and properties [45].

The ECLASS dataset is divided into two blocks: the corpus and the query. The corpus includes a cluster of ECLASS properties related to centrifugal and positive displacements pumps. On the other hand, the query comprises paraphrased versions of the corpus entities.



	name	description
corpus-element	Product type	Characteristic to differentiate between different products of a product family or special variants
paraphrases	Product Type	Characteristic to differentiate between different products regarding their usage, function and fabrication
	Item type	the type of product, an item can be assigned to
	Device type	describing a set of common specific characteristics in products or goods
	Type of the product	group of products which fulfill a similar need for a market segment or market as a whole

Figure 22: ECLASS dataset [47]

The aim of this approach is to match each paraphrased query with its corresponding corpus entry through semantic alignment. Currently there are 672 corpus elements and 1711 associated paraphrases [47].

3.3.3 Syntactic interoperability

Syntactic interoperability focuses on the structure of the exchanged information between the components of the system. It involves communication mechanisms and packaging techniques that aim to support machines and devices to efficiently interpret message representations during data transmission. Therefore, the primary objective of semantic interoperability is to define standard data formats, such as XML and JSON. Except from that, the system's devices should be equipped with interoperable interfaces that convert raw information into standardized data formats and also recognize syntactic errors. Moreover, syntactical and grammatical rules should be established and followed in the encoding and decoding processes by sending and receiving devices accordingly. In the context of industrial environments, the main challenges regarding syntactic interoperability are the heterogeneity of the devices and the difficulty in establishing standard communication protocols [35].

3.3.3.1 XML

The Extensible Markup Language (XML) is a widely adopted W3C standard intended to represent structured documents. Its simplicity and flexibility make it appealing for various operating systems, applications and browsers. The approach of XML is tag-based, like HTML, however providing the possibility of defining custom tags. Moreover, its tags, unlike HTML, have no specific semantics. XML documents involve both plain text and markups, and their structure can be visualized as ordered labeled trees. Its structure can include different types of nodes, like document nodes, elements, attributes and namespaces [4]. Originally, XML was designed to address the complexity of exchanging large amounts of data while also supporting customization. However, its capabilities were extended in order to handle the transfer of diverse types of information across heterogeneous distributed systems [34].

3.3.3.2 JSON

JavaScript Object Notation is a lightweight data format built upon JavaScript programming language's defined structures. In its core JSON documents consist of key-value pairs, where the name describes the represented information and value is the actual data. The value could also be another JSON document, supporting in this way an arbitrary level of nesting [30]. JSON constitutes the cornerstone of web applications, as it is easily understood by both developers and machines, and the main way to send and receive API requests is over the HTTP protocol [31]. Regarding its structure, JSON represents data as objects or as arrays. An object is described as an unordered collection of name-value pairs, where within these pairs information is contained that describes the object. The definition of the object starts with an opening brace and ends with a closing brace. On the other hand, an array is an ordered list of values, indexed by the position of the value in the list. The definition of an array starts with an opening bracket and ends with a closing bracket [32].

3.3.4 Technical Interoperability

Technical interoperability refers to the exchange of information process between sensing and actuating elements within the industrial system, a procedure described as functional networking. It ensures that data sharing complies with specific standards, reinforcing the provision of service quality, while also maintaining data integrity and performance. In the context of industrial environments, technical interoperability facilitates seamless data exchange between heterogeneous devices based on established specifications and standards. To achieve this, standardized protocols, open-source platforms, and interoperable network standards should be followed [35].

3.3.4.1 Standardized protocols

3.3.4.1.1 HTTP

The Hypertext Transfer Protocol (HTTP) is a protocol designed to support distributed, collaborative and hypermedia information systems and constitutes a cornerstone of the World Wide Web since 1990.

Its first version, HTTP/0.9, was a basic protocol regarding raw data exchange over the Internet. The next version, HTTP/1.0, introduced additional attributes such as MIME-style messages formatting, allowing control over request and response semantics. Despite its improvements, HTTP/1.0 lacked various important features like hierarchical proxies, caching, persistent connections and virtual hosting. Furthermore, inconsistencies issues were presented along with incomplete implementations that complicate the identification of each application's capabilities. These concerns paved the way for the development of a revised version that ensures the implementation of its features, HTTP/1.1.

This protocol is aligned with the needs of the modern information systems that require except from data retrieval, search, front-end update and annotation, offering an open-ended set of methods and headers. Regarding messaging, a format similar to the one defined by the Multipurpose Internet Mail Extensions (MIME) is adopted. Moreover, it acts as a general-purpose protocol, enabling communication between user agents and gateways. These interactions ensure the access to resources across several internet-based applications [39].

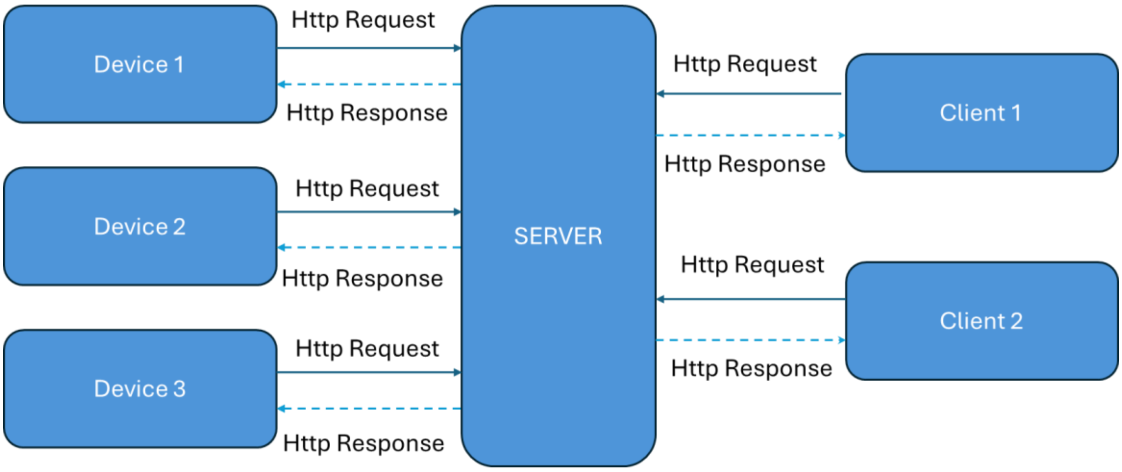


Figure 23: HTTP flow diagram [12]

Similarly to CoAP, HTTP utilizes Universal Resource Identifier (URI) rather than a topic-based approach regarding communication between server and client. Acting as a text-based protocol, HTTP does not impose strict limits on headers and message payloads; these aspects are defined by the web server. HTTP primarily uses TCP as transport protocol and TLS/SSL for security purposes [40].

3.3.4.1.2 REST

Representational state transfer (REST) refers to web services aiming to ensure interoperability between computer systems on the internet. REST utilizes HTTP, due to its popularity, supporting its standard operations including GET, POST, PUT, DELETE. It was introduced in 2000 by Roy Fielding, alongside the development of HTTP/1.1, and it was built upon the structure of HTTP/1.0.

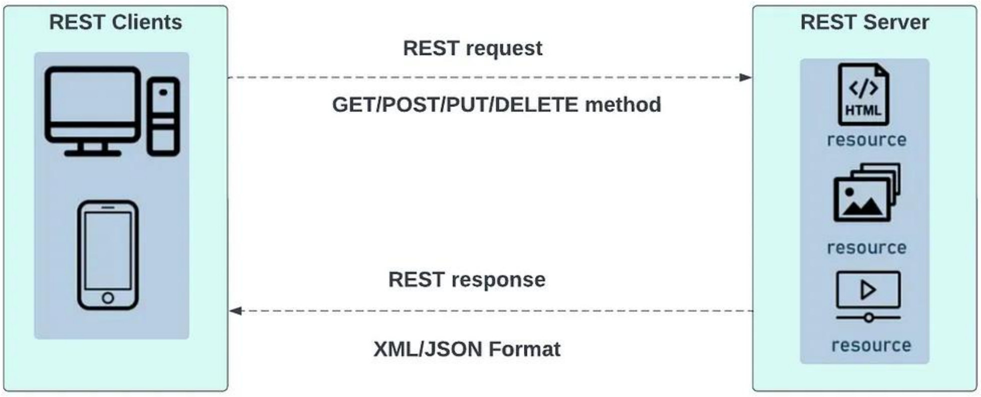


Figure 24: REST [42]

RESTful architectures leverage stateless communication and standardized operations in order to assure high-level performance, scalability and robustness.

3.4 Conclusions

Following the approach mentioned in the introductory Section 1 and combining the SotA with the Dow to fulfill the requirements from D1.1, the following statements are provided as conclusions to base the proposed conceptual architecture:

- All generic architectures (RAMI, IIRA and IIoT) are highly relevant and specific aspects of these architectures should be considered. Most relevant are the three tiers of IIRA combined with the IIoT capabilities and finally the RAMI architecture and the 4.0 smart component, the AAS.
- For cross company communication,data sovereignty plays a crucial role. The development of a dataspace seems most suitable. Further investigation should be performed in the dedicated task (T2.4) regarding its implementation and its functionalities.
- Interoperability is addressed adequately by the Asset Administration Shell and its series of specifications. Semantic interoperability and syntactic interoperability are supported by the “Details of the Asset Administration Shell - Part 1” specifications with the Concept Description notion and by the serialization specifications respectively. Technical interoperability is supported by the “Details of the Asset Administration Shell - Part 2” specifications via the detailing of the API for exchanging information.

These statements above will serve as the backbone of the RAASCEMAN architecture.

4 Conceptual Architecture

Continuing the work from the use cases and requirements analysis with some logical groupings (mainly on the non-business functional layers) we can derive a group of 7 main components on business layer, 2 main components on the infrastructure (including security, persistence and integration layers and slightly touching the Information Layer) and 2 main components on the Information Layer. In the below subsections each component is listed and analysed in terms of functionalities provided that fulfil its respective requirements. Finally on the last subsection an overall conceptual architecture is presented depicting all components and putting the basis for the information exchange and collaboration between them.

4.1 Business Layer components

In this layer components are grouped into two categories, the components that focus on intra-company functionalities (Factory level support tools) and components that focus on the cross-company functionalities (Supply chain level support tools).

4.1.1 Supply Chain level support tools

Cross-company or supply chain support tools support decision making and resilience. The “Impact Prediction” tool provides the impact of specific events along with their probability to happen while “Decision support tool” offers the capability of evaluating different supply-chain based scenaria for helping determining counter measures by analyzing a network of manufacturers/service providers for potentials collaborations. The “Trustworthiness Audit Tool” or simply “Audit Tool” works as a entry point of new participants in the network but also as an evaluation tool of the quality of the services offered by a specific network participant. Finally, the “Recommendation Engine” crawls the participants of the network to find the most suitable list of service providers that can fulfill a specific service request.

These tools are to be developed under work package 3 and each of these tools are to be developed under a dedicated task as depicted in Figure 25. More details are provided in the following subsections.

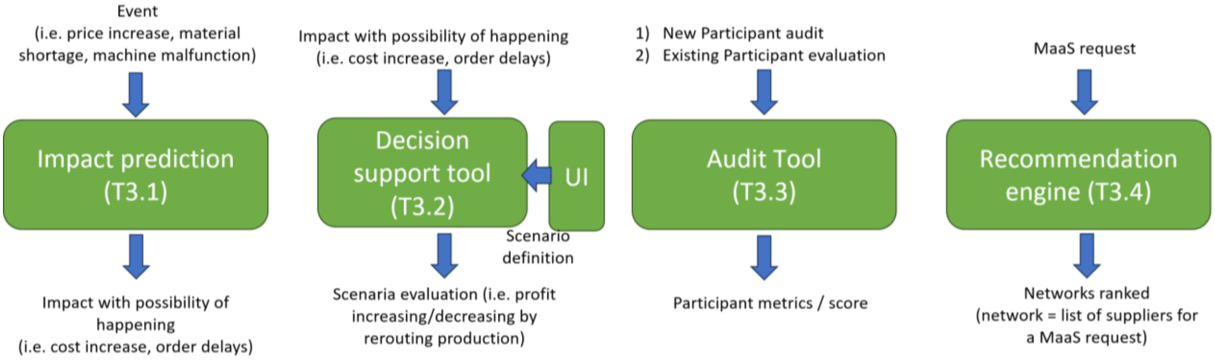

Event
(i.e. price increase, material shortage, machine malfunction)
Impact prediction (T3.1)
Impact with possibility of happening
(i.e. cost increase, order delays)
Impact with possibility of happening
(i.e. cost increase, order delays)
Decision support tool (T3.2)
UI
Scenario definition
Scenaria evaluation (i.e. profit increasing/decreasing by rerouting production)
1) New Participant audit
2) Existing Participant evaluation
Audit Tool (T3.3)
Participant metrics / score
MaaS request
Recommendation engine (T3.4)
Networks ranked
(network = list of suppliers for a MaaS request)

Figure 25: Supply Chain Level support tools

4.1.1.1 Impact Prediction Tool

The goal of the impact prediction tool is to build a probabilistic framework, built upon algorithms such as the Bayesian inference network, to quantify how various unforeseen events (external, such as supply-chain or market fluctuations, and internal, such as machine unavailability or workforce changes) may propagate through the production flow and affect manufacturing outcomes. First, different types of disruptive events are identified (drawing on information from the RAASCEMAN data model), and these events serve as inputs to the AI-based model, which can represent the causal relationships among factors influencing production. Conditional probabilities are assigned to each network node to model the likelihood and severity of each event's impact. The deployed network then allows users to perform "what-if" analyses given a scenario (e.g., a sudden supplier failure or a spike in customer demand) via the decision support tool of WP3. The impact prediction tool computes the probability distributions over downstream effects (e.g., delayed delivery, increased costs, or quality deviations). A high-level overview of the impact prediction tool is presented in Figure 26.

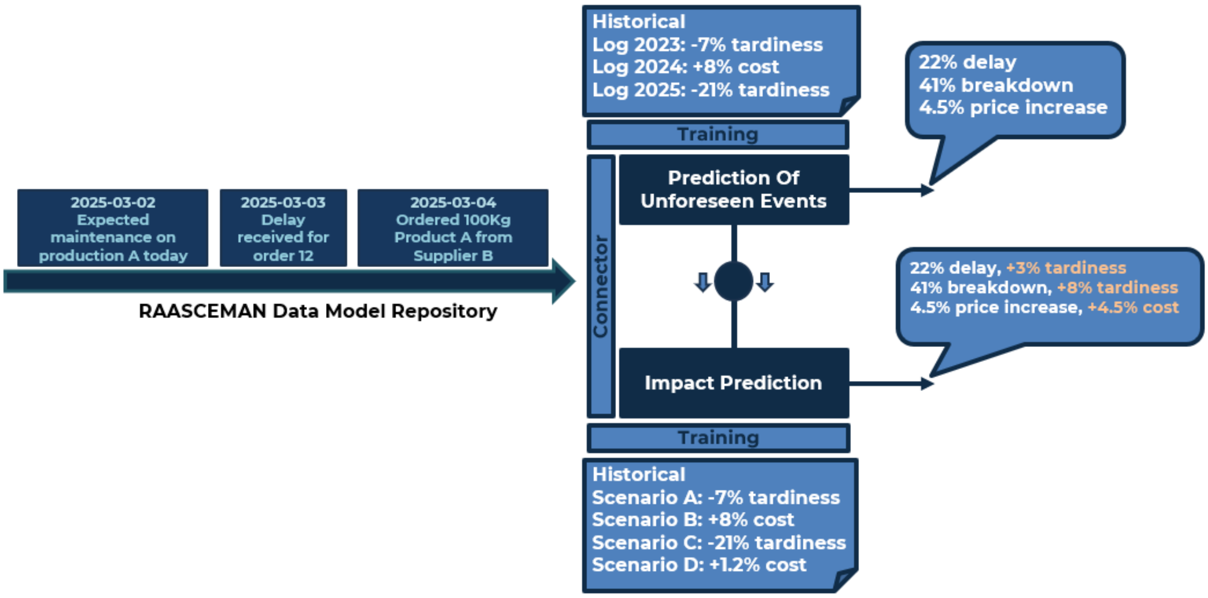

Historical
Log 2023: -7% tardiness
Log 2024: +8% cost
Log 2025: -21% tardiness
Training
22% delay
41% breakdown
4.5% price increase
2025-03-02 Expected maintenance on production A today
2025-03-03 Delay received for order 12
2025-03-04 Ordered 100Kg Product A from Supplier B
RAASCEMAN Data Model Repository
Connector
Prediction Of Unforeseen Events
Impact Prediction
22% delay, +3% tardiness
41% breakdown, +8% tardiness
4.5% price increase, +4.5% cost
Training
Historical
Scenario A: -7% tardiness
Scenario B: +8% cost
Scenario C: -21% tardiness
Scenario D: +1.2% cost

Figure 26: High-level overview of the impact prediction tool.

4.1.1.1.1 Relationship with other RAASCEMAN tools

The impact prediction tool has a relationship with the following RAASCEMAN components:

- **Data model repository**: Data needed for the tool will be accessed through the data model repository of RAASCEMAN. From there, the tool will access information on the occurrence of foreseen and unforeseen events. The predicted impact will also be provided to the data model repository.
- **Decision support tool**: The decision support tool will consume the outputs of the RAASCEMAN impact prediction tool. To facilitate the optimal interaction of the impact prediction tool with the decision support tool of RAASCEMAN, further refinement of the tool’s outputs will be performed in the context of WP3, Task 3.1.

4.1.1.1.2 Expected impact

The impact prediction tool provides a percentile estimation of the impact of internal and external unforeseen events. This functionality is essential in the supply chain level reconfiguration that RAASCEMAN will provide, as it will be a core input to the decision support tool of the project. This way, manufacturers will have an estimation of the events impact on their manufacturing operations, helping them to build more resilient supply chains.

4.1.1.2 Decision Support Tool

The Decision Support Tool (DST) developed by Flanders Make (FLM) plays a central role in enabling resilient and adaptive decision-making within the RAASCEMAN MaaS ecosystem. This tool supports manufacturers in evaluating complex, multi-faceted scenarios involving supply chain disruptions, capacity limitations and sustainability trade-offs.

4.1.1.2.1 Tool Capabilities

The DST provides manufacturers with a data-driven decision-making environment by integrating inputs from product parts provided by the Product Digital Twin (T2.2), internal company planning and capability matching (T4.1 and T4.2), impact predictive model(s) covering disruptive events (T3.1) and is connected to the recommendation engine (T3.4) as depicted in Figure 27. The tool supports trade-offs between various scenarios and consists of the following core functionalities:

- **Scenario Simulation & What-if Analysis**: Users can simulate events (e.g., supplier disruptions, machine failures) and assess their effects on key performance indicators (KPIs) like cost, carbon footprint, lead time, and production throughput.
- **Multi-Criteria Decision Support**: The tool allows to make trade-offs among conflicting manufacturing goals (e.g., speed vs. cost vs. emissions), enabling strategic prioritization under uncertainty.
- **Risk and Impact Integration**: By connecting to the Impact Prediction Tool, the DST incorporates probabilistic event likelihoods and estimated impacts, enriching the decision context with forward-looking intelligence.
- **Dynamic Data Integration**: It ingests structured data from Asset Administration Shells (AAS), product/process Digital Twins, and IIoT devices, ensuring alignment with live factory and supply chain status.
- **User-Driven Configuration & Visualization**: The web-based interface allows users to configure goals, define event parameters, and visualize simulation outcomes.

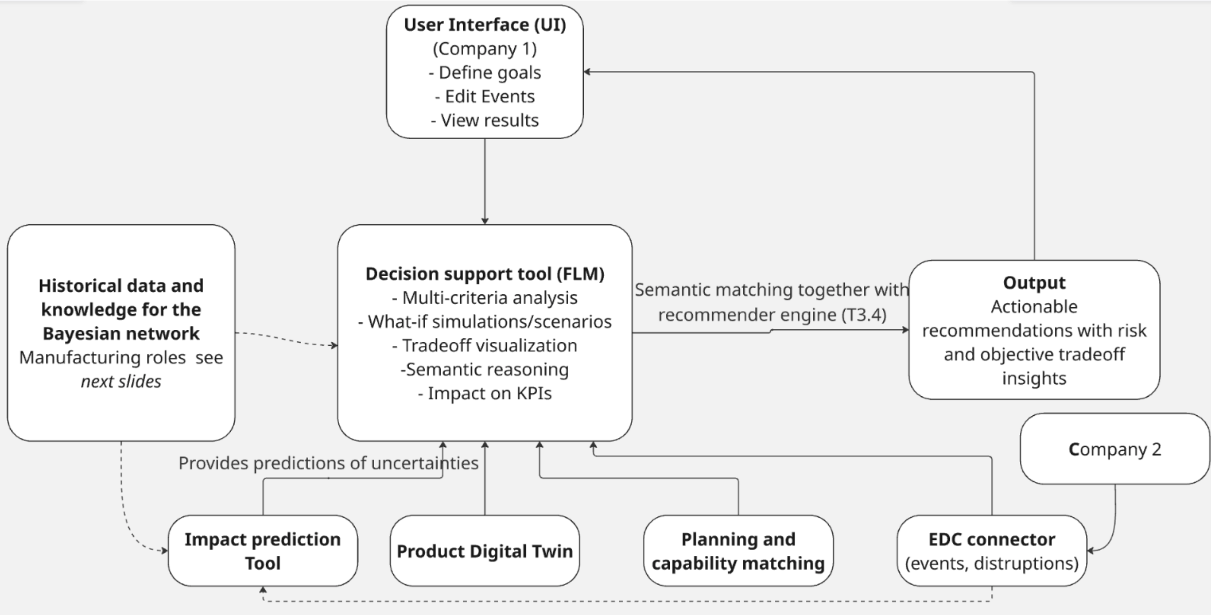

User Interface (UI)
(Company 1)
- Define goals
- Edit Events
- View results
Historical data and knowledge for the Bayesian network
Manufacturing roles see next slides
Decision support tool (FLM)
- Multi-criteria analysis
- What-if simulations/scenarios
- Tradeoff visualization
-Semantic reasoning
- Impact on KPIs
Semantic matching together with recommender engine (T3.4)
Output
Actionable recommendations with risk and objective tradeoff insights
Company 2
Provides predictions of uncertainties
Impact prediction Tool
Product Digital Twin
Planning and capability matching
EDC connector
(events, distruptions)

Figure 27: Positioning of the Decision support tool (DST) in relation to other tools and inputs

4.1.1.2.2 Interoperability and Architecture Role

Technically, the DST serves as a business logic layer component within the RAASCEMAN software architecture. Throughout this architecture, we refer to a layered model (as established in WP1 and based on Table 2), grouping components into logical layers such as Information, Integration, and Persistence. The DST operates across these layers as follows.

- Information Layer: via Digital Twin models for product and process information.
- Integration Layer: through AAS/IIoT communication gateways.
- Persistence Layer: accessing historical data for KPI benchmarking and validation.

4.1.1.2.3 Expected Impact

By offering a semantic reasoning interface for simulation, evaluation, and recommendation, the DST enables manufacturers to respond proactively to disruptions. It supports reconfiguration of supply chains and internal production plans, improving overall system agility and strategic alignment with resilience and sustainability objectives.

4.1.1.3 Audit Tool

A supplier audit tool tackles the issue of trustworthiness in a supply chain network. The core of the tool is a similarity measure comparing products manufactured in the past with offers for new product requests. The objective of the suppliers' audit tool is to verify their service descriptions using historical data, such as quotes, delivered products and other relevant documents, providing insights into the achievable quality, delivery time and eco-friendliness of the supplier.

4.1.1.3.1 Tool Capabilities

The goal of an audit tool is twofold:

1. ***Existing participant evaluation***. Given a customer order and the production history, it provides a producer rating, which is an evaluation score that a producer can produce this product or provide this service with the confidence of n%.
2. **New customer audit (onboarding)**. Given the list of existing capabilities types, taken from the standard and norms, production history and the list of production resources, it estimates the producer's capability instance with the parameters values as ranges, with the confidence of n%.

Both functionalities use the same principle of evaluating similarity between various variables. Figure 28 and Figure 29 give a high-level view of the tool. In both cases, we take various artifacts, such as production history, data from various sources, e.g., CAD drawings, product specifications, etc., codebase, and measure the similarities between these pieces of information and, in the first case, the order, and in the second case, the list of the capabilities types typical for the domain. After the first stage, we get either the list of similar products or the list of similar capabilities respectively. In the second stage, we compare them to the current production resources available on the production floor to generate a producer rating or a capability instance.

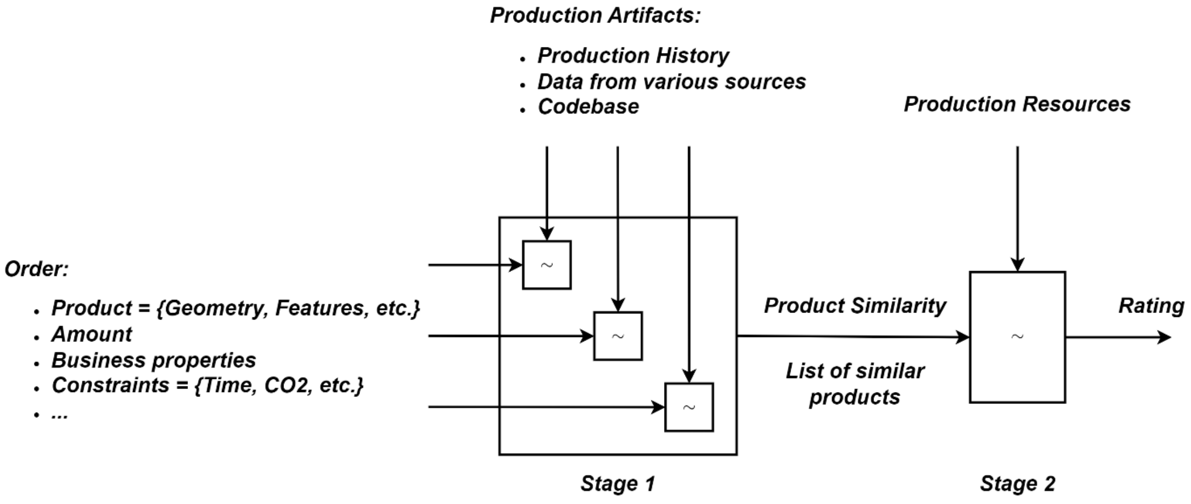


Figure 28: Evaluation of an existing partner

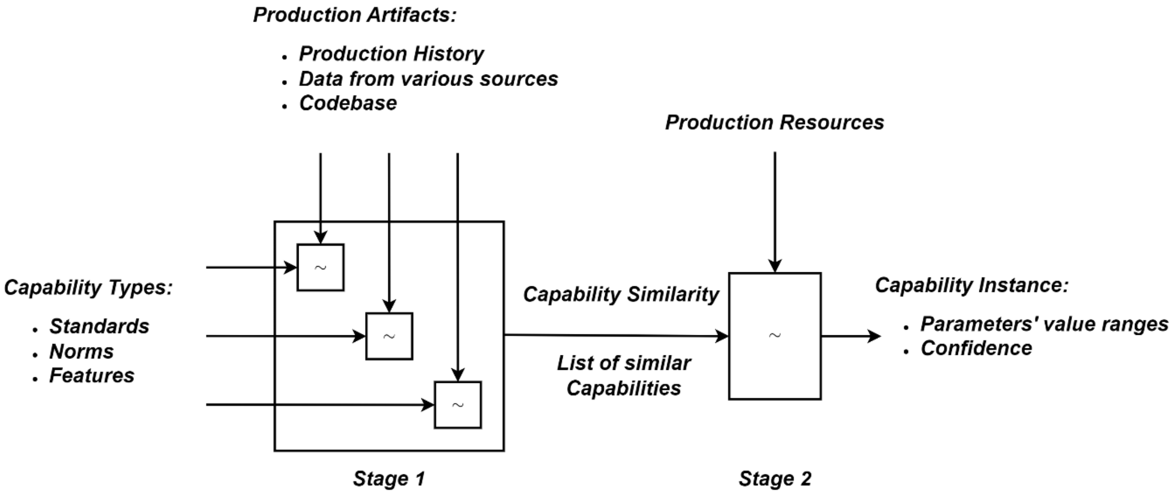


Figure 29. Onboarding of a new partner

4.1.1.3.2 Relationship with the other RAASCEMAN tools

- ***Data models repository.*** The audit tool will use the information from the standard AAS submodels defined during the project, e.g., capabilities types, product features, business information, etc.

- ***Recommendation engine.*** The tool will interact with the recommendation engine and provide it with the producer rating, that is, an aggregated evaluation of the producer's capability to provide a required service.
- ***Procedure and capability matching tool.*** The audit tool will require information about the existing production capabilities and resources, which can be taken from the capability matching tool.

4.1.1.3.3 Expected Impact

The audit tool tackles the issue of reliability and trustworthiness in the supply chain network. By providing the recommendation engine with the evaluation measure of how the supplier's service description matches its actual capabilities it increases the trust between the MaaS network participants and supports informed decision making.

4.1.1.4 Recommendation Engine

A recommendation engine for dynamic supply chain generation will be capable of generating supply chain alternatives and providing recommendations such as finding and selecting suitable service providers and performing automated negotiations.

4.1.1.4.1 Tool Capabilities

The goal of the recommendation engine is to generate the alternative supply chains based on the order, proposals from the manufacturing services providers and KPIs. First, the engine builds the search space by sending the service requests to the MaaS network. Each participant of the MaaS network has an audit agent which is a part of the RAASCEMAN audit system The agent makes an audit as described previously and generates the score, which is send back together with the proposal. The RAASCEMAN network also provides a rating system analogous to the marketplaces where all the participants can evaluate each other. Based on the information received from the audit agents and the rating system the recommendation engine generates possible paths in the search space "Service – Service Provider" (see Figure 30). We assume that a manufacturing service can be provided by several manufacturers.

4.1.1.4.2 Relationship with the other RAASCEMAN tools

- ***Data models repository.*** The recommendation engine will use information about the production services from the standard AAS submodels defined during the project.
- ***Audit tool.*** The recommendation engine will use the information received from the audit tool for building possible trustworthy supply chains.
- ***Dynamic planning and scheduling tool:*** The recommendation engine will need feedback concerning technical feasibility and commercial properties of a service request.

4.1.1.4.3 Expected Impact

The recommendation engine will support the user in making informed decisions when choosing the required services providers by generating various supply chain alternatives and providing recommendations such as finding and selecting suitable service providers and even performing automated negotiations.

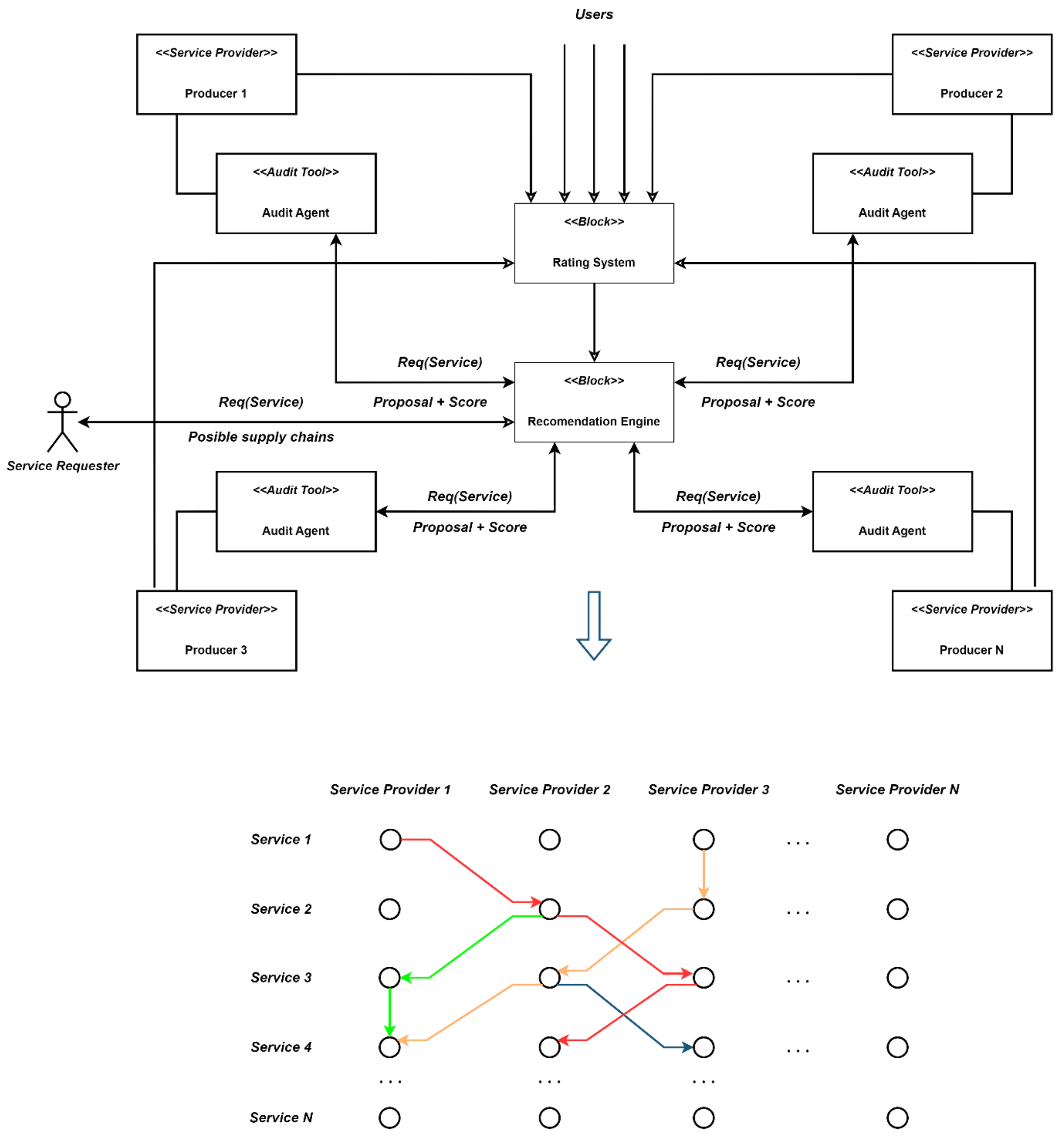


Figure 30: Recommendation and rating system.

4.1.2 Factory level support tools

Factory Level support tools focus on optimizing operations and processes within individual production sites. For the context of RAASCEMAN these tools focus mainly on continuously adjusting production ("Dynamic Planning & Scheduling" tool) with the help of a tool which aims to offer different manufacturing sequences based on identifying resource suitability from required skills ("Capability Matching Engine"). Finally, the "Dynamic Execution" tool will be responsible to instruct the shopfloor machines of the needed adaptation and monitor the production status.

These tools are to be developed under work package 4 and each of these tools are to be developed under a dedicated task as depicted in Figure 31. Moreover, an orchestrator component will be responsible to use the "Capability Matching" service and "Dynamic planning & scheduling" services to act as a "gateway" and "communication point" between the "Factory level support tools" and the "Supply Chain level support tools".

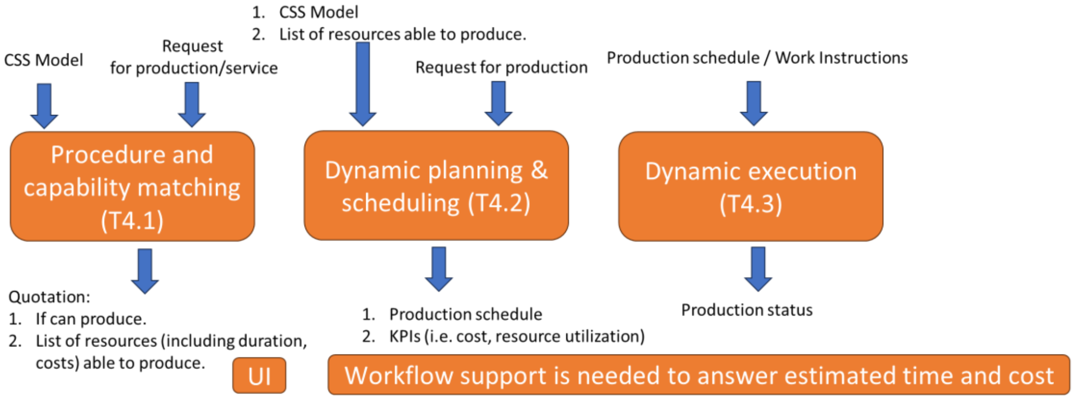

1. CSS Model
2. List of resources able to produce.
CSS Model
Request for production/service
Request for production
Production schedule / Work Instructions
Procedure and capability matching (T4.1)
Dynamic planning & scheduling (T4.2)
Dynamic execution (T4.3)
Quotation:
1. If can produce.
2. List of resources (including duration, costs) able to produce.
1. Production schedule
2. KPIs (i.e. cost, resource utilization)
Production status
UI
Workflow support is needed to answer estimated time and cost

Figure 31: Factory level support tools

4.1.2.1 Procedure and capability matching

The Procedure and Capability Matching Tool (PCMT) is the factory-level gateway that converts high-level MaaS or MES service requests into resource-aware process steps. Sitting in the Business-Logic layer, it consumes semantic models from the Information layer (CSS, AAS submodels) and real-time shop-floor status via the AAS/IIoT Communication Gateway, then forwards ranked capability matches to the Dynamic Planning & Scheduling tool while factoring in trust scores from the Audit Tool. This tight coupling ensures every match reflects current machine states, operator availability, and supplier reliability.

PCMT normalises heterogeneous inputs, executes SPARQL queries over a GraphDB of ⟨resource, skill, parameter⟩ triples, filters results with hard constraints, and re-weights them using cost, lead-time, and trust criteria. It returns a Capability-Match Manifest – task ID, ranked resources, and diagnostics – satisfying REQ5.1–5.1.4 and REQ4.1.2.5. By aligning semantic data with live factory conditions, the tool raises machine utilization, shortens replanning loops, and provides a natural-language interface for intuitive "what-if" exploration, forming the semantic backbone of RAASCEMAN's resilient, adaptive production workflow.

PCMT inputs

- Service Request: product/process requirements from MaaS network or internal MES – including tolerances, due date, KPI weights.
- Semantic Models: CSS model, Service/Capability/Skill submodels, Product DT, AAS submodels.
- Live Shop-floor Data: machine states, tool wear, operator availability via AAS/IIoT Gateway (OPC UA, MQTT).
- Trust & Risk Signals: supplier/manufacturer trust scores from Audit Tool, impact-prediction risk flags (optional).

PCMT outputs

- Capability-Match Manifest (JSON)
  - Task ID & required process steps
  - Ranked candidate resources with match scores
  - Constraint/feasibility diagnostics (missing skill, overload, etc.)
- Alerts: "no feasible match" or "data missing" messages to Orchestrator.
- Metrics Stream: matching latency, query statistics to Monitoring dashboard.
- Human-readable Summary: NL explanation of top matches for UI.

4.1.2.2 Dynamic planning and scheduling

The goal of the dynamic planning and scheduling component is to create new production schedules according to the current factory state and the provided information about replanning. The component uses the available information about intra-factory resources that needs to be planned or reordered and the result from the capability matching engine that specifies which processing steps needs to be executed considering given constraints. The result is a new plan that contains both the already planned steps and the new steps. If it is not possible to find a suitable plan, the component reports that no schedule exists to fulfil all constraints (e.g. finish within a given deadline).

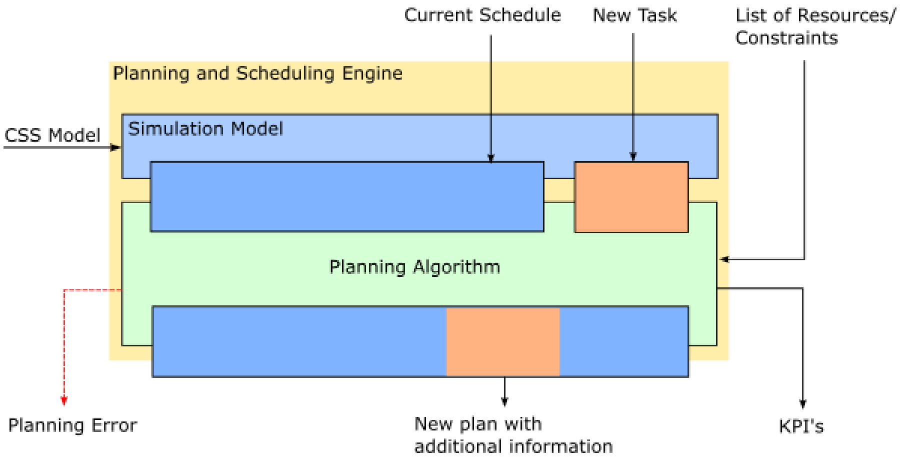

Current Schedule
New Task
List of Resources/ Constraints
Planning and Scheduling Engine
CSS Model
Simulation Model
Planning Algorithm
Planning Error
New plan with additional information
KPI's

Figure 32: Dynamic Planning and Scheduling Engine

The component needs different inputs as a basis for correct planning, and they can be differentiated by their type. The first type are static inputs that are not changing over time and the second type are dynamic inputs that needs to be determined when the planning engine starts to calculate. The static input is the provided CSS model that builds the foundation for the planning algorithm and is used to set up a simulation model. Building this model is independent from the planning calculation itself and should be established as early as possible. Furthermore, it can be rebuilt if the state within a factory changes and the model needs to be validated again (e.g., after machine breakdowns or new available machines). The dynamic inputs are the current schedule, the new task, the list of available resources, and the determined constraints that needs to be fulfilled. Thereby, the constraints specify the planning goal (e.g., finish asap) and provides further information about the time frame in which the task has to be processed. After determination of all dynamic inputs, the simulation model allows to explore different planning states to evaluate possible results through simulating different scenarios (see Figure 32).

The outputs of the component are the newly calculated production plan and the resulting KPI's. Furthermore, there can be a flag that signals if no solution is found. However, this could also be derived from the results of the planning algorithm and will be reported to the requesting service. The resulting production plan contains the new schedule and provides further information like expected start and end time which can be used on higher level (supply chain level) decision. If applicable, the component can also provide a list of best results that will be collected during planning task to enable flexibility and avoid recalculation steps if results become invalid in the meantime.

4.1.2.3 Dynamic execution

The interface of the dynamic execution for the orchestrator inside the factory enables seamless coordination within the production environment. It takes input parameters that describe the task, ensuring accurate execution. During operation, it streams process-relevant data to MES and ERP systems for real-time monitoring. After execution, it provides output data as feedback for MES and ERP, supporting traceability and continuous process optimization.

The tool aims to demonstrate key use cases in an industrial shop-floor environment, focusing on seamless human-robot collaboration for tasks such as assembly and maintenance, alongside coordinated multi-robot systems engaged in object sorting. These scenarios reflect typical challenges in agile manufacturing and provide a foundation for evaluating collaborative autonomy in dynamic settings.

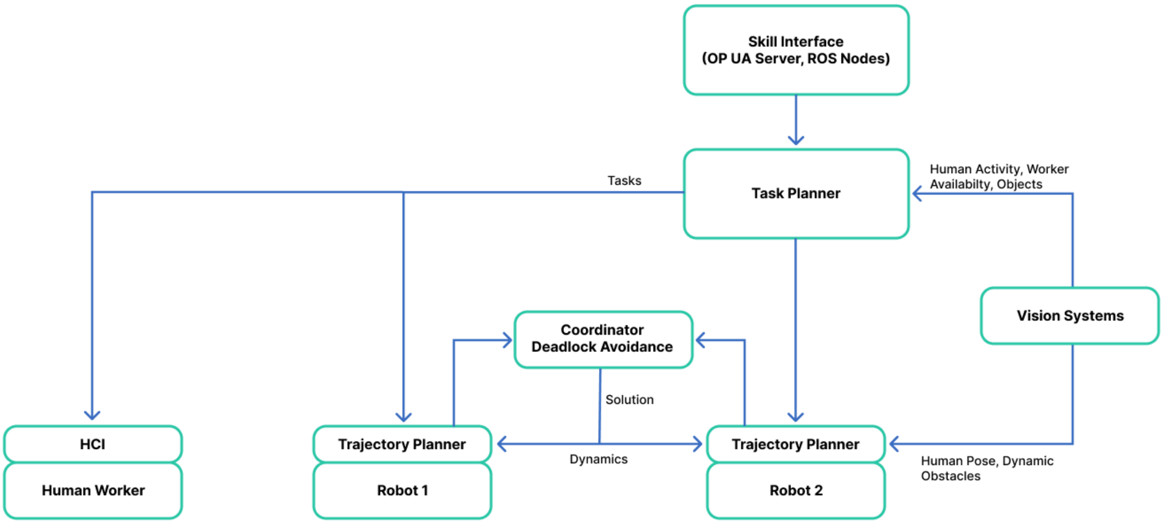


Figure 33: Dynamic Execution System Architecture

Besides having two robots that are capable of executing shared collaborative tasks, the system also deals with workers and can dynamically allocate tasks to according to their availability. Computer vision plays a major role, as shown in Figure 33, in perceiving the environment and eventually extracting information that can be used by the task planner to orchestrate tasks or even steps of tasks between different available actors in a shopfloor.

4.2 Information Layer components

These components aim to act as interoperability enablers defining a common information model (CIM) containing supporting almost all the parts of the information exchange between components in the context of RAASCEMAN project. From one hand, we identified the implementation of the CSS (Capability, Service, Skill) model to be extended to support the RAASCEMAN cases describing offered services of network participants but also offered services from each factory's shopfloor, and from the other hand a Product Digital Twin (PDT) model that could be used as specifications of a service request in the network and as documentation of a manufactured product offering functionalities of a Digital Product Passport (DPP).

These two information models will be detailed under WP2 and more specifically on each respective task T2.1 “Service, capability and skill modelling” and T2.2 “Product digital twin”.

4.2.1 CSS model

The Capability-Skill-Service (CSS) model, developed as an extension to the established Product-Process-Resource (PPR) framework, enables a capability-based approach to engineering by clearly distinguishing between product design and production planning. Within this model, services encapsulate high-level functional descriptions, incorporating commercial considerations for requests and quotes as well as linking to production capabilities available on the shop floor. These services are underpinned by capabilities, which are abstract, resource-independent descriptions of production functions defined in the context of specific processes. Capabilities, in turn, are realized by Skills, which are concrete implementations of functions on physical resources, with defined parameters, inputs, outputs, and mechanisms for executing and monitoring of the production step. The CSS model supports two key perspectives: the required product-side view, representing the customer's needs, and the offered resource-side view, representing the supplier's manufacturing capabilities. This dual structure needs semantic matching between required and available services and capabilities within a Manufacturing-as-a-Service (MaaS) platform. For this matching to work effectively, capabilities from Cyber-Physical Production Modules (CPPMs) must be clearly described and explicitly linked to the skills that implement them. This description requires a digital representation, such as the AAS, to display the offered capabilities in the AAS submodel. In supporting the matching algorithm, it is helpful to include standards such as ECLASS or DIN 8580 in explaining these capabilities. In addition, the Product Digital Twin (PDT) includes a submodel that outlines the required capabilities, further supporting this matching process. Overall, the CSS model is a fundamental element in enabling flexible, efficient and interoperable manufacturing ecosystems, in line with the objectives of the digital transformation initiatives supported by the EU. In Figure 34 is the simplified overview of the CSS model with the most important aspects shown.

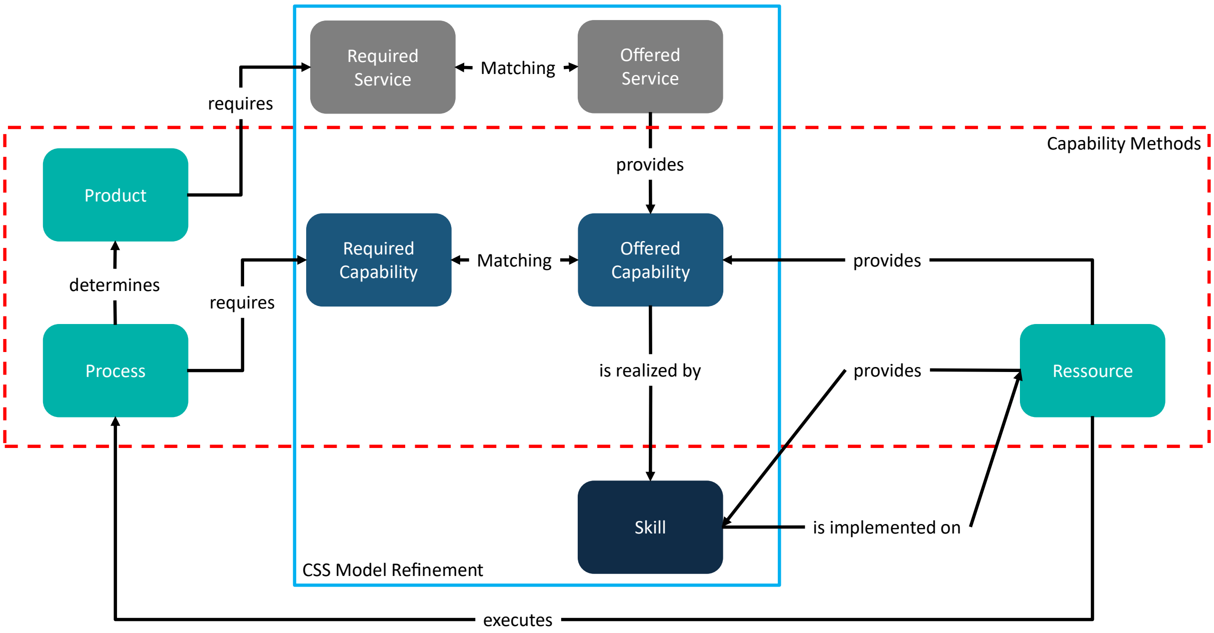


Figure 34: Simplified overview on most important aspects of the CSS model based on [58]

4.2.2 Product Digital Twin (PDT)

The PDT information model is one of the core components of the project, as it provides common data semantics about products and serves as a reference for all involved architectures and applications. The PDT information model deployed in the project has fundamental and optional parts. The fundamental part is modelled mainly by the AAS vocabularies, being extended with ECLASS and other ontologies proposed by the CIRPASS & CIRPASS 2 projects funded by the European Commission. The PDT AAS information model is composed of submodels representing different aspects of a product needed by different applications and use cases (see Figure 35). These submodels contain information needed for three basic groups:

- Information required by RAASCEMAN Tasks: Inputs for the seven basic applications defined in RAASCEMAN, which are: *(T3.1) Tool for impact prediction of disruptive events, (T3.2) Decision support Tool for companies in a dynamic MaaS network, (T3.3) Audit Tool for suppliers in MaaS network, (T3.4) Recommendation engine for dynamic supply chain generation, (T4.1) Tool for matching procedure and capability matching, (T4.2) Tool for dynamic planning & scheduling, (T4.3) Dynamic execution of tasks on the shopfloor.*
    - Note that some applications do not require information from PDT at all, for example the work developed in T3.3.
- Information for product lifecycle: Inputs for a DPP generator to specify a DPP for a product/lot of products instances. Such information can describe a life cycle of a product/lot of products

in the five steps: (1) design, (2) manufacturing, (3) distribution & logistics, (4) use & maintenance, and (5) end-of-life or recycling.

- Information for other features & applications: Materials for the other applications mentioned in the project called but not being defined concretely. Some of them are virtual product design, virtual commissioning, quality control.

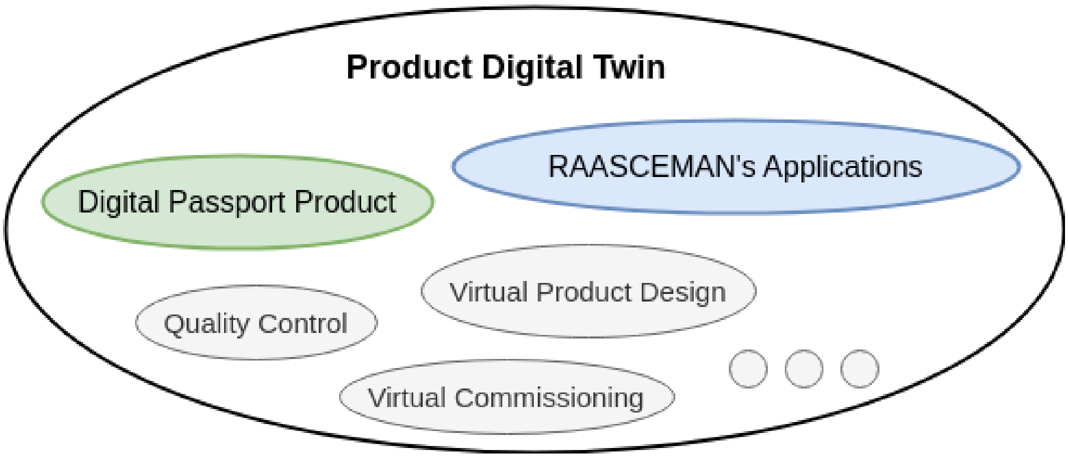


Figure 35: Product Digital Twin's information containers

Each submodel must follow strictly the AAS standard and should inherit from an AAS submodel template defined by the official submodels hub of IDTA. The optional part is an ontology that helps to link elements of the PDT information model with elements of other contexts (e.g. CSS). This ontology can support the reasoning feature of other tools such as matchmaking and production scheduling.

4.3 Infrastructure Layer components

The Infrastructure Layer components are the "AAS Infrastructure" and the "MaaS Platform" corresponding to the Factory level support and Supply Chain level support respectively.

The "AAS Infrastructure" provides from one hand IIoT functionalities by data collection from the shopfloor and on the other provides a communication gateway to serve as a shopfloor monitor to extract real-time information of shopfloor status by implementing a direct connection with each shopfloor asset. Finally, the information is provided in an interoperable way allowing other components to seamlessly acquire shopfloor data independent of each data source communication protocol.

The "MaaS Platform" aims at supporting the cross-company communication. Its "gateway" functionality supports the ability to exchange information between the different network participants. "Data Sovereignty" is of high importance in this module and it has an ability to log data exchange transactions.

These components will be developed under the context of work package 2 and more specifically under tasks T2.3 “Information infrastructure” and T2.4 “Data platform extensions” respectively.

4.3.1 AAS Infrastructure

As previously stated, this component will support the Factory level tools by providing IIoT functionalities but also shopfloor context functionalities in terms of real-time data acquisition from shopfloor’s assets. Moreover, taking into account the SotA conclusions interoperability is implemented by an integration of an AAS repository and all these moderated by a security mechanism. The Figure 36 shows the main components needed for implementing the AAS Infrastructure while also depicting the components that confront the afore mentioned aspects, Interoperability, IoT, Context and Security.

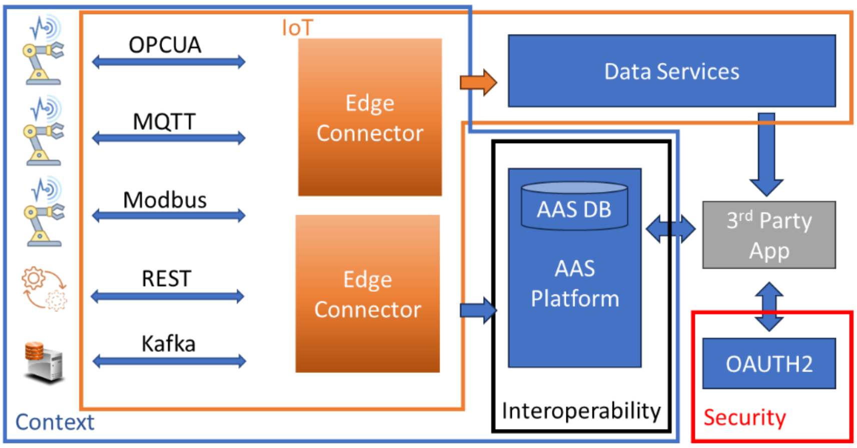


Figure 36 AAS Infrastructure detailed architecture

The schema above will be further detailed on the dedicated task “T2.3 - Information infrastructure”.

4.3.1.1 Interoperability

The adoption of the Industrie 4.0 specification regarding the definition and usage of the i4.0 smart component “Asset Administration Shell” provides interoperability on multiple layers. Semantic interoperability is provided by the use of Concept Description notion along with ECLASS semantics. Syntactic interoperability is provided by the Part 1 of the “Asset Administration Shell” specification regarding the AAS representation in JSON format. Finally Technical Interoperability is provided by the implementation of a set of RESTful services to interact with the AAS models defined in Part 2 of the “Asset Administration Shell” specification.

4.3.1.2 IIoT / IoT

A dedicated set of components should play the role of an IoT infrastructure. For these components the main functionalities offered is the implementation of a variety of industrial protocols for collecting shopfloor data serving as a normalization of different data sources. The main challenge of this component is the ability to serve high frequency / high volume data in an efficient manner. Scalability also plays a vital role as industrial production lines tends to host hundred machines each one able to produce a vast volume of data. This role is undertaken by the Edge Connector component. The Edge connector by design should be able to scale supporting multiple shopfloor equipment but also the architecture allows seamlessly the deployment of more than one Edge Connector following the "divide and conquer" technique. The other component that complements the IIoT part of the AAS Infrastructure is the "Data Services" which provides a moderated way to access historical data for the shopfloor equipment.

4.3.1.3 Shopfloor Context

In a manufacturing environment and more specifically the shopfloor, the "Context" refers to the real-time state under which manufacturing operations take place. Complemented by the "Edge Connectors" data derived from the shopfloor not only is stored for later processing (analytics etc.) but also update the shopfloor digital model aka the shopfloor related Asset Administration Shells. This way the main component that provides interoperability also serves as the "single source of truth".

4.3.1.4 Security

Finally comes Security, a cross-component functionality that provides a moderated way for data access. Out of the many implementations and based on the technical interoperability decisions (RESTful services) one of the most suited Authentication frameworks is OAuth. With a dedicated implementation/tweaking, OAuth also can support authorization. The decision of using OAuth for authorization would require the OAuth authorization scheme to be supported on the other two components namely the AAS Platform (AAS repository) and the "Data Services" component.

4.3.2 MaaS Platform

The MaaS Platform serves as a communication mechanism for secure transactions. The requirements for such a space are standard interfaces, data sovereignty and transactions logging (historical data). Moreover, there is a requirement for onboarding participants on the network.

These requirements are typically provided by and implemented through the main components of dataspaces. More specifically, the dataspace architecture is designed to provide security, sovereignty, and standardized data exchange between participants of a dataspace. The main component of such a system is a "Connector" component acting as the gateway to the dataspace infrastructure, but other infrastructure components are needed to ensure the requirements are fulfilled. The "Connector" component provides the standard interfaces and protocols (such as REST and IDS Information Model) that ensure interoperability. Moreover, data sovereignty is supported using common "vocabulary" model with semantic concepts (e.g., roles, contract terms) used for the verification of participants and provided by the "Identity Provider" component as part of a Dynamic Attribute Token (DAT) that verifies the participant along with scoped data. The DAT which accompanies each participant request is again validated from a "Policy Enforcement" mechanisms within the connectors. The "Logging Service" component logs registrations and data exchange transactions in a secure manner providing historical data crucial for traceability. Finally, the “Service Metadata” component supports participant discovery and registration/onboarding by maintaining an updated registry of available connectors and their metadata.

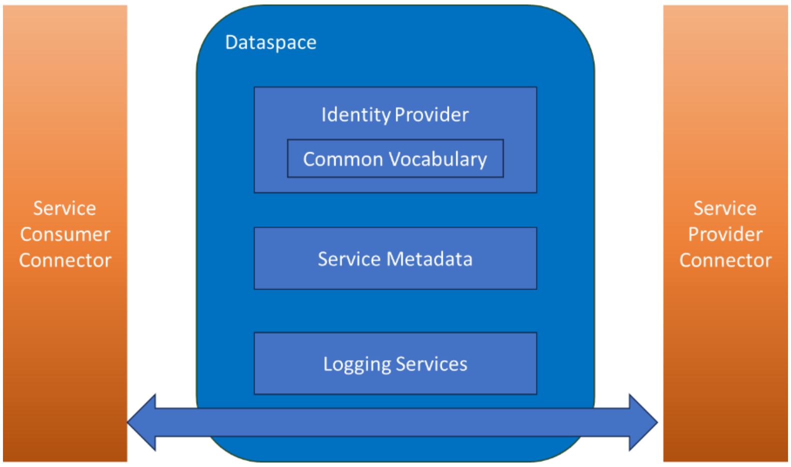


Figure 37 Main components of the MaaS Cross-company Infrastructure

Based on the components, we can define two main flows of actions between them, one is the “onboarding/registration” flow of actions where the participants wants to register its provided service and the other is the actual “service call” initiated from a service consumer.

The section below describes such communications and their interactions.

4.3.2.1 “Onboarding/Registration” flow

This flow is mainly for the service provider meaning the participant of the network that wants to make available a specific service. It starts with an interaction with the “Identity Provider” to acquire access to the dataspace, continues to the “Metadata Service” to register the service and closes with the “Logging Service”. For these transactions the service provider’s connector will be utilized.

- **Service Provider Connector → Identity Provider:** *Connector requests a Dynamic Attribute Token (DAT) from “Identity Provider” to prove its identity and access scope. On the same step “Identity Provider” consults the Comon Vocabulary to validate semantic concepts (e.g., roles, contract terms) used in attribute verification.*
- **Service Provider Connector → Metadata Service:** The Service *Provider Connector registers its services along with contracts, and metadata in the Metadata Service’s catalogue for future discovery by Service Consumers. The Metadata Service makes sure that the provided DAT allows this action to be completed.*

- **Service Provider Connector → Logging Service:** *The Service Provider logs the publication or updates of its service metadata with the "Logging Service" for traceability and making sure that the service registration adheres to appropriate/applicable rules.*

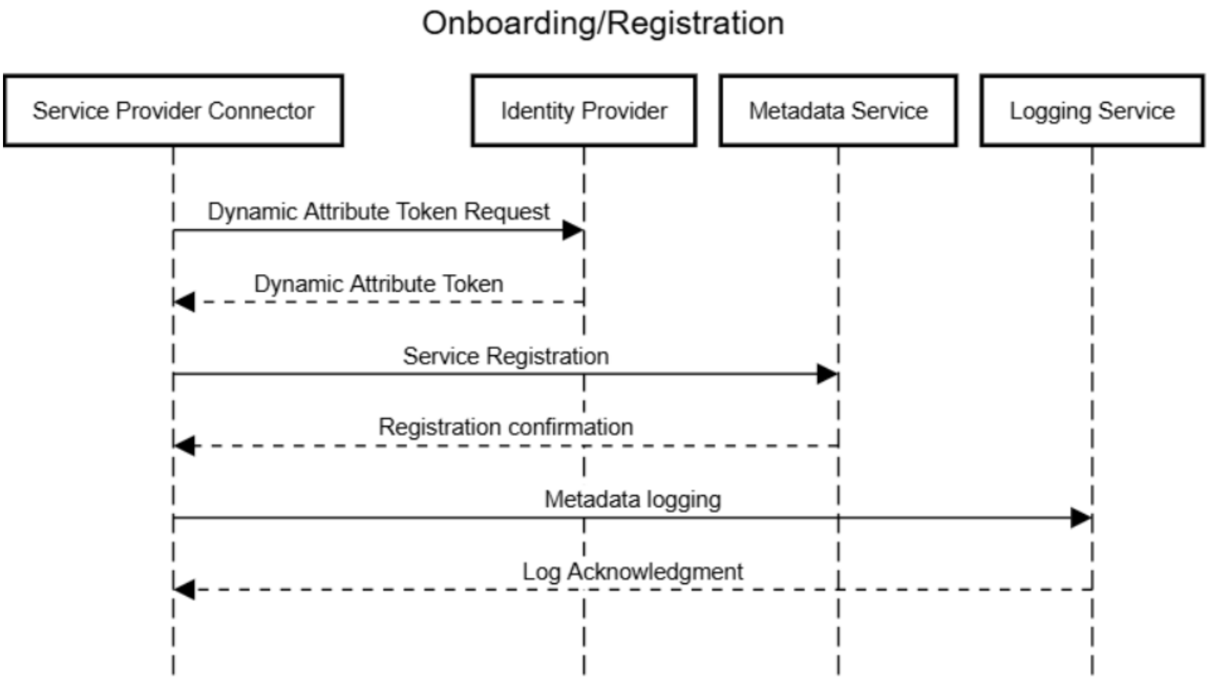


Figure 38: Onboarding/Registration flow

4.3.2.2 "Service Call" flow

- **Service Consumer Connector → Identity Provider:** The Service Consumer Connector requests a Dynamic Attribute Token (DAT) from DAPS to prove its identity and access scope.
  - **Identity Provider → Common Vocabulary:** Identity Provider consults the Common Vocabulary to validate semantic concepts (e.g., roles, contract terms) used in attribute verification.
- **Service lookup:** The following steps happens simultaneously
  - **Service Consumer Connector → Broker Service Provider:** With a valid DAT, the Service Consumer Connector queries the Broker to discover available service providers, and available metadata.
  - **Service Consumer Connector → Logging Service:** The Service Consumer Connector logs its discovery request with the Logging Service for auditing and traceability purposes.

- **Service Consumer Connector → Service Provider Connector:** Upon finding the suitable service to be "consumed" the Service Consumer Connector sends a request to the Service Provider, including the DAT and intended contract terms, requesting access to the registered service.
- **Service Provider Connector → Common Vocabulary:** The Service Provider validates semantic validity of the request using the Common Vocabulary.
- **Service Provider Connector → Logging Service:** The Service Provider logs the finalized contract to the Logging Service.
- **Service Provider Connector → Service Consumer Connector:** Upon validation and contract agreement, the Service Provider Connector securely transfers/provides the requested service to the Service Consumer Connector.

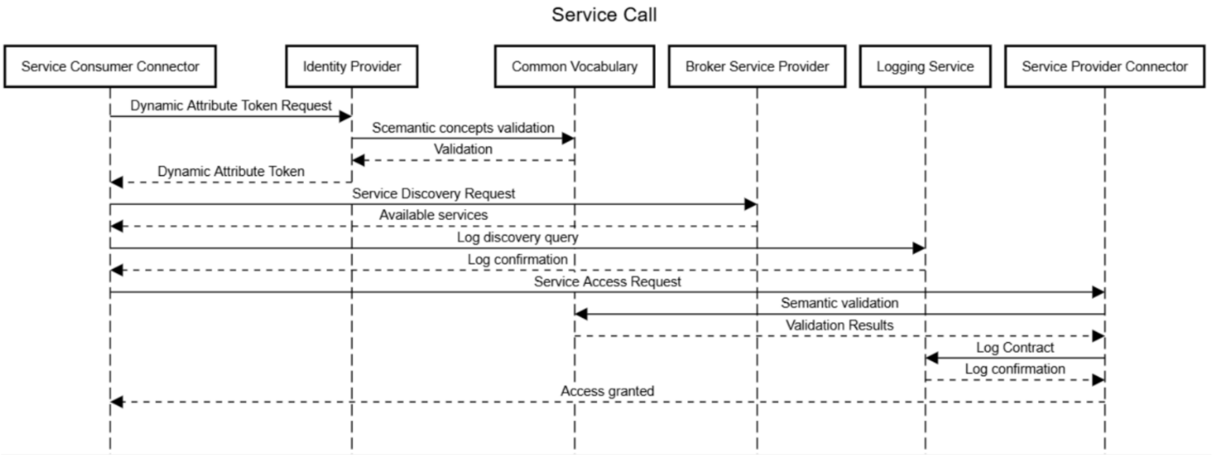


Figure 39: Service Call flow

4.4 Conceptual Architecture Implementation

The final step of the conceptual architecture is the implementation of the specific use cases to develop the RAASCEMAN platform. In the previous sections 4.1, 4.2 and 4.3 an overview of the building blocks of the architecture have been defined. This section concentrates on the interactions between the components and the infrastructure.

As a generic approach we assume that each participant of the RAASCEMAN MaaS network has an intra-company infrastructure composed by RAASCEMAN factor level support tools (section 4.1.2), and the AAS Infrastructure. The AAS Infrastructure dictates the interoperability protocol as REST calls and hosts the information layer components (section 4.2) as data models to be instantiated and populated per participant with its own specificities.

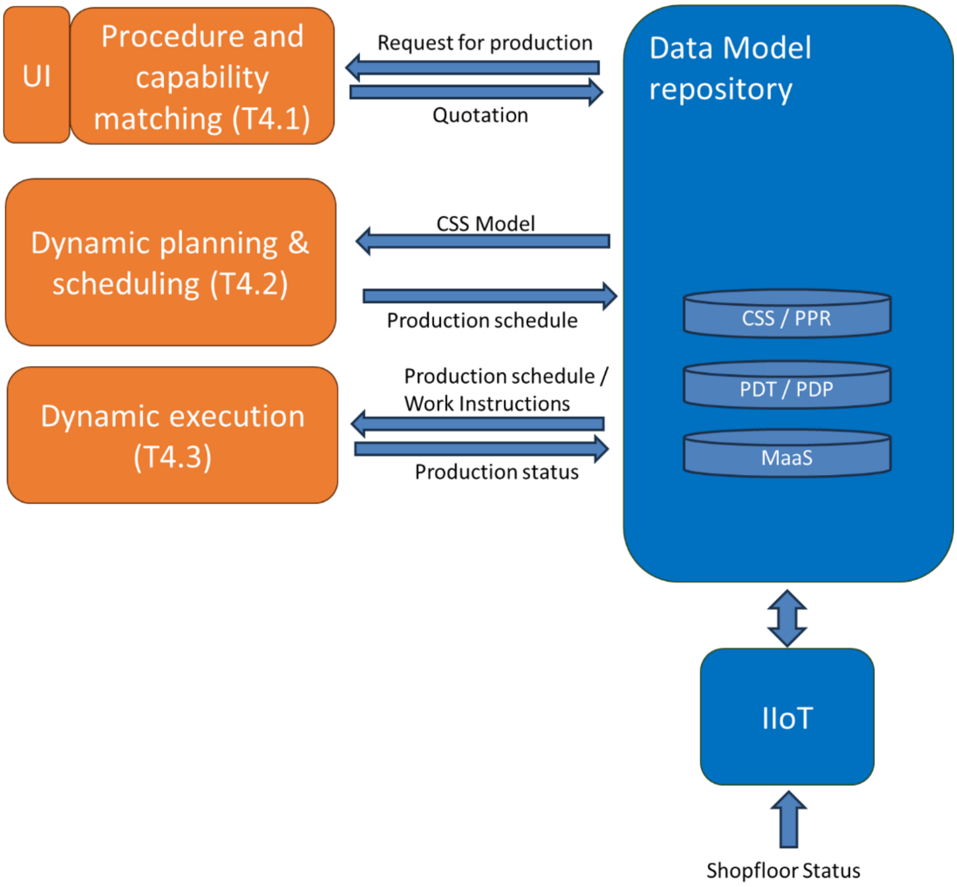


Figure 40 "AAS Infrastructure" Communication Architecture

In Figure 40**Error! Reference source not found.** we can distinguish the AAS Infrastructure composed by an IoT part providing context and historical data but also a data model repository hosting the CSS and DPT information models. These models are accessed and utilized by the Factory level support tools that read context data but also populate PDT/PDP and CSS data for providing contextual information.

The exact same information is to be utilized by the MaaS network participants. The idea here is to encapsulate the MaaS network required services as an asset having a dedicated AAS incorporating MaaS network operations like registration, request for quotation, order monitoring etc. Such an AAS would require a dedicated connection with the other participants of the network which is provided by the MaaS Gateway (i.e. an IDS Connector).

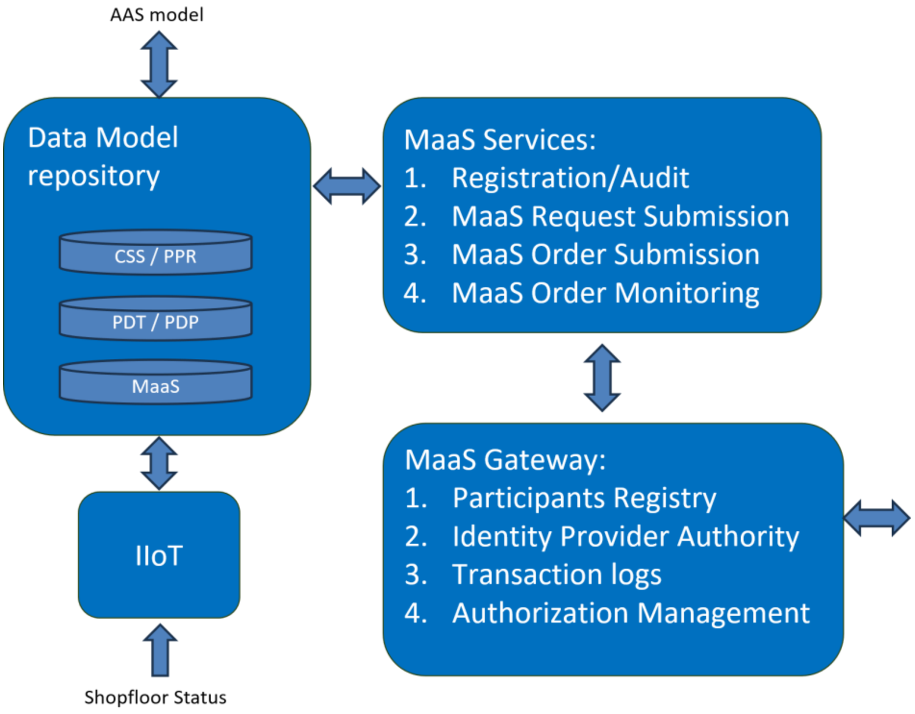


Figure 41 MaaS as an asset

Finally, supply chain level tools need to interact with the MaaS network participants in a seamless way to access the MaaS services provided by the MaaS AAS.

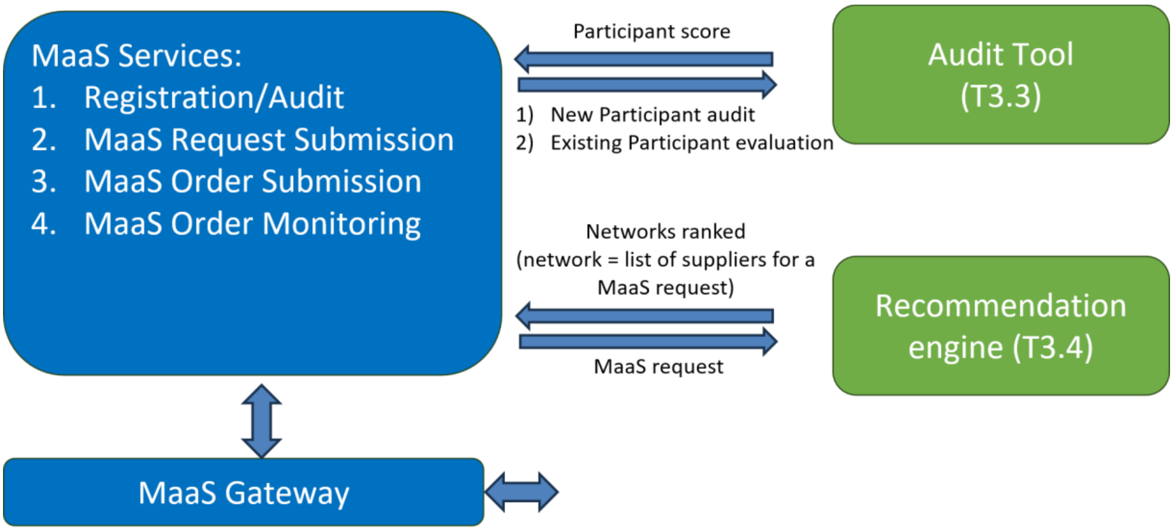


Figure 42 "MaaS Platform" Communication Architecture

Putting it all together the Figure 43 displays the infrastructure components of each RAASCEMAN MaaS network participant containing also all the components developed under RAASCEMAN context.

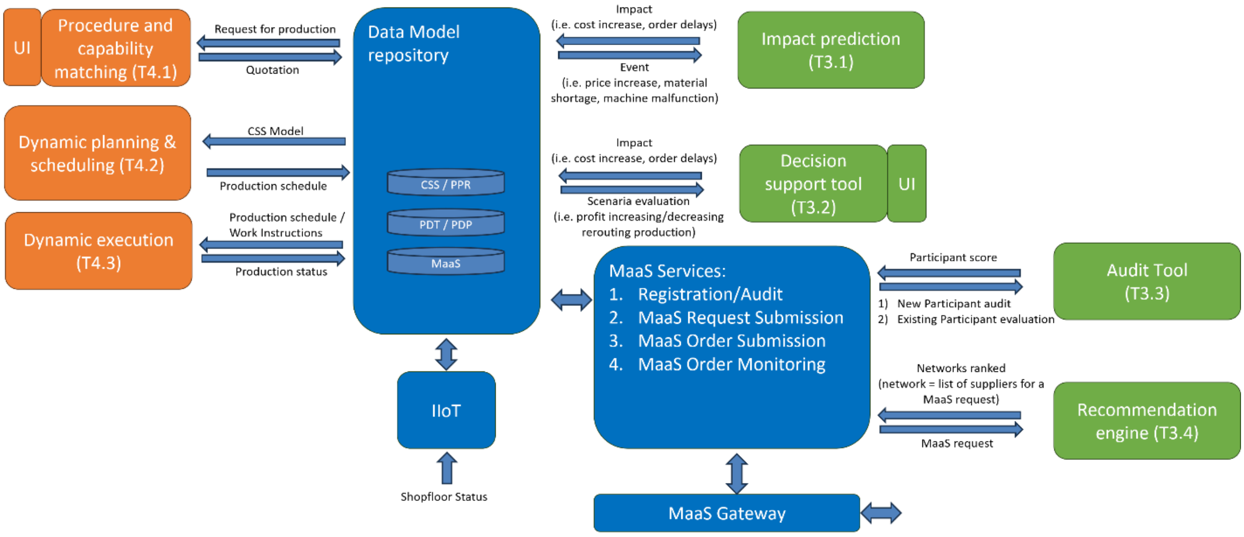

UI
Procedure and capability matching (T4.1)
Request for production
Quotation
Dynamic planning & scheduling (T4.2)
CSS Model
Production schedule
Dynamic execution (T4.3)
Production schedule / Work Instructions
Production status
Data Model repository
CSS / PPR
PDT / PDP
MaaS
IIoT
Shopfloor Status
Impact (i.e. cost increase, order delays)
Event (i.e. price increase, material shortage, machine malfunction)
Impact prediction (T3.1)
Impact (i.e. cost increase, order delays)
Scenaria evaluation (i.e. profit increasing/decreasing rerouting production)
Decision support tool (T3.2)
UI
MaaS Services:
1. Registration/Audit
2. MaaS Request Submission
3. MaaS Order Submission
4. MaaS Order Monitoring
Participant score
1) New Participant audit
2) Existing Participant evaluation
Audit Tool (T3.3)
Networks ranked (network = list of suppliers for a MaaS request)
MaaS request
Recommendation engine (T3.4)
MaaS Gateway

Figure 43 MaaS Network participant Infrastructure

The schema above provides a landscape of all the component required by a participant of the RAASCEMAN MaaS network to fully exploit RAASCEMAN functionalities without been required to support its participation with high end ERP and/or MES systems or other CAx systems. Nevertheless, the existence of similar systems (i.e. ERP, MES etc.) and other legacy systems will be supported by developing specific data bridges from the legacy systems towards the components with overlapping roles.

5 Conclusion

This deliverable sets the basis for RAASCEMAN's software and information architecture, based on the stakeholders' requirements. Through the examination of state-of-the-art models, such as RAMI4.0 and IIRA and via the link of technical needs with functional components, a multi-layered conceptual architecture model has been developed. This approach addresses various significant aspects like service capability modelling, digital twins, secure data exchange and interoperability. The resulting architecture supports intra- and cross- company collaboration and enables dynamic reconfiguration of manufacturing processes in response of disruptions.

This architectural groundwork paves the way for the development of the next work packages. Specifically, WP2 will develop the infrastructure components including support for both inter-company communication (AAS Infrastructure) and cross-company communication (MaaS Platform) and implement consistent data models (Product Digital Twin and CSS). These models will encompass various layers based on the needs defined by the architecture that was presented in this deliverable. WP3 and WP4 are responsible for the development of cross-company tools and intra-company tools respectively. These tools will be built upon the conceptual architecture and the layers that presented.

6 REFERENCES

[1] IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.

[2] ISO/IEC/IEEE 29148:2018, Systems and Software Engineering—Life Cycle Processes—Requirements Engineering.

[3] Rozanski, N., & Woods, E. (2011). Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives. Addison-Wesley.

[4] INCOSE. (2015). Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities (4th ed.).

[5] Sommerville, I. (2011). Software Engineering (9th ed.). Addison-Wesley.

[6] Hankel, Martin, and Bosch Rexroth. "The reference architectural model industrie 4.0 (rami 4.0)." *Zvei* 2.2 (2015): 4-9.

[7] Pisching, Marcos A., et al. "An architecture based on RAMI 4.0 to discover equipment to process operations required by products." *Computers & Industrial Engineering* 125 (2018): 574-591.

[8] Kirmse, Andreas, et al. "How to rami 4.0: Towards an agent-based information management architecture." *2019 International Conference on High Performance Computing & Simulation (HPCS)*. IEEE, 2019.

[9] André Pomp, Alexander Paulus, Andreas Kirmse, Vadim Kraus and Tobias Meisen, "Applying semantics to reduce the time to analytics within complex heterogeneous infrastructures", *Technologies*, vol. 6, no. 3, 2018.

[10] P. Monteiro, M. Carvalho, F. Morais, M. Melo, R. J. Machado and F. Pereira, "Adoption of Architecture Reference Models for Industrial Information Management Systems," *2018 International Conference on Intelligent Systems (IS)*, Funchal, Portugal, 2018, pp. 763-770, doi: 10.1109/IS.2018.8710550.

[11] Mirani, A.A.; Velasco-Hernandez, G.; Awasthi, A.; Walsh, J. Key Challenges and Emerging Technologies in Industrial IoT Architectures: A Review. *Sensors* **2022**, *22*, 5836. https://doi.org/10.3390/s22155836

[12] P. Leitão, S. Karnouskos, T. I. Strasser, X. Jia, J. Lee and A. W. Colombo, "Alignment of the IEEE Industrial Agents Recommended Practice Standard With the Reference Architectures RAMI4.0, IIRA, and SGAM," in *IEEE Open Journal of the Industrial Electronics Society*, vol. 4, pp. 98-111, 2023, doi: 10.1109/OJIES.2023.3262549

[13] Helmann, Alexandre, Fernando Deschamps, and Eduardo de Freitas Rocha Loures. "Reference architectures for industry 4.0: Literature review." *Transdisciplinary Engineering for Complex Socio-technical Systems–Real-life Applications* (2020): 171-180.

[14] M. Alabadi, A. Habbal and X. Wei, "Industrial Internet of Things: Requirements, Architecture, Challenges, and Future Research Directions," in *IEEE Access*, vol. 10, pp. 66374-66400, 2022, doi: 10.1109/ACCESS.2022.3185049.

[15] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman and D. O. Wu, "Edge Computing in Industrial Internet of Things: Architecture, Advances and Challenges," in *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2462-2488, Fourthquarter 2020, doi: 10.1109/COMST.2020.3009103

[16] Boyes, Hugh, et al. "The industrial internet of things (IIoT): An analysis framework." *Computers in industry* 101 (2018): 1-12.

[17] P. Satyavolu, et al., Designing for Manufacturing's 'Internet of Things'. CognizantReport. p.4 [online], (2014) Available: https://www.cognizant.com/InsightsWhitepapers/Designing-for-Manufacturings-Internet-of-Things.pdf.

[18] Pourghebleh, Behrouz, and Nima Jafari Navimipour. "Data aggregation mechanisms in the Internet of things: A systematic review of the literature and recommendations for future research." *Journal of Network and Computer Applications* 97 (2017): 23-34.

[19] S. Sirsikar and S. Anavatti, "Issues of data aggregation methods in wireless sensor network: A survey", *Proc. Comput. Sci.*, vol. 49, no. 1, pp. 194-201, 2015.

[20] Chandra, J. Vijaya, Narasimham Challa, and Sai Kiran Pasupuletti. "Authentication and authorization mechanism for cloud security." *International Journal of Engineering and Advanced Technology* 8.6 (2019): 2072-2078.

[21] H. Kim and E. A. Lee, "Authentication and Authorization for the Internet of Things," in *IT Professional*, vol. 19, no. 5, pp. 27-33, 2017, doi: 10.1109/MITP.2017.3680960.

[22] Egala, Bhaskara Santhosh, and Ashok Kumar Paradhan. "Access Control and Authentication in IoT." *Internet of Things: Security and Privacy in Cyberspace*. Singapore: Springer Nature Singapore, 2022. 37-54.

[23] M. Alramadhan and K. Sha, "An Overview of Access Control Mechanisms for Internet of Things," *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, Vancouver, BC, Canada, 2017, pp. 1-6, doi: 10.1109/ICCCN.2017.8038503.

[24] A. Niruntasukrat, C. Issariyapat, P. Pongpaibool, K. Meesublak, P. Aiumsupucgul and A. Panya, "Authorization mechanism for MQTT-based Internet of Things," *2016 IEEE International Conference on Communications Workshops (ICC)*, Kuala Lumpur, Malaysia, 2016, pp. 290-295, doi: 10.1109/ICCW.2016.7503802.

[25] Barkley, John. "Comparing simple role based access control models and access control lists." *Proceedings of the second ACM workshop on Role-based access control*. 1997.

[26] Danwei, Chen, Huang Xiuli, and Ren Xunyi. "Access control of cloud service based on ucon." *IEEE International Conference on Cloud Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[27] Ameziane El Hassani, Abdeljebar, et al. "Integrity-OrBAC: a new model to preserve Critical Infrastructures integrity." *International Journal of Information Security* 14 (2015): 367-385.

[28] Q. Wang, X. Feng, L. Wang, H. Wu and B. Düdder, "FECAC: Fine-Grained and Efficient Capability-Based Access Control for Enterprize-Scale IoT Systems," in *IEEE Internet of Things Journal*, vol. 12, no. 7, pp. 8669-8684, 1 April1, 2025, doi: 10.1109/JIOT.2024.3504825

[29] F. Yang and S. Manoharan, "A security analysis of the OAuth protocol," *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, Victoria, BC, Canada, 2013, pp. 271-276, doi: 10.1109/PACRIM.2013.6625487.

[30] Bourhis, Pierre, et al. "JSON: data model, query languages and schema specification." *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI symposium on principles of database systems*. 2017.

[31] Pezoa, Felipe, et al. "Foundations of JSON schema." *Proceedings of the 25th international conference on World Wide Web*. 2016.

[32] Lennon, Joe. "Introduction to JSON." *Beginning couchdb*. Berkeley, CA: Apress, 2009. 87-105.

[33] Bikakis, Nikos, et al. "The XML and semantic web worlds: technologies, interoperability and integration: a survey of the state of the art." *Semantic hyper/multimedia adaptation: Schemes and applications* (2013): 319-360.

[34] Sharma, Sugam, et al. "Towards XML interoperability." *Advances in Computer Science, Engineering & Applications: Proceedings of the Second International Conference on Computer Science, Engineering and Applications (ICCSEA 2012), May 25-27, 2012, New Delhi, India, Volume 1*. Springer Berlin Heidelberg, 2012.

[35] Hazra, Abhishek, et al. "A comprehensive survey on interoperability for IIoT: Taxonomy, standards, and future directions." *ACM Computing Surveys (CSUR)* 55.1 (2021): 1-35.

[36] Sengupta, Kunal, and Pascal Hitzler. "Web ontology language (OWL)." *Encyclopedia of Social Network Analysis and Mining* (2014).

[37] Needleman, Mark H. "Rdf." *Serials Review* 27.1 (2001): 58-61.

[38] Hogan, Aidan, and Aidan Hogan. "Resource description framework." *The Web of Data* (2020): 59-109.

[39] Fielding, Roy, et al. "RFC2616: Hypertext Transfer Protocol--HTTP/1.1." (1999).

[40] N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," *2017 IEEE International Systems Engineering Symposium (ISSE)*, Vienna, Austria, 2017, pp. 1-7, doi: 10.1109/SysEng.2017.8088251.

[41] B. Wukkadada, K. Wankhede, R. Nambiar and A. Nair, "Comparison with HTTP and MQTT In Internet of Things (IoT)," *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, 2018, pp. 249-253, doi: 10.1109/ICIRCA.2018.8597401.

[42] Penmetsa, C. (2024, January 14). *Representational State Transfer (REST) and design principles*. Medium. https://medium.com/codenx/representational-state-transfer-rest-and-design-principles-98640faa1ab4

[43] A. Kadadi, R. Agrawal, C. Nyamful and R. Atiq, "Challenges of data integration and interoperability in big data," *2014 IEEE International Conference on Big Data (Big Data)*, Washington, DC, USA, 2014, pp. 38-40, doi: 10.1109/BigData.2014.7004486.

[44] M. Pajpach, M. Šlauka, R. Pribiš, P. Drahoš, E. Kučera and O. Haffner, "Asset Administration Shell – Key-enabling technology of interoperability in Industry 4.0," *2025 Cybernetics & Informatics (K&I)*, Mikulov na Morave, Czech Republic, 2025, pp. 1-6, doi: 10.1109/KI64036.2025.10916454.

[45] Geibel, Fabian. "Digital Twin in Industrial Applications–How Model-Based Systems Engineering (MBSE) and Asset Administration Shell (AAS) complement each other." *Engineering for a changing world: Proceedings: 60th ISC, Ilmenau Scientific Colloquium, Technische Universität Ilmenau, September 04-08, 2023*. 2023.

[46] Pribiš, R.; Beňo, L.; Drahoš, P. Asset Administration Shell Design Methodology Using Embedded OPC Unified Architecture Server. *Electronics* **2021**, *10*, 2520. https://doi.org/10.3390/electronics10202520

[47] J. Beermann, R. Benfer, M. Both, J. Müller and C. Diedrich, "Comparison of Different Natural Language Processing Models to Achieve Semantic Interoperability of Heterogeneous Asset Administration Shells," *2023 IEEE 21st International Conference on Industrial Informatics (INDIN)*, Lemgo, Germany, 2023, pp. 1-6, doi: 10.1109/INDIN51400.2023.10218154.

[48] A. A. Malik, H. Anwar and M. A. Shibli, "Federated Identity Management (FIM): Challenges and opportunities," *2015 Conference on Information Assurance and Cyber Security (CIACS)*, Rawalpindi, Pakistan, 2015, pp. 75-82, doi: 10.1109/CIACS.2015.7395570.

[49] Kallela, Jyri. "Federated identity management solutions." *TKK T-110.5190 seminar on internetworking*. 2008.

[50] C. Martella, A. Martella and A. Longo, "European data spaces for urban digital twins: user-and implementation-driven recommendations," *2024 IEEE International Conference on Big Data (BigData)*, Washington, DC, USA, 2024, pp. 5496-5505, doi: 10.1109/BigData62323.2024.10826100.

[51] Poikola, A., Takanen, V., Laszkowicz, P., & Toivonen, T. (2023). The technology landscape of data spaces.

[52] Martella, Angelo, Cristian Martella, and Antonella Longo. "Designing Data Spaces: Navigating the European Initiatives Along Technical Specifications." *arXiv preprint arXiv:2503.15993* (2025).

[53] Arjona Aroca, J., Beltran Blanco, L., Blasco Roca, M., Saez Domingo, D., & Bernabeu Auban, J. M. (2024, October). Enabling EDIHs as Data Space intermediaries. In *Proceedings of the 4th Eclipse Security, AI, Architecture and Modelling Conference on Data Space* (pp. 18-24).

[54] Dam, T., Klausner, L. D., Neumaier, S., & Priebe, T. (2023). A Survey of Dataspace Connector Implementations. *arXiv preprint arXiv:2309.11282*.

[55] I. Matsunaga, T. Michikata and N. Koshizuka, "ITDT: International Testbed for Dataspace Technology," *2023 IEEE International Conference on Big Data (BigData)*, Sorrento, Italy, 2023, pp. 4740-4747, doi: 10.1109/BigData59044.2023.10386196.

[56] Galij, S., Pawlak, G., & Grzyb, S. (2024). Modeling Data Sovereignty in Public Cloud—A Comparison of Existing Solutions. *Applied Sciences*, *14*(23), 10803. https://doi.org/10.3390/app142310803

[57] Lopes, P. M., Guimarães, P., Pereira, T. F., & Machado, R. J. (2024). Gaia-X & Fiware: Implementation of a Federated Data Platform in Smart Cities. *Procedia Computer Science*, *239*, 1506-1515.

[58] https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/2025-i40-capabilities.pdf?__blob=publicationFile&v=5

[59] https://eclipse-edc.github.io/documentation/for-adopters/