Ref. Ares(2025)4963194 - 22/06/2025

ACCURATE

D6.1 – High-level ecosystem architecture

Actual Submission Date: 12/04/2025
Produced by: SIMAVI

Accurate

https://accurateproject.eu/

HORIZON 1.0 – 2023-12-04

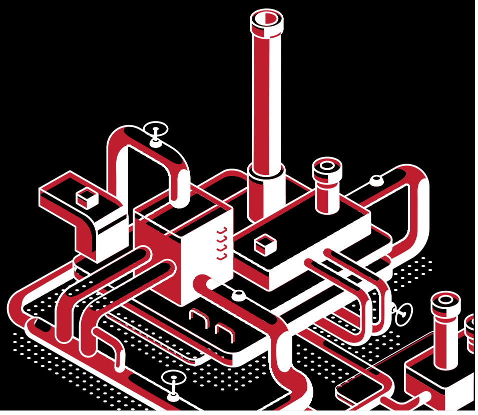
ACCURATE – HORIZON-CL4-2023-TWIN-TRANSITION-01

Grant Agreement no.: 101138269

Start date of project: 01122023 - Duration: 36 months

Funded by
the European Union

The ACCURATE project is funded by the European Union, under Grant Agreement number 101138269. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Health and Digital Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.



DELIVERABLE FACTSHEET

Deliverable D6.1	
Nature of the Deliverable:	R (Report)
Due date of the Deliverable:	M14 – 31/01-2025
Actual Submission Date:	M17 – 12/04-2025
Produced by:	SIMAVI
Contributors:	SIMAVI, IMT Atlantique, AU, EnginSoft
Work Package Leader Responsible:	SIMAVI
Reviewed by:	IMT Atlantique, SIMAVI

Dissemination level	
X	PU = Public
	PP = Restricted to other programme participants (including the EC)
	RE = Restricted to a group of the consortium (including the EC)
	CO = Confidential, only members of the consortium (including the EC)

Contents

Figures

Terms and abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
DSS	Decision Support System
ERP	Enterprise Resource Planning
EVM	Ethereum Virtual Machine
MaaS	Manufacturing-as-a-Service
MES	Manufacturing Execution System
WP	Work Package
IDSA	Industrial Designers Society of America
GDPR	General Data Protection Regulation
GUI	Graphical user interface
DT	Digital Twin
UI	User Interface
REST	Representational State Transfer
VM	Virtual Machine
UML	Unified Modeling Language
ODRL	Open Digital Rights Language
KPI	Key Performance Indicator

Public Summary

The current deliverable presents the common work of ACCURATE Consortium regarding the conceptualization of the ACCURATE architectural solution, comprising cutting-edge technologies and specific elements of the business field to which the solution is addressed.

The deliverable is the first report addressing the specific requirements of *WP6 - Decision Support and MAAS framework development* and presents the conceptual architecture of the ACCURATE ecosystem, with details on technical, logical, application and business approaches.

The report corresponds to *Task 6.1 - Service architecture, system modelling and infrastructure* and depicts the high-level architecture of the ACCURATE federated ecosystem, which aims to enable Manufacturing-as-a-Service (MaaS) and Decision Support Systems (DSS) across decentralized industrial actors. The architectural model is designed to be Gaia-X-compliant, supporting data sovereignty, modular deployment, and semantic interoperability between stakeholders.

One of the major challenges of designing the architectural solution of ACCURATE was finding the most suitable architectural model, taking into consideration both the specific of the Use cases scenarios and the complexity of the technologies involved.

Thus, the system architecture is structured into three main layers:

1. Infrastructure Layer – leveraging distributed ledger technology and compute-to-data mechanisms to ensure secure and decentralized data processing.
2. Middleware Layer – based on the Ocean Protocol stack, managing metadata, policies, and federated access control.
3. Application Layer – hosting the MaaS platform, DSS modules, digital twin registry, and user-facing portals.

Significant work was dedicated to applying the appropriate methodology of building the conceptual architectural model, as presented in ***Chapter 2 - Methodology for ACCURATE ecosystem architecture***. The design allows each industrial partner to retain control over its data and execution logic while participating in a shared ecosystem.

The architecture was developed through iterative co-design sessions with the end-users, ensuring alignment with real-world constraints and use cases, such as the AIRBUS pilot. Key features include support for pilot-specific configurations, modular workflows and processes, and integration of simulation-based digital twins.

The proposed architectural model will serve as the basis for the next deliverables D6.4 and D6.5 which present the integration process of the ACCURATE framework and for evaluation of the system demonstrators including the primary scenarios, services, state transitions, configurations and results.

1 Introduction

1.1 About this deliverable

The current deliverable was designed such as to meet the requirements of *Task 6.1 - Service architecture, system modelling and infrastructure* and describes how the systems and components work to build the ACCURATE federated ecosystem, in an integrated, interoperable and standardized manner. Different sections are dedicated to the methodological approach, architectural design, deployment environments, data flows and processes, technical components and specific Use case demonstrations.

The deliverable is the first report dedicated to the ACCURATE integrated prototype and focuses on designing the architectural model of ACCURATE solution.

The proposed solution for the ACCURATE federated ecosystem serves as the technical foundation for implementing, integrating, and scaling ACCURATE services across the European manufacturing domain.

1.2 Document structure

The document is structured as follows:

Chapter 1. Introduction: Presents a summary concerning the scope of the deliverable, its structure and the relation with other tasks and deliverables.

Chapter 2. Methodology for ACCURATE ecosystem architecture: Presents the methodological approach and specific architectural principles of designing the ACCURATE ecosystem architecture.

Chapter 3. High-Level Architecture Design: Presents the relevant aspects regarding the architectural solution of the ACCURATE ecosystem. Significant aspects about the business and storage layers of the ACCURATE framework’s components are also depicted.

Chapter 4. Technical components: Provides an overview of the software components that make up the ACCURATE ecosystem, including a global view of their organization, the data flows between them, and individual descriptions of their functionality, inputs/outputs and deployment logic.

Chapter 5. Use Cases Demonstration: Provides examples of Use Cases reflecting operational scenarios which present the interaction of components in the ACCURATE ecosystem.

Chapter 6. Conclusions: Presents the conclusions and findings corresponding to the implementation of this particular stage of the project.

1.3 Relation with other tasks and deliverables

The specific work presented in Task 6.1 is correlated with the activities and outcomes from all WPs dedicated to the definition and development of the ACCURATE technical solution: WP2 – Products and processes ontology-based matchmaking, WP3 – Digital Twins Supporting MAAS Production Adaptation, WP4 – Supply chain resilience design and stress-testing and WP5 – Data space design and implementation. A preliminary input for the Use Case Demonstration has been provided by the work from WP7 – Pilot Campaign.

Furthermore, considering the prospects for innovation, standardization and exploitation of the project results, the definition of the architectural model and the development of ACCURATE ecosystem are closely connected with the activities of WP8 – Market Uptake & business impact, Dissemination and Awareness.

2 Methodology for ACCURATE ecosystem architecture

2.1 Overview

This section presents the methodological approach and specific architectural principles of designing the ACCURATE ecosystem model. The objective was to define the architecture in an optimal way that ensures the availability of data related to the major components of the solution and their relationships.

The methodology used to design the ACCURATE architecture combines **best practices in software architecture modeling**, **Agile development cycles**, and **stakeholder-driven requirements engineering**. The overall approach is structured into three complementary dimensions:

2.1.1 Architectural Modelling using 4+1 View Model

To ensure modularity, scalability, and clarity, the ACCURATE ecosystem architecture is structured following the **4+1 view model** for standardized software architectures by Philippe Kruchten, as presented below.

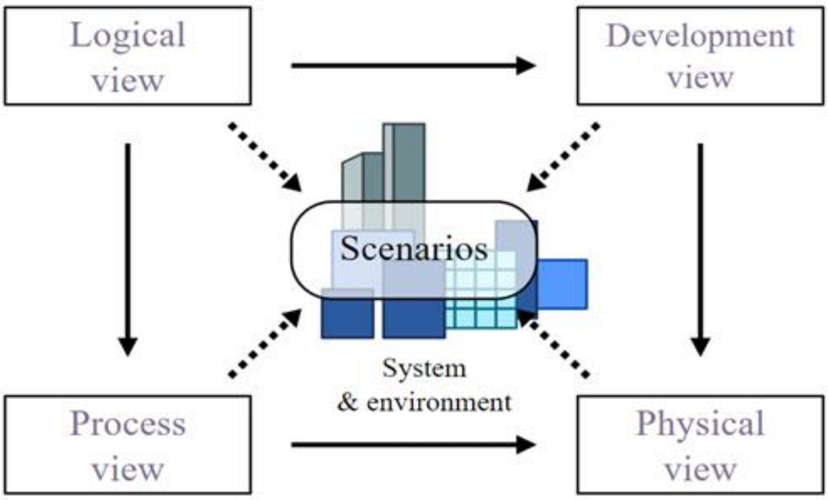


Figure 1. Architectural modelling using 4+1 View Model

This method captures the system's structure and behavior through five interrelated views:

- **Logical View** – Describes the main components (e.g., DSS, MaaS, Semantic Services) and their relationships.
- **Process View** – Shows the runtime behavior, workflows, and data flows between components.
- **Development View** – Specifies the codebase organization and microservices modularity in the development environment (Dockerized containers).
- **Physical View** – Illustrates deployment topologies per pilot, infrastructure requirements, and federation points.

- **Scenarios (Use Cases)** – Serves two major purposes: to discover the architectural elements during the architecture design and validate the proposed solution after the architecture design is completed. Concrete examples (e.g., AIRBUS pilot) are presented that validate the other views through real execution paths.

This multi-view approach ensures consistency and traceability from technical decisions to business requirements.

2.1.2 Agile and Pilot-Driven Co-Design

The architectural design follows an **iterative and Agile methodology**, aligned with the broader SCRUM-based development workflow used in ACCURATE. Key features include:

- Continuous feedback loops from WP2/WP3 (requirements, use cases, DSS concepts).
- Co-design sessions with stakeholders from each pilot (e.g., AIRBUS, EMO, TNO).
- Early prototyping and evaluation of architecture components in simulated environments.

This approach guarantees that architectural choices are grounded in real-world constraints and stakeholder needs.

2.1.3 Alignment with Federated and Gaia-X Principles

From the outset, the architecture has been developed with **federation** and **data sovereignty** in mind:

- Leveraging **Ocean Protocol** and **Gaia-X Trust Framework** for identity, access control, and policy enforcement.
- Ensuring that all data exchanges respect on-premises control and compute-to-data patterns.
- Supporting **semantic interoperability** via shared ontologies and terminology services.

The combination of agile execution, structured modeling, and federation-aware design ensures that the ACCURATE architecture is not only technically robust but also organizationally viable for cross-border industrial ecosystems.

2.2 Software global architecture methodology

The design of the ACCURATE ecosystem architecture is grounded in a *set of core principles* that ensure the system is secure, scalable, interoperable, and aligned with the European initiatives such as **Gaia-X**, **IDSA**, and **Industry 4.0**. These principles provide a foundation for enabling **cross-organizational collaboration** while ensuring **data sovereignty**, **semantic compatibility**, and **distributed intelligence**.

1. Federated and Decentralized Design

ACCURATE's architecture avoids central data aggregation by enabling **federated service execution**, **distributed deployment**, and **local data processing**. Participants maintain full control of their digital assets while integrating via common interfaces and shared governance.

2. Data Sovereignty by Design

Sensitive data (e.g., production KPIs, supplier relationships, cost structures) never leaves the organizational perimeter. Instead, the system relies on **compute-to-data** strategies and smart contracts to run analytics locally and share only insights or aggregated KPIs.

3. Semantic Interoperability

Interoperability is not just syntactic but **semantic**. All services and data assets are described using shared ontologies, metadata standards (e.g., JSON-LD, RDF), and classification models. A common vocabulary ensures compatibility across domains and organizations.

4. Modularity and Loose Coupling

Components are developed as **microservices**, independently deployable and versioned. This allows flexibility in configuration, scalability, and reuse. Interfaces follow **RESTful standards** and use **OAuth2/OpenID** for secure access control.

5. Layered and Scalable Architecture

A **layered design** separates infrastructure, middleware, orchestration, and application logic. Each layer can scale horizontally and evolve independently. The architecture can integrate additional services and participants without significant rework.

6. Trust and Compliance

Trust is embedded via **decentralized identity management**, verifiable credentials, and policy-driven access control. Compliance with **Gaia-X Trust Framework** and **EU data protection regulations (e.g., GDPR)** is ensured by design.

7. Co-Design with Industrial Pilots

The architecture is developed in **close collaboration with pilot partners** (e.g., AIRBUS Atlantic, CONTINENTAL, TRONICO), ensuring that the design reflects real operational constraints, deployment environments, specific requirements and optimization needs.

2.2.1 **Kruchten** 4+1 Methodology short description – Technical insight

The **Kruchten 4+1 View Model** is a software architecture framework that organizes a system's design into five interrelated views, each addressing specific stakeholder concerns. It promotes **separation of concerns** and ensures the architecture is well-documented, understandable, and maintainable.

- **Logical View**: Focuses on the system's functionality as perceived by end users. It models the core abstractions (e.g., components, services, data) and their relationships.
- **Development View**: Also known as the implementation view, it shows how the software is structured in terms of modules, packages, or layers. It is of particular interest to developers.
- **Process View**: Describes the system's dynamic behavior, including concurrency, synchronization, communication, and workflows between components.
- **Physical View**: Also called the deployment view, it maps the software onto the underlying hardware infrastructure. It describes node configurations, communication networks, and physical distribution.
- **Scenarios (Use-Case View)**: Ties the other four views together by showing how the system behaves in typical usage scenarios. It serves to validate the architecture through real-world examples.

In ACCURATE, this methodology ensures that the architecture is **multi-perspective**, **use-case-driven**, and **aligned with deployment and operational realities** of federated manufacturing ecosystems.

2.2.2 Motivation

The adoption of the **Kruchten 4+1 View Model** in ACCURATE is motivated by the need to address the **complexity and heterogeneity** of a federated manufacturing ecosystem. Given the involvement of multiple stakeholders—each with different roles, systems, and technical constraints—a multi-view architectural approach is essential to ensure clarity, alignment, and interoperability across all design layers.

Key motivations of the chosen design model:

- **Stakeholder-Centric Perspective**

  Each architectural view maps to the concerns of a specific stakeholder group (e.g., end users, developers, infrastructure managers, business analysts), enabling better communication, validation, and design consistency.

- **Support for Federation and Modularity**

ACCURATE must support decentralized, independently managed components (e.g., MaaS nodes, DSS engines, DT registries). The 4+1 model helps modularize these concerns across logical, process, and physical layers.

- **Integration of Use Cases into Design**

  The "+1" view — real-world scenarios — grounds the architecture in pilot-specific needs such as the AIRBUS use case. This ensures that all views are aligned with concrete requirements and expected system behavior.

- **Traceability and Maintainability**

  By clearly separating structure, behavior, and deployment concerns, the architecture remains adaptable to future updates, partner extensions, and changes in industrial context.

- **Alignment with Reference Models**

  The 4+1 methodology complements other architectural frameworks used in ACCURATE, such as Gaia-X, IDS RAM, and RAMI 4.0, ensuring conceptual and technical coherence.

2.3 Software component's description methodology

To ensure consistent understanding, traceability, and reuse across the ACCURATE ecosystem, all software components are described using a **structured methodology** that defines their functionality, interfaces, dependencies, and deployment context. This approach enables modular design, easier integration, and better alignment with pilot requirements. Within the deliverable a preliminary description of the components is made; the detailed presentation will be part of the integrated vision of ACCURATE ecosystem in further deliverables D6.4 and D6.5.

1. Standardized Component Template

Each component in ACCURATE is described using the following structured template:

Attribute	Description
Name	Unique identifier for the component
Category	Core type (e.g., MaaS, DSS, DT, Middleware, UI, Integration)
Functional Description	Summary of the component's main role and responsibilities
Inputs / Outputs	Expected input and output data formats (incl. semantic types, APIs)
Interfaces	REST, gRPC, MQTT, WebSocket, or file-based interfaces
Dependencies	Required external services, libraries, or other components

Execution Context	Whether the component runs centrally or at a local (pilot) node
Deployment Format	Docker container, VM, native app, etc.
Security & IAM	Authentication/authorization methods, token usage, ID federation support
Monitoring / Logging	Available observability metrics and interfaces

This template is used throughout the documentation, including system design, integration guides, and deployment playbooks.

2. Semantic Annotation & Ontology Alignment

All components are linked to semantic metadata to ensure:

- Discoverability across the ecosystem
- Compatibility with Gaia-X and Industry 4.0 vocabularies
- Alignment with ontologies in the ACCURATE knowledge graph

3. Lifecycle Tagging

Each component is versioned and tagged across lifecycle stages:

- **Draft / Prototype**
- **Alpha / Beta**
- **Stable**
- **Deprecated**

This supports continuous evolution and maturity tracking of software assets during the project.

4. Visual Modeling

Where applicable, components are documented with:

- **UML Component Diagrams**
- **Sequence Diagrams**
- **Deployment Views (per pilot or generic)**

These visuals support alignment across technical teams and simplify onboarding of new contributors.

2.4 Development methodology

The development methodology adopted in ACCURATE combines **Agile principles**, **modular architecture design**, and **DevSecOps practices** to ensure that software components are developed, validated, and deployed efficiently across the federated ecosystem.

1. Agile, Incremental Delivery

Development is organized around **sprint-based cycles**, using SCRUM or Kanban workflows depending on team preferences. Each sprint delivers:

- New or updated features for core components (e.g., DSS, MaaS modules)
- Integrations with pilot-specific configurations
- Feedback loops with stakeholders (technical & business)

Each feature is traceable to **user stories and requirements**, typically derived from real-world pilot scenarios (e.g., AIRBUS production Use case).

2. Modular Microservice Architecture

All components are built using **loosely coupled microservices**, each of which:

- Performs a single, well-defined function
- Is exposed through **RESTful APIs**, **OpenAPI specifications**, or **message queues**
- Can be independently deployed, tested, and scaled

This allows **reuse across pilots**, faster development iteration, and easier maintenance.

3. DevSecOps Integration

Security and automation are embedded into the development lifecycle through:

- **CI/CD Pipelines** (GitHub Actions, GitLab CI) for automated testing, building, and packaging
- **Containerization** using Docker, enabling environment-independent deployment
- **Secrets & Identity Management** handled via integrated vaults and Gaia-X-compliant brokers
- **Static Analysis & Vulnerability Scans** embedded in build chains.

4. Documentation & Semantic Alignment

Each component is:

- Described using the standardized template (see Section 2.3)
- Linked to ontologies and semantic descriptors to enable federated discovery
- Versioned and tracked through centralized Git-based repositories.

5. Pilot-Driven Feedback Integration

Feedback from pilots is incorporated during:

- Sprint reviews and demos
- Cross-WP validation sessions
- Dedicated pilot integration weeks.

This ensures that development decisions are grounded in real-world feasibility, and the architecture evolves in tandem with operational constraints.

3 High-Level Architecture Design

3.1 Overview

The ACCURATE ecosystem is built upon a federated architecture designed to support decentralized, secure, and scalable collaboration between multiple manufacturing stakeholders. This architectural choice reflects the need to enable data and service interoperability without enforcing centralization, in alignment with Gaia-X principles such as data sovereignty, transparency, and trust.

The high-level architecture is structured to allow each participant (or node) to retain control over its digital assets — including production data, services, simulation models, and optimization logic — while enabling collaboration through well-defined interfaces and shared semantic frameworks. The federation is governed by shared policies, common ontologies, and trust mechanisms, which ensure that only authorized, authenticated, and auditable exchanges take place.

Key to this architecture is the distinction between shared metadata, which is distributed and discoverable across the ecosystem, and private operational data, which remains under the control of each participant. This balance between discoverability and confidentiality is achieved through a combination of technologies such as distributed ledgers, compute-to-data execution, and semantic matchmaking services.

The architecture also separates concerns between the main components of ACCURATE, the Manufacturing-as-a-Service (MaaS) and Decision Support Systems (DSS), allowing each to evolve independently while still enabling interaction. MaaS focuses on service discovery, matchmaking, and orchestration across the manufacturing value chain, while DSS focuses on scenario modeling, optimization, and simulation-based decision making.

The following sections introduce the core principles that guide the design of federated architectures, the main components involved, the technical stack adopted in ACCURATE, and the specific architecture patterns used for MaaS and DSS.

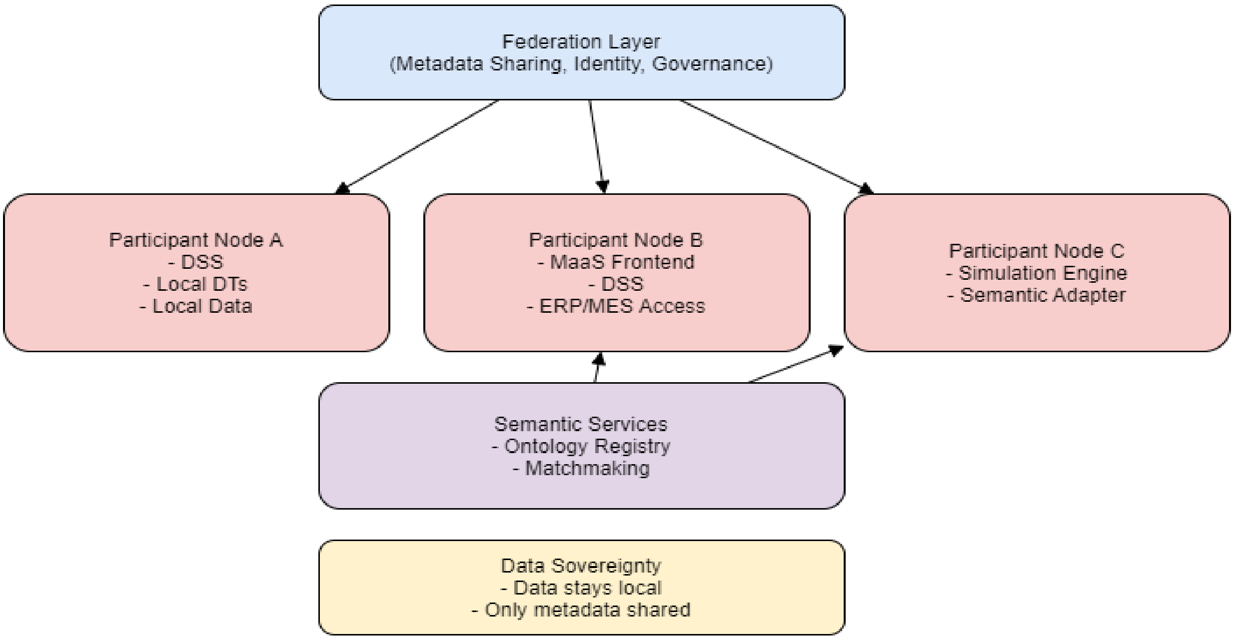


Figure 2. High-level architecture design of ACCURATE framework

3.1.1 Key Principles of Federated Architecture

The ACCURATE architecture adheres to a set of foundational principles that define how participants interact in a decentralized, collaborative, and trusted environment. These principles ensure that the ecosystem remains scalable, secure, and aligned with industrial realities such as data sensitivity, organizational autonomy, and technological heterogeneity.

3.1.1.1 Autonomy of Subordinate Systems

Each participant (e.g., manufacturer, integrator, technology provider) retains full control over its own infrastructure, data, services, and business logic. This principle enables flexibility and independence, allowing participants to join or leave the ecosystem without disrupting others. Components like Digital Twins or Decision Support Systems can be deployed locally and integrated into the federation via standard interfaces.

3.1.1.2 Interoperability

Interoperability is achieved using open standards (e.g., REST APIs, JSON-LD), shared ontologies, and semantic metadata. This ensures that diverse systems can communicate, interpret, and act on shared information even if they use different underlying technologies.

3.1.1.3 Decentralized Data Ownership

Data remains under the control of the original provider. Only metadata is published into the federated layer (e.g., through the Ocean Protocol stack), enabling discovery without compromising ownership or confidentiality. For sensitive data, compute-to-data execution allows algorithms to run locally on protected assets.

3.1.1.4 Scalability

The federated model supports the inclusion of new participants and services without centralized bottlenecks. Microservice-based architecture allows components to be scaled horizontally and deployed in various configurations, tailored to each use case or industrial setting.

3.1.1.5 Security

Security is embedded throughout the architecture via decentralized identity management, policy enforcement, and end-to-end encryption. Authentication and authorization are handled through systems like Keycloak or Gaia-X compliant Identity & Trust Frameworks, ensuring that only validated actors can interact with services and data.

3.1.2 Main Components of the Federated Architecture

A federated architecture in the context of ACCURATE requires a well-defined set of core components that collectively enable collaboration, data protection, and service orchestration across distributed participants. These components are loosely coupled, modular, and interoperable, ensuring flexibility and independence across organizational and technical boundaries.

3.1.2.1. Federation Layer

This layer provides the foundational infrastructure for governance, identity management, and metadata coordination, and comprises the following components:

- **Distributed Ledger**: Used to register and track metadata, service offers, and access policies (e.g., via EVM-compatible blockchain).
- **Decentralized Identity (DID)**: Ensures secure identification and authentication of actors in the ecosystem.
- **Access Control & Policy Engine**: Enforces usage rules using smart contracts or policy frameworks like ODRL.

3.1.2.2. Metadata & Semantic Services

Central to interoperability and discoverability, this layer includes:

- **Ontology Registry**: Manages domain-specific vocabularies to enable semantic alignment.
- **Metadata Registry (e.g., Aquarius)**: Stores structured descriptions of services, datasets, and digital twins.
- **Semantic Matchmaking Engine**: Enables intelligent service discovery based on capability, context, and requirements.

3.1.2.3. Local Execution Environments

Each participant maintains its own execution infrastructure, including:

- **Digital Twin Engines**: Used for local simulation, modeling, and prediction tasks.
- **DSS Modules**: Host scenario management, optimization workflows, and decision support analytics.
- **Data Connectors / Adapters**: Interface with ERP, MES, or sensor data sources in a standardized way.

3.1.2.4. Workflow Orchestration Layer

Facilitates the chaining and execution of federated processes:

- **Workflow Manager**: Defines and executes multi-step operations involving DTs, DSS, and external APIs.
- **Execution Manager**: Schedules and runs simulation jobs, manages configuration parameters, and gathers KPIs.

3.1.2.5. User Interface Layer

Provides stakeholders with visual access to platform functionalities:

- **MaaS Portal**: Allows service providers and consumers to interact through discovery, pricing, contracting interfaces.
- **DSS GUI**: Exposes scenario builders, dashboards, and analytics results.
- **Federation Monitoring Tools**: Track operational status, compliance, and traceability across nodes.

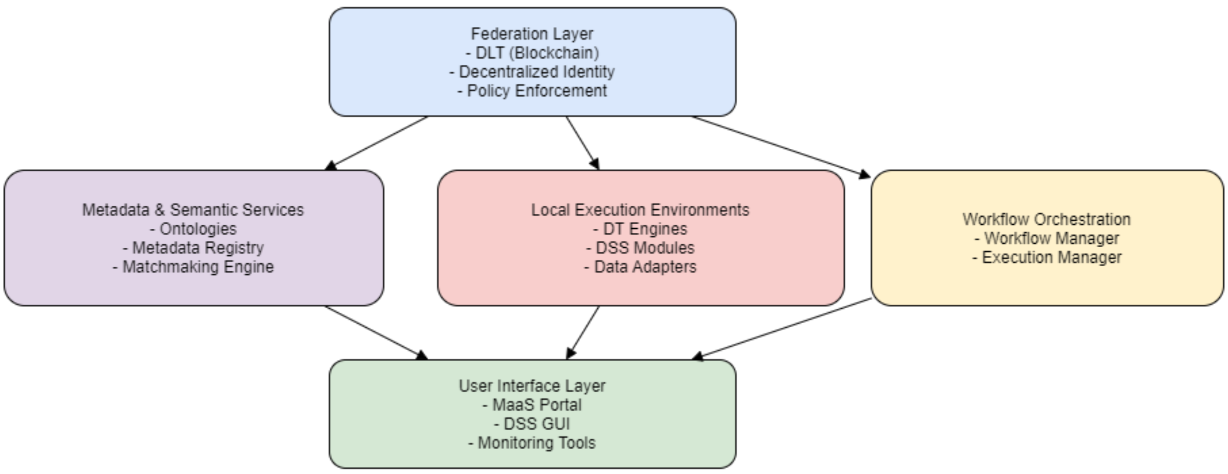


Figure 3. Main components of the ACCURATE Federated Architecture

3.1.3 Technical Stack and Tools

The ACCURATE federated architecture leverages a modern, modular, and open-source technology stack designed to ensure interoperability, scalability, and compliance with Gaia-X data sovereignty principles. The selected tools and frameworks cover infrastructure, data access, orchestration, simulation, and user-facing services.

3.1.3.1. Infrastructure and Network Layer

- **Blockchain Layer**:
  - Based on an **EVM-compatible distributed ledger** (e.g., Pontus-X, Polygon, Oasis Sapphire).
  - Manages smart contracts, transaction logs, and access policies.
- **Compute-to-Data (C2D)**:
  - Ensures that data remains local, and algorithms are moved to the data.
  - Used to enforce data privacy and avoid IP leakage.

3.1.3.2.. Middleware Layer – Ocean Protocol Stack

- **Aquarius** – Metadata storage and retrieval system.
- **Provider** – Handles authorization and enforces compute-to-data workflows.
- **DataTokens (ERC20)** – Represent access rights to datasets or services.
- **Smart Contracts** – Automate registration, consumption, and payments.
- **Metadata Cache** – Optimizes semantic search and indexing.

3.1.3.3. Service Layer Components

- **MaaS Platform**:
  - Built using microservices (e.g., Spring Boot, Node.js).
  - Provides matchmaking, contract generation, and scheduling features.
- **DSS Backend**:
  - Executes simulations and scenario analysis.
  - Integrates Digital Twins, workflow definitions, and KPI analytics.
- **Digital Twin Engines**:
  - Simulation environments based on Python, MATLAB, or proprietary tools.
  - Parameterized and wrapped for API-based invocation.

3.1.3.4. Orchestration and API Integration

- **Workflow Manager**:

    - Uses tools such as **Apache Airflow**, **Camunda**, or custom BPM engines to define and execute multi-step processes.
- **Execution Manager**:
    - Interfaces with Dockerized simulation environments.
    - Manages versioning, configuration, and result aggregation.

3.1.3.5. Semantic & Interoperability Layer

- **Ontology Definitions**:
    - OWL, JSON-LD, or RDF formats, used for standardization.
- **SPARQL/SHACL Engines**:
    - Validate semantic integrity and enable advanced matchmaking.
- **Terminology Services**:
    - Assist in aligning pilot-specific vocabularies with shared domain ontologies.

3.1.3.6. Identity and Access Management

- **Keycloak** or Gaia-X compatible ID brokers are used for:
    - Authentication (OAuth2, OpenID Connect).
    - Authorization and policy enforcement.
    - User and role federation across nodes.

3.1.3.7. Deployment & DevOps

- **Containerization**: All components are delivered as Docker containers.
- **CI/CD Pipelines**: Implemented with GitLab CI or GitHub Actions.
- **Monitoring & Logging**: Prometheus, Grafana, and ELK stack for observability.

3.1.4 Federated Architecture for Manufacturing as a Service

The **Manufacturing as a Service (MaaS)** architecture within ACCURATE enables distributed manufacturing providers and consumers to interact seamlessly across organizational boundaries. The design supports **service discovery**, **semantic matchmaking**, **collaborative production**

workflows, and **contractual agreements** — all while respecting **data sovereignty** and system autonomy.

3.1.4.1. Architectural Objectives for MaaS

- Enable buyers to search and evaluate providers across the federation based on capabilities, availability, and KPIs.
- Allow manufacturing providers to advertise services semantically (e.g., machine types, processes, certifications).
- Ensure transactions (negotiations, contracts, pricing) are secure, traceable, and policy compliant.
- Avoid centralized data collection by enabling compute-to-data and on-premises execution.

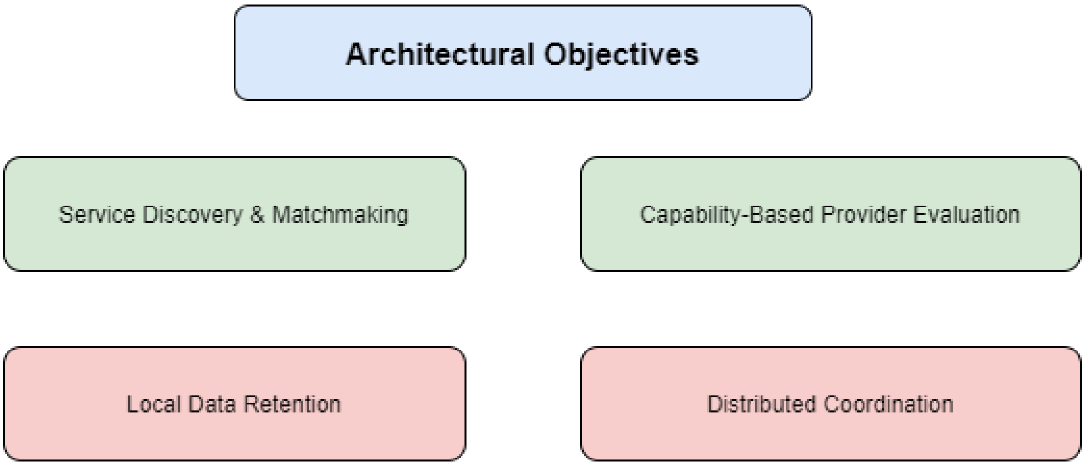


Figure 4. Architectural objectives of MaaS

3.1.4.2. Core Functionalities of MaaS

Function	Description
Service Publishing	Providers register their capabilities and constraints into the metadata registry.
Semantic Matchmaking	Ontology-driven discovery engine matches buyer needs to service offerings.
Scheduling & Orchestration	Coordinated workflows allocate production tasks to selected providers.
Contract Management	Smart contracts and policy engines define how services are consumed and monitored.

3.1.4.3. MaaS Component Interactions

- **MaaS Portal**: The user interface where buyers and suppliers interact with the system.
- **Semantic Services**: Support the matchmaking engine with ontological alignment and query resolution.
- **Workflow Manager**: Chains selected services and orchestrates their execution.
- **ERP/MES Adapters**: Integrate with legacy systems at provider sites to assess feasibility and trigger production.

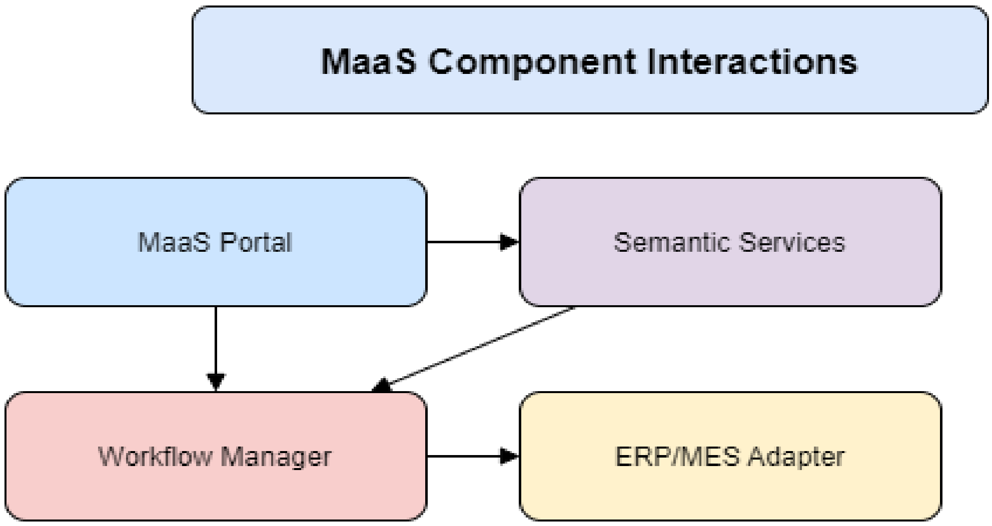


Figure 5. MaaS Component Interactions

3.1.4.4. Federated Deployment Pattern

Each provider hosts a **local MaaS node** that includes:

- Metadata registry and matchmaking endpoint
- ERP/MES connector
- Contract enforcement logic
- Identity and access management interfaces.

Meanwhile, the **federation layer** provides:

- Discovery and routing of queries
- Shared ontology registry
- Distributed logging and policy enforcement via blockchain

This architecture enables scalable, secure, and economically viable distributed manufacturing, turning Europe's fragmented industrial landscape into an agile, connected value network.

3.1.5 Federated Architecture for DSS

In the ACCURATE ecosystem, the **Decision Support System (DSS)** plays a critical role in enabling scenario modeling, simulation, and data-driven decision-making across distributed manufacturing actors. The federated architecture for DSS ensures that simulations and optimizations can be executed **locally** or **collaboratively**, depending on the operational context, while **preserving data confidentiality** and **enabling semantic interoperability**.

3.1.5.1. Architectural Objectives for DSS

- Allow each participant to host its own DSS engine and integrate it with local Digital Twins (DTs), ERP/MES systems, and data sources.
- Enable federated scenario execution, where results from multiple nodes can be aggregated without exposing sensitive business logic.
- Maintain extensibility to support custom workflows, KPIs, and optimization strategies per pilot.

3.1.5.2. Key Architectural Elements of DSS

Component	Role
DSS Core Engine	Manages execution of decision scenarios, integrates DTs and KPIs.
Scenario Manager	Defines and stores templates for manufacturing decision scenarios.
Digital Twin Interface	Connects to local or remote DTs and passes context/configuration.
Execution Manager	Orchestrates multi-step simulations, handles compute-to-data logic.
KPI Aggregator	Collects local KPIs and combines results into a federated view.
DSS Portal / API	Allows users to define, run, and analyze decision scenarios.

3.1.5.3. Data & Execution Flow

1. A user defines a decision scenario (e.g., production optimization) via the DSS Portal.
2. The DSS invokes semantic services to match required DT models and data sources.
3. Local nodes execute simulations through DT engines (possibly using surrogate models).
4. Results are validated and aggregated using predefined KPIs.

5. Feedback is provided to the user through visual dashboards or APIs.

This model supports **both single-node execution** (for local decision support) and **multi-node federated scenarios** (e.g., supply chain-level simulation).

3.1.5.4. Privacy and Federation

The DSS architecture complies with ACCURATE's principles of **data sovereignty** by:

- Using **compute-to-data** to execute models without exporting raw data
- Sharing only **aggregated or anonymized results** across nodes
- Securing all transactions via **Gaia-X identity and policy services**

3.1.5.5. Integration with MaaS and Other Layers

The DSS is fully integrated into the ACCURATE ecosystem via:

- **Common metadata and ontologies** shared with MaaS and DT registries
- **Workflow Manager** for triggering simulations as part of service compositions
- **Shared logging and monitoring** via the federated control plane

This ensures DSS remains aligned with the broader federated architecture and supports end-to-end traceability of decision-making processes.

3.2 ACCURATE System Architecture

The ACCURATE system architecture defines how the various components of the federated MaaS and DSS ecosystem interact across layers, organizations, and technical boundaries. The design integrates concepts from **Gaia-X**, **Ocean Protocol**, and **federated analytics**, aligning them with real-world requirements from pilot use cases like AIRBUS Atlantic.

3.2.1 Layered Architecture Overview

The system is organized into ***five logical layers***, each with distinct responsibilities and technical roles:

1. ***Infrastructure & Trust Layer***
    - EVM-compatible blockchain (e.g., Pontus-X) for transaction logging, identity, and policy control.
    - Gaia-X-compatible identity federation, credential verification, and access logging.
2. ***Federated Middleware Layer***
    - Ocean Protocol stack for metadata management, contract enforcement, and compute-to-data execution.

- Semantic layer with ontology registry and metadata interpretation services.

3. ***Service Orchestration Layer***
    - MaaS orchestration platform that supports matchmaking, scheduling, and production coordination.
    - DSS workflow and execution manager for running simulation-driven optimization pipelines.
4. ***Application Layer***
    - Decision Support System (frontend/backend), MaaS portal, and user dashboards.
    - Digital Twin simulation engines and KPI aggregation interfaces.
    - ERP/MES adapters and integration gateways.
5. ***Pilot-Specific Deployment Layer***
    - Modular containers deployed in each pilot (e.g., AIRBUS), including localized DTs, DSSs, and data connectors.
    - Workflow extensions and pre-configured assets tailored for each industrial setup.

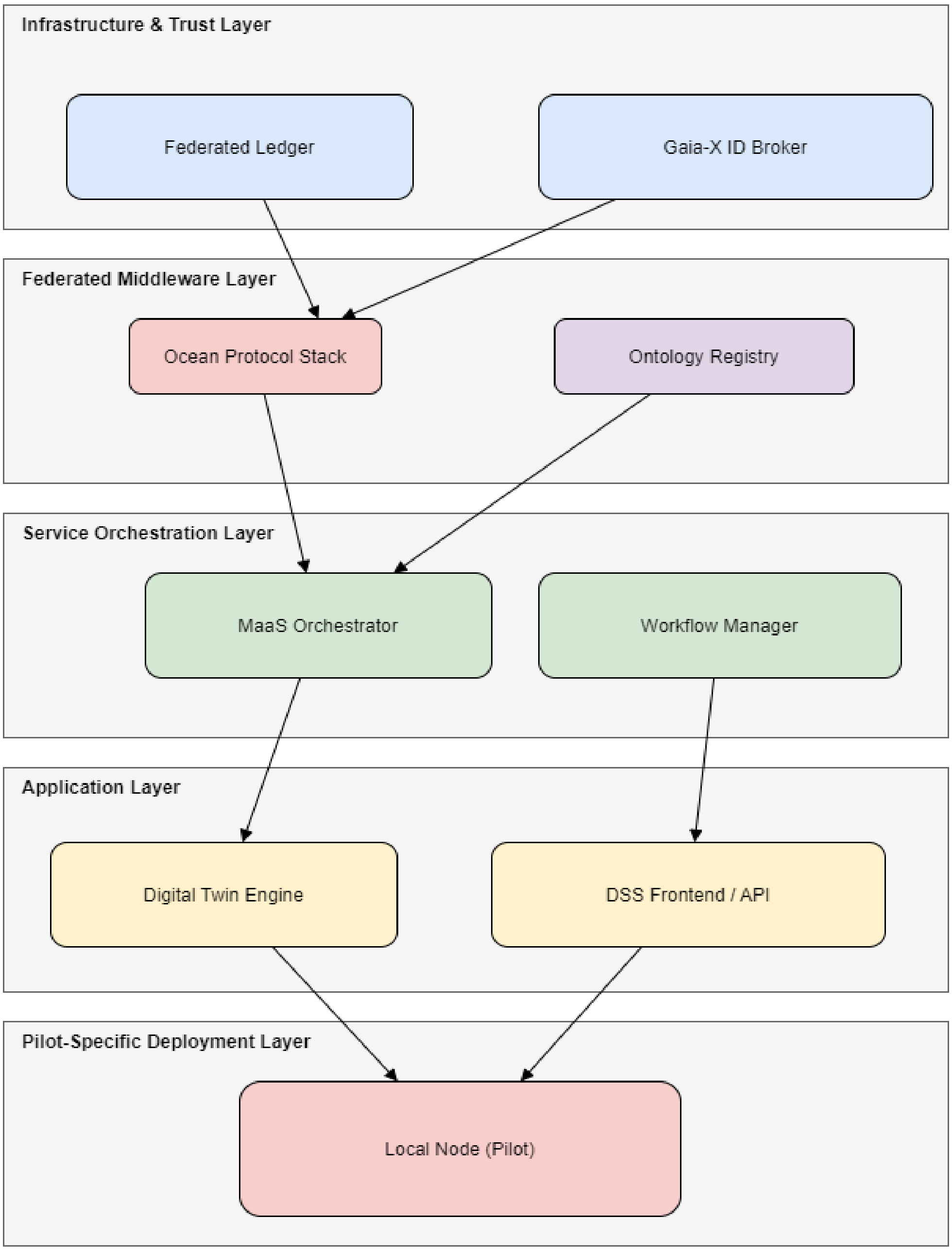


Figure 6. Layered architecture overview of the ACCURATE ecosystem

3.2.2 Deployment and Execution View

Each participant in the ACCURATE ecosystem can operate a local node that includes:

- A **metadata registry** (Aquarius)
- A **provider module** (for secure access control and compute)
- DT simulation environment and KPI management

- DSS components and API gateway

The **federated layer** acts as a backbone to:

- Publish metadata and ontologies
- Route queries across organizations
- Enforce access policies and monitor interactions

All data-sensitive computations take place on-premises using **compute-to-data**, while results and analytics are aggregated using secure APIs.

3.2.3 Component Highlights

Component	Role
Federation Ledger	Stores immutable transactions, logs, and smart contract-based policies
Semantic Matchmaker	Resolves ontologies and ranks service matches
Workflow Manager	Coordinates scenario execution and step orchestration
DSS Engine	Simulates scenarios using DT inputs and optimization models
MaaS UI Portal	Interface for users to discover, contract, and monitor service execution
Local ERP Connector	Exposes real production schedules and capacity to the MaaS engine

3.2.4 Federation Control Plane

In addition to operational interactions, ACCURATE defines a **control plane** that supports:

- Service onboarding and credential management
- Monitoring and observability
- Semantic alignment and update propagation
- Versioning of workflows and simulation configurations

This architecture ensures long-term extensibility, auditability, and trust, critical for cross-enterprise manufacturing scenarios.

3.3 Logical View

3.3.1 Introduction

The **Logical View** focuses on the **functionality** of the ACCURATE system from a **system-level perspective**, independent of physical deployment or runtime behavior. It identifies the **main functional components**, their relationships, and how responsibilities are divided between **central coordination services** and **local (subordinate) nodes** deployed at each pilot site.

This view is especially relevant for understanding the **semantic, orchestration, and integration logic** that underpins federated operation across Decision Support Systems (DSS), Manufacturing-as-a-Service (MaaS), and Digital Twin (DT) components.

3.3.2 Central Component and Subordinates

The ACCURATE ecosystem is logically structured into a **central federated component** and multiple **subordinate (local) components**. This separation reflects how responsibilities and data are distributed in a **privacy-preserving**, **autonomous**, and **interoperable** manner.

- **Central Component (Federation Layer)**:
  - Provides shared services such as:
    - Metadata registry (Aquarius)
    - Ontology registry
    - Matchmaking engine
    - Federated monitoring and orchestration coordination
  - Maintains a **global semantic index** and **contract layer** for MaaS and DSS.
- **Subordinate Components (Local Nodes)**:
  - Operate independently at each pilot/organization
  - Host Digital Twin Engines, local DSS modules, and ERP/MES integration points
  - Interact with the central layer via secure APIs
  - Maintain control over data and simulation logic.

3.3.3 Structural Overview: Central Component

The **central component** includes the following logical modules:

Module	Function
Federation Gateway	Entry point for requests, enforces identity and policies
Metadata Registry	Stores service descriptions, ontological mappings
Semantic Matchmaker	Performs ontology-based service discovery

Contract Engine	Manages service-level agreements using smart contracts
Monitoring Hub	Collects health/status metrics from all subordinate nodes
Workflow Coordinator	Triggers cross-node workflows involving multiple DSS/DTs

All components are **stateless and scalable**, allowing for horizontal expansion and fault-tolerance.

3.3.4 Structural Overview: Subordinate Component

Each **subordinate node** includes its own logical architecture tailored to pilot-specific requirements:

Module	Function
Local DSS Engine	Executes decision scenarios using pilot-specific models
Digital Twin Engine	Runs simulation, surrogate models, or AI pipelines
Execution Manager	Coordinates workflow steps within the local context
ERP/MES Connector	Bridges factory systems and provides live data to simulations
Data Vault / Proxy	Handles secure, controlled data access for federated calls
Node Gateway	Communicates with central services (REST APIs, tokens, logs)

These components ensure **semantic alignment**, **federated participation**, and **full local control**.

3.3.5 Comparison between Central and Subordinate Components

Aspect	Central Component	Subordinate Component
Governance	Managed at federation level (neutral coordination)	Managed by pilot or organization
Responsibility	Metadata, semantic matchmaking, orchestration	Execution of services, local optimization

Data Access	No access to raw data	Full control of local data
Autonomy	Limited – coordination role	Full autonomy
Workload	Stateless, coordination-heavy	Stateful, computation-heavy
Deployment	Typically cloud-hosted or consortium-owned	On-premises, pilot-specific

3.4 Development View

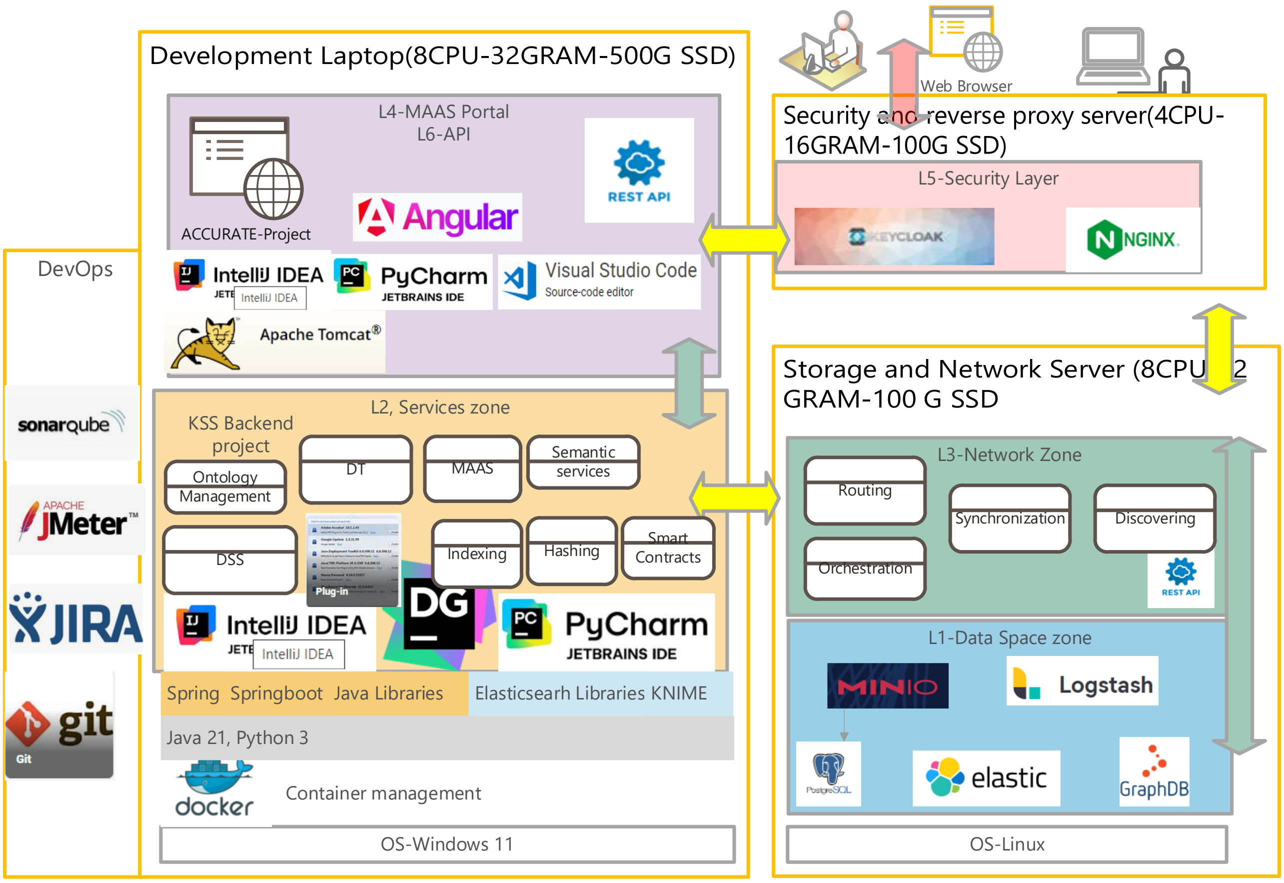


Figure 7. Development view of the architectural model

3.4.1 Key Components of the Development View

The Development View describes how the ACCURATE system is structured from a **software engineering perspective**, including development tools, technologies, and component layering.

Key components identified in the development environment, organized on the representative layers (L1 ... L6), are the following:

1. Code & Project Structure

- **Frontend Layer (L4 & L6)**:
  - Angular used for the MaaS portal and RESTful API consumers

  - Developed and tested in Visual Studio Code
- **Backend Layer (L2 Services)**:
  - Core services (DSS, DT, Semantic Services, Ontology Manager) implemented in Java and Python
  - Managed as microservices within a Spring Boot-based backend
- **Integration & Orchestration**:
  - Local components communicate via REST APIs
  - Hashing, semantic mapping, and contract enforcement modules developed in KNIME or custom Python logic

2. Development Tools & IDEs

- IntelliJ IDEA, PyCharm, and Visual Studio Code used for service development
- Version control managed through Git and GitHub
- Dependency and container management through Docker

3. DevOps & Quality Tools

- **Continuous Integration / Testing**: JIRA, JMeter, SonarQube used for tracking, performance testing, and code quality
- **CI/CD Pipelines**: Configured via Git-based workflows, triggered from the shared repositories

4. Security & API Gateway

- **Keycloak** for authentication, authorization, and identity federation
- **NGINX** used as a reverse proxy for secure access to backend services

5. Runtime Environment

- **OS Contexts**:
  - Development laptop running **Windows 11** with Docker containers
  - Backend services hosted on **Linux-based servers**
- **Data Layer Tools**:
  - MINIO, Logstash, ElasticSearch, GraphDB used for federated logging, data storage, and semantic data space services

3.4.2 How It Applies to Both Central and Subordinate Components

The development architecture supports both **central coordination services** and **subordinate local deployments**, using a consistent tooling and deployment model:

Aspect	Central Components	Subordinate Components
Service Development	Same technology stack (Java, Python, Spring Boot)	Same stack with extensions for pilot-specific logic
Frontend & UI	Angular-based MaaS Portal and DSS interface	Custom or embedded GUIs possible at node level
Security Stack	Centralized Keycloak and NGINX for access federation	Local instances possible for offline pilots
Integration Services	Ontology registry, semantic services, matchmaking	ERP connectors, DT engines, DSS modules
Data Storage	Federated metadata and logs using GraphDB, Elastic	Local databases and in-memory storage per pilot
DevOps Toolchain	Unified Git/CI pipeline for all partners	Forked branches or isolated stacks per organization

This setup ensures **reusability of engineering assets**, **standardized deployment** using Docker containers, and **traceable versioning** for every ACCURATE software component.

3.5 Deployment View

The **Deployment View** describes how ACCURATE software components are physically deployed across the system infrastructure — including development machines, pilot nodes, federation services, and security gateways. It captures the **runtime topology**, **inter-node communication**, and **infrastructure profiles** used for running both central and subordinate services.

3.5.1 Deployment Architecture Overview

The deployment of ACCURATE components follows a **hybrid model** combining:

- **Centralized services** (e.g., metadata registry, semantic services, federation gateway)
- **Distributed local nodes** (one per pilot site or industrial partner)
- **Secure access layer** for federated identity and API gateway management

All services are containerized using **Docker** and orchestrated using **Docker Compose** or **Kubernetes** (depending on complexity and scale).

3.5.2 Infrastructure Roles

Node Type	Description
Development Laptop	Used by engineers to develop, test, and package microservices locally
Storage & Network Server	Hosts production-ready containers, databases, and orchestrators (Linux OS)
Security Gateway Server	Hosts **Keycloak** (IAM) and **NGINX** reverse proxy for API routing/security

Each environment reflects different layers of the system (L1–L6), mapped onto physical hosts with CPUs, RAM, and SSD as required (see the architectural solution described in Figure 7).

3.5.3 Deployment by Layer

Layer	Host	Key Components
L1 – Data Space	Server (Linux)	MinIO, Logstash, ElasticSearch, GraphDB
L2 – Service Zone	Laptop / Server	DSS, DT, Semantic Services, Contract Manager, ERP Connectors
L3 – Network Zone	Server	Routing, Synchronization, Service Discovery, Workflow Execution
L4 – UI Portal	Laptop / Server	Angular-based MaaS Portal, Scenario Builder GUI
L5 – Security Layer	Security Server	Keycloak, NGINX, API Token Validators
L6 – API Layer	All nodes (via NGINX)	REST APIs for internal and external service communication

3.5.4 Deployment Considerations

- **Federated Isolation**: Each node (pilot or partner) runs its own stack locally. No raw data is sent externally; only metadata and results may be shared.
- **Compute-to-Data Compatibility**: Simulation workflows run where the data resides. DSS orchestrators schedule execution across the network.
- **Security Hardening**: Identity federation and API access policies enforced through Keycloak and reverse proxy configurations.
- **Containerization and Portability**: All services are Dockerized to ensure uniformity across different environments.

3.5.5 Monitoring and Observability

- Each deployment includes a monitoring stack (Prometheus + Grafana or Elastic Stack) to capture:
    - Service uptime and errors
    - Resource usage (CPU, memory, I/O)
    - Workflow and DSS activity logs

This enables **traceability, accountability, and resilience** within each deployment, whether it is a central federation node or a local manufacturing partner.

3.6 Process View

The **Process View** describes the **dynamic behavior** of the ACCURATE system — focusing on **runtime interactions**, **message flows**, and **execution workflows** between components across the federation. It captures **how the system works**, not just how it's structured.

This view is essential to understand the **sequence of operations**, **cross-node orchestration**, and **data flow control** for Decision Support System (DSS) scenarios, Manufacturing-as-a-Service (MaaS) interactions, and semantic service coordination.

3.6.1 Execution Workflow: Federated DSS Scenario

This process involves running a decision-making simulation using Digital Twins hosted at multiple organizations.

Actors:

- End user (DSS Portal)
- Local DSS Engines (pilot nodes)
- Federation services (semantic layer, orchestrator)
- Semantic Registry & DT Metadata

Steps:

1. **Scenario Definition**
   User defines a decision scenario via the DSS UI, selecting process constraints and KPIs.
2. **Semantic Resolution**
   The DSS requests matchmaking via the **semantic services**, identifying appropriate DTs and data endpoints based on metadata.

3. **Workflow Construction**
   A federated **execution plan** is generated and validated by the Workflow Manager.
4. **Compute-to-Data Execution**
   Simulation jobs are dispatched to **local DSS engines**. Each executes the model locally, preserving data privacy.
5. **Result Aggregation**
   KPIs from each node are collected via secure APIs and aggregated by the DSS core.
6. **Feedback Delivery**
   Results are visualized for the user through dashboards and exported via DSS APIs.

3.6.2 Execution Workflow: MaaS Discovery and Contracting

This process focuses on how a manufacturer discovers and engages services from another participant in the federated marketplace.

Steps:

1. **Service Query**
   A user searches for a manufacturing capability (e.g., laser cutting) via the MaaS Portal.
2. **Semantic Matchmaking**
   The matchmaking engine queries the metadata registry and returns a ranked list of providers.
3. **Capability Check & Availability Validation**
   The orchestrator queries selected providers to confirm availability and constraints.
4. **Contract Proposal**
   A smart contract is generated based on the offer and business terms.
5. **Execution or Scheduling**
   If accepted, the workflow is launched; otherwise, negotiation is iterated.

3.6.3 Runtime Synchronization and Monitoring

To ensure end-to-end coordination across all services:

- Synchronization agents ensure event alignment between orchestrator and local nodes.
- Logs are streamed via Logstash or OpenTelemetry to a central or federated monitoring hub.
- Errors and threshold breaches trigger alerts through the federation control plane.

3.6.4 Security and Policy Enforcement (Cross-cutting)

Across all processes, the following security mechanisms are applied:

- **Identity Verification** via Keycloak before any API interaction
- **Token-based Authorization** for federated service calls
- **Smart Contract Logging** to ensure non-repudiation of service agreements
- **Audit Trails** for DSS scenarios and contract executions

3.7 Use Case View

The deliverable focuses mainly on the technical aspects of the ACCURATE ecosystem, but it also identifies the requirements and challenges on operational and business level, alongside with facing the upcoming implementation activities.

The 4+1 view model for standardized software architectures describes the logical, physical, process and development views correlated with the Use case scenarios, as presented in *Chapter 5 – Use Case demonstration*. Use case view is related to the functional and business levels and implies the end-user validation of the architectural approach.

As depicted in deliverable D7.1, the ACCURATE ecosystem will be evaluated and validated through three pilot demonstrators, each of them reflecting a different manufacturing environment: Airbus Atlantic, Tronico and Continental.

Each pilot has defined their own Use Cases, based on their real needs and requirements. The business processes have been analyzed, described and formalized in Use case scenarios. Chapter 5 presents in a comprehensive form the appropriate scenarios and the specific use cases for each Pilot, with the corresponding technical components of the ACCURATE platform.

4 Technical components

This section provides a detailed overview of the software components that make up the ACCURATE ecosystem, including a global view of their organization, the flow of data between them, and individual descriptions of their functionality, inputs/outputs, and deployment logic. This section presents a global view of the components; the technical details will be presented further, in deliverables D6.4 and D6.5 dedicated to the integration process of ACCURATE ecosystem.

4.1 Global View of the Components

The ACCURATE architecture is composed of modular, interoperable components distributed across **central federation services** and **subordinate local nodes**. These components interact to support key capabilities such as semantic discovery, workflow orchestration, Digital Twin integration, and decision-making.

Global Component Categories:

Category	Examples
Federation Services	Metadata Registry, Ontology Manager, Contract Engine
Semantic Services	Matchmaking Engine, Ontology Resolver, Semantic Validator
Orchestration Tools	Workflow Manager, Execution Scheduler, Result Aggregator
MaaS Services	Service Discovery, Capability Description, Pricing Modules
DSS Components	DSS Core Engine, Scenario Manager, KPI Aggregator
Digital Twins (DTs)	Simulation Engines, Surrogate Models, Data Ingest Interfaces
Adapters & Connectors	ERP/MES Bridge, Data Connectors, C2D Execution Wrappers
Access & Security	Keycloak, NGINX, Token Issuer, IAM API
Monitoring & Logging	Prometheus, Logstash, ElasticSearch, GraphDB

Each component is either hosted centrally (cloud/federation node) or locally at pilot sites.

4.2 Data Flow Between Components

The data flow within ACCURATE follows the **principles of data sovereignty**, meaning that **no sensitive raw data** is transferred between nodes. Only metadata, semantic queries, or anonymized results are exchanged.

Federated DSS Flow (example)

1. **User Input**: Scenario initiated via DSS Portal (input params + objectives)
2. **Metadata Query**: Semantic services match inputs to compatible DTs
3. **Workflow Trigger**: Execution Manager builds and launches local jobs
4. **Compute-to-Data**: DSS logic executed locally with secure KPI output
5. **Result Aggregation**: KPIs collected via APIs, visualized centrally

MaaS Contracting Flow (example)

1. **Discovery**: Client queries semantic registry for service capabilities
2. **Matchmaking**: Compatible offers are returned from metadata registry
3. **Negotiation & Contracting**: Smart contracts are proposed, signed, and logged
4. **Orchestration**: Production or simulation tasks are scheduled and launched

5. **Monitoring**: Execution status and metrics fed to the dashboard and logs

Secure flows are enforced with:

- OAuth2 tokens (via Keycloak)
- NGINX-protected APIs
- Smart contract logging
- Semantic payload validation

4.3 Component Description

Each component is described using a structured template as defined before, in section 2.3. Below is presented a summary of some key components:

Example: DSS Core Engine

Attribute	Description
Function	Executes multi-step simulation scenarios using DTs and inputs
Inputs	Scenario parameters, simulation configs, DT models
Outputs	KPIs, result datasets, trace logs
Interfaces	REST API, JSON-LD
Execution Context	Local node
Security	Authenticated API access, logs stored via Elastic
Deployment	Docker container, Linux-based

Example: Ontology Manager

Attribute	Description
Function	Hosts and manages domain ontologies used in semantic matchmaking
Inputs	RDF/OWL models, updates via API
Outputs	Resolved terms, alignment suggestions
Interfaces	SPARQL, REST
Execution Context	Central federation node
Deployment	Docker container, GraphDB

5 Use Cases Demonstration

This section presents the Use Cases perspective of the architectural model, with focus on the integration and interaction of the components of ACCURATE ecosystem. The applicability of technical components to the business processes is graphically represented in Figure 8.

As presented in deliverable D7.1, the ACCURATE solution provides a comprehensive framework that spans the entire flow, from physical manufacturing assets to business processes. This is achieved through the development of *a suite of digital capabilities* (also called *functions*) and services, based on Decision Support Systems (DSS) and Manufacturing-as-a-Service (MaaS). These functions and services will leverage data from manufacturing software systems (such as MES, ERP, PLM, etc.), integrated within an open, federated, and trusted data space.

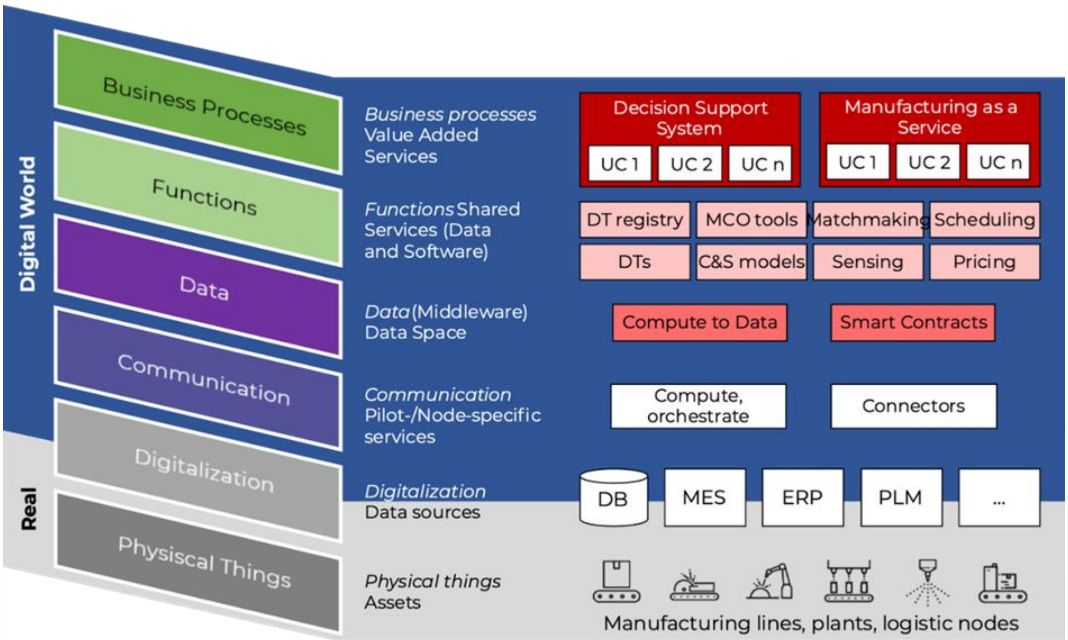


Figure 8. Use Cases in the High-level architectural model of ACCURATE

The ACCURATE platform will be tested and validated through three pilot demonstrators, each in different industrial settings: Airbus Atlantic, Tronico, and Continental. These demonstrators will showcase how the ACCURATE methodology, digital tools, decision support systems (DSS) and MaaS services can improve manufacturing operations management at the shop floor level, End-to-End supply chain operations, and orchestration of networks of supply chains.

Demonstrator at Airbus Atlantic

The focus targeted by the ACCURATE demonstrator for Airbus Atlantic relies on a better understanding of the impacts caused by disruptive events, enabling the design of optimized solutions throughout the development of DT-based industrial systems.

The ACCURATE demonstrator for Airbus Atlantic will showcase the added value of:

- **Transition from static models to a data-centric approach:** Currently, models such as Manufacturing Flow Charts, Build Processes, and Value Stream Maps are defined in charting software, resulting in static representations disconnected from ERP, MES, and other operational systems. This leads to data discontinuities and requires manual input of information already available across systems like ERP, MES, resource skill databases, and asset registries. A data centric model will be demonstrated by allowing to deliver a sign source of truth for the needed data, harmonized definition of each parameter or criterion, and better-informed decision making.
- **Industrial system modelling through Digital Twins (DTs):** The demonstrator will showcase the use of DTs to simulate various normal and abnormal operating scenarios, enabling the assessment of the impact of disruptive events across the value chain. The demonstration will highlight how this capability supports decision-making (including e.g., capacity planning and scheduling, system reconfiguration, re-ranking of supply chain orders, and forecasting of industrial performance) under normal and abnormal operating conditions.

  In addition, the demonstrator will feature scenario-based explorations to support disruption mitigation decisions, such as implementing dual sourcing of critical components or sub-assemblies from geographically diverse suppliers to reduce lead times and mitigate risks.

- **DT (and MaaS) integration for adaptive responses:** The demonstrator will showcase how the integration of digital twins (and MaaS) enables the industrial system to respond swiftly and effectively to unexpected events through corrective or preventive actions. The ACCURATE platform, built on a federated architecture, will demonstrate its ability to dynamically reconfigure the supply chain by leveraging available capacity and capabilities across distributed manufacturing and logistics nodes and provide answers to the following questions: Which resources are missing or are exposed to risk of becoming unavailable (human skills, components/critical components, jigs, tools)? What is the status of work at the plant and supply chain level and what will be the next ones? How to maintain adherence to the production plan through dynamic reconfiguration of assembly sequence?, etc.). This will support adaptation of production plans, supplier orders, and efficient allocation of resources in response to changing conditions.

Manufacturing decision-makers will benefit from post-processed data generated by the Digital Twin functions (and MaaS services).

Demonstrator at Tronico

At Tronico, the demonstrator will showcase a human-centered and automated approach to decision-making for both internal manufacturing operations and external supply chain management:

- **Smart manufacturing operations management:** The demonstration will show how DSS (and MaaS) components support human decision-makers (such as program, production, and supply chain managers), by enhancing their ability to understand complex situations, generate insights, and take informed actions. The system will demonstrate decision analytics capabilities including production control (e.g., dispatching, scheduling), production planning (e.g., batching), and demand-supply matching. These functionalities will be underpinned by performance- and resilience-driven will integrate core ACCURATE technologies such as trusted data spaces and multi-level Digital Twins.

- **External supply chain resilience**: The demonstrator will also showcase how scenario-based simulations and stress tests can be used to assess the impact of external events on the supply chain of Tronico. Through these simulations, the system will help identify critical components and define both conventional (siloed redundancy, diversification, agility, flexibility) and MaaS-based preventative and reactive measures enabled by the federated data space and ecosystem services of the ACCURATE platform.

Demonstrator at Continental

At Continental, the demonstrator will highlight how the integration of DT and DSS enables resilient and collaborative decision-making across manufacturing and supply chain operations.

- **Human-centric decision support:** The demonstration will show how the DSS integrates multi-scale and multi-level digital twins to support human decision-makers in handling complex manufacturing and logistics challenges. It will highlight how the system enables real-time insight generation and supports both preventative and corrective actions (conventional and MaaS-based), helping decision-makers respond swiftly and effectively to disruptions.
- **System integration and modular supply chain representation:** The demonstrator will also showcase the integration of data from existing systems (including SAP, MES, and other enterprise platforms) within the ACCURATE environment. This integration will enable the creation of a modular, dynamic representation of the supply chain and manufacturing nodes. The demonstration will show how this modular view supports early detection of disruptions and enables timely and effective response strategies, thereby strengthening overall supply chain resilience, performance, and sustainability.

Across all three demonstrators, the ACCURATE platform will be validated as a comprehensive solution for streamlining data flows across systems, supporting both manufacturing and supply chain operations, enhancing responsiveness to disruptions, and improving manufacturing resource sharing and sustainability across a given supply chain (DSS-oriented functions) or network of supply chains (MaaS-oriented services). The pilots will provide practical evidence of effectiveness and scalability of the ACCURATE ecosystem in different industrial settings.

6 Conclusion

This deliverable presents relevant aspects of the ACCURATE architectural solution, focusing on the methodological, logical and technological approaches. Additionally, the use case view has been described to reflect the specific business needs of the Pilots and the mapping with the technical components of the ACCURATE ecosystem. The architectural model will serve as ground for the next deliverables D6.4 and D6.5 which present the integration process of the ACCURATE framework and for evaluation of the system demonstrators including the scenarios, services, state transitions, configurations and results.

Bibliography

Kruchten, P. (1995). Architectural Blueprints—The “4+1” View. *IEEE Software 12 (6)*, 42-50.

SIMAVI. (2021). QA-Procedures and Work Instructions (In Romanian). Bucharest, Romania.