Grant Agreement Number: 957204 (H2020-ICT-38-2020)
Project Acronym: MAS4AI
Project Start Date: 1st October 2020
Project Full Title: Multi-Agent Systems for Pervasive Artificial Intelligence for assisting Humans in Modular Production

MAS4AI

D3.1 - Implementation of knowledge models

Dissemination level:	PU (Public)
Date:	March 2022
Deliverable leader:	TNO
Contributors:	DFKI, Semaku, Tecnalia, Volkswagen, Flexis, LMS
Reviewers:	DFKI
Type:	Report
WP / Task responsible:	WP3 / Task 3.1
Keywords:	AAS, Ontology, Semantics, Knowledge model

Executive Summary

The purpose of this document is to describe the formal knowledge models for the agent framework in MAS4AI. It includes the formalization of agents for the use cases, representing relevant agent knowledge and the knowledge model used to support the orchestration of the agents at the production level process.

This document describes the approach taken to model the agents, the mapping to the AAS structure that underpins the models and the actual AAS templates as a result. The document also describes the ontology that is defined to allow orchestration, its role in the MAS4AI framework and the approach taken to define the model.

Document History			
Version	**Date**	**Contributors**	**Description**
0.1	20-02-2022	TNO	Initial structure
0.5	23-03-2022	All	Draft texts AAS
0.6	24-03-2022	TNO	Draft texts ontology
0.9	31-01-2022	TNO	Initial version for review
1.0	13-04-2022	TNO, DFKI	Review comments processed, final version

Table of Contents

Table of Figures

1 Introduction (TNO)

Formal knowledge models are one of the underpinnings of the MAS4AI project. MAS4AI implements a Multi Agent System (MAS), with many assets in the manufacturing processes represented by agents. Knowledge models provide a common reference for knowledge among the agents and function as a basis for a common language in the use cases and in the project.

There are two different sources of knowledge for the agent in MAS4AI: (1) knowledge about a manufacturing application domain, which is defined in the ontology and (2) characteristics and properties of individual agents, which are defined in the Asset Administration Shell (AAS). Within MAS4AI the latter serves to communicate more specific semantics and defines the messages sent between agents.

The design approach to knowledge models of both the ontology and the AAS is grounded in principles of reusability and extensibility. This is reflected in the fact that the ontology is based on existing models for manufacturing and aims to extends those. The AASs in turn are engineered according to a methodology that focusses on iterative design in multiple phases and a component-based approach. Close collaboration with the use case owners to determine the intended use of agents and what components should be part of the AAS models ensures alignment with the use cases.

The main purpose of this deliverable is twofold:

- It describes the approach taken to define the ontology, its role in the MAS4AI framework and the application of the ontology in the use cases.
- It describes the approach taken to model the AASs for the agents, the structure of the AAS models for the eight agent types that have been defined in the project and the descriptions of the various components that constitute these models.

The document concludes with considerations on the role of the ontology, some design challenges with respect to the AASs and the application of the generic AAS types in use cases, which may require use case specific extensions.

2 Model requirements

2.1 Role of the ontology in the MAS4AI framework

The ontology in the MAS4AI framework is designed for the purpose of enabling knowledge sharing and reuse: it stores (semantic) knowledge about the manufacturing process of an organization and makes the knowledge-based organization in a MAS unambiguous.

A single agent in MAS4AI requires at least two different sources of knowledge [Zini99]:

- domain knowledge - information about a manufacturing application domain
- agent knowledge - characteristics and properties of other agents

The agents in MAS4AI manipulate information related to some manufacturing application domain (Figure 1). Knowledge about this domain can be incorporated into the MAS4AI ontology. Whereas individual agents have a scope that spans their own characteristics and computational knowledge, the MAS4AI ontology stores knowledge about the manufacturing process outside the scope of the knowledge of the individual agents. In order to do so, the ontology needs to define concepts of the MAS4AI process and their relations: it represents knowledge that relates the types of individual agents, describes their capabilities and ability to execute certain tasks that need to be performed in the process of the domain.

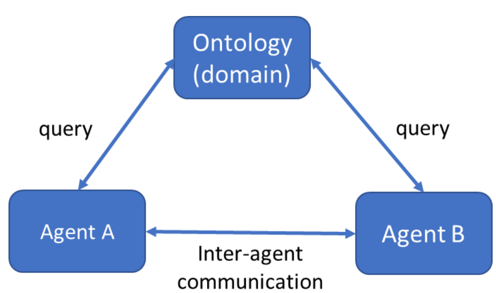

Figure 1 Role of a domain ontology in MAS

The ontology is implemented in an RDF knowledge base (RDF store), which means that the overall structure and concept definitions adhere to the RDF schema language. The knowledge base can be regarded as a semantic extension to an AAS hosting - and AAS registry platform in the MAS4AI framework: it is both a means to create process definitions and query the process on a schema level. Whereas an AAS hosting platform can handle simple queries on AASs and serve as a yellow pages directory, the ontology can handle more complex queries about the process. Types of queries are e.g., finding an agent that has the skill to perform a certain activity, while not being an agent of type X etc. Complexity of queries includes ones over multiple concept relations and e.g. negations, as is supported in the RDF schema language.

The ontology and the RDF store are not meant to store instance information, i.e., current properties, statuses of agents or processes. This information is available in the asset administration shells (AAS) that are created for the individual agents (cf. subsequent sections). The only instance information available, is the reference to agents by means of a (semantic) ID, which connects the ontology and the RDF knowledge base to the AAS hosting platform and the AAS instances.

2.2 Role of the AAS

While the MAS4AI ontology provides a way to query all agents registered in the MAS4AI framework and describes some high-level semantics, the AAS is used to define individual agents and the properties of that specific agent. Within MAS4AI it serves as a way to communicate more specific semantics and defines the messages being sent between agents, as agents may request or provide properties as defined in the AAS.

In this project the AAS does not store data regarding the workflow or environment, and is only used to describe agent types in AAS templates, and agent instances through the instantiation of those AAS templates.

A variety of semantic modelling methods could have been used for this purpose, including the RDF ontology-based approach used for the general MAS4AI overview. However, as the AAS standard is already gaining traction in industry it is a more familiar way for companies to interact with semantic models and its presentation closer matches the perception of decision makers in industry. Moreover, the AAS allows the creation of new open models which are interoperable with already existing models.

3 Ontology

This chapter describes the design process of the ontology, the base structure of the ontology and its functional use when implemented in the RDF store as part of the MAS4AI framework applied in the use cases.

3.1 Design process

The MAS4AI ontology aims to build on existing ontologies available in the domain of manufacturing, in order to reuse existing (and well-founded) knowledge models and prevent reinventing definitions where established ones already exist.

In the manufacturing domain, one can characterize a typical manufacturing system according to three notions of a PPR model: (1) Product, (2) Process, and (3) Resources [Cao19]. There are multiple domain specific ontologies that provide definitions for these concepts in various combinations, like PSL and MSDL. We will turn our attention to the MASON ontology [Lema06], which aims at providing a common semantic model in the manufacturing domain by conceptualizing these three core concepts, defined as: Entities, Operations and Resources. Typical applications of MASON are in multi-agent systems for manufacturing, which is also the scope of MAS4AI.

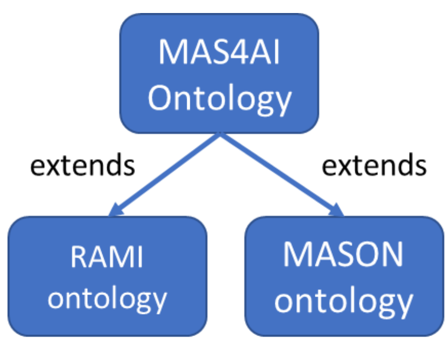

Figure 2 The MAS4AI ontology as an extension of MASON and RAMI

In addition, a semantically enriched data model of RAMI has been defined, based on the Platform Industry 4.0 specification, “Details of the Asset Administration Shell - Part 1” [AAS18]. This RAMI Ontology [RAMI19] represents the Reference Architecture Model for Industry 4.0 (RAMI), including the concept of the Administration Shell I4.0 Component. This ontology provides basic notion of the AAS, an asset, etc.

The MAS4AI ontology aims to build on and extend the concepts of both MASON and RAMI (Figure 2), by including notions relevant to the scope of the project.

3.2 Application of the ontology in use cases

The MAS4AI ontology is stored in an RDF store, allowing agents to query for information on the manufacturing process or for users to update reference information on the agent(s) currently active in the use case. The ontology and the RDF knowledge base aim to provide three specific functions in the MAS4AI framework:

- Provide an overview of all the agents required, specifically those to be coordinated by the holonic agent.
- Allow updates (reference IDs) for the agents currently active (within the holon).
- Provide references to all agents suitable to perform a certain activity.

Interaction type 1: Overview of agents

The first request in the lifespan of the holon, is the query that requests all agent types required for spawning within the current holon agent (Figure 3). The ontology then provides the list of agent types.

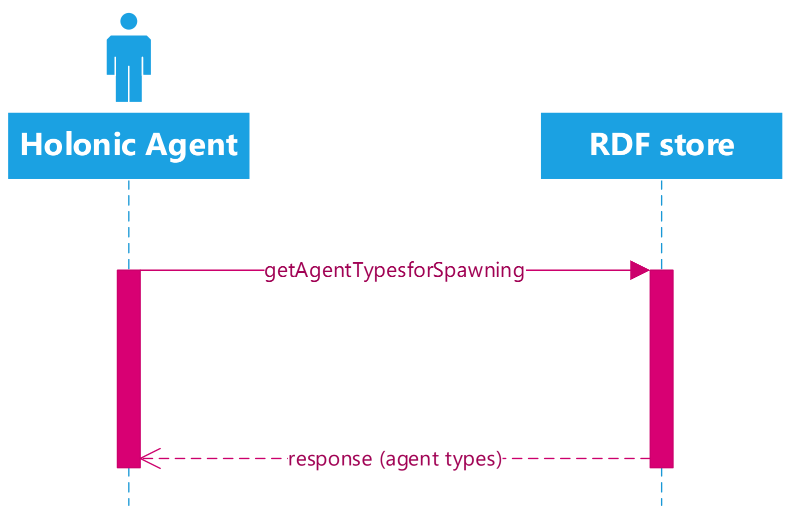

Figure 3 Query: all agent types for spawning within a holon

Interaction type 2: updates (reference IDs) for agents

The second type of request is an update query in order to register the reference ID used by the active agent(s). These updates can be made by the holon, i.e., the registration of its own reference as is visualized in Figure 4:

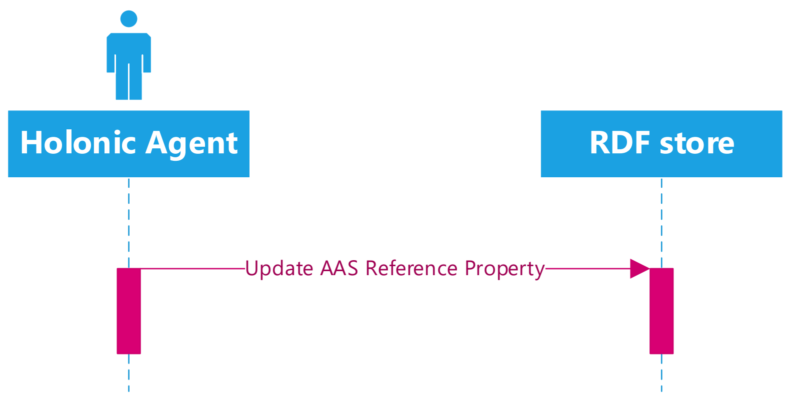

Figure 4 Ontology interaction: update holon agent reference

The registration can also be made by the BaSyx server in order to update the list of active agents (Figure 5).

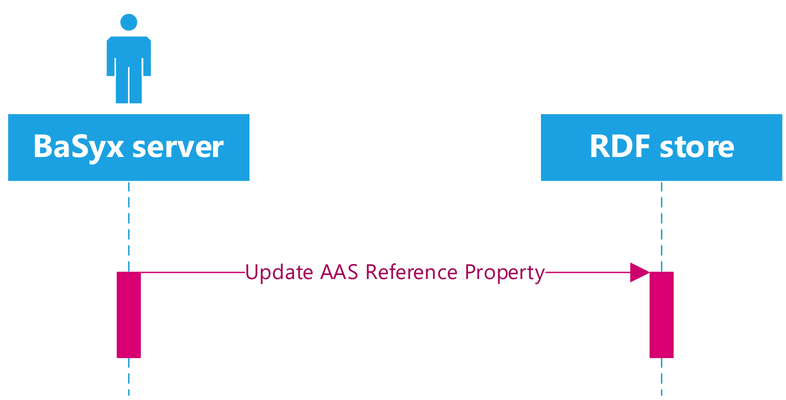

Figure 5 Ontology interaction: update active agent reference

Interaction type 3: query for agents to perform a certain activity

The third type of request is a query for agent types that have the capability to perform a certain activity (Figure 6). The ontology relates agents and their skills to activities, so the holon agent can identify the right agents registered (by reference). With these reference IDs, the holon in turn can access the (AASs of) the actual agents in BaSyx.

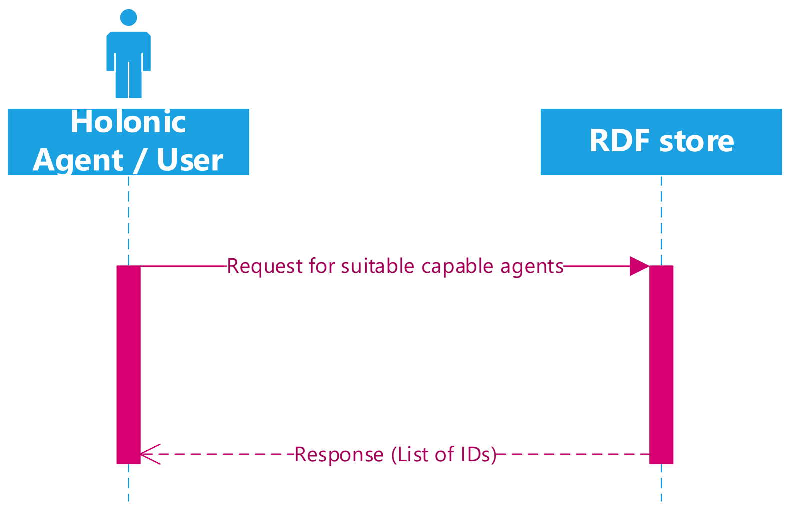

Figure 6 Ontology interaction: query for capable agents

The RDF store provides the functionality for these three types of interaction as a set of stored SPARQL queries, which are available as REST APIs.

3.3 Main concepts in the MAS4AI ontology

The interactions of section 3.2 serve as requirements to the MAS4AI ontology: it should be able to handle these queries and provide results accordingly. Within the scope of MAS4AI and the application of the ontology in mind, we will focus on the definition of processes, agents, assets and the AAS. These four concepts are defined and related from:

- Process: An **Operation** constitutes all processes related to manufacturing (MASON)
- AAS: the Admin Shell (**AAS)** is a standardized digital representation of the asset (RAMI) [Gloss22].
- Asset: Entity which is owned by or under the custodial duties of an organization, having either a perceived or actual value to the organization (RAMI) [Gloss22].
- Agent: An **Agent** is a computational process that implements the autonomous, communicating functionality of an application [FIPA02].

The MAS4AI ontology integrates the concepts from both the MASON and RAMI ontologies according to the diagram in Figure 7.

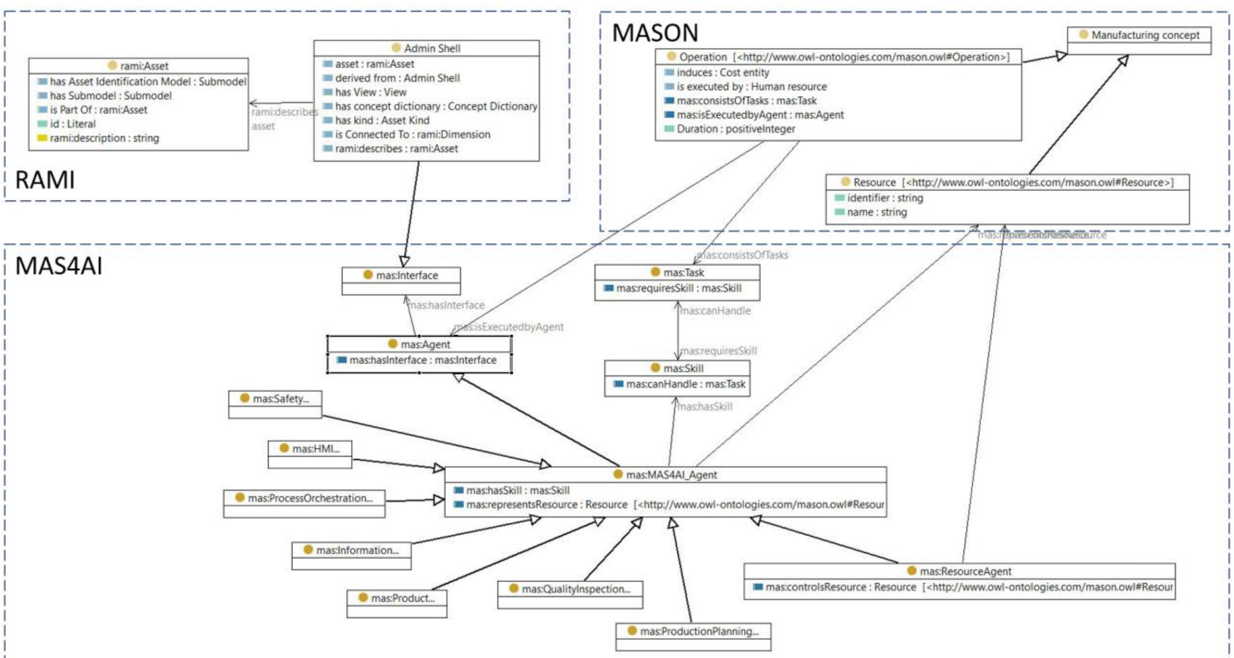

Figure 7 Relation between RAMI, MASON and MAS4AI ontologies

The MASON ontology formalizes the Process notion by introducing a class named Operation. The ontology uses this class and its subclasses to specify a set of manufacturing-related processes, including manufacturing operations (e.g., machining operations, control, assembly), logistic operations (e.g., maintenance, handling), human operations (e.g., scheduling, programming).

The notion of AAS and asset are defined in the Industry 4.0 glossary and embedded in the RAMI ontology. The AAS identifies the Administration Shell and the assets represented by it, holds digital models of various aspects (submodels) and describes technical functionality exposed by the Administration Shell or respective assets.

FIPA provides a definition for the term agent but does not provide an ontological model that embeds the agent concept. The MAS4AI ontology introduces the concept based on the FIPA definition and extends it to include the eight different types of agents that have been defined for MAS4AI in WP1.

The eight types of *MAS4AI agents* have an *Interface*, that is implemented by an *Admin Shell* (AAS). The agents have a certain *Skill*, to perform a *Task* in a manufacturing process. On the other hand, an *Operation*, essentially a process in manufacturing, consists of tasks. At the same time, MAS4AI agents represent and in case of the *Resource Agent*, control *Resources*.

This basic model allows us to ask about AASs for agents that have the skill to perform a requested task, allows us to request certain types of agents and insert reference IDS for the AASs of agents.

4 AAS

In this chapter the current status of the AAS semantic models of the MAS4AI framework are explained. We go into how they are designed and provide a functional level description of the used shells. As the modelling process is ongoing, not all AAS models are of equal maturity level, however all agents defined in WP1 have a provided rudimentary model.

4.1 Design process

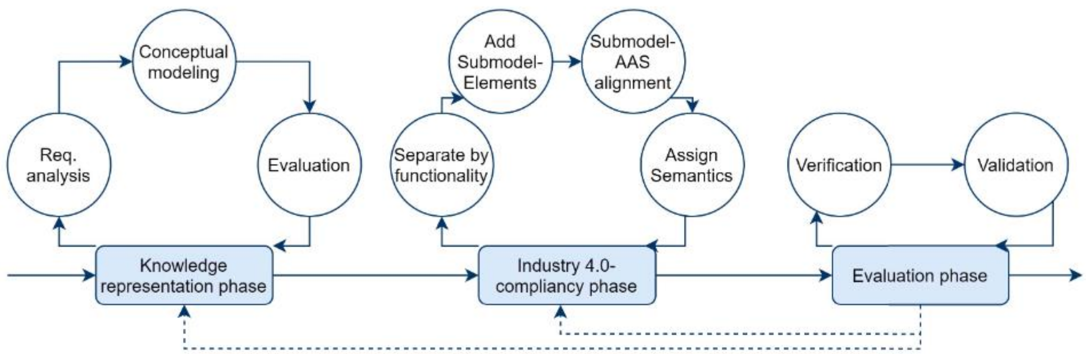

Figure 8: Overview of the methodology

The AAS models developed in MAS4AI aim to be both generically applicable to all current and future MAS4AI use cases. Furthermore, they should be rich enough to be used in their standard form, or with very minor additions. Recently, an approach outlining how to create AAS models was published as a result from the DIMOFAC project [Bout21]. This methodology (shown in Figure 8) is geared specifically towards use in industry and can therefore be used to achieve the abovementioned goals.

The methodology focusses on iteratively building a model while keeping in mind three distinct general modelling phases. First is a conceptual (knowledge representation) phase. It focusses on what is needed from the model and which concepts it should contain. Within this phase there has been close collaboration with the use case owners to determine how they intend to use the agents and what the use case generic concepts are versus the use case specific components.

After the conceptual level is agreed upon the models enter the formalization phase (alternatively called Industry 4.0 compliance phase). In this phase the ideas and high-level models from phase one are broken into separate AAS submodels, which are filled with the specific elements and then formalized. Finally, the Semantic identifiers are aligned with pre-existing models and concepts. This process, like the one in phase one can be performed in an agile way where insights gained during the process feed back into the next iteration of the development cycle. Within this phase a lot more collaboration with the RTO's and agent developers in the project took place. As they can provide more detailed input on which properties are needed for a specific agent.

The final phase in the used methodology is the evaluation phase. In this phase the models are used for the development of agents, and feedback is gathered. Depending on this feedback, the design process may go back to an earlier phase to improve the model. Evaluating the MAS4AI models is scheduled for the upcoming period, and no model has yet completely gone through the design process. Within the third phase we are expecting collaboration with all stakeholders, including the integrators of the agents, to determine what works or may not work within the created AAS descriptions.

Some models are currently in the formalization cycle where semantics are aligned and the details are fleshed out (i.e., orchestration, planning, resource and product agent models), other models are still in the conceptual stage where requirements are gathered, and a conceptual understanding of the agent is developed. More about those agents will be described in the following section.

4.2 Models

In MAS4AI deliverable D1.3 eight distinct types of agents are, as shown in table 1 below. Each one focuses on a particular part of the manufacturing process, monitoring and/or managing different assets. Every agent model essentially describes a generic set of type of agents, which have the same or similar capabilities and thus can be expressed using a generic information model. The next sub-sections explain every agent type in detail, outlining the inputs/outputs and working aim.

Table 1: Generic agent types

Type	**Description**
Production planning	Agents that dynamically optimize scheduling and reschedule planning.
Process orchestration	Agents that dynamically coordinate entities involved in the production process.
Resource	Agents that monitor and optimize resources availability and requirements.
Product	Agents that track product requirements during their lifecycle.
Safety	Agents that monitor and assess risks dynamically.
Information	Agents that collect and analyze information from different sources and optimize information exchange.
HMI/HMC	Agents aiming to improve human-machine interaction by optimizing the information workflow between agents and humans and providing explainable solutions and instructions to operators.
Quality inspection	Agents aimed at monitoring and/or optimizing the quality of produced parts.

4.2.1 Product Agent

The product agent serves to represent a product throughout the regardless of the product's life cycle. It exposes the product attributes as would be expected from an AAS but may add logic or functionality. The structure of the agent can be seen in Figure 9.

Input

The inputs of the product agent contain information describing the product of interest. Firstly, there is detailed technical information provided containing the specific product design (for example in the form of 3D models) and corresponding relevant details. The technical data of the product is also available within the model. Aside from that, there is also information about how the product in question should be manufactured.

Secondly, the product agent also contains information regarding the lifecycle of the given product. Depending on the concrete situation this can contain the current cycle stage of the production process and the stages left to be executed.

Standard to all MAS4AI agents, the product agent also has a general set of agent models for identification, some general nameplate properties, and its MAS4AI framework specific attributes such as which RDF instance it relates to. Furthermore, the agent also has two specific submodels dedicated to tracking the current agent's status and its life cycle.

Output

The product agent's exact output can differ per specific use case. Standardly, the agent is able to provide information regarding its status or expose its input descriptions to other agents. An active output of this agent can be the current product step and product status which may be after that used to initiate the next production step or have the product ask for its own next production step.

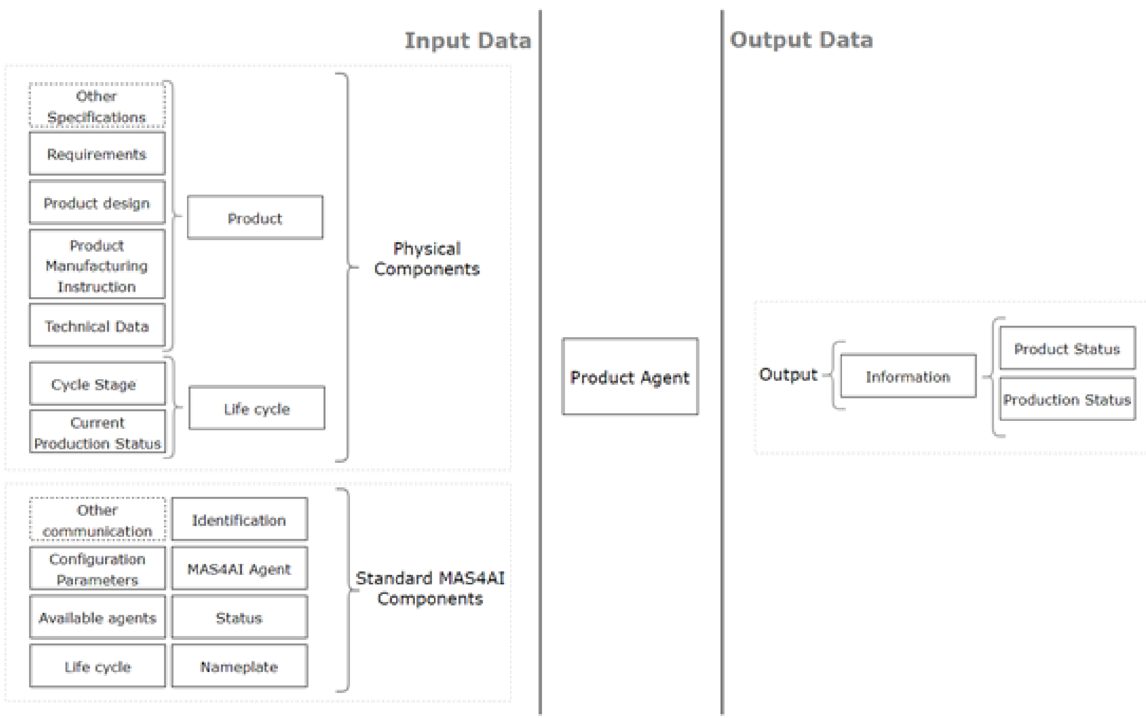

Figure 9 Information Model of the Product Agent

4.2.2 Resource Agent

The resource agent fulfills the role of representing a (manufacturing) resource in the MAS4AI framework. It exposes properties of the resource as is normally the case with an AAS but may enrich these with e.g., logic or functions to invoke the resource. The structure of the resource agent can be seen in Figure 10.

Input

The resource agent's inputs contain descriptions pertaining to its general usage. The manufacturing equipment model can describe the resource itself and the operations it may perform. Depending on the type of resource, there may also be a model containing information about the human operator involved. The resource gets some information from the HMI agent, such as the operator skill level. However, as the data can describe an individual person, it is privacy sensitive. As such, it highly depends on the situation which properties can be used within the model.

The resource's energy consumption is provided in its own submodel. This provides both the designed consumption and, in more dynamic deployments, the actual consumption.

Further, certain physical inputs of the manufacturing resource are provided:

- A component of the final product, e.g., when assembling multiple semi-finished products, these can reference another product.
- Raw resource which does not have a unique identifier for its discrete product e.g., in the case where a machine regularly consumes refilled oil or cooling liquids.

A resource is further associated with a department which may have a specific storage facility. If the resource in question is specifically a transport resource, there are some specific models available as well, representing the different types of transport vehicles that can be used.

Depending on the situation, the resource agent can also get a specific task to be executed. This can come from the process orchestration agent or from another external party.

Standard to all MAS4AI agents, the resource agent also has a general set of agent models (for identification, general nameplate properties, MAS4AI framework specific attributes such as which RDF instance it relates to). Furthermore, the agent also has two specific submodels dedicated to tracking the current agent's status and its life cycle.

Output

A resource agent does currently not provide a specific output as it does not have to trigger any other agents. However, it may send a confirmation of its action back to the invoker or can expose some additional status details besides the data already used to describe the resource in the 'input' column. Specifically, data related to how the product was manufactured and the current status is made available through submodels.

In the next phase of the project new requirements or use cases may be identified which would lead to the resource agent having additional outputs for new logic exposed by the agent.

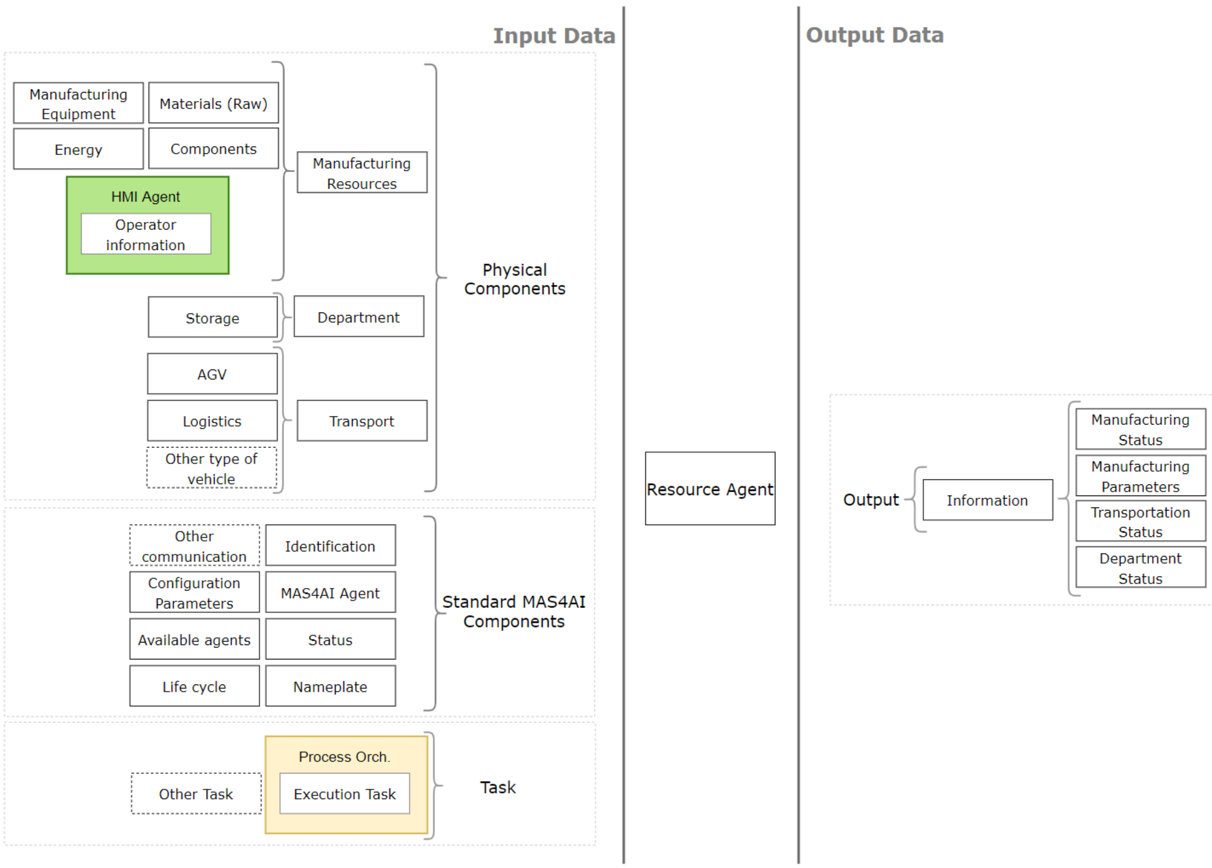

Figure 10 Information Model of the Resource Agent

4.2.3 Planning Agent

The planning agent is responsible for creating an adequate production plan for a given set of tasks for manufacturing of set of products. It takes the incoming orders and breaks these up into distributable and executable plans for the manufacturing environment. The structure of the planning agent can be seen in Figure 11. This generic model is expanded upon in WP4, where specific planning agent instances are developed and agent modelling requirements are fulfilled through additional models.

Input

The main input for the planning agent is a set of orders that need to be executed. This, depending on the exact use case, can contain different types of information relevant for the concrete situation (e.g. precedence graphs defining the different processes an order may go through to achieve the desired results). Aside from the task, the agent can also refer to the product agent of the product of interest to receive more information about the product description and manufacturing requirements.

The given order needs to be fulfilled within a specific set of constraints, specifically the available manufacturing resources and transport capacity which are defined by the resource agents themselves to which the production planning agents model references. Similarly, the production planning agents need to know where these resources are available i.e., which department operates them and requires knowledge regarding the available storage of the organization.

Standard to all MAS4AI agents, the planning agent also has a general set of agent models for identification, some general nameplate properties, and its MAS4AI framework specific attributes such as which RDF instance it relates to. Furthermore, the agent also has two specific submodels dedicated to tracking the current agent's status and its life cycle.

Output

The planning agent outputs a production plan, applying to a specified time period and number of products, which can be distributed to the relevant departments and be executed by them. Depending on the manufacturing environment, this may contain a machine plan, operator schedule for HR, transport plan to prepare logistics. The format of this plan is specified by the developers of the planning agent in work package 4.

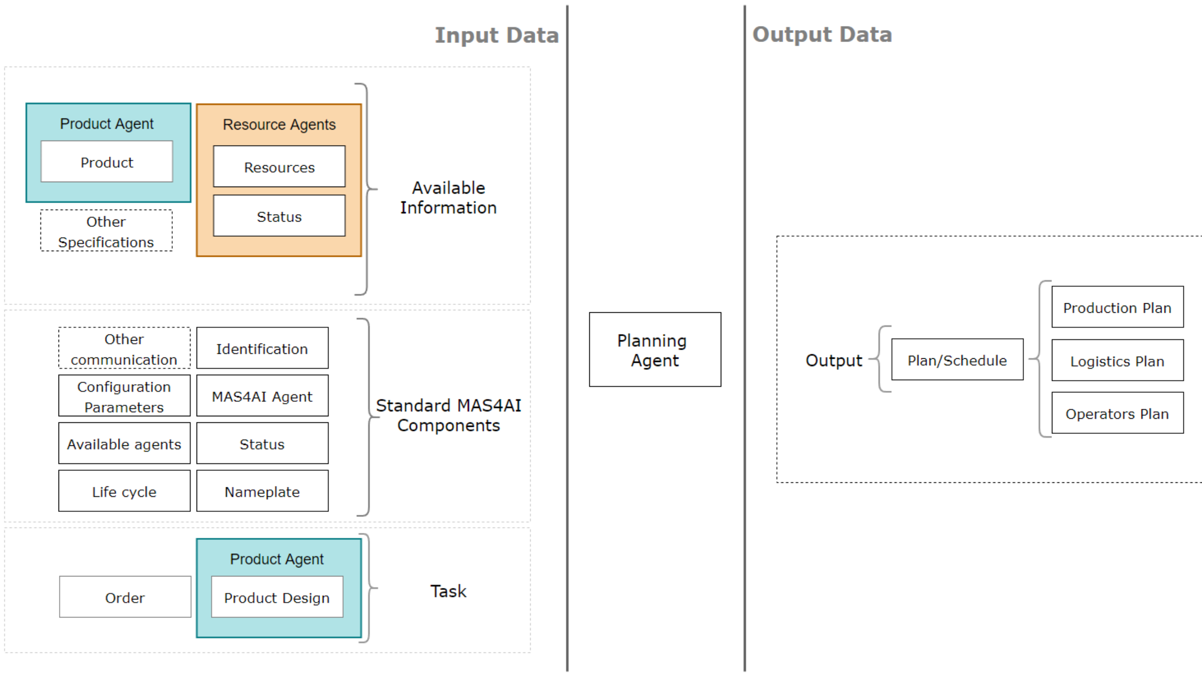

Figure 11 Information Model of the Planning Agent

4.2.4 Process Orchestration Agent

The process orchestration agent takes incoming work, in the form of production planning, and breaks it down into individual tasks which are sent to the agents responsible for those tasks. Through this process the agent is responsible for orchestrating work between several agents. Every MAS4AI holon can have its own orchestration agent which manages the agents within that holon. The structure of the process orchestration agent can be seen in Figure 12.

Input

The orchestration agent takes as input sets of work, for example in the shape of a production plan from the planning agent. It then determines how the subtasks should be distributed among the set of available agents (resource, quality, safety).

To do that it first looks at the available resources and splits the given tasks among them. Depending on the complexity of the production plan, the orchestration agent may also consider which semi-manufactured products are needed to perform the planned steps. This information can be found through the corresponding product agent.

Standard to all MAS4AI agents, the process orchestration agent also has a general set of agent models for identification, some general nameplate properties, and its MAS4AI framework specific attributes such as which RDF instance it relates to. Furthermore, the agent also has two specific submodels dedicated to tracking the current agent's status and its life cycle.

Output

The output of the orchestration agent is a task, or set of tasks, which can be executed by set of other agents. This makes the process orchestration agent effectively responsible for making sure that the right agents perform the right tasks on the right products. Depending on the specific situation the process orchestration agent can directly provide the tasks to required agents.

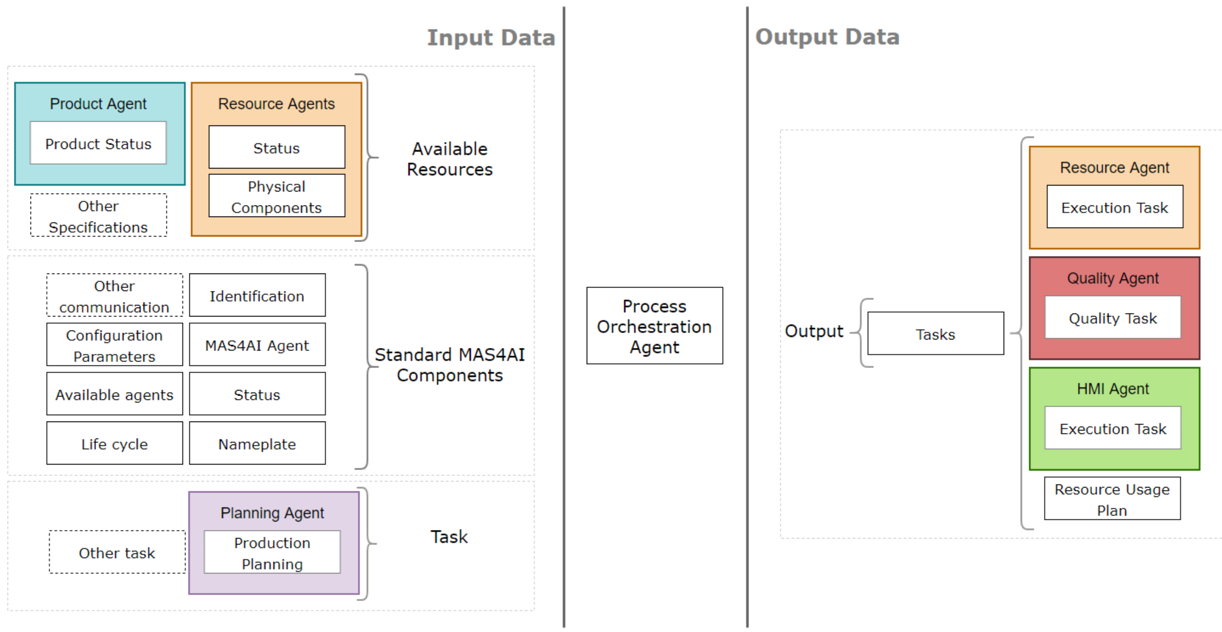

Figure 12 Information model of the Process Orchestration Agent

4.2.5 Safety Agent

The safety agent's focus is on keeping track of the safety and risks within a factory. It monitors the current level of risk at a given point of production and suggests (triggers) and adjustment to the current regulations should that be required. Depending on the specific case the scale of the safety monitoring can vary – from looking only a single machine to looking at a whole production line. The structure of the safety agent can be seen in Figure 13.

Input

The first input for a safety agent would be a set of certain safety parameters and elements which can be used as configuration of the aimed task. Those specifications can also contain information about the current risk levels and corresponding concrete thresholds.

As the agent needs to assess dynamically the risk level in the system, it actively analyzes the information from the resource and product agents. It may gather data from the product agent about the product being made that could influence the safety level, the type of materials, risk level, moving parts etc. From the resource agent the safety agent may extract the current safety requirements for the specific resource, the used safety sensors and any others relevant risk factors. This is all the information that could be required to perform a proper risk analysis of a given situation. There may also be interaction with the planning agent, as performing tasks in a certain order may influence the risks resulting from these tasks. However, so far this application is only theoretical and no specific use case or requirement has been identified. As such, this interaction is not modelled. But it may be added in the second phase of the project.

Standard to all MAS4AI agents, the safety agent also has a general set of agent models for identification, some general nameplate properties, and its MAS4AI framework specific attributes such as which RDF instance it relates to.

Output

The outputs of the safety agent are different analysis values. Depending on the specific situation the exact measurements can vary. The agent provides if required information about the current risk level in the factor/production line/machine and possible actions/alerts should this apply to the given situation.

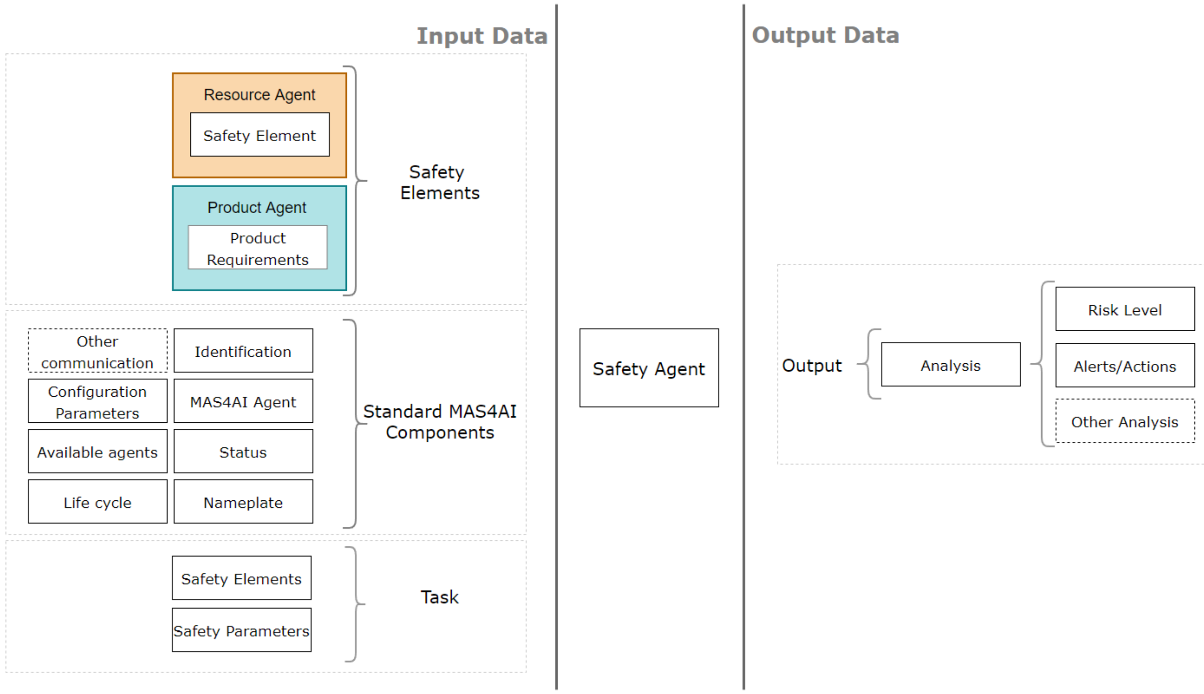

Figure 13 Information Model of a Safety Agent

4.2.6 Quality Agent

The quality agent focuses on monitoring the quality of a given product inline of the production process. It analyses the current state and if required looks for a way to optimize the quality of the product. Furthermore, the quality agent can also monitor the exact resource processes, so that it may be able to perform root cause analysis for any product quality issues. The structure of the quality agent can be seen in Figure 14.

Input

As the concept of quality is connected to a specific product, the main input for this agent comes from a product and resource agent. Depending on the products which are assigned, the quality agent extracts information regarding the product specifications, and the actual one's product realization. Aside from that, the agent can also inquire about the current resource that is being used from the corresponding resource agent, so that it is able to track the product's manufacturing process.

Standard to all MAS4AI agents, the quality agent also has a general set of agent models for identification, some general nameplate properties, and its MAS4AI framework specific attributes such as which RDF instance it relates to. Furthermore, the agent also has two specific submodels dedicated to tracking the current agent's status and its life cycle.

Output

The quality agent's main output would be information regarding the current quality of the product(s) of interest. This quality can, if required, be linked to the processes and resources involved in the manufacturing process of interest. In the situation when the quality is below a certain threshold, the agent could also output a sort of trigger signal towards the process orchestration agent to signal the possible discrepancy or send that directly towards a resource agent depending on the type of quality issue.

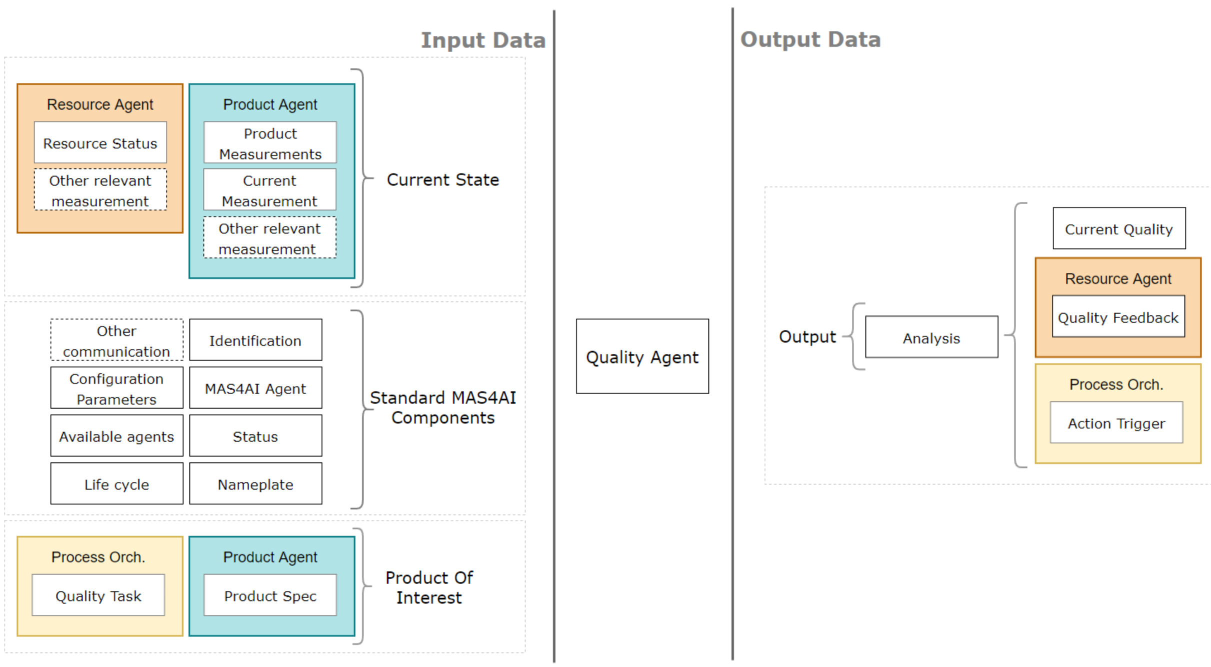

Figure 14 Information Model of a Quality Agent

4.2.7 HMI Agent

The HMI agent aims to improve the human-machine interaction by optimizing the information flow between agents and humans. It can have different type of specified purposes. For example, within the VDL use cases the HMI agent specializes in supplying clear work instructions to operators of a given machine. The structure of the HMI agent can be seen in Figure 15.

Input

The HMI agent receives three distinct types of input. First the agent gets information about the exact task (job) that needs to be executed. This may be provided by the process orchestration agent. Based on this task the HMI agent can requests product specifications from the product agent which represents the product to manufacture. (This could include information such as current job, model data, CAD file etc.)

Aside from that, the HMI agent may further communicate with the product agent to receive the exact manufacturing instructions which must be followed. Furthermore, it can receive information from the resource agent, regarding the machine and its operator

Standard to all MAS4AI agents, the HMI agent also has a general set of agent models for identification, some general nameplate properties, and its MAS4AI framework specific attributes such as which RDF instance it relates to. Furthermore, the agent also has two specific submodels dedicated to tracking the current agent's status and its life cycle.

Output

The HMI agent does not have a specific output, but it may supply information about its current status, the current product state, and the operator assessment (level) depending on what is relevant.

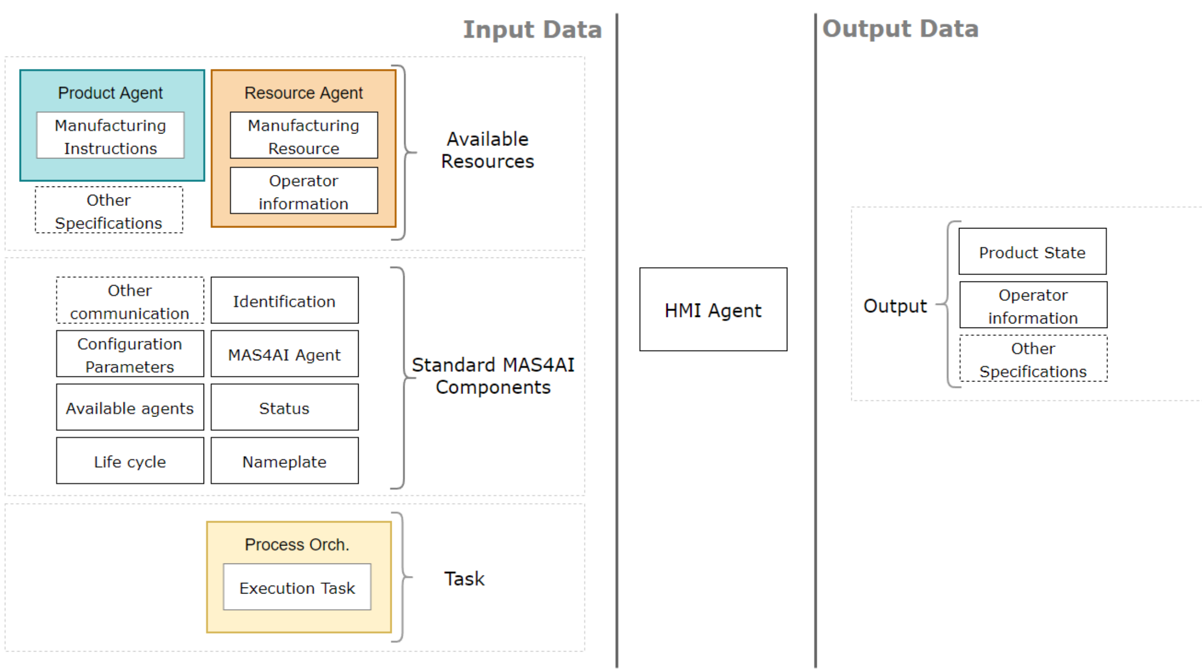

Figure 15 Information Model of an HMI Agent

4.2.8 Information Agent

The information agent gathers, monitors and analyses the information flow within the multi agent framework. This can be used to optimise the flow of information by the agent itself, or to pre-process and if required analyse the information for the purpose of presenting it to parties outside of the MAS (such as legacy systems, human operators, data analysts). The structure of the information agent can be seen in Figure 16.

Input

The information agent takes input from all other agent types, and as such can reference all of those to request information as well. Specifically, the generic information agent model contains example references to the product, resource, planning, safety and quality agents. Using these references, the information agent may know where to find the other agents and request information from them. To find which agents are active on the MAS framework the information agent may query the RDF knowledge base, through this it has another way of finding the locations of other agents, besides them being explicitly stated in the agent description.

Standard to all MAS4AI agents, the information agent also has a general set of agent models for identification, some general nameplate properties, and its MAS4AI framework specific attributes such as which RDF instance it relates to. Furthermore, the agent also has two specific submodels dedicated to tracking the current agent's status and its life cycle.

Output

The information agent may output commands towards other agents to optimize their working or provide some of the gathered information in a certain pre-processed form. Besides that, the information agent also exposes a data storage which may be used for analysis purposes or by other agents which may query the database. This can either be a local storage, or an external database which is populated by the information agent (for e.g., an RDF store).

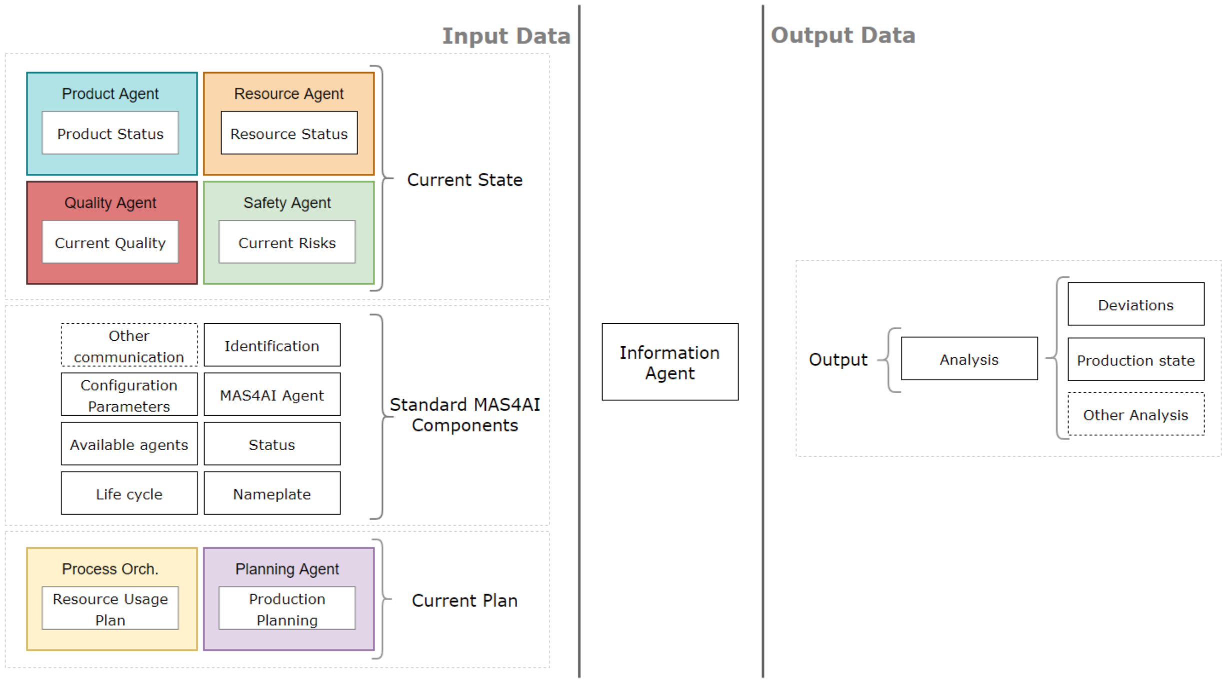

Figure 16 Information Model of an Information Agent

5 Conclusions

The descriptions of the agents in this document are provided in a general form, that marks the beginning of the development process. In the next stages of the project for every use case specified agents will be developed, using the provided structure here as a base for the agents' semantic model. As the development in this project is agile, it is expected that the general descriptions from here will go through multiple iterations before a finalized version is achieved. Some of the changes are likely to be about the addition of properties, while some will be regarding the alignment of the current models with already existing standards.

The ontology described in this document is fit for its purpose for essentially all the use cases in the MAS4AI project. We do not foresee any major extensions or additions to its structure, since all the essential concepts in MAS4AI are covered. Specific requirements by the use cases may however result in additions in e.g. concept hierarchies of activities.

5.1 Role of the ontology

The role of the ontology is to serve as a (semantic) extension to the agent framework and the AAS hosting platform, providing additional capabilities to specify and query knowledge about the manufacturing process beyond the scope of individual agents. While AAS hosting components like BaSyx could theoretically be customised to provide similar functionality, it is not a natural fit. The ontology in the RDF store provides a well-defined component that has the logical foundation and reasoning capabilities to support formulating simple queries for potentially complex questions about the manufacturing process. The current use of the ontology is mainly to provide schematic knowledge, i.e., knowledge about types of concepts. This is a clear alignment with the fact that operational (instance) information is available in the AASs of the individual assets and agents.

5.2 AAS Design Challenge

The main challenge when designing an AAS Information model lies in the fact that there are very few existing models which can be used. Currently there are no general repositories or specific established patterns that can be followed when creating a new shell. This is even a bigger challenge when trying to model non-physical assets, which the agents are, as the predominant focus of the AAS modelling up until now is on modelling physical elements from a factory. Considering this challenge, the iterative design approach by Cornelis et al. ([Bout21]). served the modelling well so far. The approach supported quick interaction with use case owners without getting bogged down in finding and aligning with existing models before starting the modelling work. This has prevented the AAS modelling from becoming a bottleneck in the project and provides structure to continue work in the second half of the MAS4AI project.

Furthermore, the methodology supported quickly developing small submodels which can be iterated and expended upon as needed. This ended up creating highly reusable submodels which can be shared between AAS models.

Another challenge within the design of the models was providing a clear and understandable way for the different involved parties to comprehend the created structure. The current asset administration shell tooling is not intuitive to everyone, especially those, who do not have extensive semantic modelling experience. Therefore, the choice was made to create high level visuals, which helped significantly with the communication with different stakeholders and allowed the gathering of feedback early in the process. To further work on this issue, the MAS4AI project is collaborating with the HORIZON DIMOFAC project [DIMO22] to develop a web application which makes AAS models and model templates more accessible to a wider audience.

5.3 Agent Modifications

In work package 1 eight different agent types were defined. These were based on the common needs of different use cases where each of the types fulfils distinct functionalities which collaboratively can achieve the desired use case's business function.

However, when the details of the agents were further explored with the use cases it appeared that this ideal situation does not necessarily hold up in practice. In some cases conceptual agents may be combined into single technical implementations. For example, an agent which combines the planning and process orchestration capabilities within one system. Especially in the situation, when the planning is done on a smaller scale with less resources, this can provide a more concise algorithm. This combination does not compromise either of the capabilities, but instead creates an agent which is able to handle more specific tasks. This consideration on the scoping of components is broader than the MAS4AI project, as can be seen in discussions on planning and control systems for example.

Another possible direction is simplification of an agent. In certain cases, it may not be needed to have for example a complete resource agent, as the actual resource does not possess any specific *smart* capabilities or does not require any pre/post processing. In such a case the active AAS of the agent can be simplified to the form of a passive AAS containing only the relevant resource data and omitting the agent specific models. This would still enable the other agents to keep the same references as if there was an actual resource agent, since the AAS elements holding the resource information are still going to be present.

6 References

[Zini99] Zini, F. & Sterling, L. (1999). *Designing Ontologies for Agents.* Joint Conference on Declarative Programming, AGP'99, L'Aquila, Italy, September 6-9, 1999

[Cao19] Cao, Q., Zanni-Merk, C., & Reich, C. (2019). *Ontologies for manufacturing process modeling: A survey*. Smart Innovation, Systems and Technologies, 130, 61–70. https://doi.org/10.1007/978-3-030-04290-5_7

[Lema06] Lemaignan, S., Siadat, A., Dantan, J. Y., & Semenenko, A. (2006). *MASON: A proposal for an ontology of manufacturing domain*. Proceedings - DIS 2006: IEEE Workshop on Distributed Intelligent Systems - Collective Intelligence and Its Applications, 2006(July), 195–200. https://doi.org/10.1109/DIS.2006.48

[AAS18] *Details of the Asset Administration Shell - Part 1*, https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/2018-verwaltungsschale-im-detail.html

[RAMI19] S. Bader (2019). *An Ontology to represents the Reference Architecture Model for Industry 4.0 (RAMI)*. https://github.com/i40-Tools/RAMIOntology

[Gloss22] *Glossary - Plattform Industrie 4.0*, https://www.plattform-i40.de/IP/Navigation/EN/Industrie40/Glossary/glossary.html

[FIPA02] *FIPA Agent Management Specification*, http://www.fipa.org/specs/fipa00023/SC00023J.html

[Bout21] C. Bouter, M. Pourjafarian, L. Simar and R. Wilterdink (2021). *Towards a Comprehensive Methodology for Modelling Submodels in the Industry 4.0 Asset Administration Shell.* 2021 IEEE 23rd Conference on Business Informatics (CBI), 2021, pp. 10-19, doi: 10.1109/CBI52690.2021.10050.

[DIMO22] *Digital Intelligent MOdular FACtories*. HORIZON DIMOFAC Project, https://cordis.europa.eu/project/id/870092