Specification "Demonstrator I4.0-Language" v3.0

Technical Report · July 2019

CITATIONS
0

READS
72

2 authors:

Alexander Belyaev

17 PUBLICATIONS **11** CITATIONS

SEE PROFILE

Christian Diedrich

Otto-von-Guericke-Universität Magdeburg

182 PUBLICATIONS **1,050** CITATIONS

SEE PROFILE

FAKULTÄT FÜR
ELEKTROTECHNIK UND
INFORMATIONSTECHNIK

INSTITUT FÜR

AUTOMATISIERUNGSTECHNIK

Specification „Demonstrator I4.0-Language“ v3.0

Alexander Belyaev, Christian Diedrich

Technical Report

IFAT-LIA 07/2019

09. July 2019

Universität Magdeburg
Fakultät für Elektrotechnik und Informationstechnik
Institut für Automatisierungstechnik
Postfach 4120, D-39016 Magdeburg
Germany

Table of content

1 Scenario Description

The concept of intelligent production includes new functionalities such as negotiations between I4.0 components in order to fulfill a specific task with defined characteristics in a defined period of time, with a defined quality of service, and a defined cost range. To enable I4.0 components to cooperate with each other and to perform the tasks in a cooperative manner, they must be able to speak a common language. The recently published VDI 2193 guideline suggests a concept of a I4.0 language. The I4.0 language is considered to be a three-level rule-based system that defines a vocabulary, a message structure, and semantic interaction protocols.

In this document a demonstrator of the I4.0 Language (VDI/VDE 2193) is presented, which was developed at the chair of Integrated Automation of the University of Magdeburg together with GMA 7.21 / UAG AG1 (Semantic and Interaction of I4.0 Components) of Plattform I4.0.

1.1 I4.0-Component

I4.0-Component	Worldwide uniquely identifiable communication-capable participant consisting of asset and administration shell
Asset Administration Shell (AAS)	Virtual digital and active representation of an Asset of the I4.0 component in the I4.0 system.
Asset	An object that has a value for an organization

What are the implementation variants for an administration shell?

AAS can be provided in different forms (7). A distinction can be made between passive and active AAS. The term passive and active refers to the role that the AAS plays in the value chain.

Administrative shells that take on a passive role provide all their information content without initiating their own actions.
A management shell plays a passive role if it is exchanged as a file between partners.
A passive role, however, is also played by administration shells that are servers in a client-server relationship or slaves in a master-slave relation and can be accessed, for example, via an IP-based API.

Administrative shells that interact with each other via the I4.0 language play an active role. This corresponds to the interaction pattern peer-to-peer. The active AAS establish contact with each other and perform cooperative tasks without higher-level, centrally controlling, non-AAS-based applications.

Passive AAS in file format - as described in VWSiD [4] in XML or JSON format - provide a standardized way to make all information associated with the asset available to authorized user groups according to the details approved by the asset owner. This concept represents a new quality, as it enables standardized information exchange across all life phases of assets.

Passive AAS with IP/API-based access basically have the same information content as AAS in file format. The difference lies in the fact that the inner structure of AAS (e.g. submodels) is not visible (as with the

file-based AAS) and is only made available via an interface. The interface design depends on the chosen technology. A CRUD-oriented specification is favored for this.

In addition to the possibilities of the CRUD-oriented administration shell, **active administration shells can participate in horizontal protocol-based interactions**, as defined for example in the VDI/VDE 2193 guideline for the bidding process (see also [1]). The I4.0 language is designed for both types of the interaction patterns (vertical and horizontal). The goal is to design decentralized processes that are based on a certain autonomy or decision-making ability of the administrative shells.

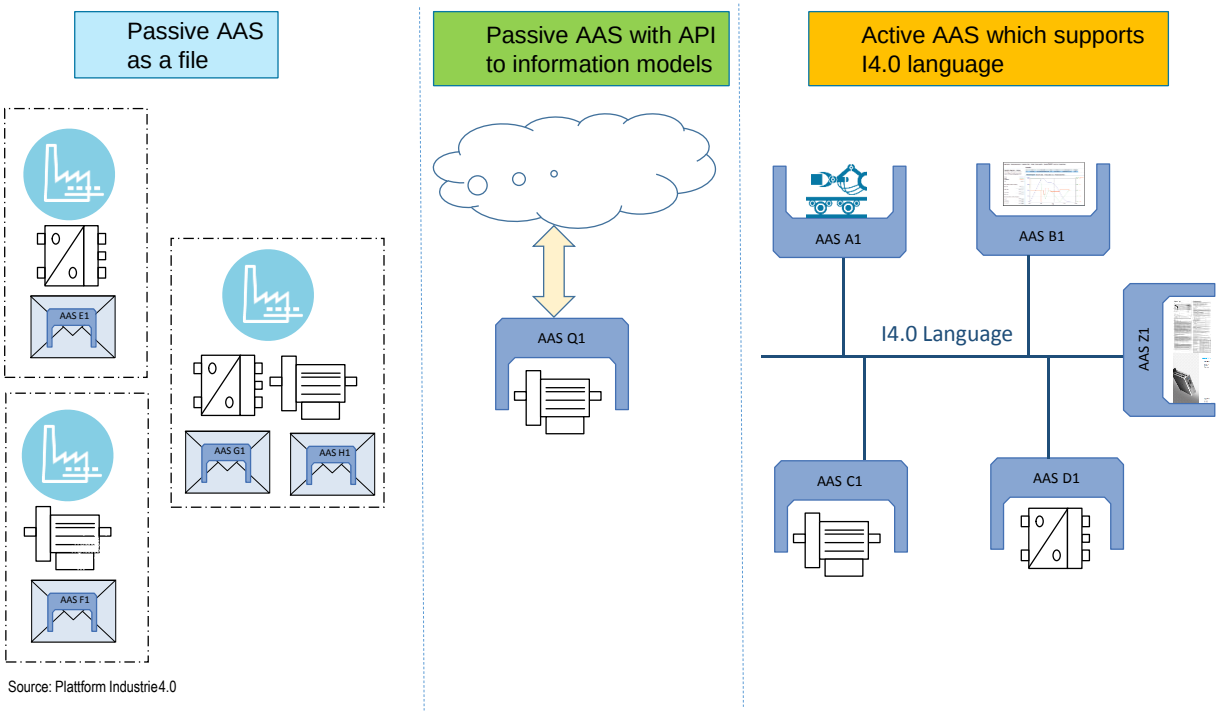

Figure 1: Various forms of the administrative shell (7)

The difference between the passive and active AAS can be illustrated by a placement in the RAMI4.0.

The passive administration shells contain a description of properties, parameters, variables and process capabilities in the form of so-called submodels. This abstraction of assets can be accessed, read and manipulated by other components. Passive aspect of such kind of AAS means these ability to respond to external requests and commands and inability to take the initiative and to make the decisions to achieve own aims.

Active, on the other hand, refers to the autonomous activation of interaction with external AASs, e.g. on the basis of an objective pursued (e.g. to act as economically as possible).

In the business layer, technical and economic decisions are made with the help of more or less complex algorithms. They thus represent business processes within an AAS. Not all AAS need to provide business services. The AAS that provide such kind of business services are called active AAS in this document.

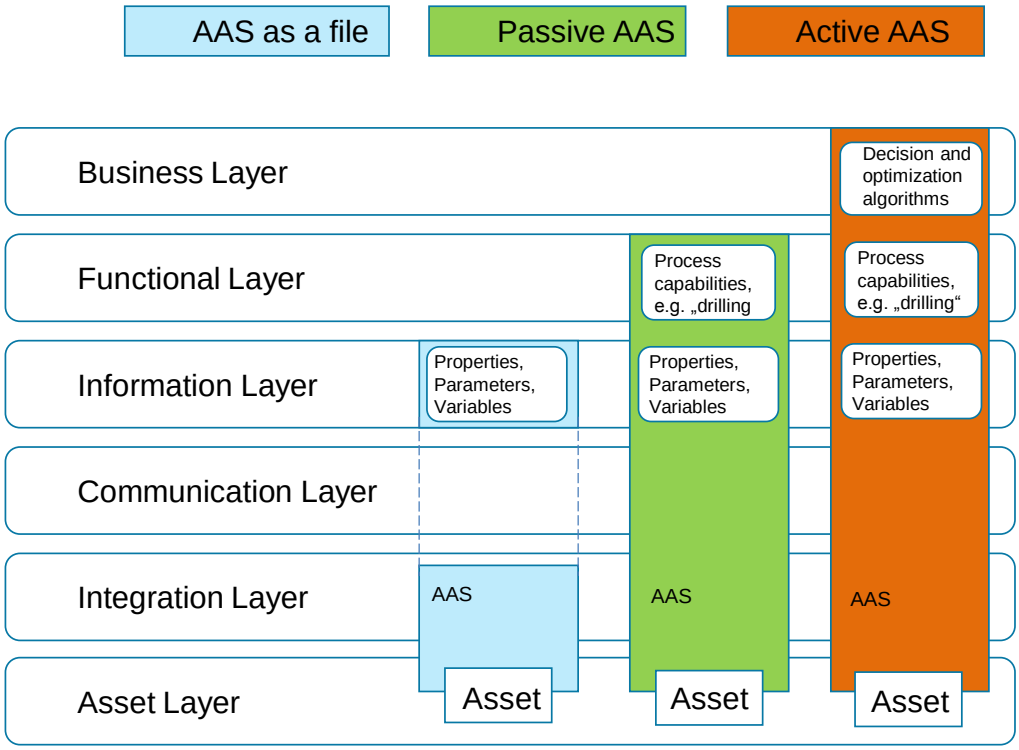

Figure 2: Assignment of the active AAS in the RAMI4.0-Model

1.2 Concept of I4.0 language

Information exchange between I4.0-Components is message-based. VDI/VDE 2193-1 defines the structure and types of messages in the I4.0 language. A sequence of single messages in a dialog of two or more I4.0components can be defined by different interaction protocols, , so called semantic interaction protocols.

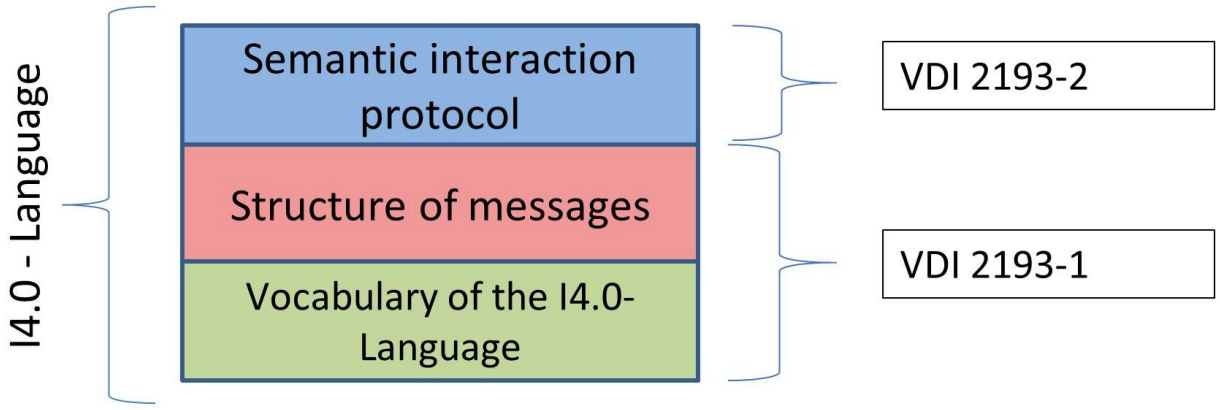

Figure 3: Concept of I4.0-Language [5]

In the part 2 of VDI/VDE 2193, two interaction protocols are considered that organize a cooperation of I4.0-Components to fulfil the bidding process. The aim is to reach legally binding value chains across company boundaries in a direct way between two or more I4.0 components horizontally, in which each participating I4.0-Component takes over a task agreed in bidding process.
If the I4.0-Components support the interaction protocol “bidding process", they can write out the tasks for other I4.0-Components or take over the tasks themselves by reacting to calls for proposals.
The special feature compared to the central coordination mechanisms is that there is no central control over the allocation of tasks and the relationships between the I4.0 components are not defined by the central element of the system. Depending on the task, each I4.0 -Component can assume the role of a service requester or service provider and spontaneously interact with any other I4.0 components. This is needed to fulfill the tasks taken over in agreed, legally binding cooperation with other I4.0 components that take over tasks defined in the bidding process.

1.3 Concept of an active AAS

The core of the passive administrative shell are standardized submodels. These describe the properties and functionalities of assets and make them available in a machine-readable format. The AAS act as an standardized representative of assets for usage of other AAS or non I4.0 applications. The submodels and their elements can thus be read and the functions/asset skills can be started or stopped.
A semantically clear, machine-readable description of the properties and capabilities of assets is an important step towards increasing the interoperability and integration of automation technology components.
However, there are certain industry 4.0 application scenarios that require a certain autonomy and intelligence of I4.0 components (e.g. order-driven production, adaptable factory). These include dynamic optimization of production load, avoidance of downtimes due to machine failures, order-driven production, as well as dynamic orchestration of production resources for cost-optimized production in one-lot size.
What is striking now is that the existing concepts of the administration shell do not clarify the question at which point an I4.0-Component makes decisions, even if only this feature allows the I4.0 components to interact autonomously.
Furthermore, it has not yet been discussed which entity is the location of the Interactions and the potentially necessary decisions and rules.

In this demonstrator architecture of an active autonomous intelligent administration shell is introduced.

Active means the possibility to react to changes in the environment and in the asset and to take the initiative to interact with other AAS with configurable strategies (e.g. "as fast as possible" or "as cheap as possible") in order to achieve one's own goals.
Intelligence means the ability to make decisions.
Autonomous means the ability to define one's own relationships and interaction needs with other components.

Active AAS can only become active on the basis of an appropriate knowledge base. Exactly this knowledge base is provided by the passive part of the AASs. Therefor the active AAS consists of two parts: a passive part and an active part.

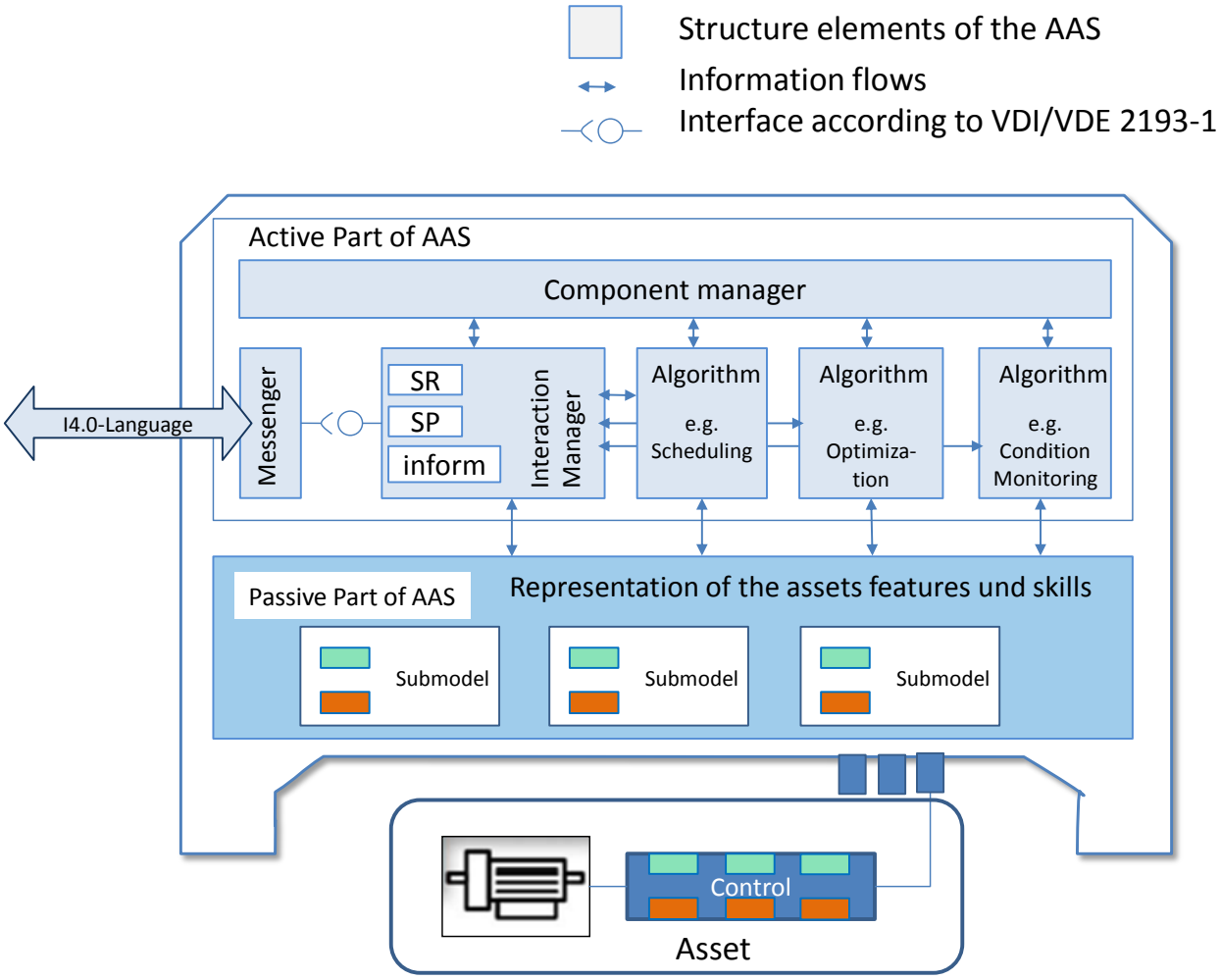

Figure 4: General architecture of the active AAS introduced in [8]

Proposed architectural elements of an active part of AAS:

- **Interaction Manager (IM),** host the interactions state machines.

  Interaction Manager is responsible for the implementation of different semantic interaction protocols [VDI/VDE2193-1/2] and the call of necessary decision and optimization algorithms. Each I4.0 component hosts a specific set of state machines which implements the different interaction protocols. These interaction patterns correspond to the purpose of I4.0-Component, i.e. these functional content, position in a value chain and life cycle phase [3].

VDI/VDE 2193-2 defines one of the important interaction protocols “Bidding process” (german: einfaches und erweitertes Ausschreibungsverfahren). The other interaction protocols will be defined in further parts of the VDI/VDE 2193.

- **Messenger** is an interface of AAS, takes over the transport of messages.

  The I4.0-Language describes the interactions independent from underlying communication systems related to OSI. The I4.0 language describes interactions between I4.0 components at application level according to the OSI Reference Model (Figure 5). The VDI/VDE 2193 specifies the structure, the sequence of messages and their possible JSON-serialization. The messenger shown at the Figure 4 implements the protocol stack and takes over the transport of messages (in this demonstrator - MQTT). The Interface between the messenger and the interaction manager are the JSON-objects representing the messages according to the VDI/VDE 2193-1.

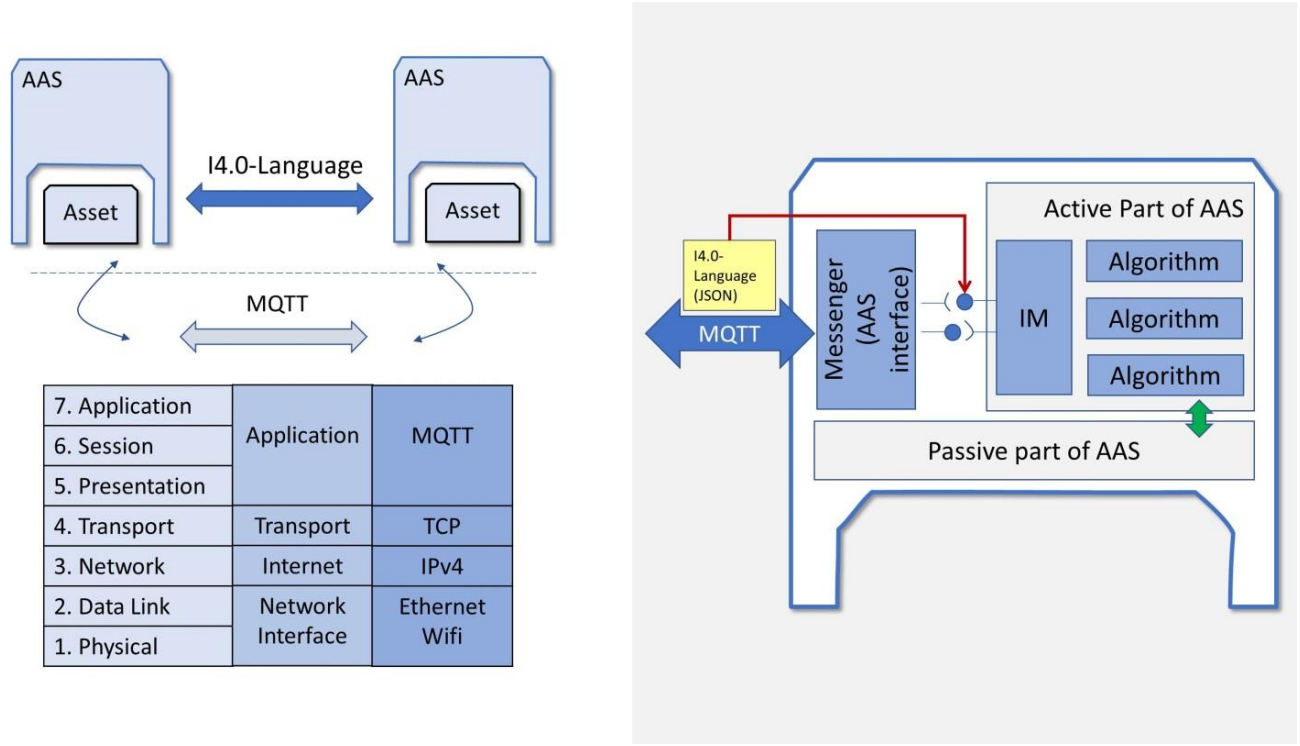

Figure 5: I4.0-Language describes the interaction independent of the data transport

- **Component manager**: orchestrates the internal interaction of elements of active part of AAS.

  In this demonstrator, the component manager is kept very simple, it decides only when the component has to assume which role (the role of service requester or service provider).

- **Algorithms**: necessary for certain decisions in certain negotiation steps.

  The algorithms depends on the configured objective of an AAS (e.g. feasibility check, availability check (e.g. by scheduler), price calculation, optimization algorithm to find the best offer etc. (Figure 4 and Figure 6)).

The decision-making algorithms of the negotiating parties that lead to the submission of an proposal or the acceptance of an proposal are not legally prescribed. These decision-making processes take place in the private black boxes, which have to be individually designed. For the implementation of independently acting contract negotiation modules in the I4.0 components, the negotiation rules must be defined individually.

The decision and optimization mechanisms will be the essential factors in determining the success of negotiations. These lie in the area of competition and thus outside the scope of standardization activities.

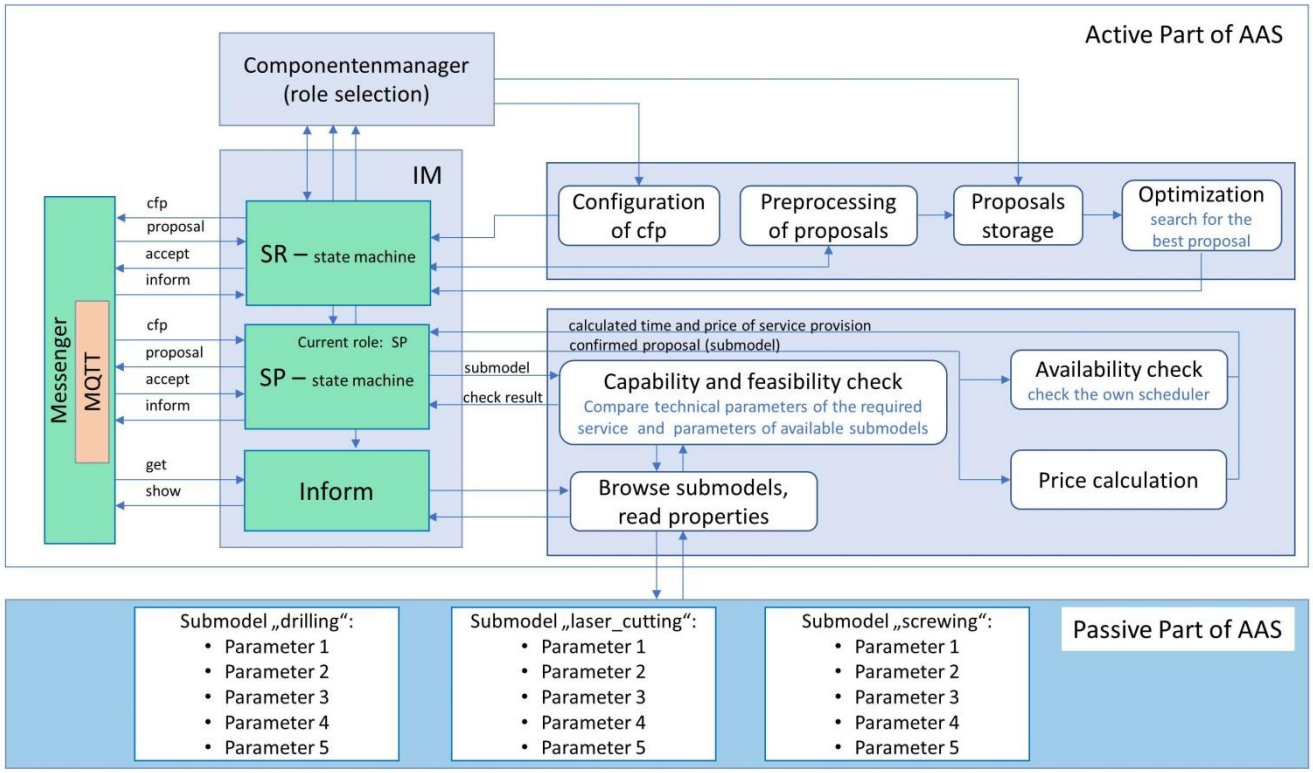

Figure 6: Posible architecture of an active AAS

Features of an active AAS:

- Intelligence as ability to make decisions according to own goals
- Autonomy as ability to define the relationships and interaction needs with other components itself
- Has a multiple decision/ optimization algorithms
- Supports both vertical and horizontal interaction mechanisms [1]
- Can participate in a bidding process according to VDI/VDE-2193-2

Assets of active I40-Components in this demonstrator can have different skills, e.g. “drilling”, “laser cutting”, “screwing”.

Through the combination of assets and active AAS, these machines become intelligent autonomous service units that provide their capabilities/skills to the other I4.0-Components in the form of services (German: Dienstleistungen).

1.4 Demonstrator purpose and showcase

Main goals of the demonstrator are:

- Show the characteristic of the I4.0 language according to VDI/VDE 2193
- Show message/protocol based approach for the interaction between the AAS
- Show the implementation of horizontal interactions [1] with the I4.0 language according to VDI/VDE 2193
- Show independence of the interactions from underlying communication systems related to OSI
- Show the structure and implementation of an active AAS
- Show the difference between an active AAS and a passive AAS

In this version of demonstrator, the active administration shell is implemented in a simplified way. The goal is to introduce the concept of the active AAS, their interaction using the I4.0 languge (VDI/VDE 2193) and to give the user the feeling of what kind of decisions a fully functional active AAS should be able to make, or which optimization algorithms it should contain.
The main show case is a bidding process (German: Ausschreibungsverfahren) between multiple active I4.0-Components. The interaction protocol "bidding process“ presented in VDI/VDE 2193-2 offers the possibility of a highly flexible generation of cooperation relationships between I4.0-Components, especially if the tasks have to be allocated. Exactly such dynamic relations are to be realized in the sense of the flexible order production, in order to be able to integrate free internal or external manufacturing capacities as independently and as automated as possible into the own business processes.

1.5 Implemented scenario

The I4.0-Components implemented in this demonstrator are considered as independent service units that based on local knowledge and objectives act autonomously in order to achieve their own goals. Service units means that the components can provide some services (in the economic sense of the word. German: Dienstleistungen), e.g. “drilling”, “laser cutting”, “screwing”.
Such service units are responsible for its own schedule, can verify the ability to perform certain production services and are able to determine their availability to complete the process by the desired deadline and then make proposals. They are able calculate the price of their service and include it in proposals.
Therefor there are two roles, which can be taken over by active I4.0-Component participating in a bidding process:

- Service Requester (initiator of a bidding process, want to order some service, sends “call for proposal”)
    - “call for proposal” contains a description of a required service
- Service Provider (provide some services, responds to a “call for proposal”, can send a “proposal”)
    - “proposal” contains a description of a service that can be provided

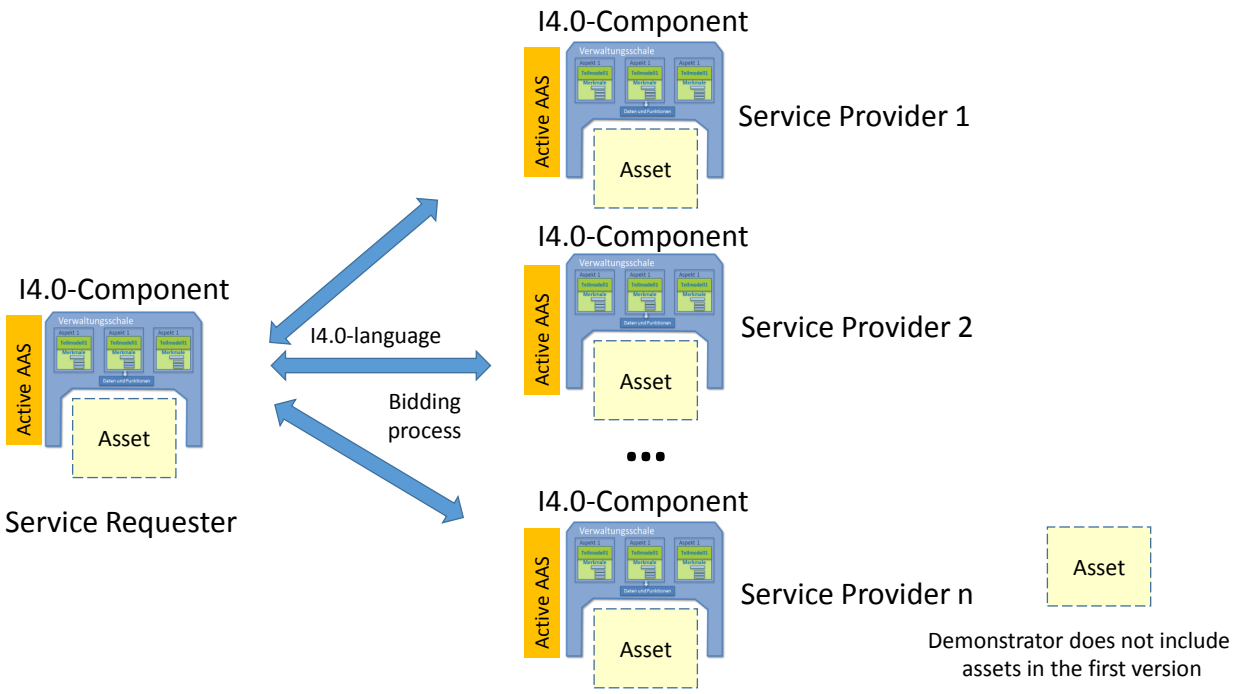

Figure 7: Basic demonstrator structure

The I4.0-Component consists of an asset and its asset administration shall (AAS). The AAS that support the bidding process are the active AAS. The concept of an active AAS is discussed in detail in further chapters.

The active I4.0-components that participate in the bidding process are divided into two groups. One I4.0-component plays the role of the service requester (SR) i.e. the initiator of a bidding process. The other I4.0 components become the role of subcontractors (service provider - SP). SR sends the “call for proposal” and waits for the answers from service providers. SP receive the “call for proposal”, evaluate it (prove if they are able to execute the required service, if they are available in required time, calculate the price of service providing) and sends their answers to the service requester back. The possible answers of the SP are:

- “Proposal”
- “not understood” (if requested submodel (service) is not known)
- “refuse” (if other parameters of requested service do not fit/ if the feasibility check failed)
- it is also possible not to react.

The SR selects the best proposal and informs the I4.0 component which has sent it. In this version of the demonstrator, the SR selects the best proposal on the basis of the price and the time of service provision. According to VDI/VDE 2193 the bidding process has to follow the following interaction protocol (Figure 11).

2 Interactions between I4.0-Components

2.1.1 Structure of messages

According to VDI/VDE2193-1 the I4.0-Components interact message-based (Figure 8).

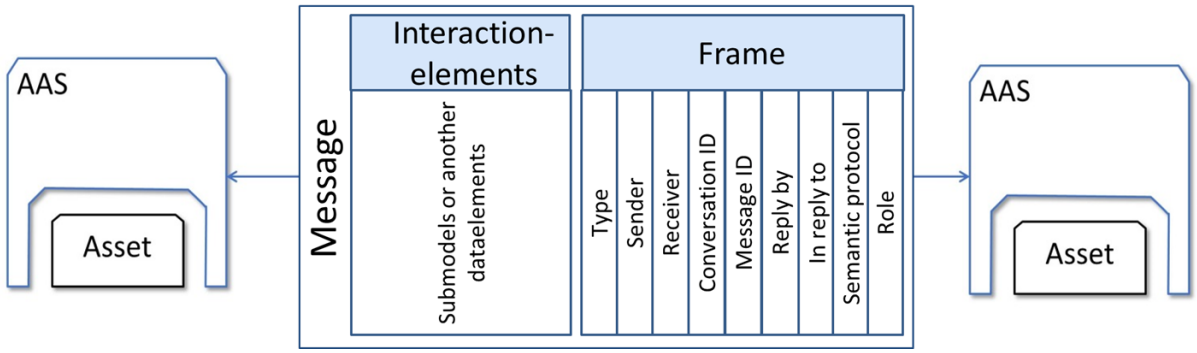

Figure 8: Message-based interaction between I4.0-Components

The structure of the message is shown in the Figure 9 and the Table 1.

Table 1: Message structure (VDI/VDE 2193-1)

Message-area	**VDI/VDE 2193-1** Message-element	**Description**	**Use**	**Demonstrator** Message-element
Interaction-elements	Interaktions-element	Represents objects of a message. Receiver must evaluate the content himself.	Optional	`interactionElements`
Frame	Typ	Typ	Mandatory	`type`
	Sender	Sender of a message (administration shell ID)	Mandatory	`sender`
	Empfänger	Message receiver (administration shell ID)	Optional[1]	`receiver`
	KonversationID	ID of a conversation (dialog)	Optional	`conversationId`
	NachrichtenID	ID of a message	Mandatory	`messageId`
	Als-Antwort-auf	Expression that references the original message.	Optional	`inReplyTo`
	Antworten-bis	Time by which the answer must be received	Optional	`replyBy`

[1] Bei Broadcastnachrichten braucht keine Empfängeradress-ID angegeben werden

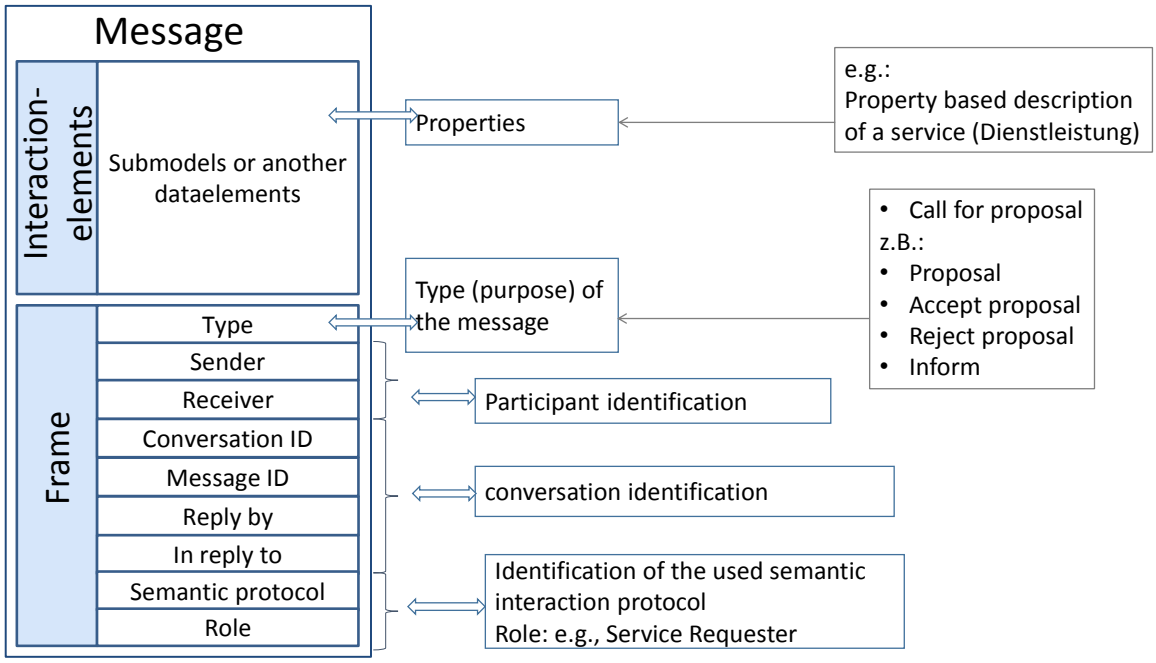

Figure 9: Message structure

The VDI/VDE 2193 is in German language. For the purpose of this demonstrator the purposes of messages are used in English language. Table 2 provides the relations between the German and the English terms.

Table 2: Relation between German and English terms

VDI/VDE 2193 „Typ“	**Sender/Receiver Action**	**Demonstrator „Type“**
Horizontal interactions		
Absage	I refuse to take the action you requested.	**refuse**
Fehler	I couldn't execute your request.	**failure**
Nicht verstanden	I didn't understand you	**not_understood**
Ausschreibung	I would like to receive offers in response to the following call for proposals.	**call_for_proposal**
Angebot	I offer a proposal	**proposal**
Angebotsannahme	I accept your proposal	**accept_proposal**
Angebotsablehnung	I reject your proposal.	**reject_proposal**
Informieren	Believe me, the following is true or not true in my world model.	**Inform**

2.1.2 Interactions between I4.0-Components

The I4.0 components interact with each other to execute I4.0 scenarios. These interactions can be horizontal, i.e. between components of the same plant and factory level, or vertical, i.e. from the product and sensor level to the business level within or even beyond the boundaries of a company (Figure 10). Vertical interaction represents hierarchies in which the information-providing interaction partners act as servers and the requesting interaction partners are the clients.

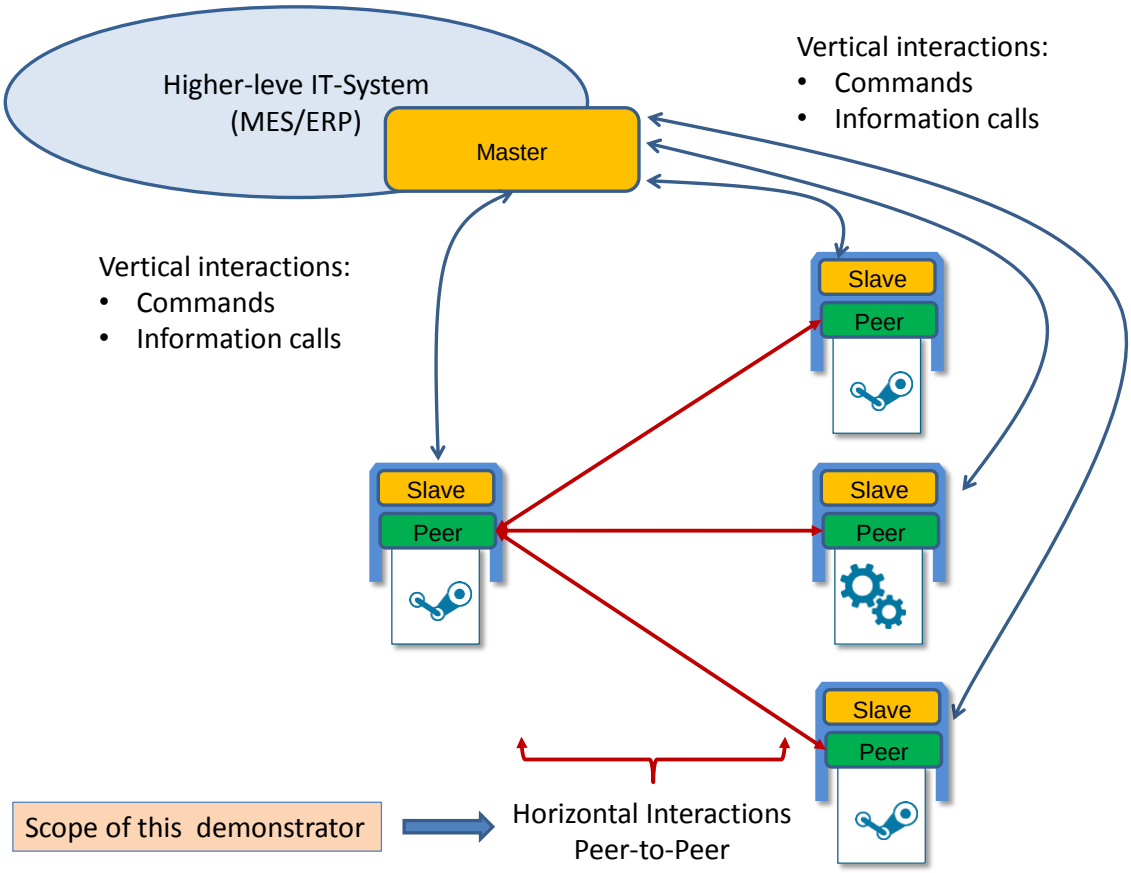

Figure 10: Vertical and horizontal interactions between I4.0-Components [1]

An example of vertical interactions is the exchange of information between product resources and higher-level IT systems. MES and ERP fulfil the tasks of monitoring, planning, documentation and control of the production process and interactions with resources take place in the form of commands and information calls. In this scenario, an administration shell acts as a standardized interface of assets that contains the standardized information models and enables access to the parameter. The behavior of accessed I4.0 components is clearly deterministic from the point of view of higher-level systems. Such AAS are called passive in this document.

It is different in the scenarios where certain autonomy is required from the interaction partners, i.e. production systems or their components, e.g. dynamic optimization of production utilization, avoidance of downtimes due to machine failures, order-driven production, as well as the dynamic orchestration of production resources for cost-optimized production in batch size one. In these scenarios, interaction

partners meet, who both act as equal interaction initiators. For this reason, this is also referred to as horizontal interaction. In this interaction, both partners act as "peers".
The administration shells of I4.0 components can offer both forms of interaction. Accordingly, the I4.0 language allows both horizontal and vertical interactions between I4.0 components.

2.1.3 Horizontal interactions between active AAS (Bidding Process)

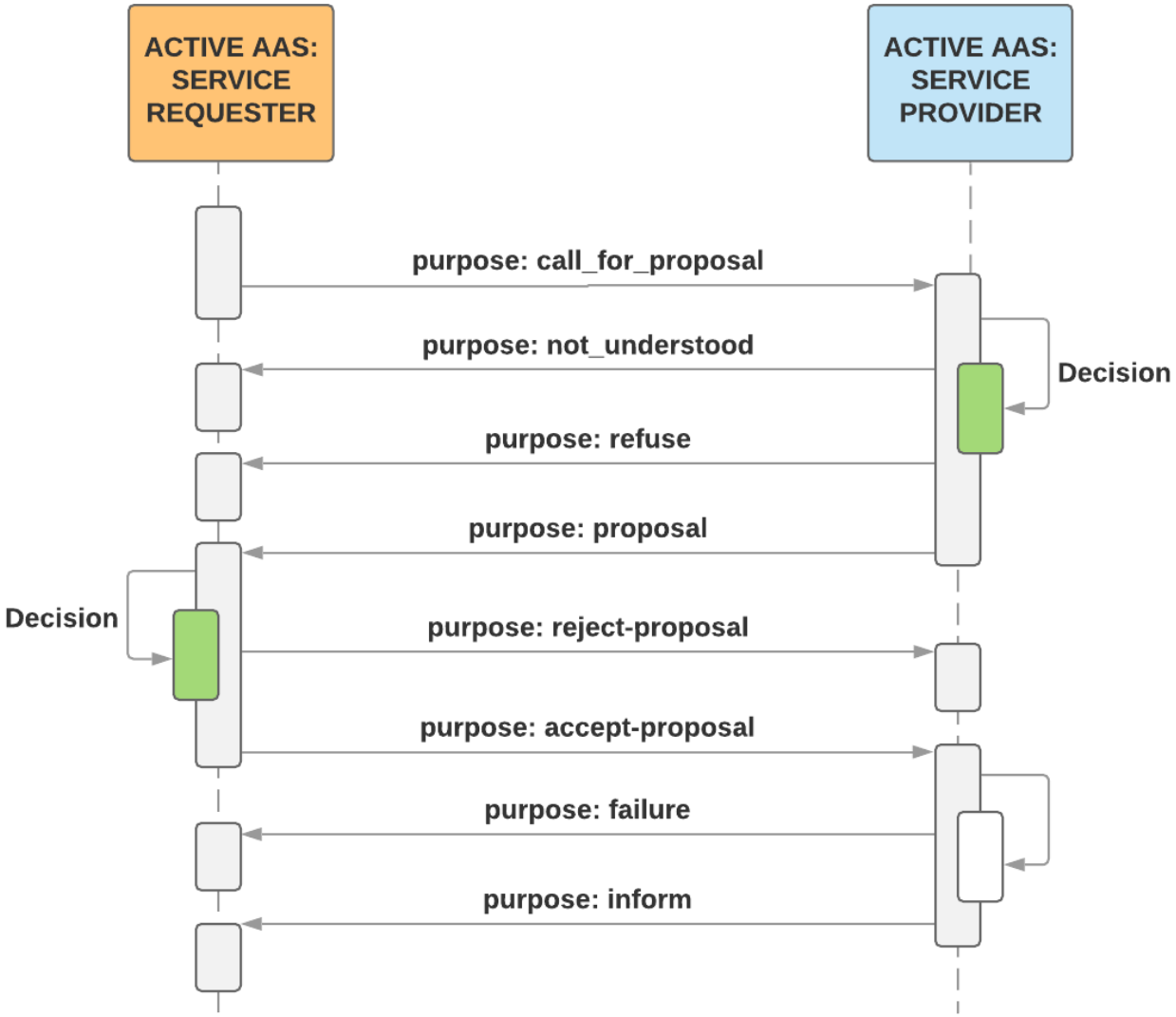

Figure 11: Interaction protocol “Bidding” according to VDI/VDE 2193-2

The VDI/VDE 2193 defines the sequence of messages in the dialogs between I4.0 components. VDI/VDE 2193-2 describes an important interaction protocol, the “bidding process”. The bidding process is described in detail in chapter 1.3.
If the cooperation of I4.0 components is orchestrated according to the interaction protocol "bidding process", at least two situations occur in which administrative shell have to make decisions.

On the one hand, this relates to the decision whether an asset is able to process a requested service. Technical and commercial aspects that require a decision are looked at. This is the case, for example, when a proposal is prepared. On the other hand, the requesting component must decide which of the received offers will be accepted. The decision/ optimization algorithms can pursue several goals. For example, the best economic conditions should be negotiated.

In this version of demonstrator only horizontal interactions are implemented.

2.1.4 Interactions between AAS in the demonstrator

The Figure 12 illustrates the interactions between several active AAS, which participate in the bidding process, the interaction with the passive AAS of the Robot and changing the role of SP to SR after the winning in a tendering process.

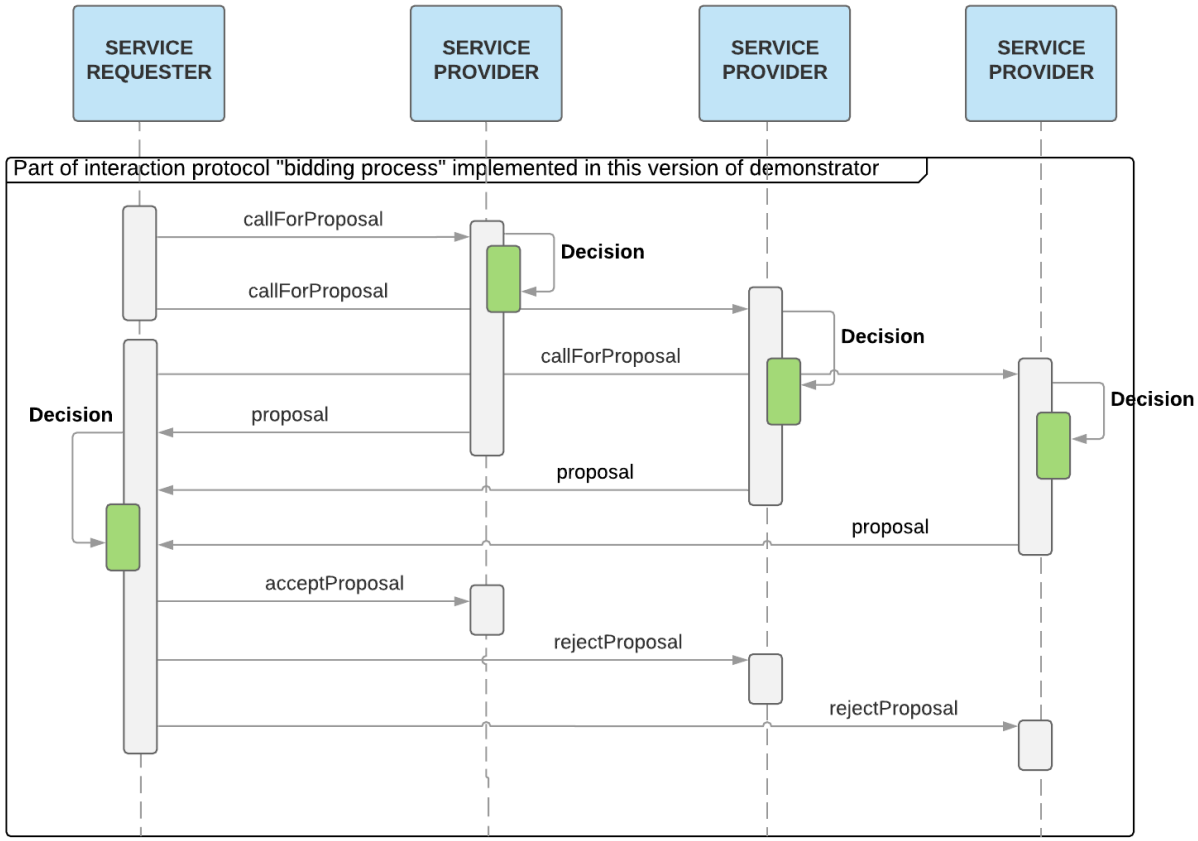

Figure 12: Example of interaction between components in the demonstrator

3 Property-based description of a service (German: Dienstleistung)

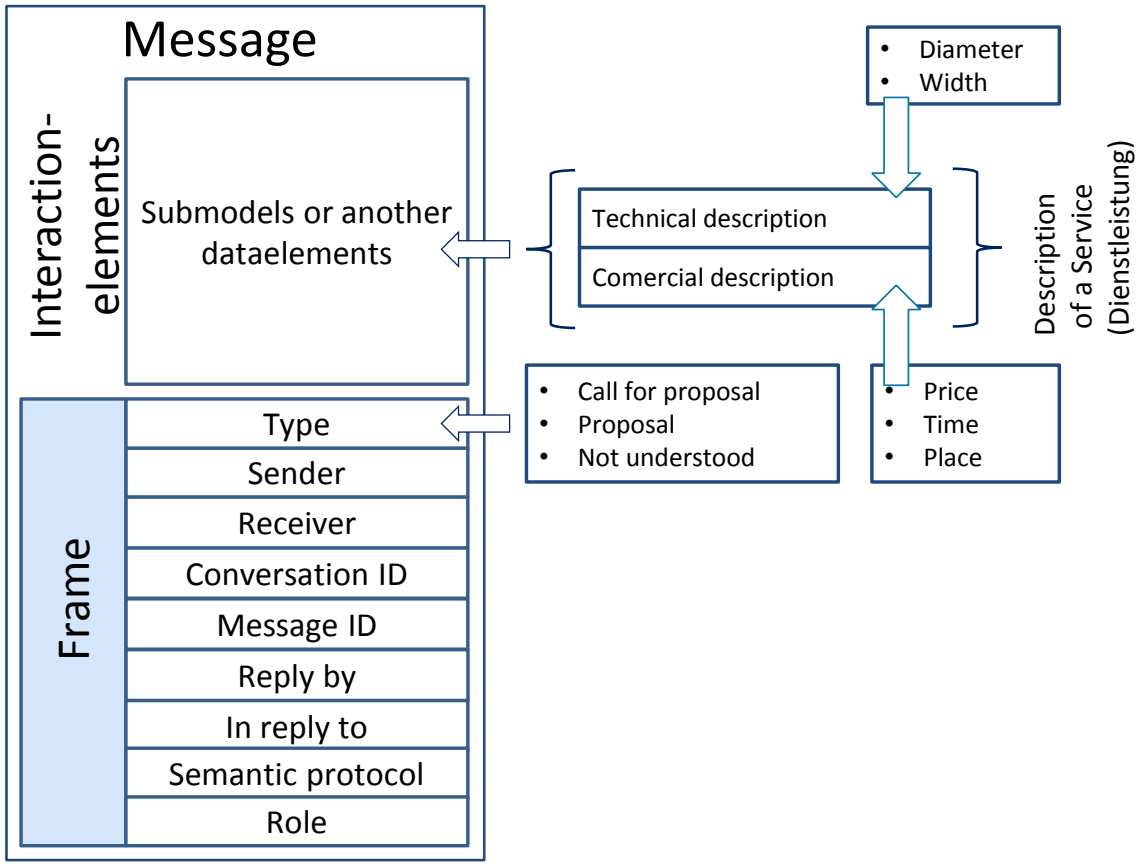

Figure 13: Content of a message 5

As described above the active AAS in this demonstrator provides the capabilities of their assets to other I4.0-Components in the form of services[3] (German: Dienstleistungen). In [2] the concept of the description of services with IEC 61360-based properties is introduced. According to this concept includes the property-based description of a service a technical and the commercial part. The commercial part describes the conditions of the service providing and includes the specification of a time, a place and a price of a service provision. This schema is used to describe the required services (German: Dienstleistungen) of service requester (call for proposal-message) and services which can be provided by service provider (proposal-message).

[3] in the economic sense of the word

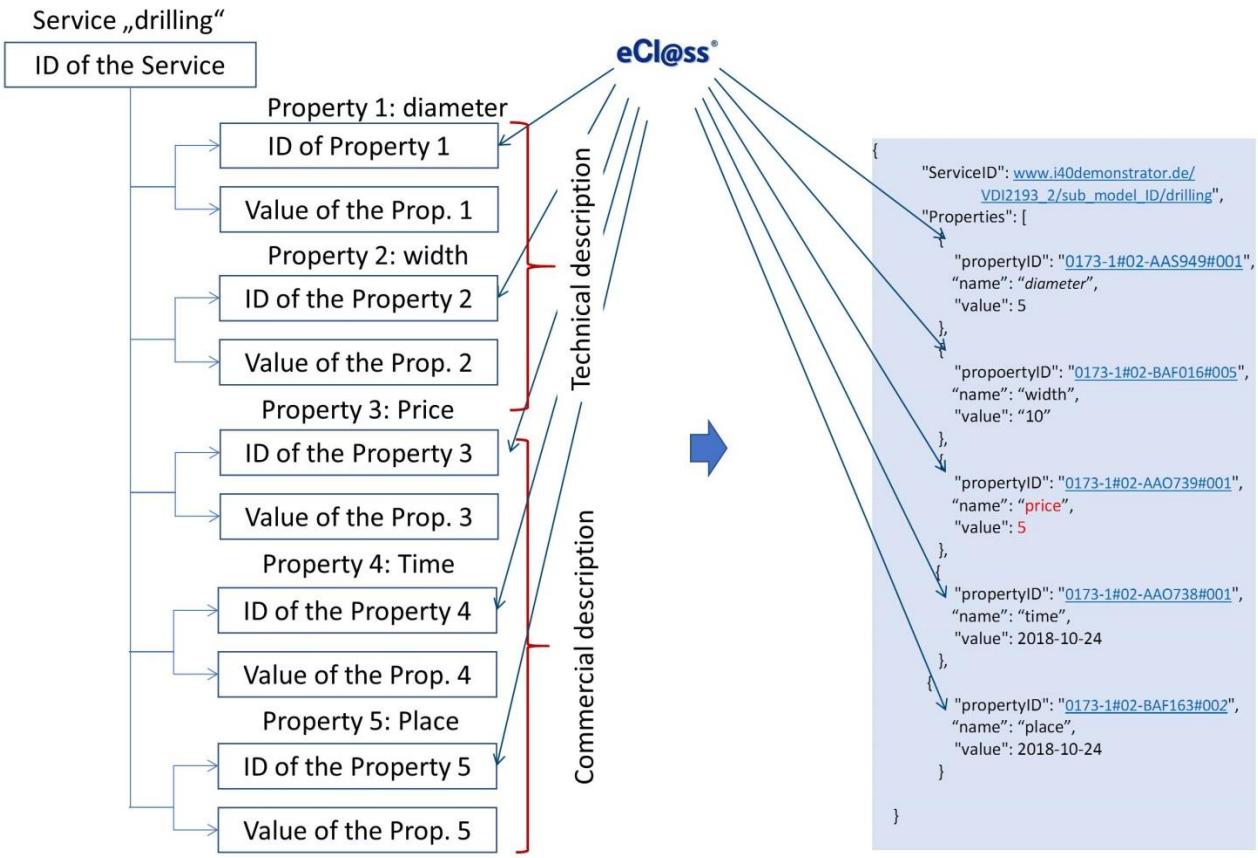

Figure 14: Description of a service "drilling" with eCl@ss-Properties. The serialization of the submodel shown in this Figure is not up-to-date. The serialization of the submodel corresponding to the document “administration shell in detail”, is described in chapter 5.

4 Behavior of Service Requester and Service Provider by bidding

For the description of behavior of Service Requester and Service Provider implemented in this demonstrator the UML activity diagrams are shown in the Figure 15 and Figure 16. These activity diagrams are exemplary and meant to support developers. Other equivalent state machines and activity diagrams can be more complex (in terms of number of states and sub states) or simpler.

The first state after deploying of the Service Requester implemented in this demonstrator is the configuration of a new call for proposal. After SR has sent the proposal, he waits for new proposals.

After the call for proposal was sent by interaction manager, the AAS changes to the next state “wait for proposals”. This state is complex and contents other states. If an proposal is received, it must be validated (it will be submodel ID, IDs of the properties, values of the properties checked). If the received proposal is not valid, an error code will be returned. If the proposal is valid, it will be stored in proposal storage.

After the waiting time is over and at least one proposal received, the AAS changes to the state “proposal evaluation”. If the SR did not receive any proposals, he sends “call for proposal” again. In this state, an optimization problem is solved by explicitly calling an optimization algorithm.

After the best proposal has been chosen, interaction manager sends to the winner “accept-proposal” message. The remaining components (SP) receive “rejectProposal” messages.

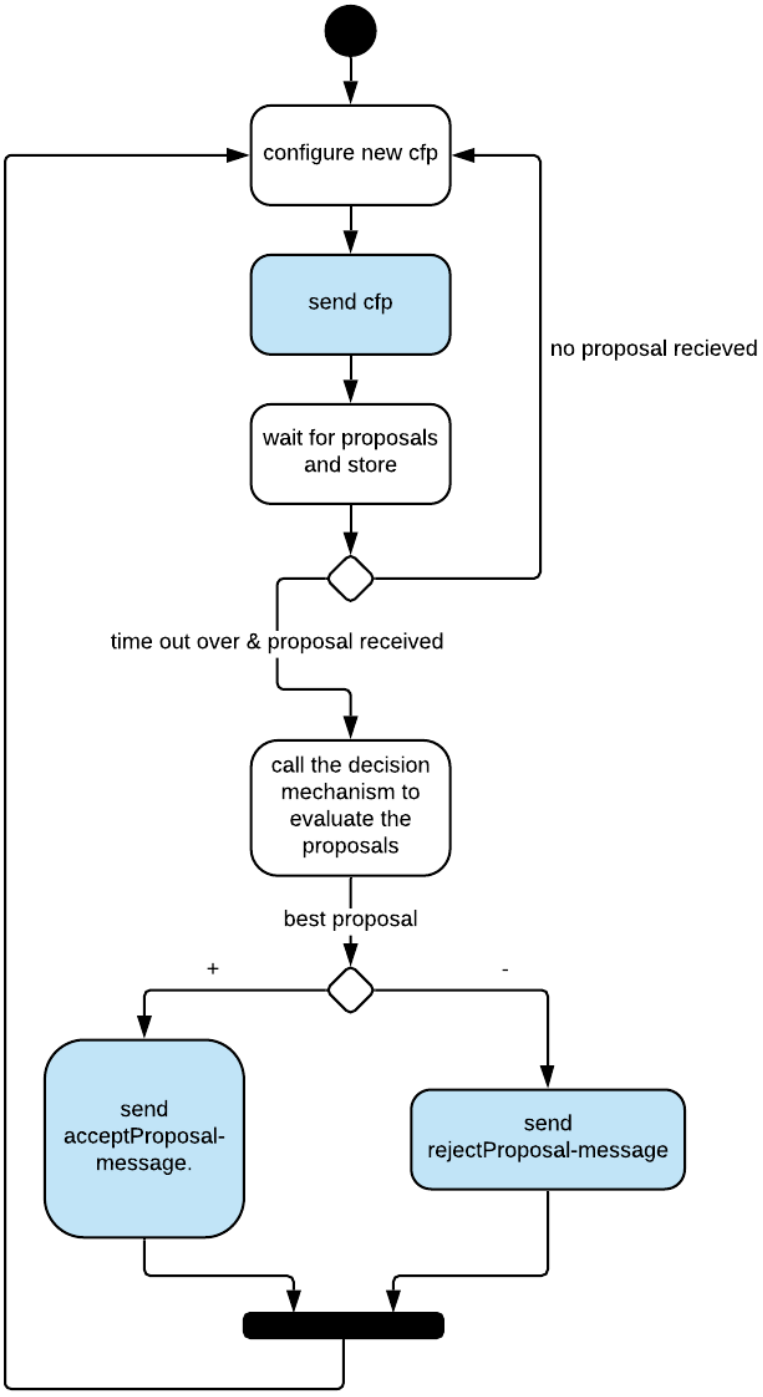

Figure 15: UML Activity diagram describing the behavior of service requester

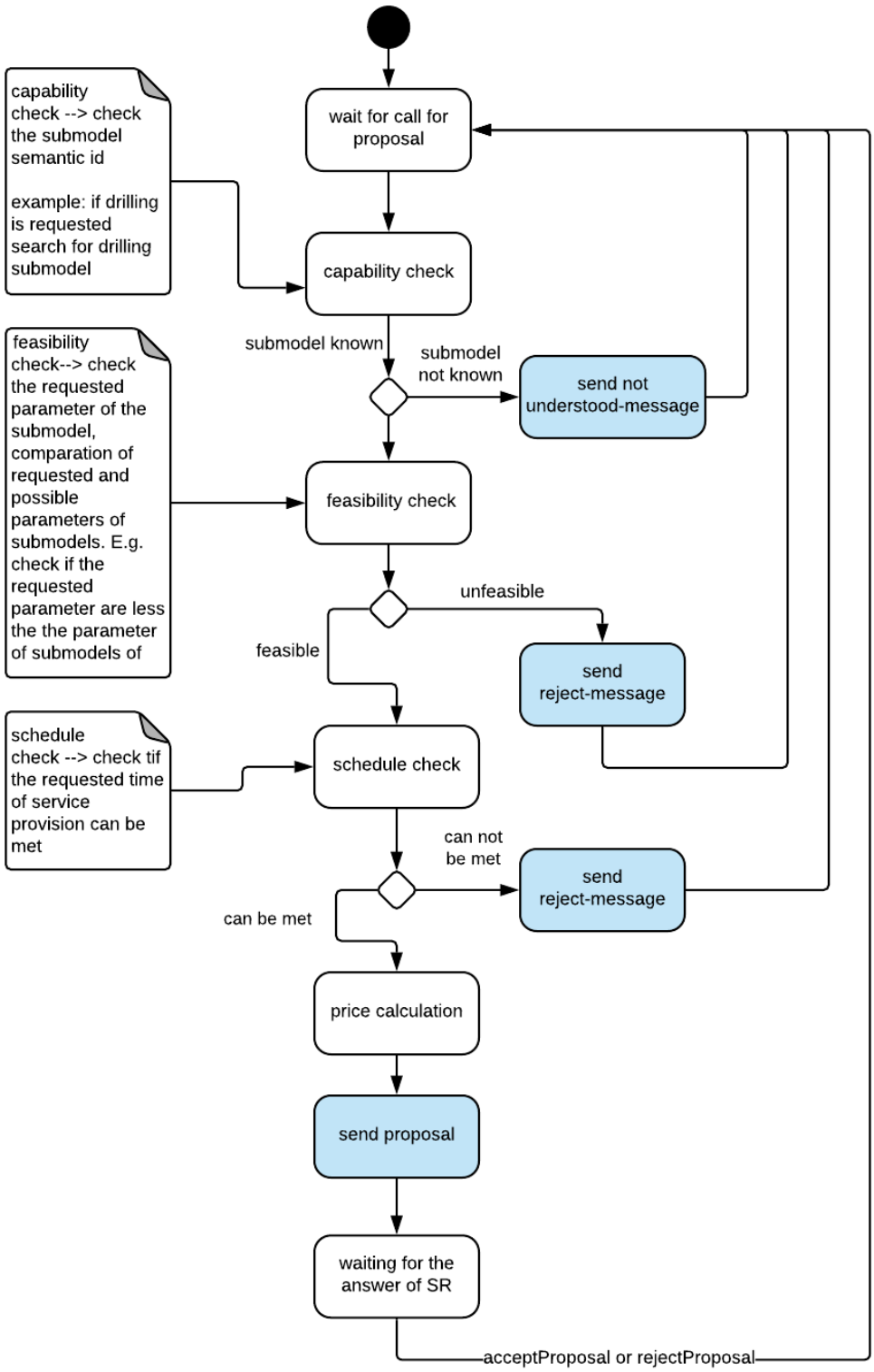

Figure 16: UML activity diagram describing the behavior of service provider

The first state after deploying of the Service Provider implemented in this demonstrator is the waiting for new call for proposal. After SR has sent the proposal, he waits for new proposals.

After component manager has selected the role SP, the AAS is in the state “wait for call for proposal”. After receiving the call of proposal, the interaction manager calls the capability check algorithm. The capability check algorithm browses the passive Part of AAS and proves if the requested submodel exists in his knowledge base (is known). If SP knows the submodel, in the next step it checks the technical parameters of the requested service (feasibility check). If the parameters match, the interaction manager calls the economical algorithm, which calculates the price and the time of service provision. After the price and the time are calculated, the interaction manager of SP sends the “proposal” to SR. If the requested submodel is not known, the SP sends the message “not understood” to the SR.

After the sending of proposal, SP waits for the decision of the SR (accept proposal or reject proposal message). After the answer is received Sp changes to the state “wait for call for proposal”

5 Submodel description

In this version of demonstrator the AAS of SR and SP includes only one submodel.

Table 3: Exemplary description of the submodel “drilling”

Submodel ID	submodelElement ID	Short ID	Description	Value
0173-1#01-AKG243014	X	X	eCl@ss-Klassifikation: 25-09-07-02 Bohren (Lohnfertigung) [AKG243014] English: Boring (subcontracting)	X
	0173-1#02-BAB216	Senkdurchmesser	Diameter of the cylindrical bore of a hole in the center of a tool or tool holder that can receive a screw head to create a connection	-
	0173-1#02-AAA014	Senktiefe	Depth of the cylindrical countersinking of a hole in the center of a tool or tool holder that can receive a screw head to create a connection	-
	0173-1#02-BAF163#002	Ort	Place of service provision	-
	0173-1#02-AAO738#001	Zeit	Time of service provision	-
	0173-1#02-AAO739#001	Preis	Value of a product or service according to the price list	

The JSON serialization of the submodel “Boring” shown and used in the demonstrator was created with the tool “AASX Package Explorer”.

JSON serialization of the submodel “Boring” :


```
{
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#01-AKG243014",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "identification": {
                "idType": "URI",
                "id": "www.company.com/ids/sm/7341_5170_7091_4616"
            },
            "idShort": " Boring ",
            "modelType": {
                "name": "Submodel"
            },
            "kind": "Instance",
            "submodelElements": [
                {
                    "value": "50",
                    "semanticId": {
                        "keys": [
                            {
                                "type": "GlobalReference",
                                "local": false,
                                "value": "0173-1#02-BAB216",
                                "index": 0,
                                "idType": "IRDI"
                            }
                        ]
                    },
                    "idShort": "durchmesser",
                    "category": "PARAMETER",
                    "modelType": {
                        "name": "Property"
                    },
                    "valueType": {
                        "dataObjectType": {
                            "name": "string"
                        }
                    }
                },
                {
                    "value": "100",
                    "semanticId": {
                        "keys": [
                            {
                                "type": "GlobalReference",
                                "local": false,
                                "value": "0173-1#02-AAA014",
                                "index": 0,
                                "idType": "IRDI"
                            }
                        ]
                    },
                    "idShort": "tiefe",
                    "category": "PARAMETER",
                    "modelType": {
                        "name": "Property"
```

```
                },
                "valueType": {
                    "dataObjectType": {
                        "name": "string"
                    }
                },
                "kind": "Instance"
            },
            {
                "value": "9F4H4JRR+5F",
                "semanticId": {
                    "keys": [
                        {
                            "type": "GlobalReference",
                            "local": false,
                            "value": "0173-1#02-BAF163#002",
                            "index": 0,
                            "idType": "IRDI"
                        }
                    ]
                },
                "idShort": "ort",
                "category": "PARAMETER",
                "modelType": {
                    "name": "Property"
                },
                "valueType": {
                    "dataObjectType": {
                        "name": "string"
                    }
                }
            },
            {
                "value": "1558461600",
                "semanticId": {
                    "keys": [
                        {
                            "type": "GlobalReference",
                            "local": false,
                            "value": "0173-1#02-AAO738#001",
                            "index": 0,
                            "idType": "IRDI"
                        }
                    ]
                },
                "idShort": "zeit",
                "category": "PARAMETER",
                "modelType": {
                    "name": "Property"
                },
                "valueType": {
                    "dataObjectType": {
                        "name": "string"
                    }
                }
            }
        ]
}
```

6 Serialization of messages

The VDI/VDE 2193-1 introduces a simple form of serialization of messages. The ZVEI SG2 working group "models and standards“ is developing a formal model “administrations shell in detail". In the next steps, the serialization of the messages must be adapted to this document.

6.1 JSON-Serialization

Based on the VDI/VDE 2193-1, the following serialization scheme is used in this demonstrator:

```
{

  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "www.admin-shell.io.schema.json.1.1",
  "type": "object",
  "additionalProperties": false,
  "properties": {
    "frame": {
      "$ref": "#/definitions/frame"
    },
    "interactionElements": {
      "type": "array",
      "items": {
        "$ref": "#/definitions/Referable"
      }
    }
  },
  "required": ["frame", "interactionElements"],
  "definitions": {
    "frame": {
      "additionalProperties": false,
      "type": "object",
      "properties": {
        "semanticProtocol": {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/Reference"
        },
        "type": {
          "type": "string"
        },
        "conversationId": {
          "type": "string"
        },
        "messageId": {
          "type": "string"
        },
        "sender": {
          "$ref": "#/definitions/conversationMember"
        },
        "receiver": {
          "$ref": "#/definitions/conversationMember"
        },
        "replyBy": {
          "type": "number"
        },
        "inReplyTo": {
          "type": "string"
        }
      },
      "required": ["semanticProtocol", "type", "messageId"]
    },
```

```
    "conversationMember": {
      "type": "object",
      "properties": {
        "identification": {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/Identifier"
        },
        "role": {
          "type": "object",
          "properties": {
            "name": {
              "type": "string"
            }
          }
        }
      }
    },
    "Referable": {
      "oneOf": [
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/AssetAdministrationShell"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/Submodel"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/ConceptDescription"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/Asset"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/Property"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/SubmodelElementCollection"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/File"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/Blob"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/ReferenceElement"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/ConceptDictionary"},
        {
          "$ref": "www.admin-shell.io.vwsid.schema#/definitions/View"}
      ]
    }
  }
}
```

In chapters 6.1.1 – 6.1.6 the concrete expressions of serializations of messages of the I4.0 language are represented according to this schema.

6.1.1 callForProposal


```
{
    "frame": {
        "semanticProtocol": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "idType": "URI",
                    "value": "www.admin-shell.io/interaction/bidding",
                    "local": false
                }
            ]
        },
        "type": "callForProposal",
        "messageId": "1",
        "sender": {
            "identification": {
                "id": "www.admin-shell.io/aas/1",
                "idType": "URI"
            },
            "role": {
                "name": "serviceRequster"
            }
        },
        "receiver": {
            "identification": {
                "id": "www.admin-shell.io/aas/2",
                "idType": "URI"
            },
            "role": {
                "name": "serviceProvider"
            }
        },
        "replyBy": "",
        "inReplyTo": "0",
        "conversationId": "68175af8-8826-49f6-8953-5749fc0a45bd"
    },
    "interactionElements": [
        {
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#01-AKG243014",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "identification": {
                "idType": "URI",
                "id": "www.company.com/ids/sm/7341_5170_7091_4616"
            },
            "idShort": "drilling",
            "modelType": {
                "name": "Submodel"
            },
            "kind": "Instance",
            "submodelElements": [
                {
                    "value": "50",
```

```
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-BAB216",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "durchmesser",
            "category": "PARAMETER",
            "modelType": {
                "name": "Property"
            },
            "valueType": {
                "dataObjectType": {
                    "name": "string"
                }
            }
        },
        {
            "value": "100",
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-AAA014",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "tiefe",
            "category": "PARAMETER",
            "modelType": {
                "name": "Property"
            },
            "valueType": {
                "dataObjectType": {
                    "name": "string"
                }
            },
            "kind": "Instance"
        },
        {
            "value": "9F4H4JRR+5F",
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-BAF163#002",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "ort",
            "category": "PARAMETER",
```

```
                    "modelType": {
                        "name": "Property"
                    },
                    "valueType": {
                        "dataObjectType": {
                            "name": "string"
                        }
                    }
                },
                {
                    "value": "1558461600",
                    "semanticId": {
                        "keys": [
                            {
                                "type": "GlobalReference",
                                "local": false,
                                "value": "0173-1#02-AAO738#001",
                                "index": 0,
                                "idType": "IRDI"
                            }
                        ]
                    },
                    "idShort": "zeit",
                    "category": "PARAMETER",
                    "modelType": {
                        "name": "Property"
                    },
                    "valueType": {
                        "dataObjectType": {
                            "name": "string"
                        }
                    }
                },

            ]
        }
    ]
}
```

6.1.2 proposal


```
{
    "frame": {
        "semanticProtocol": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "idType": "URI",
                    "value": "www.admin-shell.io/interaction/bidding",
                    "local": false
                }
            ]
        },
        "type": "proposal",
        "messageId": "1",
        "sender": {
            "identification": {
                "id": "www.admin-shell.io/aas/2",
                "idType": "URI"
            },
            "role": {
                "name": "serviceProvider"
            }
        },
        "receiver": {
            "identification": {
                "id": "www.admin-shell.io/aas/1",
                "idType": "URI"
            },
            "role": {
                "name": "serviceRequester"
            }
        },
        "replyBy": "",
        "inReplyTo": "0",
        "conversationId": "68175af8-8826-49f6-8953-5749fc0a45bd"
    },
    "interactionElements": [
        {
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#01-AKG243014",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "identification": {
                "idType": "URI",
                "id": "www.company.com/ids/sm/7341_5170_7091_4616"
            },
            "idShort": "drilling",
            "modelType": {
                "name": "Submodel"
            },
            "kind": "Instance",
```

```
"submodelElements": [
    {
        "value": "50",
        "semanticId": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "local": false,
                    "value": "0173-1#02-BAB216",
                    "index": 0,
                    "idType": "IRDI"
                }
            ]
        },
        "idShort": "durchmesser",
        "category": "PARAMETER",
        "modelType": {
            "name": "Property"
        },
        "valueType": {
            "dataObjectType": {
                "name": "string"
            }
        }
    },
    {
        "value": "100",
        "semanticId": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "local": false,
                    "value": "0173-1#02-AAA014",
                    "index": 0,
                    "idType": "IRDI"
                }
            ]
        },
        "idShort": "tiefe",
        "category": "PARAMETER",
        "modelType": {
            "name": "Property"
        },
        "valueType": {
            "dataObjectType": {
                "name": "string"
            }
        },
        "kind": "Instance"
    },
    {
        "value": "9F4H4JRR+5F",
        "semanticId": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "local": false,
                    "value": "0173-1#02-BAF163#002",
                    "index": 0,
                    "idType": "IRDI"
                }
            ]
```

```
            },
            "idShort": "ort",
            "category": "PARAMETER",
            "modelType": {
                "name": "Property"
            },
            "valueType": {
                "dataObjectType": {
                    "name": "string"
                }
            }
        },
        {
            "value": "1558461600",
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-AAO738#001",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "zeit",
            "category": "PARAMETER",
            "modelType": {
                "name": "Property"
            },
            "valueType": {
                "dataObjectType": {
                    "name": "string"
                }
            }
        },
        {
            "value": "5",
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-AAO739#001",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "preis",
            "category": "PARAMETER",
            "modelType": {
                "name": "Property"
            },
            "valueType": {
                "dataObjectType": {
                    "name": "string"
                }
            }
        }
    ]
}
```

```
    ]
}
```

6.1.3 notUnderstood

```
{
    "frame": {
        "semanticProtocol": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "idType": "URI",
                    "value": "www.admin-shell.io/interaction/bidding",
                    "local": false
                }
            ]
        },
        "type": "notUnderstood",
        "messageId": "1",
        "sender": {
            "identification": {
                "id": "www.admin-shell.io/aas/2",
                "idType": "URI"
            },
            "role": {
                "name": "serviceProvider"
            }
        },
        "receiver": {
            "identification": {
                "id": "www.admin-shell.io/aas/1",
                "idType": "URI"
            },
            "role": {
                "name": "serviceRequester"
            }
        },
        "replyBy": "",
        "inReplyTo": "0",
        "conversationId": "68175af8-8826-49f6-8953-5749fc0a45bd"
    }

}
```

6.1.4 refuseProposal

```
{
    "frame": {
        "semanticProtocol": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "idType": "URI",
                    "value": "www.admin-shell.io/interaction/bidding",
                    "local": false
                }
            ]
        },
        "type": "refuseProposal",
        "messageId": "1",
        "sender": {
            "identification": {
```

```
                "id": "www.admin-shell.io/aas/2",
                "idType": "URI"
            },
            "role": {
                "name": "serviceProvider"
            }
        },
        "receiver": {
            "identification": {
                "id": "www.admin-shell.io/aas/1",
                "idType": "URI"
            },
            "role": {
                "name": "serviceRequester"
            }
        },
        "replyBy": "",
        "inReplyTo": "0",
        "conversationId": "68175af8-8826-49f6-8953-5749fc0a45bd"
    }

}
```

6.1.5 rejectProposal

```
{
    "frame": {
        "semanticProtocol": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "idType": "URI",
                    "value": "www.admin-shell.io/interaction/bidding",
                    "local": false
                }
            ]
        },
        "type": "rejectProposal",
        "messageId": "1",
        "sender": {
            "identification": {
                "id": "www.admin-shell.io/aas/1",
                "idType": "URI"
            },
            "role": {
                "name": "serviceRequester"
            }
        },
        "receiver": {
            "identification": {
                "id": "www.admin-shell.io/aas/2",
                "idType": "URI"
            },
            "role": {
                "name": "serviceProvider"
            }
        },
        "replyBy": "",
        "inReplyTo": "0",
        "conversationId": "68175af8-8826-49f6-8953-5749fc0a45bd"
    }

}
```

6.1.6 acceptProposal


```
{
    "frame": {
        "semanticProtocol": {
            "keys": [
                {
                    "type": "GlobalReference",
                    "idType": "URI",
                    "value": "www.admin-shell.io/interaction/bidding",
                    "local": false
                }
            ]
        },
        "type": "acceptProposal",
        "messageId": "1",
        "sender": {
            "identification": {
                "id": "www.admin-shell.io/aas/1",
                "idType": "URI"
            },
            "role": {
                "name": "serviceRequester"
            }
        },
        "receiver": {
            "identification": {
                "id": "www.admin-shell.io/aas/2",
                "idType": "URI"
            },
            "role": {
                "name": "serviceProvider"
            }
        },
        "replyBy": "",
        "inReplyTo": "0",
        "conversationId": "68175af8-8826-49f6-8953-5749fc0a45bd"
    },
    "interactionElements": [
        {
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#01-AKG243014",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "identification": {
                "idType": "URI",
                "id": "www.company.com/ids/sm/7341_5170_7091_4616"
            },
            "idShort": "drilling",
            "modelType": {
                "name": "Submodel"
            },
            "kind": "Instance",
            "submodelElements": [
                {
                    "value": "50",
```

```
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-BAB216",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "durchmesser",
            "category": "PARAMETER",
            "modelType": {
                "name": "Property"
            },
            "valueType": {
                "dataObjectType": {
                    "name": "string"
                }
            }
        },
        {
            "value": "100",
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-AAA014",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "tiefe",
            "category": "PARAMETER",
            "modelType": {
                "name": "Property"
            },
            "valueType": {
                "dataObjectType": {
                    "name": "string"
                }
            },
            "kind": "Instance"
        },
        {
            "value": "9F4H4JRR+5F",
            "semanticId": {
                "keys": [
                    {
                        "type": "GlobalReference",
                        "local": false,
                        "value": "0173-1#02-BAF163#002",
                        "index": 0,
                        "idType": "IRDI"
                    }
                ]
            },
            "idShort": "ort",
            "category": "PARAMETER",
```

```
                    "modelType": {
                        "name": "Property"
                    },
                    "valueType": {
                        "dataObjectType": {
                            "name": "string"
                        }
                    }
                },
                {
                    "value": "1558461600",
                    "semanticId": {
                        "keys": [
                            {
                                "type": "GlobalReference",
                                "local": false,
                                "value": "0173-1#02-AAO738#001",
                                "index": 0,
                                "idType": "IRDI"
                            }
                        ]
                    },
                    "idShort": "zeit",
                    "category": "PARAMETER",
                    "modelType": {
                        "name": "Property"
                    },
                    "valueType": {
                        "dataObjectType": {
                            "name": "string"
                        }
                    }
                },
                {
                    "value": "5",
                    "semanticId": {
                        "keys": [
                            {
                                "type": "GlobalReference",
                                "local": false,
                                "value": "0173-1#02-AAO739#001",
                                "index": 0,
                                "idType": "IRDI"
                            }
                        ]
                    },
                    "idShort": "preis",
                    "category": "PARAMETER",
                    "modelType": {
                        "name": "Property"
                    },
                    "valueType": {
                        "dataObjectType": {
                            "name": "string"
                        }
                    }
                }
            ]
        }
    ]
}
```

7 Interaction framework

7.1 Overview

This demonstrator uses MQTT as interaction framework to transport the messages between the I4.0 components. There is no other requirement for the implementation of the AAS of the I4.0 component. The initial component implementation is based on node-red implementation (OvGU).

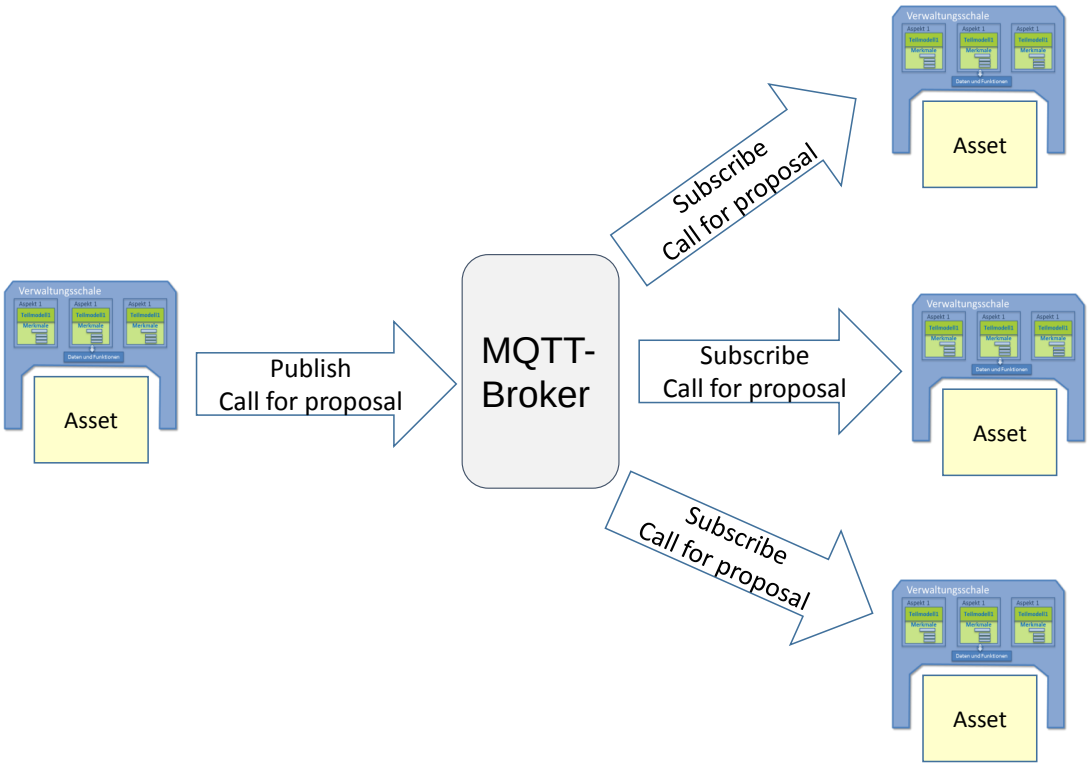

Figure 17: MQTT messaging

The MQTT broker address is:
ifatnode.et.uni-magdeburg.de: 1883

Login / Passwort:

- industrie40 / %industrie:4.0

7.2 MQTT-Topics

For the exchange of messages between active AAS three MQTT topics are created (Figure 18).

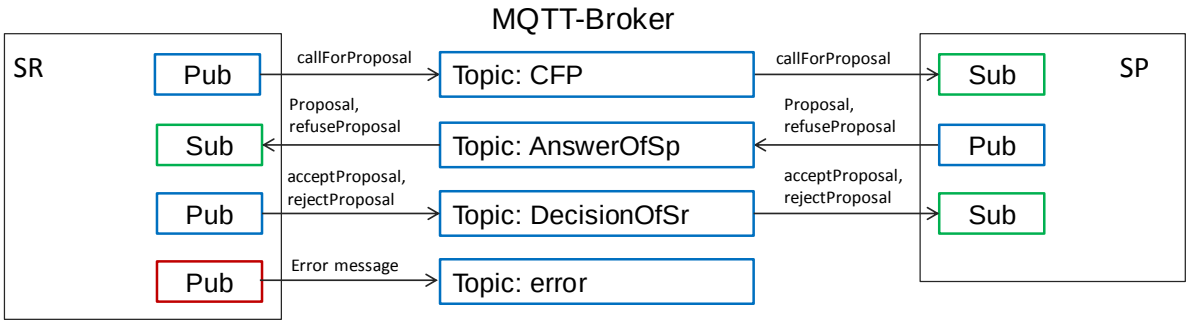

Figure 18: MQTT-Topics

There is an additional “error” topic. There the error messages are published, which can occur when sending a proposal.

8 Literaturverzeichnis

[1] C. Diedrich, „Semantik der Interaktionen von I4.0-Komponenten,“ in *AUTOMATION – Leitkongress der Mess- und Automatisierungstechnik*, Baden-Bden, 2018.

[2] A. Belyaev und C. Diedrich, „Erhöhung der Flexibilität und Robustheit zwischen Interaktionspartnern durch das Merkmalmodell,“ *at - Atomatisierungstechnik,* Bd. 3, Nr. 67, pp. 193-207, 2019.

[3] „I4.0-Sprache: Vokabular, Nachrichtenstruktur und semantische Interaktionsprotokolle,“ Plattform I4.0, 2018.

[4] „Details of the Asset Administration Shell. Part 1 - The exchange of information between partners in the value chain of Industrie 4.0,“ Plattform I4.0, 2018.

[5] „VDI/VDE 2193 Blatt 1. Sprache für I4.0-Komponenten,“ VDI, Düsseldorf, 2019.

[6] „VDI/VDE 2193 Blatt 2. Sorache für I4.0-Komponenten. Interaktionsprotokoll für ausschreibungsverfahren,“ VDI, Düsseldorf, 2019.

[7] „Verwaltungsschale in der konkreten Praxis - Wie definiere ich Teilmodelle, beispielhafte Teilmodelle und Interaktion zwischen Verwaltungsschalen,“ Plattform Industrie 4.0, Berlin, 2019.

[8] A. Belyaev und C. Diedrich, „Aktive Verwaltungsschale von Industrie 4.0 Komponenten,“ in *Automationkongress 2019*, Baden-Baden, 2019.