

Service and Agent based System Architectures for Industrie 4.0 Systems

Hartwig Baumgärtel
Institute of Organisation and Logistics
Ulm University of Applied Sciences
Ulm, Germany
hartwig.baumgaertel@thu.de

Richard Verbeet
Institute of Organisation and Logistics
Ulm University of Applied Sciences
Ulm, Germany
verbeet@mail.hs-ulm.de

Abstract—Industrie 4.0 is based on three pillars: Vertical Integration, Horizontal Integration and Consistency of Engineering over Product Life Cycle. The implementation of systems to realize these pillars requires an architecture enabling interoperability of distributed components and their digital representations. Potential middleware concepts for an architecture are the service-oriented architectures OPC UA, JADEX/JADEX and Arrowhead. A holistic and modular I4.0 architecture using agent technology to implement Digital Twins and thereby decouples communication middleware from processes and hardware is proposed.

Keywords—I4.0 Architecture, Service-Oriented Architecture, Middleware, JADE/JADEX, OPC UA, Arrowhead, Active Component Shell, Multiagent-System

I. INTRODUCTION

In 2012, the German Academy of Science and Engineering explained the fourth industrial revolution as "autonomous embedded systems networking wirelessly with each other and with the Internet. Cyber-Physical Production Systems (CPPS) are created with intelligent machines, storage systems and operating resources exchanging information, trigger actions and control each other independently. They can improve industrial processes in production, engineering, logistics, supply chain and life cycle management" [1]. By this statement three pillars of Industrie 4.0 "Horizontal Integration of Value Creation Networks", "Vertical Integration in Production Systems" and "Consistency of Engineering over Product Life Cycle" are introduced.

Horizontal Integration means cross-company cooperation in supply network management. It extends to all levels of a network: institutional (company cooperation), social (human cooperation), informational (information processing) and physical level (physical material flow). Horizontal Integration on information level has been implemented in many industrial branches by electronic data interchange (EDI) or via Internet platforms (webEDI). Increasingly, solutions introducing service-oriented architectures (SoA) are realized.

Vertical Integration means interlinking physical entities in production systems, traditionally along the Automation Pyramid (IEC 62264) [2]. Communication takes place exclusively between entities of same or adjacent levels by level-specific communication technologies. There is no direct communication between entities across multiple levels limiting the availability of information about the status of a production system at higher levels and transmission of control instructions for lower levels. Cyber-Physical Production Systems (CPPS) exceed this limitation by using Cyber-Physical Systems (CPS), i.e. the integration of physical objects by information and communication technology to interlink and influence each other. Methods of artificial intelligence [3] or autonomy and self-x properties can be integrated to create properties like decentralization, scalability, or data and process sovereignty. An approach to realize such components is presented in [4].

Consistent engineering over Product Life Cycle means to integrate information on product instances during their operation. Current PLM systems consider product classes, but don't integrate operational information from individual instances. Industrie 4.0 enables the interlinking of these classes and their instances during product life cycle [5]. Individual products communicate with IT systems leading to a new generation of the concept "Digital Factory" as information from the system can be integrated into a virtual model visualizing the state of a systems in real time. The term "Industrie 4.0" became a hype in recent years. However, the development of concepts for the three pillars still require further research and development in automation and information technology, as consistent communication and coordination of billions of CPS in various contexts cannot be reached with today's concepts and infrastructures.

The remainder of this paper is organized as follows. In chapter II, requirements to I4.0 systems and components are collected. Chapter III presents architectural approaches. Chapter IV introduces service-oriented architectures in general and three representatives in particular: JADE/JADEX, OPC UA and Arrowhead. In chapter V, the main contribution of this paper, a holistic I4.0 architecture is proposed. It comprises several existing concepts and orchestrates them in three layers. The paper concludes with a discussion of this architecture in Chapter VI.

II. REQUIREMENTS OF INDUSTRIE 4.0 APPLICATIONS

An I4.0 system is a group of interacting physical or non-physical components meeting the characteristics of Industrie 4.0. Approaches for the implementation of I4.0 systems can be found in the concepts of Internet of Things (IoT) / Industrial IoT (IIoT) and CPS / CPPS, which take up the concept of distributed applications and digital representation. Requirements for IoT systems and their components are described in [6]. Although these are defined in context of a multiagent-system, they can be generalized. However, security requirements regarding authentication and authorization are missing, others are only implicitly listed and hence defined as specific requirements in this article. The conceptual connection to processes is also added.

A. Required Properties of Industrie 4.0 Systems

In [6], the following requirements are listed: it should be possible to virtualize heterogeneous components as homogeneous, digital objects (*Virtualization*), components of the system should be able to communicate with each other to cooperate and interact independent of specific technical network protocols (*Communication Abstraction*), heterogeneous software should be addressed and integrated via generic and standardized interfaces (*Software Interfaces*), hardware and software components should adapt dynamically to the place and the context of their use (*Self- and Context-Awareness*), in order to enable continuous data exchange between the components, they should use uniform data and information models

to ensure interoperability (*Data Abstraction*), the design, implementation and analysis of individual components and the overall system should be enabled by suitable tools (*Development Methodology*), and I4.0 systems should be classifiable to make them comparable in terms of scalability, i.e. number of components, geographic locations and infrastructures, even if no general scalability can be guaranteed in distributed systems (*System's Scale Characterization*).

The paper on hand adds the following requirements. I4.0 systems should be able to interact with each other beyond their borders using dynamic interfaces without limiting their sovereignty of data and decision-making. This interaction should also include connections between components of these systems (*Inter-System-Communication*). They should have a standardized authentication procedure to assess reliability and validity of requests from external systems (*Authentication-Certificate*) and should be integrated in the processes of their enterprises, i.e. they should be designed, developed and implemented with the goal to support business processes and workflows (*Process-Link*).

B. Required Properties of Industrie 4.0 Components

An I4.0 component is software or a physical object that is connected to the digital world by software (digital representation). Given that the ability to connect to the digital world is such a fundamental requirement it is not listed as a separate property. The following properties of I4.0 components are required according to [6]: software should adapt to a physical object to be used for similar objects, e.g. other products of the same product class, it should also have procedures for adapting to unknown objects (*Heterogeneity and Interoperability*), components should offer their functions and properties through different interfaces (services) enabling similar components to offer different services or different components to offer the same service (*Augmentation Variation*), the decentralized calling and searching of functions (services) should be possible to adapt to changing or new tasks, infrastructures, and environments (*Decentralized Management*), components should adapt autonomously (learning) or should be configurable with little manual effort for new requirements (*Dynamic Evolution*), and it should be possible to describe physical parts (objects) of components by properties to compare them with other physical parts or objects (*Objects Characterization*).

The paper on hand adds the following requirements. Components should be able to make autonomous decisions. They must be equipped with an adequate knowledge representation and decision-making mechanism (*Autonomy*). The authorization to call functions (services) or query properties by other components should be checkable using internal lists or certificates (*Authorization*) and Components should be able to decide which information is transferred to whom and when. (*Data Sovereignty*). Moreover, the requirements of services should be checkable when searching for services. This requires a standardized description of services or the ability to evaluate them dynamically. (*Quality of Service*)

III. APPROACHES FOR INDUSTRIE 4.0 ARCHITECTURES

The basis for an implementation of I4.0 systems is a software architecture that enables the fulfilment of the requirements listed above. A software architecture is a structured or hierarchical arrangement of system components and a description of their relationships, properties and capabilities.

One widely used architecture model is the Reference Architecture Model Industrie 4.0 (RAMI4.0) [7]. This model comprises three dimensions, which are referred to as "Hierarchy Levels", "Life Cycle Value Stream" and "Layers". Hierarchy Levels addresses the hierarchical structure of production systems and is based on the standards IEC 62264 (automation levels in discrete industry) and IEC 61512 (process industry). It extends them by "Products" at the bottom and "Connected World" at the top. Life Cycle Value Stream represents the standard IEC 62890, which is based on the life cycle model of ZVEI for automated systems [8]. It contains phases for an entire product or system class (product planning and development) and phases for individual instances of this classes (manufacturing, use, maintenance). Layer consists of the characteristics "Asset" (physical components and humans), "Integration" (representation of properties and capabilities of physical components), "Communication" (data formats and interaction services), "Information" (data/information models, rules for signal processing and service interfaces), "Functional" (functions, services and decision logic) and "Business" (business logic, legal rules and service orchestration).

Beside RAMI4.0, approaches for architectures of I4.0 systems are IMSA [9] and IIRA [10]. The projects IDEAS [11], PRIME [12], and GOOD-MAN [13] propose implementations by multiagent-systems. They were developed for applications in different domains, but in general they describe the interlinking of components and their control in a production system and don't define a generic approach for an I4.0 architecture. IDEAS and GOOD-MAN are based on a service-oriented architecture. The implementation of IoT systems by agents is described in [6] by ACOSO. IDEAS, PRIME and GOOD-MAN describe architectures for production systems, ACOSO and IIRA realize the concept of Internet of Things. RAMI4.0, on the other hand, is an abstract but more integrated architecture that is better suited as a generic template for I4.0 architectures. The characteristics of the dimension Layers of RAMI4.0 can be aggregated to the following layers:

- *Process Layer (Business + Functional)*
At this level, functions for fulfilling business processes are executed by IT systems (PLM, ERP, MES, BI systems, ...) based on defined rules and logics.
- *Information Layer (Information + Communication)*
This level ensures the interoperability of physical (machine, load carrier, conveyor technology, ...) and non-physical (IT systems, such as ERP or MES, orders, ...) system components.
- *Technology Layer (Integration + Asset)*
At this level, the physical and digital worlds are connected (Digital Twin, Device Shadow, ...) to enable the interaction of assets.

The Process Layer contains software applications implemented according to predefined logic and rules to provide functions for the fulfillment of business processes. The Technology Layer describes the physical objects that are necessary for the execution of many of these functions. In traditional IT systems, these layers communicate with each other hierarchically via bus systems (fieldbus, plant/process bus), Ethernet (LAN) or local radio technologies (WLAN, BLE), as described by the Automation Pyramid according to ISA 95. There are no direct connections across multiple layers. A middleware for an integrated mediation between the two worlds "process" and "technology" through the digital Information

Layer must meet new requirements in the context of I4.0 to enable Vertical Integration and Horizontal Integration. Service-oriented architectures are an approach for the implementation of such a middleware.

IV. SERVICE-ORIENTED ARCHITECTURES

A service-oriented architecture (SoA) is a style of software design modularizing previously monolithic IT systems and able to fulfill many of the requirements of I4.0 systems. Representatives from different domains based on this style are JADE/JADEX, OPC UA and Arrowhead. A SoA is based on the concept of a service, a mechanism providing access to a capability of a system through a predefined interface in accordance with restrictions and policies defined in a service description [14]. In [15], services are concretized to Industrie 4.0 services if they realize functions of an I4.0 component. Such a component consists of a physical asset with a digital representation, a so-called Asset Administration Shell. Services can be related to physical entities and only their description and access mechanisms are placed in the digital world, e.g. a service can activate a micro controller, setting actuators of a physical system in motion or machines in a production systems can provide services to access their capabilities. The following considerations focus on the digital elements of service-oriented architectures according to the concept of Industrie 4.0 services.

A. JADE/JADEX

JADE (Java Agent Development Framework) [16] is a framework for the design of multiagent-systems according to FIPA standard [17] to ensure the interoperability of agents. A JADE agent platform provides the components *Agent Management System* (AMS) (Agent Management), *Directory Facilitator* (DF) (Service Registry), and *Agent Communication Channel* (ACC) (Middleware). A JADE application consists of agents interacting with each other by exchanging messages. They are implemented on a platform that provides basic functions for running the agents, e.g. transferring messages. Each platform implements an AMS and a DF agent. An AMS Agent manages the components of the agent platform (white pages). It can create and delete other agents or can terminate the entire platform. A DF Agent manages the service register (yellow pages) of the SoA of the platform, enable other agents to provide and search services.

Agents can communicate with each other regardless of their native platform via ACC. The communication is based on the asynchronous transmission of messages whose format is defined by the Agent Communication Language (ACL) according to FIPA containing information about sender and receiver, the type of message, its content and ontology. Communication between platforms is realized by MTP modules (Message Transport Protocol), which receive and send ACL messages. The protocols HTTP and IIOP are offered for message transfer. Every MTP module must know the network addresses of the other MTP modules and the transfer of messages is verified by a user password query. Another integrated functionality of a JADE platform is provided by a Web Service Integration Gateway (WSIG) module. It transforms local services to web services described by WSDL and translates SOAP calls to ACL messages and vice versa.

JADEX (JADE eXtension) [18] is a JADE-based agent framework integrating the goal- and context-oriented Belief-Desire-Intention (BDI) concept into agents. In addition, it ex-

tends the functionality of JADE regarding interoperability between components and agent platforms. A JADEX agent platform possess a functionality called "Platform Awareness" by which other JADEX-platforms can be discovered in local and global networks. Services provided for global usage are made local accessible on other platforms by the creation of local proxy agents offering these services.

B. OPC UA

OPC UA (Object Linking and Embedding for Process Control Unified Architecture) is a multi-layered service-oriented architecture. It is designed for standardized networks of systems in local IT and automation domains. The concept based on OPC UA Servers and OPC UA Clients interacting by services and executed on a hardware device called host. On each host several standard components are implemented, such as "Local Discovery Service Application" (LDS) or an application for authentication and authorization. The LDSs within a local network use a mechanism to share provided services to enable the search for applications.

Three core components of OPC UA are the address space and information model of an OPC UA Server, standardized services provided by these servers and consumed by OPC UA Clients, and a binary communication protocol based on TCP to exchange data. The address space of an OPC UA Server consists of nodes of pre-defined types, e.g. object, variable and method. Objects are used to map processes, system components and systems. Variables represent data variables or describe properties of nodes. Methods represent "light-weight" functions and can be called by the standard service "Method Call". Information models supplement address spaces by objects, variables and methods related to a specific context. While the address space of an OPC UA Server is context-dependent, the standard services are fixed. Due to these standardized services no individual services need to be registered or searched for. The registration and search of services are designed for software applications, not for single services. Therefore registration is very compact but searching for specific capabilities is quite difficult. The capabilities of CPS are not provided as services, but as methods in an address space. A dynamic search for capabilities at runtime (late binding) is impossible unless the OPC UA Servers use a priori known information models.

In [10] OPC UA is mapped to RAMI4.0. It is placed at communication layer and is used for instances of production systems in the second phase of a product life cycle. Related to Hierarchy Levels in a production system, OPC UA can be used for products and field devices up to stations in local automation domains. Entire corporate or cross-company IT-networks are not addressed by OPC UA. The development of processes for product and production system classes are not covered so far [10].

C. Arrowhead

Arrowhead is a framework for the design and implementation of systems of systems based on the concept of a local cloud and a SoA. Various functions are provided by the local cloud by standardized services. An Arrowhead service is an exchange of information between the provider and the user of a service. It can be implemented and accessed by any number of applications. Its interface contains a description that provides a common understanding of its effects and parameters. An application can use several services, process the received information and provide services itself to fulfill complex

tasks. Each local cloud must provide the core components *Service-Registry-and-Discovery-System* (application management, service registry), *Authentication-and-Authorization-System* (security, service access), and *Orchestration-System* (service allocation by request-response sequence) to ensure communication and interaction of applications (Fig. 1) [19].

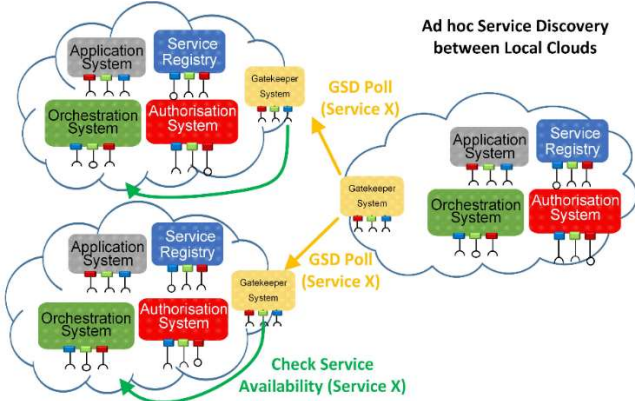


Fig. 1. Core Components of local clouds and interoperability by Gatekeeper and Gateway Systems in an Arrowhead network [20]

A connection to another local cloud can be initiated via the Global Service Discovery (GDS) service offered by the Arrowhead application Gatekeeper System (Fig. 1). The Gatekeeper System forwards a service request from a local application to Gatekeeper Systems of other local clouds. They check their local Service Registry to evaluate if there are suitable service providers. If there is a provider, it is reported back to the requesting Gatekeeper. After the service provider is selected and the authorization of the requester is confirmed, a connection is established by another Arrowhead component started in both local clouds, called Gateway Systems.

V. PROPOSAL FOR A HOLISTIC I4.0 ARCHITECTURE

The approaches presented in chapter 4 meet many requirements for an I4.0 architecture, but don't offer a holistic concept for Horizontal and Vertical Integration. Therefore an architecture based on the concepts of the Asset Administration Shell (AAS) and BDI agents realizing an Active Component Shell (ACS) and implemented by a BDI multiagent-system is proposed. It will be shown that this architecture can use each of the presented middleware approaches.

A. Asset Administration Shell

An I4.0 architecture needs a concept for a digital representation of components at Technology Layer, so that they can be addressed by Process Level. One approach is the I4.0 component of Plattform Industrie 4.0. [21] define an I4.0 component as a combination of one or more objects with an Asset Administration Shell. It is a digital representation of a physical and non-physical object, e.g. product, machine or order, contains information about properties and functions of the represented object and can communicate with other AAS. It is divided into a publicly visible header, which contains a unique ID, designation of the object and offered services, and an internal body, whose sub-models each define a hierarchy of properties describing data or functions. [22] present the sub-models *Index data item* (unique reference), *Property value statement* (static properties and data), *Documentation* (status information), *Communication* (mechanism and protocols), and *Function* (capabilities). A component manager manages access and update of these properties. Thus, an AAS forms a mediation layer between different components, applications and people.

B. BDI Agents

A technical agent is a delimitable unit (hardware and/or software) with defined objectives. A technical agent strives to achieve these goals through autonomous behaviour and interacts with his environment, including other agents [23]. Agents that communicate and interact with each other in an environment are referred to as multiagent-system. The BDI concept (Belief, Desire, Intention) was implemented as a procedural reasoning system, which reduces the abstract description of desires and intentions (Desire) to the concrete fulfillment of goals and plans (Intention) on the basis of available information (Belief) [24]. A BDI agent can store information about itself and its environment in an internal model. Each instance of this model describes a state of the environment which corresponds to its individual view (belief). A sequence of actions to consciously change the environment is called a plan (intention). The execution of plans is controlled by goals (Desire) describing a specific state of the environment. Thereby an agent strives to transform the current state of the environment into these target states. The mechanism to select plans can go beyond a rule-based logic to enable agents to act even if a goal cannot be reached by a single plan. Therefore BDI agents are capable of proactive behaviour, since their internal model enables them to evaluate the effects of their decisions. A BDI agent can implement an AAS as an Active Component and integrate the process layer of an I4.0 architecture by its cognitive complexity [4].

C. Active Component

The concept of an Active Component [25] combines the concepts of object, component, service and agent orientation for the implementation of distributed systems. Object orientation models real scenarios using objects and interfaces as a concept to create an abstraction of real objects and processes, whereas component orientation extends the object-oriented concept by designing independent entities with defined functions and properties to provide more modularity and reusability. Service orientation integrates a business and technical perspective through workflows that represent processes and call services to realize activities. Agent orientation is a paradigm that describes agents as abstractions for autonomous entities with control their status and actions.

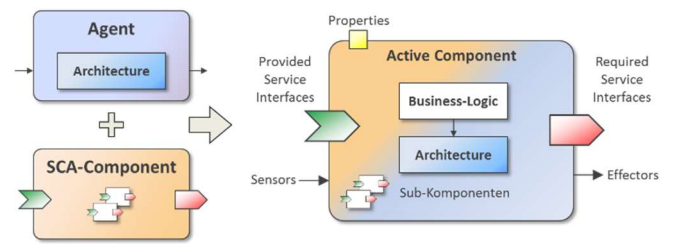


Fig. 2. Active Component as provider and consumer of services [18]

The concept of an Active Component is based on the Service Component Architecture (SCA) and extends it by using agents to transform a passive SCA component into autonomous service providers and consumers (Fig. 2). An Active Component realized by BDI agents enables the modeling of distributed systems by autonomous components that can interact and communicate with each other by services and are capable of goal- and context-oriented problem solving. Therefore it can be used to implement an AAS of an I4.0 component. The header of an AAS is realized by capabilities of the agent framework and its body by a BDI structure. An ACS realized

in JADEX can also fulfill the requirements for the implementation of an AAS and its sub-models specified by [22]:

- **Identification:** Each JADEX agent receives a unique ID from the AMS of a platform; each platform is also identifiable via a unique key. In combination, each agent is uniquely identifiable.
- **Representation:** A JADEX agent can be implemented with any Beliefs that refer to dynamic or static properties of the asset and can be retrieved via services or integrated into their description.
- **Communication:** JADEX agents are connected to all agents on their platform, the platform itself can connect to other platforms to reach the agents there. Communication is possible through direct messages or services.
- **Lifecycle Stage:** JADEX agents can be created and deleted, but it would be appropriate to maintain the digital representation of an asset throughout its lifecycle and provide an agent with properties and functions for all phases. It is also possible to move a JADEX agent between platforms.
- **Function:** A JADEX agent can be connected to all functions of an asset and offer them as a service. Any functions can be programmed as plans.
- **Interoperability:** FIPA messages contain a data field for ontologies. If services are to be used instead of direct messages, meta services can be set up to exchange ontologies.
- **Index data item:** A JADEX agent monitors its Beliefs (data) and plans (functions) independently, so that no a priori known references are necessary. If external access is required, they can be classified and referred according to any information model.
- **Property value statement:** A JADEX agent is aware of its internal structure (Belief, Plan, Goal) and registers all changes of this structure.
- **Documentation:** All changes to Beliefs and plans are registered by the Reaction-Deliberation-Mechanism and can be saved. Information about agents can be requested from AMS or DF of the platform but can also be directly integrated when a JADEX agent is initialized.
- **Communication:** JADEX agents communicate via services or messages that can be retrieved and received by all other JADEX agents. Separate libraries can also be integrated to use additional communication protocols such as MQTT in addition to the platform's ACC.
- **Function:** When an asset is connected to a JADEX agent, it can link its functions to plans and integrate them into a service.

D. Industrie 4.0 Architecture

The proposed I4.0 architecture is based on the following principles: The execution of business logic and functions takes place at Process Layer, the digital representation of technical assets at Technology Layer. A middleware connects the systems of these layers, but it works independently to connect as many components as possible with each other and not have the claim to integrate functions from the other levels directly. The middleware concepts OPC UA, JADE/JADEX and Arrowhead can establish consistent connections according to their specifications between IT systems and hardware with

digital representations. However, compatibility with hardware according to other specifications is problematic because references to components, capabilities or protocols are not available (Fig. 3). Arrowhead and JADE/JADEX have similar components and functionalities in a local cloud or an agent platform that can be synchronized. An interaction with OPC UA applications, however, is difficult because OPC UA has no central components and cannot implement arbitrary service interfaces, whereby a service discovery in OPC UA is only indirectly possible.

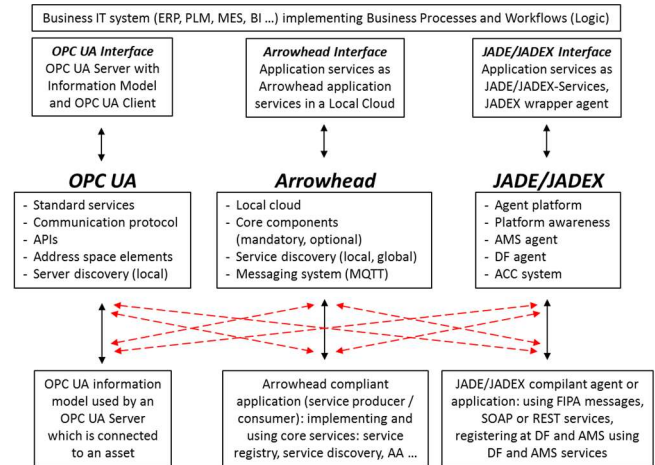


Fig. 3. Options for a cross-conceptual configuration realizing an I4.0 architecture

One approach to solve this compatibility problem is to implement an AAS as an ACS. It can be implemented with any interface for a service and inherently unite process and technology layer. An asset can be described and classified according to its integration and its complexity in terms of activity: physical object (e.g. active tooling machine or passive container), IT system (e.g. active ERP system or passive product database) and information object (active production order or passive maintenance documentation). According to [4] the interaction of BDI agents with assets and middleware can be described by the following variants: an asset can interact with middleware itself (separate), is represented by an ACS (complementary) or is fully integrated into an ACS (integrated). An ACS can be implemented with different interfaces for a service to enable it to interact with different middleware. OPC UA (SoA or Publish-Subscribe) can be used for automation technology in a production or logistics system where components and processes are known. Arrowhead enables enterprise interoperability by its security concept, core components, global service discovery and inter-cloud communication by means of Gatekeeper and Gateway Systems. JADE/JADEX acts as native middleware of BDI agents since no separate interface must be generated.

A combination of agents and OPC UA is introduced by [26]. Besides using OPC UA for agent communication specific OPC UA Servers provide white and yellow pages in a SoA to replace agent-specific software components. This approach is not very natural for both agents and OPC UA. Furthermore, it is restricted to local area networks due to the limited network scope of OPC UA. A more natural approach is the combination of BDI agents as ACS and Arrowhead. Every agent must implement interfaces to use the core services of Arrowhead (service registration/discovery, authentication/authorization) for inter-cloud communication, wide area service discovery and application collaboration. JADE/JADEX and

Arrowhead both use standard methods and formats for service description like WADL, implement REST as design principle of services and use similar architectural concepts (Arrowhead: Local Clouds, JADE/JADEX: Agent Platform).

VI. DISCUSSION

Industrie 4.0 systems can be implemented using concepts like Internet of Things or Cyber-Physical Systems that take up the concept of distributed applications and digital representation. These systems are based on an architecture consisting of layers for processes, information and technology. The information layer is realized by middleware and has an important meaning in digital and networked systems of Industrie 4.0. The concept of a service-oriented architecture is a promising approach for the realization of distributed applications as they are realized in Industrie 4.0. The communication between the process and technology layer required for Industrie 4.0 can be realized by various technological approaches. Companies can implement almost any "path" along the elements in Figure 3 analogous to a morphological box, at which cross-middleware interfaces are possible, but sometimes are very intricate.

The proposed architecture is based on agent technology. [4] show BDI multi-agent systems are capable to realize the requirements of I4.0 systems. The Asset Administration Shell is implemented by an Active Component Shell (ACS), i.e. a software artefact controlled by process logic and able to provide or retrieve services with interfaces for several middleware like OPC UA or Arrowhead. An ACS can be implemented by an BDI agent. The process layer of an architecture is realized by its Reaction-Deliberation-Mechanism. The technology layer can also be integrated into a BDI agent, but there are no generic approaches yet.

The architecture also shows that the discussion "agents or SoA" is unnecessary. A SoA describes the concept of middleware in a system, agent technology focuses on the internal logic of system components. Integrated platform communication is necessary for agent-based applications but is not suitable for realizing cross-system integration in industrial applications. On the other hand, the direct integration of intelligent control concepts into middleware is not promising.

An ACS is intended to decouple tasks such as service orchestration or digital representation from middleware and integrate them into a component so that middleware can concentrate on the transfer of information. Otherwise, the layers of an I4.0 architecture might become blurred and concepts or systems are used for tasks they shouldn't execute. Middleware plays a major role in digitized and networked systems, so it should work as transparently and efficiently as possible. Due to the heterogeneous requirements of applications, however, there will never be "the" I4.0 middleware, e.g. networking properties or reaction time may be more important. Therefore, the goal of the development of an I4.0 architecture should be modular approaches and not concepts that can only be used for a certain domain and are based on a single standard.

REFERENCES

- [1] H. Kagermann, W. Wahlster, and J. Helbig, *Deutschlands Zukunft als Produktionsstandort sichern: Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0*. [Online] Available: <https://www.acatech.de/publikation/umsetzungsempfehlungen-fuer-das-zukunftsprojekt-industrie-4-0-zwischenbericht-des-arbeitskreises-industrie-4-0/>. Accessed on: Jul. 02 2019.
- [2] ANSI, *ISA 95: Enterprise-Control System Integration 1: Models and Terminology*, 2000.
- [3] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia: Pearson Education Limited, 2016.
- [4] R. Verbeet and H. Baumgärtel, "Implementierung von autonomen Industrie 4.0-Systemen mit BDI-Agenten," in *Handbuch Industrie 4.0: Industrie-4.0-Anwendungsszenarien für die Automatisierung*, B. Vogel-Heuser, T. Bauernhansl, and M. t. Hompel, Eds., Berlin, Heidelberg: Springer-Verlag, 2019.
- [5] H. Baumgärtel, "Electronics and ICT as enabler for digital industry and optimized supply chain management covering the entire product lifecycle," Productive4.0 Project Consortium, 2019.
- [6] G. Fortino, W. Russo, C. Savaglio, W. Shen, and M. Zhou, "Agent-oriented cooperative smart objects: From IoT system design to implementation," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 99, pp. 1–18, 2017.
- [7] IEC, *Smart Manufacturing—Reference Architecture Model Industry 4.0 (RAMI4.0): IEC Standard PAS 63088*, 2017.
- [8] R. Birkhofer et al., *Life-Cycle-Management für Produkte und Systeme der Automation: Ein Leitfaden des Arbeitskreises Systemaspekte im ZVEI Fachverband Automation*. Frankfurt, M.: Zentralverb. Elektrotechnik- und Elektronikindustrie, Fachverb. Automation, 2010.
- [9] SCI 4.0, "Alignment Report for Reference Architectural Model for Industrie 4.0/ Intelligent Manufacturing System Architecture / Intelligent Manufacturing System Architecture: Sino-German Industrie 4.0/Intelligent Manufacturing - Standardisation Sub-Working Group," Federal Ministry of Economic Affairs and Energy, 2018.
- [10] S. W. Lin and S. Mellor, *Architecture Alignment and Interoperability: An Industrial Internet Consortium and Plattform Industrie 4.0 Joint Whitepaper*. IIC:WHT:IN3:V1.0:PB:20171205. [Online] Available: www.iiconsortium.org.
- [11] M. Onori, N. Lohse, J. Barata, and C. Hanisch, "The IDEAS project: plug & produce at shop-floor level," *Assembly Automation*, no. 32, pp. 124–134, 2012.
- [12] A. Rocha et al., "An agent based framework to support plug and produce," in *12th International Conference on Industrial Informatics (INDIN)*, Porto Alegre RS, Brazil, 2014, pp. 504–510.
- [13] J. Barbosa et al., "Implementation of a Multi-Agent System to Support ZDM Strategies in Multi-Stage Environments," in *16th International Conference on Industrial Informatics (INDIN)*, 2018, pp. 822–827.
- [14] OASIS, *Reference Model for Service Oriented Architecture 1.0: Committee Specification 1*. [Online] Available: <http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf>. Accessed on: Jul. 03 2019.
- [15] DIN, *Reference Model for Industrie 4.0 Service Architectures*, 16593-1, 2018.
- [16] F. Bellifemine, A. Poggi, and G. Rimassa, "JADE: a FIPA2000 compliant agent development environment," in *Proceedings of the 5th international conference on Autonomous agents*, 2001, pp. 216–217.
- [17] FIPA, *FIPA*. [Online] Available: <http://www.fipa.org/index.html>. Accessed on: Jun. 05 2019.
- [18] L. Braubach and A. Pokahr, "Developing distributed systems with active components and Jadex," *Scalable Computing: Practice and Experience*, no. 13, pp. 100–120, 2012.
- [19] J. Delsing, *IoT automation: Arrowhead framework*. Boca Raton, FL: CRC Press, Taylor & Francis Group, 2017.
- [20] C. Hegedus, D. Kozma, G. Soos, and P. Varga, "Enhancements of the Arrowhead Framework to refine inter-cloud service interactions," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, Florence, Italy, 2016, pp. 5259–5264.
- [21] P. Adolphs et al., "Struktur der Verwaltungsschale: Fortentwicklung des Referenzmodells für die Industrie 4.0-Komponente," Bundesministerium für Wirtschaft und Energie (BMWi), Berlin, 2016.
- [22] X. Ye and S. H. Hong, "Toward Industry 4.0 Components: Insights Into and Implementation of Asset Administration Shells," *EEE Ind. Electron. Mag.*, vol. 13, no. 1, pp. 13–25, 2019.
- [23] VDI/VDE, *Agentensysteme in der Automatisierungstechnik*, 2653, 2010.
- [24] M. Georgeff and A. Lansky, "Reactive reasoning and planning: An experiment with a mobile robot," in *Proceedings of the 6th National Conference on Artificial Intelligence (AAAI 1987)*, 1987, p. 677.
- [25] A. Pokahr and L. Braubach, "Active components: A software paradigm for distributed systems," in *Proceedings of International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2011)*, 2011, pp. 141–144.
- [26] M. Hoffmann et al., "Semantic integration of multi-agent systems using an OPC UA information modeling approach," in *IEEE 14th International Conference on Industrial Informatics (INDIN)*, Poitiers, France, 2016, pp. 744–747.