Industrial Frameworks for Internet of Things: A Survey

Cristina Paniagua and Jerker Delsing

***Abstract*—The Internet of Things (IoT) has gained popularity and is increasingly used in large scale deployments for industrial applications. Such deployments rely on the flexibility and scalability of systems and devices. Heterogeneous systems need to be interoperable and work together seamlessly. In order to manage such system of systems, it is important to work with a framework that not only supports the flexible nature of IoT systems but also provides adequate support for industrial requirements, such as real-time and runtime features, architectural approaches, hardware constraints, standardization, industrial support, interoperability, and security. The selection of an appropriate framework results difficult due to the rising number of available frameworks and platforms, which offer different support for the aforementioned requirements. Therefore, this article investigates the features of seven prominent frameworks for the purpose of simplifying the selection of a suitable framework for an industrial application. The aim of this article is to present the recent developments and state-of-the-art of industrial IoT frameworks and provide a technical comparison of their features and characteristics.**

***Index Terms*—Frameworks, Industrial Internet of Things (IIoT), system of systems (SoS).**

I. Introduction

THE Internet of Things (IoT) paradigm is rapidly growing with the new development, design, and integration of technology. As a result, several IoT frameworks have been developed to fulfill some of the new requirements and needs. Frameworks intend to provide a high-level abstraction, including management, security, interoperability, flexibility and interconnection of services, systems, and devices.

Looking at the different application domains, the industrial application of the IoT, namely the Industrial Internet of Things (IIoT), is expected to play an important role. In recent years, there has been growing interest in the IIoT. Boyes *et al.* [1] analyzed the IIoT concept and related IoT taxonomies. Based on the definitions presented, the IIoT is the connection of smart assets (things) as part of a larger system of systems (SoS) in industrial environments to optimize the production value. The requirements and specifications in industrial environments are more restrictive than in consumer applications and are usually more focused on communication protocols and technologies, security requirements, quality of service (QoS), and device interoperability [2]. The frameworks presented in this survey are all considered key enabling technologies for the development of IoT applications, especially in industrial environments.

The IoT paradigm has been researched and analyzed from numerous points of view, and as a result, a plethora of scientific articles and surveys can be found in the literature. Broad topics include architectures, protocols, technologies, applications, challenges and opportunities [3]–[5], security [6], [7], and cloud and edge computing [8]. More specific domains include smart homes [9] and 5 G communications [10], among others.

Nevertheless, only a few surveys can be found in reference to IoT frameworks. In 2015, Derhamy *et al.* [11] presented a complete survey about commercial frameworks and platforms for the IoT, providing a detailed comparison in terms of utilized approaches, supported protocols, usage in industry, and hardware requirements. Due to the dynamism of the industry and the evolution of consortium and projects, some of the information introduced is currently outdated. The focus changes from cloud platforms to more generic and complete IoT frameworks that provide solutions in industrial scenarios. This survey presents an updated vision of the current IoT frameworks for the industry, including new emerging frameworks and an extensive analysis of their features. In 2018, Ammar *et al.* [12] presented an updated survey regarding the most popular cloud platforms for the IoT, which provides a comparative analysis based on security features.

The aim of this work is to broaden current knowledge of the emergent IIoT frameworks that are currently available and are not present in today's literature to provide a better understanding and a reference of the framework features and technologies, which represent a novelty in regards to other surveys of the IoT domain.

The objectives of this survey are as follows.

1) To provide a picture of the current state of the art of IIoT frameworks. The framework features and group alliances are continuously evolving and expanding, and consequently, published surveys rapidly become obsolete. The information provided in this article is updated and accurate at the time of writing and uses original specifications and documents from the included groups and projects.
2) To provide a high-level comparison of the most prominent frameworks in the industrial and automation domain that are supported by important research groups and company alliances.

Manuscript received October 28, 2019; revised February 10, 2020, April 16, 2020, and May 4, 2020; accepted May 5, 2020. Date of publication May 26, 2020; date of current version March 9, 2021. This work was supported in part by the European Commission, the Productive 4.0 project under Grant 737459 and in part by the Arrowhead Tools project Joint Undertaking under Grant 826452. *(Corresponding author: Cristina Paniagua.)*

The authors are with the Department of Computer Science, Space and Electrical Engineering, Luleå University of Technology, 971 87 Luleå, Sweden (e-mail: cristina.paniagua@ltu.se; jerker.delsing@ltu.se).

Digital Object Identifier 10.1109/JSYST.2020.2993323

3) To illustrate a comparison and analysis of IIoT frameworks in terms of fulfilling industrial IoT requirements, such as interoperability, security, adaptability, and standardization as well as community and industrial support.

This article presents a technical comparison of IoT frameworks engaged in the current industrial context that provide SoS solutions to Industry 4.0 issues. This article provides a detailed summary of the functionalities, domains, and technological features of cutting edge frameworks and a brief overview of cloud-based platforms launched by major IoT shareholders. The information presented has been gathered from official repositories and sources. The analysis criteria include key automation and digitization requirements, such as real-time and runtime features, hardware requirements, architectural approaches, entry barriers, industrial support, standardization, interoperability, and security.

The rest of this article is organized as follows. Section II presents an overview of the selected frameworks. Section III summarizes the cloud platforms available in the market. Section IV provides a detailed comparison of the frameworks features and characteristics. Finally, Section V wraps up the discussion and concludes the article.

II. Frameworks

In the following section, emerging frameworks for IIoT are presented. In this context, an IIoT framework is considered a set of principles and characteristics that enable IIoT application. The frameworks are presented in alphabetical order and were not ranked in any way.

The following frameworks have been selected based on the following criteria.

i) The industrial character and focus on the Industry 4.0 objectives.
ii) The provisioning of architectural and technical solutions for industrial contexts beyond individual IoT solutions.
iii) The targeting of SoS applications based on the IoT.
iv) The reputation of the consortia members and support from large projects.
v) Their future potential and emergence.
vi) The marked evolution from the cloud to the edge.

A. *Arrowhead*

The arrowhead framework [13] began as a part of the European project arrowhead (Artemis) and continued its development during the European project Productive 4.0. It will continue with the recently granted European project arrowhead tools. The arrowhead framework is a framework based on a service-oriented architecture (SOA) [14]. Its aim is to provide automation capabilities, such as scalability, security, real-time control, and engineering simplicity while also enabling IIoT and device interoperability at a service level. It supports SOA-based design principles, such as standardized service contracts, late binding, loose coupling, service abstraction, and autonomy.

The arrowhead framework is composed of local clouds, devices, systems, and services. A local cloud [15] is a key concept for designating the communication and computational environment capable of providing the core services for the development of automation tasks in a protected manner, a simple example is shown in Fig. 1. A local cloud communicates with other compliant local clouds through the gatekeeper. The minimal local cloud contains three mandatory core systems (service registry, authorization, and orchestration) and at least one application system. The core systems provide automation requirements: application service registration, service discovery, authorization, and orchestration. Service, system, and device registries are in charge of the storage and cataloging of the arrowhead entities information. There are more core systems that extend and provide more functionalities, including Gatekeeper, DataManager, QoS, EventHandler, and Configuration.

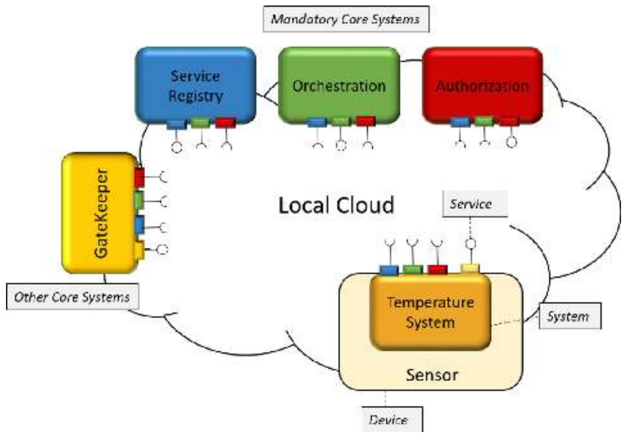

Fig. 1. Example of a simple arrowhead local cloud.

In order to facilitate the implementation of arrowhead compliant systems, a documentation structure [16] and design patterns are defined. The documentation is divided into three levels: system of systems (SoSD, SoSDD), system (SysD, SysDD), and service level (SD, IDD, SP, CP).

The main goal of the arrowhead framework is to achieve interoperability between heterogeneous systems using existing protocols, standards, and handle legacy systems. Application systems implement the details, whereas arrowhead provides a framework for governing and interconnecting services.

Arrowhead has been applied in numerous IoT automation scenarios [14], such as the efficient deployment of a large number of IoT sensors, programmable logic controller (PLC) device monitoring, replacement devices, energy optimization, and maintenance.

B. *Automotive Open System Architecture (AUTOSAR)*

AUTOSAR [17] is a layered software framework for intelligent mobility, supported by AUTOSAR, a worldwide partnership of automotive, electronics, and software industrial companies. AUTOSAR provides standards for electronic control units (ECU), consequently, the specifications of this framework differed from the more high-level oriented frameworks. The application scope of AUTOSAR is automotive ECUs, with strong hardware interaction, connected to vehicle networks (CAN,

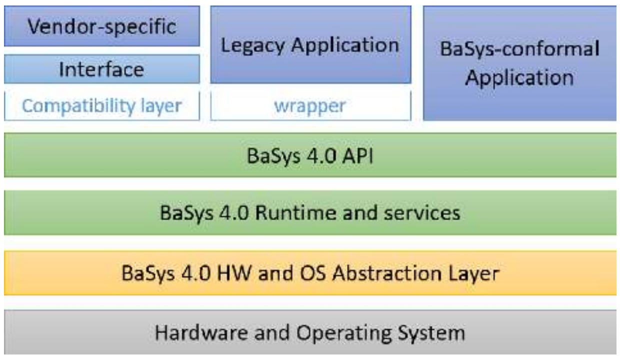
Vendor-specific
Interface
Compatibility layer
Legacy Application
wrapper
BaSys-conformal Application
BaSys 4.0 API
BaSys 4.0 Runtime and services
BaSys 4.0 HW and OS Abstraction Layer
Hardware and Operating System

Fig. 2. BaSys 4.0 functional block diagram.

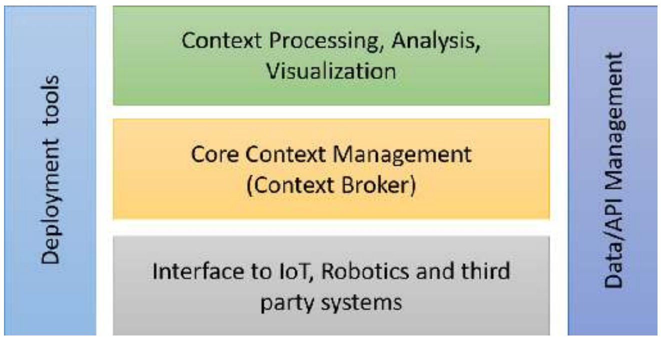
Deployment tools
Context Processing, Analysis, Visualization
Core Context Management (Context Broker)
Interface to IoT, Robotics and third party systems
Data/API Management

Fig. 3. FIWARE principal components.

LIN, Ethernet), and running in resource-constrained devices (microcontrollers) with real-time features.

AUTOSAR provides two different types of platforms: classic and adaptive. The classic platform architecture is formed by three layers: application which is hardware independent; runtime environment (RTE), which is the interface for applications; and basic software, which is divided into services, ECU abstraction, and microcontroller abstraction. The adaptive application provides the AUTOSAR runtime for adaptive applications, and SOA-based restful applications (HTTP/JSON). The adaptive platform persistency is responsible for all aspects which regard the storage and retrieval of data. The last release for both types of platforms was launched in November 2019 [18].

AUTOSAR application domains are vehicle manufacturers, suppliers, service providers and companies from the automotive electronics, semiconductor, and software industries [17].

C. BaSys

Basic system Industry 4.0 [19], commonly called BaSys 4.0, is a research project whose aim is the development of a basic system for production facilities. The first results of the project produced the BaSys 4.0 specification, which later evolved in the open source platform called Eclipse BaSyx [20] and is the result of this collaborative project launched by the German Federal Ministry of Education and Research (BMBF) in 2016 with the Eclipse foundation. A key challenge of Industry 4.0 is to address the efficient variability of production processes.

Eclipse BaSyx has been developed as a reference technology platform for next generation automation, which enables participation in the fourth industrial revolution to all types of industries, research institutes, and academia. It was created in order to address issues such as changeable production to enable individualized products, the exploitation of big data and the collaboration between heterogeneous devices and systems.

BaSys is designed based on three central pillars: process planning, the use of digital twins, and the creation of a standardized RTE. Its functional blocks are shown in Fig. 2.

BaSyx components are divided into four layers [21].

1) Field level. Automation devices, sensors and actuators form the field level; a specific BaSys conforming interface is not required.
2) Device level. Automation devices with a BaSys 4.0 compliant interface are part of the device level, including bridging BaSys 4.0 compliant devices to the field devices without an interface.
3) Middleware level. The BaSyx middleware includes the virtual automation bus (VAB), registry and discovery services, protocol gateways, and asset administration shell. The VAB implements end-to-end communication for Industry 4.0. VAB includes five primitives to map physical networks and VAB gateways to achieve internetwork communication.
4) Plant level. This consists of high-level components to manage and optimize production.

Its architecture, therefore, consists of the VAB and six closely coupled component types. The interfaces are predefined and depend on the component type.

BaSys provides components to implement Industry 4.0 applications, such as end-to-end digitization of production (shopfloor), connectivity between the shopfloor and IT, changeable production processes, big data analysis of production processes, and predictive maintenance [20].

D. FIWARE

FIWARE [22] is an open-source platform that defines a universal set of standards for context management in order to support the development of smart solutions in multiple scenarios, such as smart cities, industry, or smart energy grids. It is designed and managed by the FIWARE community.

The main distinctive feature of FIWARE is the management of contextual information. The FIWARE principal components are the interface with IoT third-party systems, the context broker, the context data/API management block and the processing, analysis, and visualization block. The context broker in conjunction with external components (Cygnus) allows the management of data, storage and analysis, and the storage of historic data. FIWARE also provides deployments tools (see Fig. 3). The gathering and management of contextual information as well as the processing of that information and its communication to external actors is the key proposal of FIWARE.

The FIWARE context broker is the core component in charge of this task, enabling systems to perform updates and access the context state. Interactions between the additional platform components and the context broker are performed by the open standard FIWARE NGSI (next generation service interface) restful API [23]. One remarkable feature of FIWARE is its flexibility. It is possible to use FIWARE components with other third-party components to design a hybrid platform. The only requisite is the use of the core system FIWARE context broker.

The FIWARE NGSI API proposes three elements: first, a data model for context information based on the notion of context entities, second, a context data interface, and third, a context availability interface that defines how to collect context information. At a technical level, FIWARE draws on the advanced communication middleware GE, providing efficient, flexible, scalable, and secure communication.

FIWARE's main application includes the automation of processes across the entire value chain. Green Route (smart cities), Wilma (smart plant), and Cloudino (IoT platform) are some examples of applications developed using this framework by INFOTEC [24]

E. Industrial Data Space (IDS)

IDS [25] is a reference architecture model sponsored by the OPC foundation and the IDS association that targets all types of enterprises. Its aim is to automate and network production and logistics by promoting system-wide interoperability. The reference architecture model is based on three major core concepts (security, certification, and governance) that are implemented across a five-layer structure (system, information, process, functional, and business). IDS provides a model that may be implemented using other frameworks. For example, in [26], Alonso *et al.* presented an IDS implementation using FIWARE.

F. Open Connectivity Foundation (OCF) and IoTivity

The OCF [27] is dedicated to supporting secure interoperability for industries, business, and consumers by providing a standard communication platform, a bridging specification, an open source implementation, and a certification program. Their aim is to enable device communication regardless of their characteristics, such as operating system, service provider, transport technologies, or domains.

The OCF proposes a framework with a homonymous name, OCF [28], which is based on the resource oriented REST architectural style. The goal of the OCF framework is the integration of devices into networks in order to address interoperability issues. In order to achieve interoperability between OCF-compliant products and current IoT devices and legacy systems, the framework provides specifications, code, and a certification program. One of the main characteristics is the definition of several resources that provide functionality to the framework. The main functional blocks are shown in Fig. 4.

The proposed specification standards promote a set of interoperability guidelines as well as a certification program for devices involved in the IoT, becoming one of the most important industrial connectivity standards organizations for IoT.

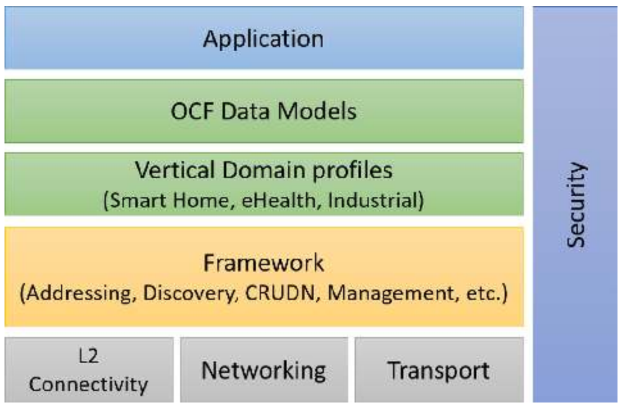

Fig. 4. Principal functional blocks are the Networking layer (interchange functionalities), Framework (core functionalities), and the Vertical domain profile (specific functionalities depending on the market segment).

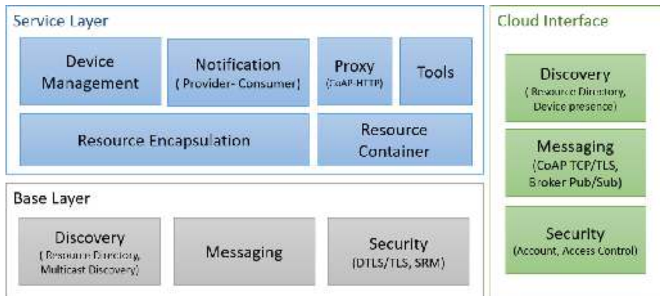
Service Layer
Device Management
Notification (Provider- Consumer)
Proxy (CoAP-HTTP)
Tools
Resource Encapsulation
Resource Container
Base Layer
Discovery (Resource Directory, Multicast Discovery)
Messaging
Security (DTLS/TLS, SRM)
Cloud Interface
Discovery (Resource Directory, Device presence)
Messaging (CoAP TCP/TLS, Broker Pub/Sub)
Security (Account, Access Control)

Fig. 5. IoTivity functional block diagram.

IoTivity [29] is an open-source software framework sponsored by OCF. IoTivity enables seamless device-to-device connectivity to address emerging IoT requisites. This framework was initially developed by the open interconnect consortium to target IoT in smart homes. The last versions are already targeting broader IoT scenarios.

There are two available implementations, IoTivity and IoTivity lite. The key functional blocks that form the framework are the base layer, the service layer, and the cloud interface. The IoTvitiy framework has security features (DTLS/TLS), IPv4 and IPv6 support, server/client and publish/subscriber architectural styles support, and its implementation is based on CoAP and extended to HTTP. IoTivity data management supports the collection, storage, and analysis of data. The functional blocks are shown in Fig. 5.

G. Open Mobile Alliance (OMA) SpecWorks-LWM2M

OMA SpecWorks [30] is a novel type of standards development organization that joins together the OMA and the IPSO alliance. Their mission is to develop technical documents, such as specifications, smart objects, and white papers to enable interoperability across networks and IoTs.

Lightweight M2M (LwM2M) is a device management standard developed by OMA that targets machine-to-machine (M2M) communications and sensor networks. With LwM2M, OMA SpecWorks seeks to provide a common standard for managing low-power devices to support IoTs in a variety of

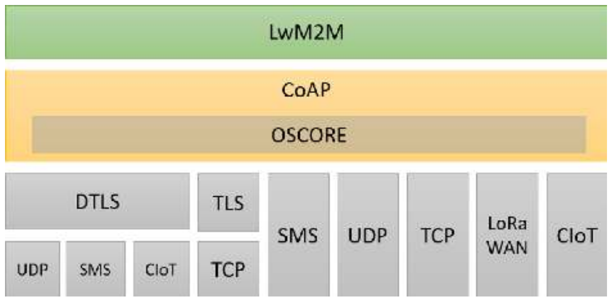

Fig. 6. LwM2M framework stack.

networks. It is important to distinguish LwM2M from its predecessor the OMA framework, which has different modules and features.

LwM2M is designed to follow modern REST architectural design, and it is built on CoAP following a bidirectional client/server style. LwM2M utilizes IETF standards to enhance security (DTLS and OCORE end-to-end). LwM2M defines an extensible data model. Its specification registers a set of objects and resources whereby clients contain objects that define resources, the objects are registered into the LwM2M registry. These objects can be instantiated by either a remoter server or the client itself. Once instantiated, the object can be used via the defined interfaces.

This framework supports UDP, TCP, and SMS as well as several encoding formats, such as TLV, JSON, plain text, and binary formats (see Fig. 6).

Vertical domains where LwM2M has been implemented include automotive, agriculture, smart cities, smart metering, and e-health. In addition, the Coiote LwM2M server has been successfully implemented in the enterprise and telecom sectors [31].

H. Potential Frameworks and Standards

New projects and technologies rapidly evolve. Projects present new points of view that may evolve in new frameworks in the future. An example of this emergence project is the project promoted by the XMPP IoT interfaces working group [32]. IEEE XMPP interfaces for the IoT is an open source project that presents a draft IEEE standard proposal, which covers different communication patterns, data models and security features for sensor and actuator connection and communication based on the standard IEEE 14.51.

III. Cloud IoT Platforms

Unlike the frameworks that are in the early stages of development, cloud IoT platforms have been growing during the last decade and have consolidated in the market during the last years. The most popular IoT platforms benefit from cloud computing. IoT platforms based on the public-cloud approach are widely available in the market. This article is focused on industrial frameworks; however, in this section a brief overview of popular cloud IoT platforms of the most reputable vendors is presented.

1) AWS IoT [33]: AWS (Amazon web services) is a cloud platform that aims at an easy and secure connection of smart devices to the AWS cloud. AWS IoT facilitates the use of other Amazon services. AWS IoT consists of three sets of components: device software (Amazon FreeRTOS and AWS IoT Greengrass), control services (AWS IoT core, device management, device defender, things graph), and data services (AWS IoT analytics, SiteWise, events).

2) Azure IoT suit [34]: Azure is the Microsoft IoT platform that is formed by a set of services that enables interaction with IoT devices, as well as, processing data via some operations, and finally their visualization. Azure supports a broad variety of devices, operating systems, and programming languages.

3) Google cloud IoT [35]: Google released its IoT platform which provides collection, process, analysis, and visualization of IoT data in real time. The main component, the cloud IoT core, allows distributed devices to securely connect and manage data. It is constituted by two main components: the device manager and the protocol bridge.

4) ThingWorx [36]: ThingWorx is a platform designed for industrial IoT. Its aim is to deliver tools and technologies to develop and deploy applications and augmented reality experiences. ThingWorx is widely used due to the inclusion of specific functionality for industrial scenarios, including scalability, connectivity, and security. In ThingWorx, the connection between the edge devices and ThingWorkx servers is provided over WebSockets-based Edge MicroServer using the ThingWorx communication protocol.

These are only some examples of IoT platforms. Other renowned platforms include Bosch IoT Suite [37], IBM Watson IoT [38], Cisco IoT cloud connect [39], Oracle IoT [40], Salesforce IoT, Kura from Eclipse [41], and SmartThings from Samsung.

IV. Discussion

An introduction and high-level overview of the key features of the selected IIoT frameworks have been described. In the following section, the frameworks are discussed using some of the most relevant IIoT and automation requirements as evaluation criteria [42]. The requisites include functional principles, entry barriers, interoperability, and security. This comparison is based on available online specifications, not on actual framework testing. IDS is considered to be an architectural model and is excluded from the technical analysis.

A. Functional Principles and Features

Requirements in industry are more restrictive than in other domains. Real-time specifications, runtime features, hardware requirements, and the distribution of their systems are key characteristics must be evaluated. Many industrial processes require real-time constraints [43]. Moreover, Industry 4.0 benefits from runtime information, which opens new possibilities and provides flexibility in production [44].

1) Real-Time Features: Real-time specifications concern the capability of the middleware to support and manage applications with real-time constraints. Real-time ability is an important aspect in many IoT applications, especially in industry and health care scenarios [45].

Industrial IoT frameworks should provide real-time services to ensure response times and overlying applications. If the operations are not performed logically and finished in a bounded time frame, the operation may be useless. Despite this most of the frameworks presented do not have real-time features. This is not the case for the arrowhead framework, which provides real-time control features, the QoS manager core system is in charge of verifying the feasibility of the request and ensuring real-time constraints [46]. AUTOSAR also provides real-time features [47], to work and analyze timing constraints.

Although they use REST architectural styles that provide fast response times, frameworks such as FIWARE, IoTivity, LwM2M, and OCF do not have support for real-time behavior, since they do not provide inherent functionality to predict response times. BaSyx [48] considers that real time is no longer essential at the process level, and instead, PLCs are in charge of the real-time production steps.

Nevertheless, within the IoTivity project, it is possible to find other complementary projects that accomplish a specific objective, including the IoTivity-constrained framework [49], which is a new lightweight implementation of the OCF standards. The main difference associated with the IoTivity project is its design targeted to resource-constrained devices that run real-time operating systems. The IoTivity-constrained framework currently runs on Linux, Zephyr, RIOT OS, Contiki, MyNewt, and Tizen.

2) Runtime Features: Runtime features refer to aspects and operations that are performed in or from a running system. Runtime features open a variety of possibilities, such as dynamic orchestration, IoT device management and monitoring, runtime verification and modification, among others. These functionalities are highly valued in industrial environments.

The arrowhead framework has runtime functionalities, and the orchestration system is capable of computing new orchestration patterns in runtime and providing dynamic orchestration and authorization [46]. The EventHandler system and the DataManager system also permit verification and monitoring during runtime. In addition, there are core systems that are under development, such as the Translator, WorkflowManager, and WorkflowExecutor [50] that provide runtime features to increase the interoperability and dynamic automation, respectively.

AUTOSAR [51] has a RTE layer, which is one of its key components. The RTE is in charge of the communication between software components and their scheduling. In addition, it dynamically links services and clients and includes an event manager and other services to trace errors in runtime. In a similar manner, the Eclipse BaSyx project [52] incorporates from other versions of BaSys a RTE based on the virtual function bus. The runtime and service layer can run hardware independent code, providing a set of services to manage the functional components.

FIWARE presents some runtime features in its specifications [53]. These features include runtime monitoring attributes of the object store, possibility to add a security runtime component to dynamically verify compliance with policies, runtime service selection and late binding based on some specification of constraints. LwM2M does not present runtime features, unlike the predecessor OMA framework, which proposed a web runtime API [54] in 2014. OCF also does not allow generic interoperability at runtime.

3) Centralized/Distributed Approach: Systems distribution is directly related to the flexibility and scalability of the SoS ecosystem. Many frameworks take a centralized approach, where the middleware provides all of the functionalities. AUTOSAR, BaSys, FIWARE, IoTivity, LwM2M, and OCF fall in this category.

The arrowhead framework proposes a new distributed approach, where the different functionalities and core services are distributed into the different core systems [15] instead of having a unique middleware that reduces the scalability. Moreover, not only are the functionalities distributed, in comparison with other frameworks where the middleware is running in the cloud, arrowhead uses different local clouds that modularize and distribute the applications and is capable of communicating in a securely with each other.

4) Hardware Requirements: The IoT and SoS involve a strong connection between the digital and physical world (cyber physical systems). Consequently, hardware constraints and requirements play an important role in the development of the frameworks on factory shopfloors.

The frameworks presented can be divided according to hardware specifications. AUTOSAR and LwM2M have been designed to fit constrained-resource device requirements. AUTOSAR can be adaptable to different 32-b MCUs [55]. LWM2M targets class 1 devices as defined by RFC 7228 (<20 kB RAM) [56], a LwM2M client example specifications could be 10 KB RAM and 100 KB flash [57]. Arrowhead, IoTivity, and OCF can be deployed in a large range of devices. Despite not being optimized for resource-constrained devices as the previous frameworks, they can be used in small-medium boards. OCF provides a light version for small devices [58], including class 2 devices (approximately 50 KB of RAM and 250 KB of flash). Arrowhead is in the process of developing light versions of the core systems and application interfaces in Python and C++ languages to fit the lighter requirements.

However, BaSys and FIWARE require a certain amount of computational power, which makes them less suitable for small devices with resource constraints. FIWARE runs as a web application, the hardware requirements for the platform would be at least 8 GB RAM.

5) Quality of Service: QoS refers to requirements regarding all the aspects of a connection, such as response time, losses, signal-to-noise ratio, or frequency response. Some of the frameworks provide tools for the monitoring of these QoS requirements. The arrowhead framework provides the QoS manager core system [59], AUTOSAR supports QoS for the Ethernet driver used in its adaptive platform, FIWARE presents the QoS and software defined networking block [60] and IoTivity and OCF allows to specify the QoS requirements for requests. On the other hand, BaSys and LwM2M do not provide their own tools but use the ones provided by other underlying protocols to ensure the QoS.

TABLE I
ACCESSIBILITY

	Specification	Code	Tutorials	Examples
Arrowhead	Easy	Easy	Medium	Easy
AUTOSAR	Easy	Difficult	Medium	Medium
Basys	Difficult	–	–	Medium
FIWARE	Easy	Easy	Easy	Easy
IoTivity	Easy	Easy	Medium	Difficult
LwM2M	Medium	Easy	Medium	–
OCF	Easy	–	Difficult	Difficult

B. Entry Barriers

Before comparing the technical characteristics of the frameworks, the entry barriers are analyzed.

We considered all the aspects that prevent a framework from entering the market and being competitive as entry barriers. Barriers to entry can include the need for licenses, industry support, accessibility, the lack of information and specifications, and bad accessibility of the code and resources. Other barriers such as the corporate image, lack of trust in the community, and other aspects that are difficult to analyze are not included in this survey.

Open-source frameworks are important for developers who look for flexibility at the time of choosing libraries, vendor platforms, and interoperability. Many cloud platforms are proprietary; however, open-source technologies are gaining importance in these frameworks. Arrowhead, Eclipse BaSyx, FIWARE, and IoTivity are all open source, and LwM2M has an open-source version of the framework. The frameworks or protocols that are not open source have less information available and usually do not show their technical specification. For example, thread specifications are only available for member companies.

In order to be competitive, frameworks need to provide easy accessibility to their resources and specifications. In Table I, an analysis of the level of accessibility to specifications, codes, tutorials, and examples are presented. The level of accessibility has been modeled into three levels (easy, medium, and difficult). Values are determined based on the facility to find the resources in their official web-pages. *Easy* is when resources were easy to find without previous knowledge of the framework. *Medium* when the resource is accessible but it required some effort and motivation to find it. And finally, *difficult* when it required some previous knowledge to find it. Some of the resources were unavailable at the time of writing.

Complexity in the implementation also represents an entry barrier, due to the high engineering effort and extra costs that any change in the industry carries. The level of complexity is difficult to measure and subjective to each specific scenario; consequently, in place of analyzing the complexity of implementation, the existence or absence of tools that facilitate development has been analyzed. The number and type of tools available increase or decrease the integration complexity of the frameworks in new scenarios. Table II enumerates the tools available for each framework.

1) Industry Support: A factor that can be decisive for the success rate of any framework is the support from industrial and research partners. The prestige of supporting partners as well as their number influence the chances of becoming a reference framework as well as their dissemination and utilization in the industry. Commonly, the frameworks are supported by large consortia and projects (see Table III).

TABLE II
DEVELOPMENT AND SIMULATION TOOLS

Framework	Tools
Arrowhead	Client Skeletons (C++, Java), Sandbox, Docker version, Management tool [61]
AUTOSAR	No available without partnership
Basys	BaSyx Java SDK [62]
FIWARE	FIWARE Lab (Sandbox) [63], Docker version [64]
IoTivity	IoTivity C SDK [65], IoTivity Simulator [66]
LwM2M	Developer ToolKit (C library, Sandbox, LabKit) [67]
OCF	Onboarding Tool Generic Client (OTGC) [68]

TABLE III
FRAMEWORKS ORGANIZATIONS SUPPORTERS

Framework	Consortium	Members
Arrowhead	Arrowhead	88
AUTOSAR	AUTOSAR	116
Basys	Basys 4.0	15
FIWARE	FIWARE Foundation	52
IDS	OPC fundation, IDS Association	610, 89
IoTivity	Open Connectivity Foundation	113
LwM2M	OMA SpecWorks	48
OCF	Open Connectivity Foundation	113

TABLE IV
MESSAGE PATTERNS

	Request-reply	Publish-subscriber
Arrowhead	✓	✓
AUTOSAR	✓	✓
Basys	✓	
FIWARE	✓	✓
IoTivity	✓	✓
LwM2M	✓	
OCF	✓	

C. Interoperability

One of the most difficult challenges of IoT paradigms is the interoperability between the devices. Myriad definitions regarding the term interoperability can be found in the literature [69]–[71]. Nevertheless, despite the variations, interoperability can be defined as the seamless connection and communication of heterogeneous systems and software components. The collaboration of heterogeneous systems with very different characteristics, protocols, and standards is a major bottleneck for the new industrial paradigm, that needs to be solved. In order to compare the interoperability of the frameworks, the enabling technologies used by each framework need to be compared and contrasted.

In Table IV, the frameworks in this survey are categorized by message pattern. The request-reply pattern is the most used in the IoT field, which is based on services and resource architectures. These are commonly implemented by servers and clients. Consequently, all of the analyzed frameworks are based on this architectural pattern. Arrowhead, AUTOSAR, IoTivity, and FIWARE support publish-subscribe message patterns as well.

TABLE V
TRANSPORT PROTOCOLS

	TCP	UDP	DTLS/TLS
Arrowhead	✓	✓	✓
AUTOSAR	✓	✓	✓
Basys	✓		
FIWARE	✓	✓	✓
IoTivity	✓	✓	✓
LwM2M	✓	✓	✓
OCF	✓	✓	✓

TABLE VI
COMMUNICATION PROTOCOLS

	HTTP	CoAP	MQTT	OPC-UA	Other
Arrowhead	✓	✓	✓	✓	
AUTOSAR	✓				
Basys	✓			✓	BaSys native
FIWARE	✓				RTPS
IoTivity	✓	✓			
LwM2M		✓			
OCF	✓	✓			

The comparison at the transport layer (see Table V) shows that all of the frameworks are based on TCP and provide more security features by using DTLS/TLS. However, this is not always the case. For example, BaSys does not support DTLS/TLS and is exclusively based on TCP. In addition, some of the frameworks such as arrowhead, AUTOSAR, FIWARE, OCF, and LwM2M are based on UDP.

One common interoperability mismatch is the difference in the application layer, where several communication protocols can be used. Table VI shows the frameworks and their popular protocols. LwM2M is only focused on low-power devices, and the only protocol that supports is CoAP. The rest of the frameworks support HTTP as the main communication protocol. BaSys supports OPCUA in their component modules (lower layers) and a specific protocol called BaSys native based on TCP. The only framework that supports the four protocols (HTTP, CoAP, MQTT, and OPC-UA) is, at the time of writing, the arrowhead framework.

Finally, the interoperability between third-party systems and legacy systems needs to be analyzed. The arrowhead framework has three levels of adaptability (native implementation, software adapter, and external gateway adapter) to make possible the interoperability between other systems. BaSys provides a three similar adaptability levels of configuration. AUTOSAR provides an adaptive platform for this purpose. FIWARE presents the possibility of using the main block context broker with third systems. However, the flexibility and adaptability in other frameworks (e.g., IoTivity, LwM2M, and OCF) are limited to the use of their own standards. The interoperability between frameworks is also a key element in the industry. For example, the arrowhead framework is in the process of developing adaptors between FIWARE and BaSys frameworks. Another example of the interaction between frameworks is FIWARE, which presents IoT agents for collaboration with LwM2M [72].

D. Security

Security has become a fundamental feature since the communication and interchange of services between different, and possibly unknown, systems are present [73]. In addition, the privacy of the data, especially when referring to personal data or confidential industrial data, is especially important. Hence, security, privacy, and trustworthiness remain important challenges in the IoT field [74]. All the different frameworks have security features and components capable of managing the security of their services and resources.

In the case of arrowhead, the core system AAA implements authentication, authorization, and accounting. It is possible to choose between two levels of security, with or without DTLS/TLS: use of the X.509 certificates and use of tokens for accounting. In addition, access control can be made by authorization rules or next generation access control policies [75]. AUTOSAR bases its security on the AUTOSAR crypto service manager and the secure onboard communication, which is in charge of resource-efficient cryptographic authentication.

FIWARE specifications present [76] the identity management enabler (IdM), which provides authentication, basic authorization, and security assertions as a service. This core security GE uses open protocols such as OAuth [77] and OASIS SAML v2.0. [78]. The IdM manages the authorization in collaboration with the PEP proxy GE and the authorization PDP GE. The PEP intercepts each access request to the resource but relies on the IdM to authenticate the request and on the PDP to authorize it (deny of permit).

The IoTivity stack includes a secure resource manager (SRM) [79], which is formed by three functional blocks and a database. The resource manager manages the security virtual resources (e.g., access control list, credential, provisioning status). The policy engine filters resource requests in order to grant or deny based on the policy, and the persistent storage interface provides storage API. The SRM [80] is configured via an OIC resource with specific properties.

LwM2M requires communications to be authenticated, encrypted, and integrity protected. For this purpose, LwM2M 1.1 specification supports an application layer security protocol called OSCORE. OSCORE (object security for constrained restful environments) protects message exchanges and provides support for proxy operations and end-to-end security, independent of the underlying transport protocols, including UDP, SMS, and TCP, with or without DTLS/TLS [81].

In the OCF security specifications [82], the OCF security enforcement points are established, including the SRM and the session protection in the connectivity abstraction layer (usually DTLS). The access control relies on predefined policies that are stored by a local access control list or an access management service in the form of an access control entry. The supported credential are pair-wise symmetric keys, group symmetric keys, asymmetric authentication keys, and certificates (X.509 format).

E. Summary

Finally in Table VII all the discussed features are presented in a summarized way. The feature *resource accessibility* has been added as a qualitative average of the features explained in Section IV-B entry barriers.

TABLE VII
SUMMARY

Features	Arrowhead	AUTOSAR	BaSyx	FIWARE	IoTivity	LWM2M	OCF
Key principles	SOA, Local Automation Clouds	Runtime, Electronic Control Unit (ECU)	Variability of production processes	Context awareness	Device-to-device communication	M2M, Constrained networks	Resource-Oriented REST, Certification
Real-time	Yes	Yes	No	No	Yes (IoTivity Constrained)	No	No
Run-time	Dynamic orchestration and authorization, monitoring, and dynamic automation	Runtime environment layer (RTE)	Runtime environment	Monitoring, dynamic service selection and verification	No	No	No
Distribution	Distributed	Centralize	Centralize	Centralize	Centralize	Centralize	Centralize
Open Source	Yes	No	Yes	Yes	Yes	Yes	No
Resource accessibility	High	Low	Very low	High	Medium	Medium	Low
Supporters	Arrowhead	AUTOSAR	Basys 4.0	FIWARE Foundation	Open Connectivity Foundation	OMA SpecWorks	Open Connectivity Foundation
Message patterns	Req/Repl, Pub/Sub	Req/Repl, Pub/Sub	Req/Repl	Req/Repl, Pub/Sub	Req/Repl, Pub/Sub	Req/Repl	Req/Repl
Transport protocols	TCP, UDP, DTLS/TLS	TCP, UDP, TLS	TCP	TCP, UDP, DTLS/TLS	TCP, UDP, DTLS/TLS	TCP, UDP, DTLS/TLS, SMS	TCP, UDP, DTLS/TLS, BLE
Communication protocols	HTTP, CoAP, MQTT, OPC-UA	HTTP	HTTP, OPC-UA	HTTP, RTPS	HTTP, CoAP	CoAP	HTTP, CoAP
3rd Party and Legacy systems adaptability	Yes	Yes	Yes	Yes	No	No	No
Required device size	Small to large	Resource-constrained	Large	Large	Small to large	Resource-constrained	Small to large
Security Manager	Authentication, Authorization and Accounting Core System	Crypto Service Manager, Secure Onboard Communication	–	Identity Manager Enabler	Secure Resource Manager	OSCORE	Secure Resource Manager
Standardization	Use of existing standards	AUTOSAR standards	Use of existing standards	FIWARE NGSI	OCF standards	Use of existing standards	OCF standards

The advantages and limitations regard the use of the presented frameworks depend on the specific application and scenario. Table VII wants to present an objective comparison to allow the selection of the most appropriate framework for each domain, pointing out the features that each framework offers.

V. CONCLUSION

Over the last decade, technology has evolved very rapidly, as well as, the functionalities and characteristics of frameworks are developing quickly along with the protocols and standards from which they are developed. Even the alliances and groups are changing over the years, grouping platforms under the same name and specifications.

This survey has presented a number of available frameworks for IIoT applications. The studied frameworks have their own approach to solving challenges; nevertheless, they all seek to aid interoperability in heterogeneous environments.

A comparative analysis of the frameworks was presented based on industrial requirements. The analysis criteria included real-time and runtime features, architectural approach, entry barriers, industrial support, standardization, interoperability, and security. The analysis highlights the general effort to solve problems such as security and interoperability on the part of the large consortia and research projects that support the development of the presented frameworks. However, there is a clear lack of industrial and automation requirements in some of the frameworks. Additional work is needed to consolidate the different standards and frameworks.

The dynamism of the industry and the evolution of the SoS requirements, frameworks, and projects make difficult to foresee coming trends and developments. Future capabilities will be determined by the identification of new industrial requirements, such capabilities may include autonomous contract execution and financial transactions, autonomous self-mitigation, autonomous information enhancements, and decision support. The monitoring of the frameworks evolution and the analysis of new industrial requirements that lead to game-changing technologies are part of future work.

REFERENCES

[1] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial Internet of Things (IIoT): An analysis framework," *Comput. Ind.*, vol. 101, pp. 1–12, 2018.

[2] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, opportunities, and directions," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4724–4734, Nov. 2018.

[3] P. Sethi and S. R. Sarangi, "Internet of Things: Architectures, protocols, and applications," *J. Elect. Comput. Eng.*, vol. 2017, 2017, Art. no. 9324035.

[4] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of Things security: A survey," *J. Netw. Comput. Appl.*, vol. 88, pp. 10–28, 2017.

[5] P. P. Ray, "A survey on Internet of Things architectures," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 30, no. 3, pp. 291–319, 2018.

[6] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, "Internet of Things security and forensics: Challenges and opportunities," pp. 544–546, 2018.

[7] E. Bertino and N. Islam, "Botnets and Internet of Things security," *Computer*, vol. 50, no. 2, pp. 76–79, 2017.

[8] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and Internet of Things: A survey," *Future Gener. Comput. Syst.*, vol. 56, pp. 684–700, 2016.

[9] B. L. R. Stojkoska and K. V. Trivodaliev, "A review of Internet of Things for smart home: Challenges and solutions," *J. Cleaner Prod.*, vol. 140, pp. 1454–1464, 2017.

[10] S. Li, L. Da Xu, and S. Zhao, "5G Internet of Things: A survey," *J. Ind. Inf. Integr.*, vol. 10, pp. 1–9, 2018.

[11] H. Derhamy, J. Eliasson, J. Delsing, and P. Priller, "A survey of commercial frameworks for the Internet of Things," in *Proc. IEEE 20th Conf. Emerg. Technol. Factory Autom.*, 2015, pp. 1–8.

[12] M. Ammar, G. Russello, and B. Crispo, "Internet of Things: A survey on the security of IoT frameworks," *J. Inf. Secur. Appl.*, vol. 38, pp. 8–27, 2018.
[13] Arrowhead, "Arrowhead official website," 2019. [Online]. Available: https://www.arrowhead.eu/, Accessed: Apr. 10, 2019.
[14] J. Delsing, *IoT Automation: Arrowhead Framework*. Boca Raton, FL, USA: CRC Press, 2017.
[15] J. Delsing, J. Eliasson, J. van Deventer, H. Derhamy, and P. Varga, "Enabling IoT automation using local clouds," in *Proc. IEEE 3rd World Forum Internet Things*, 2016, pp. 502–507.
[16] F. Blomstedt *et al.*, "The arrowhead approach for SOA application development and documentation," in *Proc. 40th Annu. Conf. IEEE Ind. Electron. Soc.*, 2014, pp. 2631–2637.
[17] AUTOSAR, "Autosar official website," 2020. [Online]. Available: https://www.autosar.org, Accessed: Jan. 15, 2020.
[18] AUTOSAR, "Requirements on core test," 2019. [Online]. Available: www.autosar.org/fileadmin/user_upload/standards/classic/19-11/ AUTOSAR_SRS_CoreTest.pdf, Accessed: Jan. 15, 2020
[19] BaSys 4.0, "Basic system industry 4.0." 2018. [Online]. Available: https://www.basys40.de/, Accessed: Apr. 11, 2019.
[20] Eclipse Foundation, "Basyx project documentation," 2018. [Online]. Available: https://wiki.eclipse.org/BaSyx, Accessed: Apr. 11, 2019.
[21] Eclipse Foundation, "About BaSyx," 2019. [Online]. Available: https://www.eclipse.org/basyx/about/, Accessed: Apr. 11, 2019.
[22] FIWARE, "Fiware: The open source platform for our smart digital future," 2019. [Online]. Available: https://www.fiware.org/, Accessed: Apr. 10, 2019.
[23] FIWARE, "Fiware: Open API specifications," 2019. [Online]. Available: http://fiware.github.io/specifications/, Accessed: Apr. 10, 2019.
[24] INFOTEC, "Fiware: Examples of fiware applications," 2016. [Online]. Available: http://www.cudi.edu.mx/presentaciones/FIWARE/5_FIWARE _examples_of_application.pdf, Accessed: Jan. 15, 2020.
[25] Industrial Data Spaces Association, "IDS reference architecture model," 2018. [Online]. Available: https://www.internationaldataspaces.org/the-principles/, Accessed: Apr. 9, 2019.
[26] Á. Alonso, A. Pozo, J. Cantera, F. de la Vega, and J. Hierro, "Industrial data space architecture implementation using fiware," *Sensors*, vol. 18, no. 7, 2018, Art. no. 2226.
[27] Open Connectivity Foundation, "Open connectivity foundation," 2019. [Online]. Available: https://openconnectivity.org/, Accessed: Apr. 10 2019.
[28] Open Connectivity Foundation, "OCF specifications," 2019. [Online]. Available: https://openconnectivity.org/specs, Accessed: Apr. 10, 2019.
[29] Linux Foundation, "Iotivity," 2019. [Online]. Available: https://iotivity.org/, Accessed: Apr. 10, 2019.
[30] OMA SpecWorks, "Lightweight m2m (lwm2m)," 2019. [Online]. Available: https://www.omaspecworks.org/what-is-oma-specworks/iot/lightweight-m2m- lwm2m/, Accessed: Apr. 10, 2019
[31] OMA SpecWorks, "Avsystem on the current status of LwM2M adoption," 2018. [Online]. Available: https://www.omaspecworks.org/avsystem-on-the-current-status-of-lwm2m-ad option/, Accessed: Jan. 15, 2020.
[32] IEEE Standards Association, "XMPP interface descriptions for the Internet of Things," 2019. [Online]. Available: https://gitlab.com/IEEE-SA/XMPPI/IoT, Accessed: Aug. 4, 2019.
[33] Amazon, "AWS IoT," 2019. [Online]. Available: https://aws.amazon.com/iot, Accessed: Apr. 10, 2019.
[34] Microsoft, "Microsoft azure," 2019. [Online]. Available: https://azure.microsoft.com/en-us/, Accessed: Apr. 10, 2019.
[35] Google, "Google cloud IoT core," 2019. [Online]. Available: https://cloud.google.com/iot-core/, Accessed: Apr. 10, 2019.
[36] PTC, "Thingworx platform," 2019. [Online]. Available: https://www.ptc.com/en/resources/iot/product-brief/thingworx-platform, Accessed: Apr. 10, 2019.
[37] BOSH, "Bosh IoT suite," 2019. [Online]. Available: https://www.bosch-iot-suite.com/, Accessed: Apr. 16, 2019.
[38] IBM, "IBM Watson," 2019. [Online]. Available: https://www.ibm.com/watson, Accessed: Apr. 16, 2019.
[39] Cisco, "Cisco IoT cloud connect," 2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/service-provider/iot-cloud-conn ect/index.html, Accessed: Apr. 16, 2019.
[40] ORACLE, "Oracle Internet of Things," 2019. [Online]. Available: https://www.oracle.com/internet-of-things/, Accessed: Apr. 16, 2019.
[41] Eclipse, "Kura," 2019. [Online]. Available: https://www.eclipse.org/kura/, Accessed: Apr. 16, 2019.
[42] J. Delsing, "Local cloud Internet of Things automation: Technology and business model features of distributed Internet of Things automation solutions," *IEEE Ind. Electron. Mag.*, vol. 11, no. 4, pp. 8–21, Dec. 2017.
[43] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Bus. Inf. Syst. Eng.*, vol. 6, no. 4, pp. 239–242, 2014.
[44] S. Weyer, T. Meyer, M. Ohmer, D. Gorecky, and D. Zühlke, "Future modeling and simulation of CPS-based factories: An example from the automotive industry," *IFAC-PapersOnLine*, vol. 49, no. 31, pp. 97–102, 2016.
[45] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 3, no. 1, pp. 70–95, Feb. 2016.
[46] J. Delsing, "Arrowhead framework core systems and services," in *IoT Automation: Arrowhead Framework*. Boca Raton, FL, USA: CRC Press, 2017.
[47] AUTOSAR, "Requirements on operating system," 2019. [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/3-0/AUT OSAR_SRS_OS.pdf, Accessed: Apr. 16, 2019.
[48] Eclipse Foundation, "Basyx features," 2018. [Online]. Available: https://wiki.eclipse.org/BaSyx.Features, Accessed: Apr. 12, 2019.
[49] IoTitivy, "Iotivity projects and functions," 2017. [Online]. Available: https://wiki.iotivity.org/projects_and_functions, Accessed: Apr. 10, 2019.
[50] J. G. Represa and J. Delsing, "Autonomous production workstation operation, reconfiguration and synchronization," in *Proc. 25th Int. Conf. Prod. Res.*, 2019, pp. 226–234.
[51] AUTOSAR, "AUTOSAR standards specifications," 2019. [Online]. Available: https://www.autosar.org/standards/, Accessed Apr. 15, 2019.
[52] Eclipse, "Eclipse Basyx," 2019. [Online]. Available: https://projects.eclipse.org/proposals/eclipse-basyx, Accessed: Apr. 10, 2019.
[53] FIWARE, "Fiware apps features," 2013. [Online]. Available: https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Category:AppsFeature, Accessed: Apr. 10, 2019.
[54] OMA, "Enabler release definition for web runtime API (WRAPI)," 2014. [Online]. Available: http://openmobilealliance.org/release/WRAPI, Accessed: Apr. 10, 2019.
[55] AUTOSAR, "Requirements on memory hardware abstraction layer," 2020. [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/19-11/AUTOSAR_SRS_MemoryHWAbstractionLayer.pdf, Accessed: Jan. 15, 2020.
[56] C. Bormann, M. Ersue, and A. Keranen, "Terminology for constrained-node networks," in *Proc. Internet Eng. Task Force*, Fremont, CA, USA, 2014, pp. 2070–1721.
[57] Open Mobile Aliance, "LwM2M v1.1," 2019. [Online]. Available: http://www.openmobilealliance.org/release/LightweightM2M/Lightweight _Machine_to_Machine-v1_1-OMASpecworks.pdf, Accessed: Jan. 15, 2020.
[58] Open Connectivity, "OCF specification overview core technology specification," OCF 2.0.1 release, 2019. [Online]. Available: https://openconnectivity.org/wp-content/uploads/2019/03/2.-OCF-Architec ture-Introduction.pdf, Accessed: Jan. 15, 2020.
[59] M. Albano, P. M. Barbosa, J. Silva, R. Duarte, L. L. Ferreira, and J. Delsing, "Quality of service on the arrowhead framework," in *Proc. IEEE 13th Int. Workshop Factory Commun. Syst.*, 2017, pp. 1–8.
[60] FIWARE, "Fiware.architecturedescription.i2nd.middleware," 2019. [Online]. Available: http://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FIWARE.ArchitectureDescription.I2ND.Middleware, Accessed: Mar. 30, 2020.
[61] Arrowhead Consortia, "Arrowhead consortia github," 2020. [Online]. Available: https://github.com/arrowhead-f, Accessed: Jan. 16, 2020.
[62] ECLIPSE Foundation, "How to build the Basyx java SDK," 2019. [Online]. Available: https://wiki.eclipse.org/BaSyx_/_Download_/_Java_Setup, Accessed: Jan. 16, 2020.
[63] FIWARE, "Fiware lab account," 2019. [Online]. Available: https://www.fiware.org/developers/fiware-lab/, Accessed: Jan. 16, 2020.
[64] FIWARE, "Fiware step-by-step," 2019. [Online]. Available: https://fiware-tutorials.readthedocs.io/en/latest/, Accessed: Jan. 16, 2020.
[65] IoTivity, "Iotivity c SDK," 2017. [Online]. Available: https://api-docs.iotivity.org/latest-c/annotated.html, Accessed: Jan. 16, 2020.
[66] IoTivity, "Iotivity tools," 2017. [Online]. Available: https://wiki.iotivity.org/iotivity_tool_guide, Accessed: Jan. 16, 2020.
[67] Open Mobile Alliance, "OMA LwM2M developer toolkit," 2020. [Online]. Available: github.com/OpenMobileAlliance/OMA_LwM2M_for_Developers/wiki, Accessed: Jan. 16, 2020.

[68] Open Connectivity Foundation, "Onboarding tool generic client (OTGC)," 2019. [Online]. Available: https://github.com/openconnectivityfoundation/development-support/tree/master/otgc, Accessed: Jan. 16, 2020.
[69] P. Wegner, "Interoperability," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 285–287, 1996.
[70] T. C. Ford, J. M. Colombi, S. R. Graham, and D. R. Jacques, "Survey on interoperability measurement," Air Force Inst. Technol., Wright-Patterson AFB, OH, USA, Tech. Rep. ADA481314, 2007.
[71] D. Gürdür and F. Asplund, "A systematic review to merge discourses: Interoperability, integration and cyber-physical systems," *J. Ind. Inf. Integr.*, vol. 9, pp. 14–23, 2018.
[72] FIWARE, "Fiware lab account," 2019. [Online]. Available: https://www.fiware.org/developers/catalogue/, Accessed: Jan. 16, 2020.
[73] P. Bull, R. Austin, E. Popov, M. Sharma, and R. Watson, "Flow based security for IoT devices using an SDN gateway," in *Proc. IEEE 4th Int. Conf. Future Internet Things Cloud*, 2016, pp. 157–163.
[74] S. Vashi, J. Ram, J. Modi, S. Verma, and C. Prakash, "Internet of Things (IoT): A vision, architectural elements, and security issues," in *Proc. Int. Conf. I-SMAC (IoT Social, Mobile, Analytics Cloud)*, 2017, pp. 492–496.
[75] K. K. Kolluru, C. Paniagua, J. van Deventer, J. Eliasson, J. Delsing, and R. J. DeLong, "An AAA solution for securing industrial IoT devices using next generation access control," in *Proc. IEEE Ind. Cyber-Phys. Syst.*, 2018, pp. 737–742.
[76] FIWARE, "Fiware openspecification. Security identity management," 2016. [Online]. Available: https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FIWARE.OpenSpecification.Security.IdentityManagement, Accessed: Apr. 9, 2019.
[77] Okta, "Oauth 2.0," 2018. [Online]. Available: https://oauth.net/, Accessed: Apr. 9, 2019.
[78] OASIS, "Oasis security services (SAML) TC," 2017. [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security#sa mlv20, Accessed: Apr. 9, 2019.
[79] Iotivity, "Iotivity security architecture overview," 2015. [Online]. Available: https://wiki.iotivity.org/iotivity_security_architecture_overview, Accessed: Apr. 9, 2019.
[80] Iotivity, "Iotivity secure resource manager design," 2017. [Online]. Available: https://wiki.iotivity.org/secure_resource_manager_functional_blocks, Accessed Apr. 9, 2019.
[81] OMA SpecWorks, "White paper lightweight M2M 1.1," 2018. [Online]. Available: https://www.omaspecworks.org/white-paper-lightweight-m2m-1-1/, Accessed Apr. 9, 2019.
[82] OMA SpecWorks, "OCF security specification," 2019. [Online]. Available: https://openconnectivity.org/specs/OCF_Security_Specification.pdf, Accessed: Apr. 9, 2019.

Cristina Paniagua received the B.Sc. degree in electronic and automation engineering and the M.Sc. degree in industrial engineering from Zaragoza University, Zaragoza, Spain, in 2015 and 2017, respectively. She is currently working toward the Ph.D. degree in industrial electronics with Luleå University of Technology, Luleå, Sweden.

Her current research interests include the Industrial Internet of Things focus on resource-constrained devices and the collaborative performance in heterogeneous service-oriented architecture (SOA).

Jerker Delsing received the M.Sc. degree in engineering physics from the Lund Institute of Technology, Lund, Sweden, in 1982, and the Ph.D. degree in electrical measurement from Lund University, Lund, in 1988.

In 1994, he was promoted to Associate Professor in Heat and Power Engineering with Lund University. Early 1995, he was appointed Full Professor in Industrial Electronics with the Lulea University of Technology, where he is currently the Scientific Head of EISLAB. His present research profile can be entitled IoT and SoS automation, with applications to automation in large and complex industry and society systems.

Prof. Delsing and the EISLAB group have been a partner and coordinator of several large to very large EU projects in the field, e.g., socrades, IMC-AESOP, arrowhead, FAR-EDGE, productive4.0, and arrowhead tools.