Local Cloud Internet of Things Automation

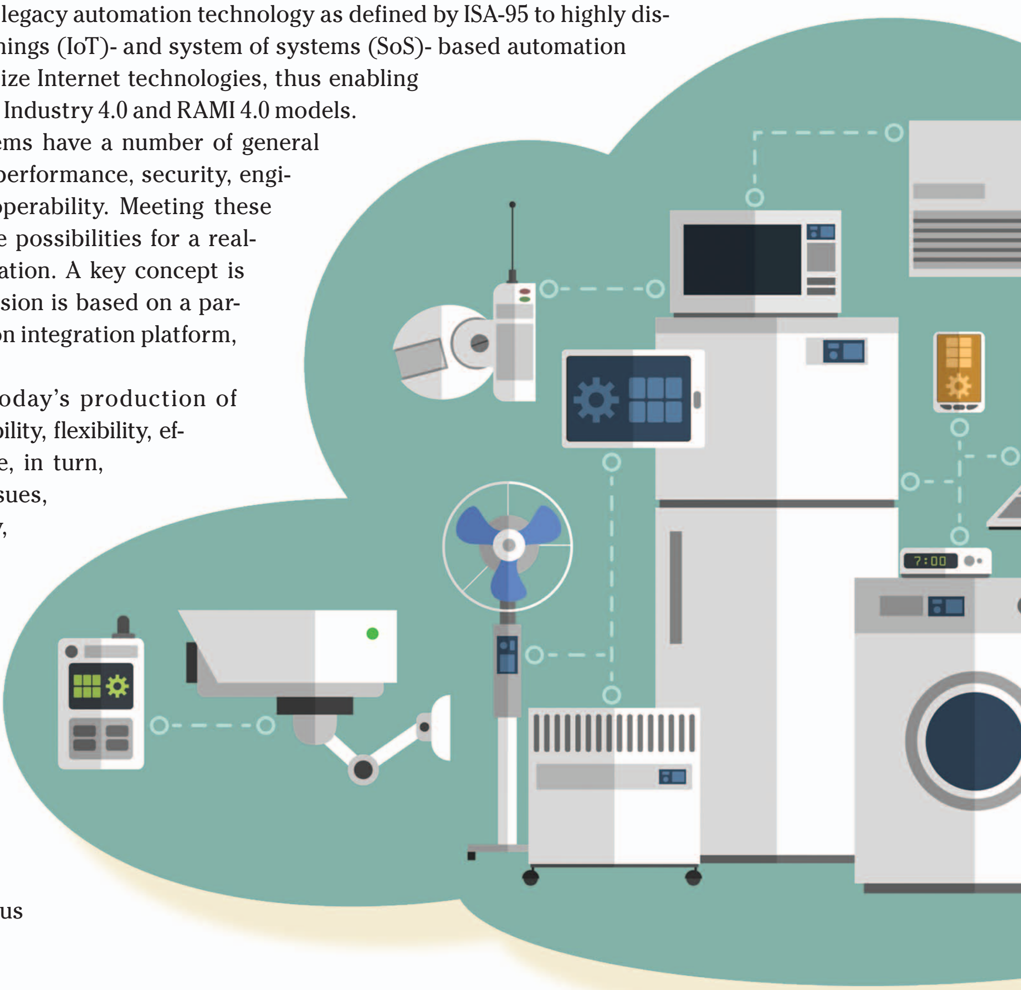
Technology and Business Model Features of Distributed Internet of Things Automation Solutions

JERKER DELSING

The hype concerning digitalization is increasing the demand for new generations of automation systems. Concepts like Reference Architecture Model Industry 4.0 (RAMI 4.0) give us models but do not tell us how to facilitate actual implementations. This article discusses the transition from legacy automation technology as defined by ISA-95 to highly distributed Internet of Things (IoT)- and system of systems (SoS)- based automation systems that fully utilize Internet technologies, thus enabling the implementation of Industry 4.0 and RAMI 4.0 models.

Distributed IoT automation systems have a number of general requirements concerning real-time performance, security, engineering cost, scalability, and interoperability. Meeting these requirements is necessary to enable possibilities for a real-world implementation of IoT automation. A key concept is local automation clouds. The discussion is based on a particular example of such an automation integration platform, the Arrowhead Framework.

High-level topics concerning today's production of goods and services include sustainability, flexibility, efficiency, and competitiveness. These, in turn, are driven by important societal issues, such as environmental sustainability, the availability of energy and other raw materials, and rapidly changing market trends. Several changes that address these topics in different ways are apparent. One change is the move from large monolithic organizations toward multistakeholder collaborations in which cooperation is fostered by market requirements. Another change is the immediate and continuous



Digital Object Identifier 10.1109/MIE.2017.2759342
Date of publication: 20 December 2017

1932-4529/17©2017IEEE

learning from previous products, other parts of the value chain, the life cycle of the product, and the product or service production process itself.

These trends are placing new requirements on the technology used to support current product and service production. For this reason, new approaches to production automation and stakeholder cooperation are being sought by many players. It is from these questions and requirements that the quest for the digitization of production is arising. Considering this situation reveals a number of gaps regarding technology, organization, cooperation structure, operational management, and related business models that need to be addressed.

IMAGE LICENSED BY INGRAM PUBLISHING

Toward Industrial and Societal Automation and Digitization

The high-profile key aspects of modern production are related to three domains defined in the collaborative management model (CMM) [1]–[3] (see Figure 1). The introduction of digitization enables multistakeholder cooperation in these domains defined in CMM [1]:

- multistakeholder product life-cycle management
- multistakeholder supply chain management
- multistakeholder production operations management.

With the move from large monolithic enterprises toward multistakeholder collaborations, management is similarly changing toward distributed multistakeholder cooperation with distributed responsibilities and decision making. Flexible cooperation in each of these three domains also opens up possibilities for dynamic learning through feedback, feedforward, and cross-linking within and between each of the three key domains of modern production (see Figure 2). A further aspect is that these domains tend to become wider (longer), thereby involving more stakeholders with diverse objectives and leading to more details and variations of the offered services or products in response to increased customer diversity and service and product quality requirements.

These ideas are currently emerging, but they are already regarded as very important in addressing the high-level topics of flexibility, efficiency, and competitiveness and, with suitable incentives, supporting sustainability. Such considerations have fostered the development of successor models to ISA-95. Proposed models that are gaining popularity include RAMI 4.0 [4] and the Industrial IoT (IIoT) [5]. The transition from ISA-95 to RAMI 4.0/Industrial Interenet Reference Architecture is visualized in Figure 3. It is also clear that the current hierarchical implementations of ISA-95 automation systems are not sufficient to address the dynamics encountered in a flexible, multistakeholder RAMI 4.0 production environment. Many companies are currently fighting high costs in making even small changes to their ISA-95-based production automation systems and thus are calling for new models and architectures for dynamic and digitized production.

In an attempt to support these developments, there are a number of technology gaps at present that seemingly cannot be addressed by the current state of the art. Consequently, a number of new technologies are emerging to fill these gaps. Several current technological trends that are receiving substantial attention include the following:

- IoT
- SoS
- cyberphysical systems (CPSs)
- clouds
- big data
- service-oriented architecture (SOA).

Despite the many new operational and organizational ideas and emerging technologies, the automation fundamentals captured by today's state-of-the-art automation technology must be maintained. Thus, the next generation of automation and digitalization technology will be required to satisfy a large set of requirements while also involving a wider scope of actors and stakeholders. This is the major challenge for the automation and digitization technology suppliers of the future. Given the rapid technological development related to automation and digitalization, such systems must be capable of evolving over time, thus further supporting the upgrading and expansion of automation.

Real-time performance and system safety are key requirements of legacy automation systems. To enable the implementation of next-generation automation systems using Internet technologies, it is proposed that the following high-level technology requirements will need to be addressed:

- real-time performance
- system safety
- IoT interoperability
- information technology security and associated system safety
- engineering simplicity to reduce design and run-time changes

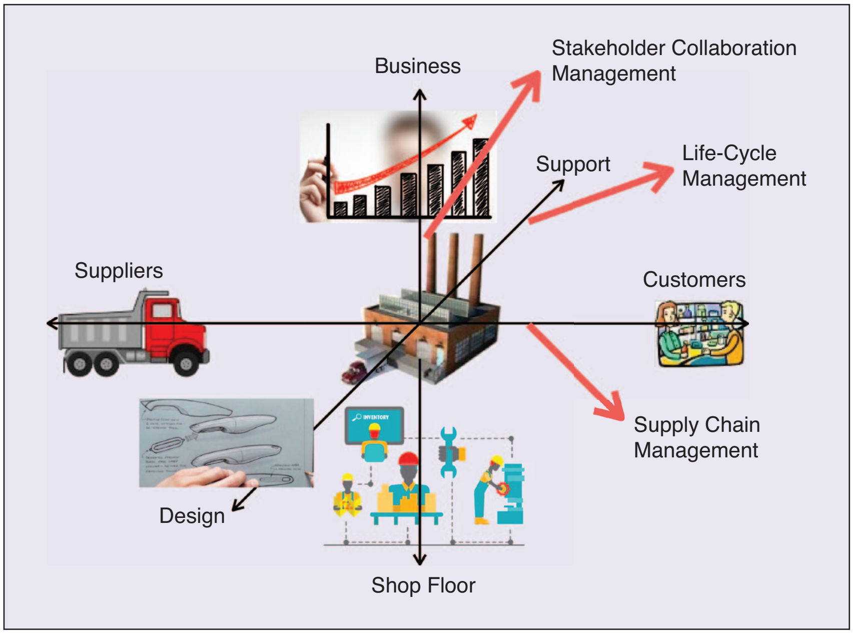


FIGURE 1 – The digitization extends production to multistakeholder production operations along the three axes of the CMM model: supply chain management, life-cycle management, and production operations management [1]–[3].

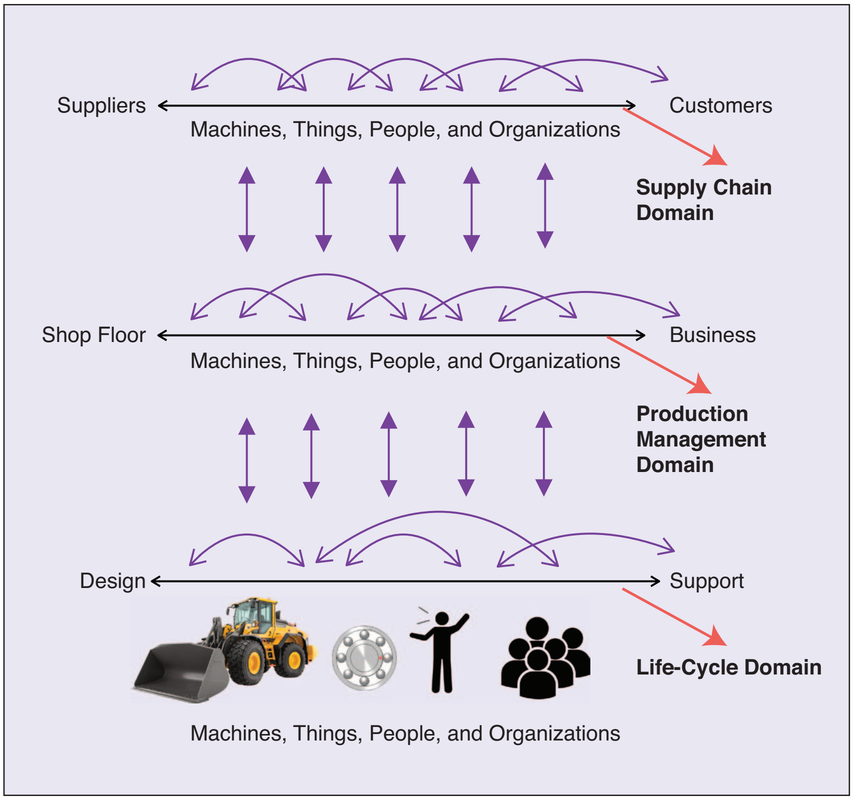

Suppliers
Customers
Machines, Things, People, and Organizations
Supply Chain Domain
Shop Floor
Business
Machines, Things, People, and Organizations
Production Management Domain
Design
Support
Life-Cycle Domain
Machines, Things, People, and Organizations

FIGURE 2 – The multistakeholder cooperation and management within and between the three key domains will require system capabilities of information feedback, feedforward, and cross-linking between machines, things, people, and stakeholder organizations.

- scalability to very large systems [> 10^6 input/output (I/O) points]
- system evolvability over time and technology generations.

Transitioning to the IoT and SoS

Current state-of-the-art automation systems are regarded as mission critical for the third generation of production systems. Here, sensors and actuators are connected to monitoring and control systems, such as distributed control systems (DCSs) and supervisory control and data acquisition (SCADA) systems, using technologies such as field buses. A hierarchical approach consisting of device, DCS, and SCADA levels (known as *ISA-95*) rapidly became the de facto architectural style for the design and deployment of industrial production systems and their detailed functionality. DCSs and SCADA systems soon became networked, thereby enabling integration between control systems and manufacturing execution systems (MESs) as well as enterprise resource planning (ERP) systems.

Today, this is the approach that is most widely used by the industry, as it has been for at least the last 20–30 years. In the 1990s, the current state-of-the-art architecture ISA-95 was established [6]. Seemingly, the size of ISA-95-based automation systems is limited to approximately 100,000 I/O points (no systems larger than ~10^5 I/Os have been built based on ISA-95, according to interviews with the world's five largest automation suppliers). The layered structure and lack of interoperability between devices both within one ISA-95 layer and between layers impose rigidity and inflexibility on such a system. The size limitation can most likely be understood as an engineering cost issue, as this rigidity and inflexibility result in enormous change costs when an automation system is to be changed due to production needs or for technological upgrades. Thus, automation engineering costs become a business bottleneck for a production industry experiencing rapid change.

In 2011, the concept of Industry 4.0 [7] was born in Germany. This concept builds upon the last generation of industrial monitoring and control systems, with the clear intent to enable greatly expanded interaction between shop-floor devices and high-level enterprise systems. In Industry 4.0, state-of-the-art research technologies, such as the IoT, CPSs, SoS, virtualization, real-time simulations, and big data analytics, are exploited. This enables improved production flexibility and breaks the classical

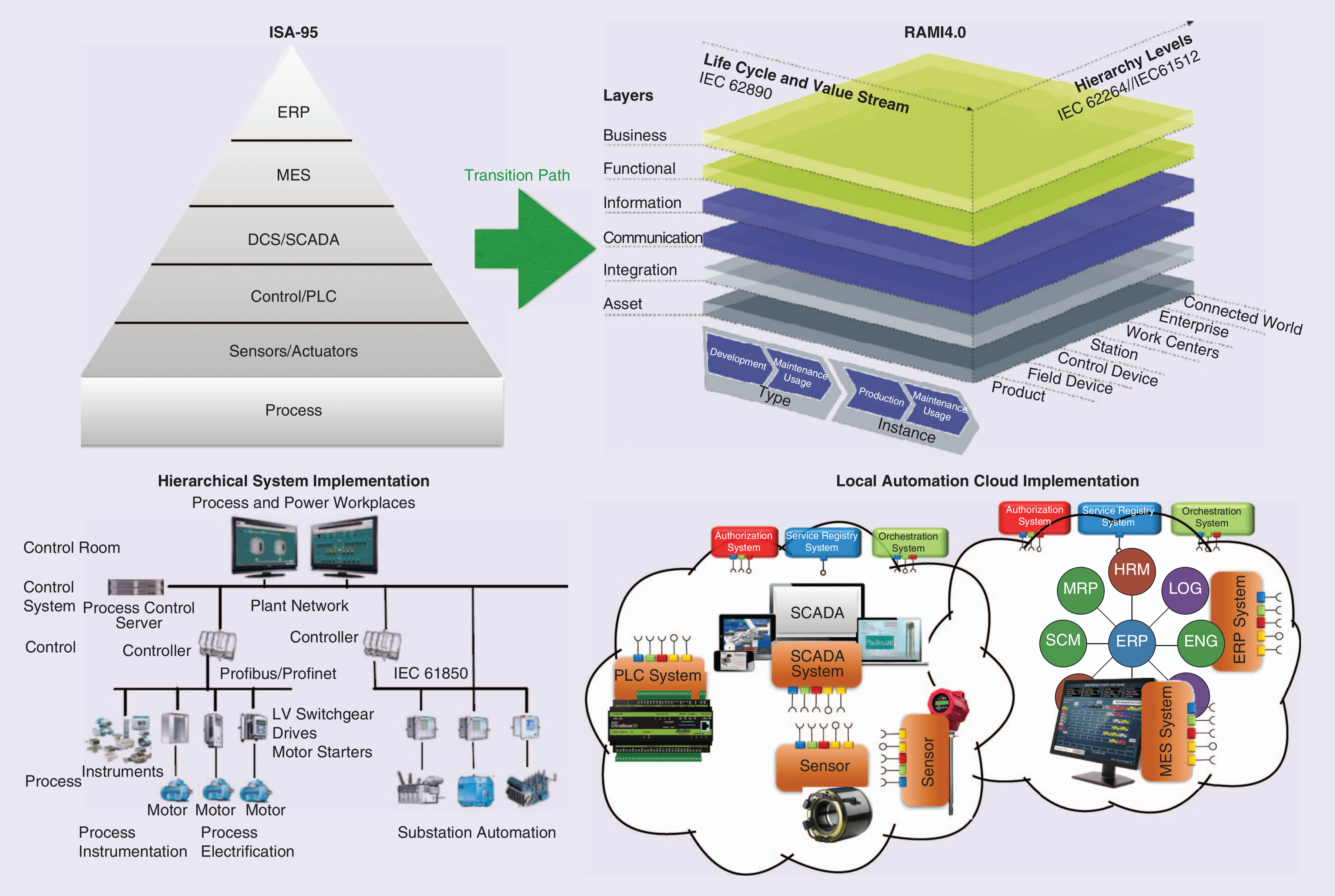


FIGURE 3 – A current digitalization trend is the transition of production automation from the ISA-95 model and its hierarchical implementation to, e.g., the more complex RAMI 4.0 model, with an Internet-Protocol-based networked/cloud implementation approach based on interoperable IoT devices/systems, thus providing, e.g., RAMI 4.0 implementations that can be dynamically modified at both design time and run time. ERP: enterprise resource planning; MES: manufacturing execution systems; SCADA: supervisory control and data acquisition; PLC: programmable logic controller; I/O: input/output; DCS: distributed control systems; ; HRM: human resource management; SCM: supply chain management; ENG: engineering; LOG: logistics and delivery; INV: inventory; FRM: finance resource management; Acc: accounting. (RAMI 4.0 image courtesy of Plattform Industrie 4.0 and ZVEI.)

strict hierarchical approach of ISA-95 [6]. When all communication is based on standard Internet protocols, i.e., the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite, information exchange becomes possible among (almost) all systems in a production facility. This opens up possibilities for new strategies in terms of, e.g., global plant optimization, operational safety, reduced energy consumption, and increased operational flexibility.

Simultaneously, the use of Internet technologies increases the visibility of information and communication technology (ICT) security threats. This visibility will facilitate a common understanding of ICT security and related impacts on automation systems. In turn, this understanding will support attempts to prevent ICT-security-related production problems. It should, however, be noted that brownfield installations will continue to use existing and well-functioning legacy communication infrastructures throughout their lifetime, and that suitable integration between legacy and IoT/SoS technology will therefore be provided.

For the development of Industry 4.0 and the digitalization of production industries, a number of clear trends related to automation systems have been identified based on different road maps and initiatives, such as Industry 4.0 [7], the Factory of the Future road map [8], and the ProcessIT Europe European Roadmap for Industrial Process Automation [9]. Key aspects of these trends are as follows:

- production flexibility and customization
- very large automation systems
- automation system security
- physically local automation
- automation engineering cost reduction.

These trends have long been under discussion by the research community. For more than ten years, discussions on the next generation of SCADA systems, DCSs, and MESs have been reported, and a multitude of research projects on this topic have been executed. Some of the more prominent works include SOFIA, SOCRADES [10], and IMC-AESOP [11]. All of these works investigated the move from the hierarchical ISA-95 approach to a more IoT-, SoS- and cloud-like approach, as depicted in Figure 4.

Regarding the transition from the ISA-95 architecture to a cloud-based approach, there are several important published works concerning system

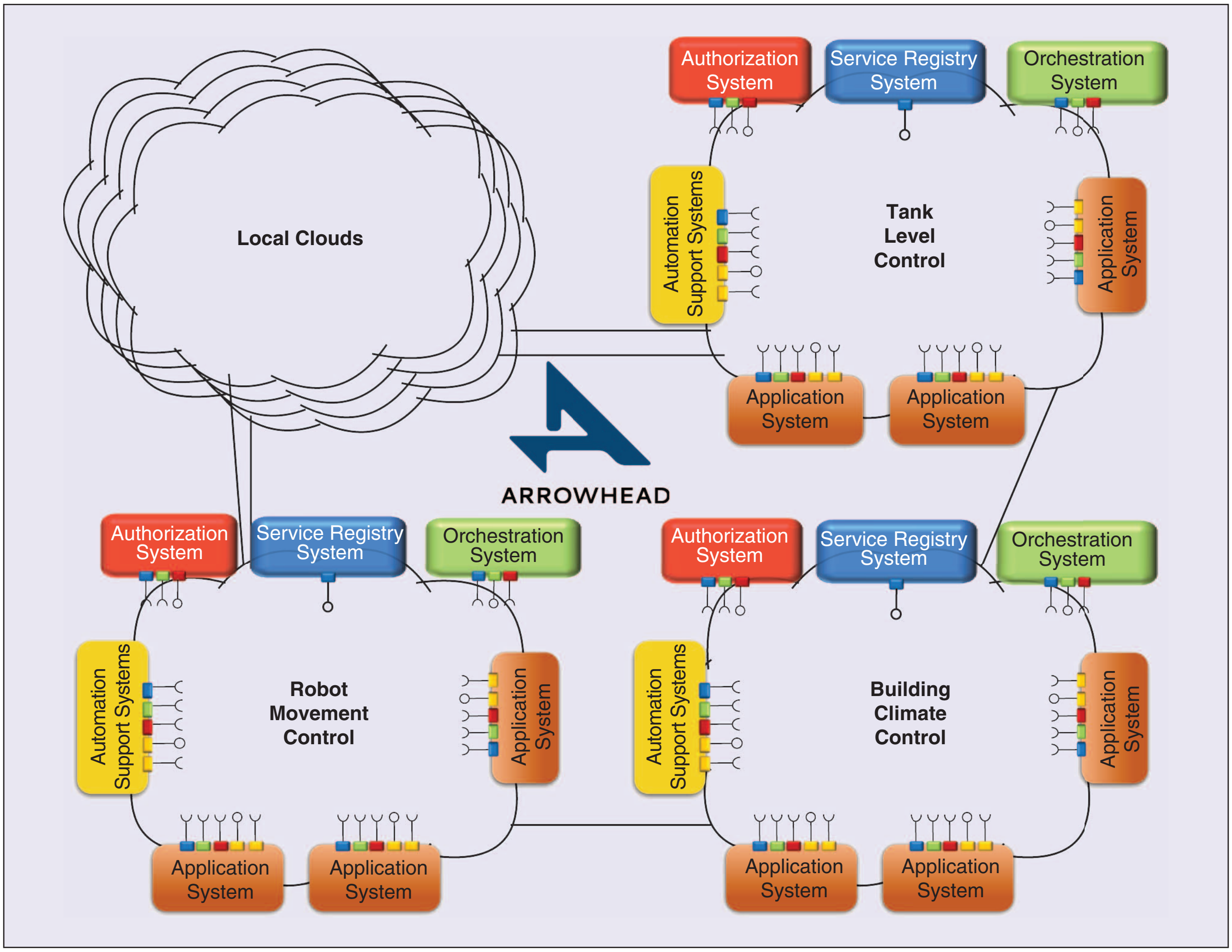


FIGURE 4 – The local cloud concept is visualized here as a number of local clouds with largely independent automation tasks. Each local cloud is self-sustained. Based on an SOA approach, each cloud provides a ServiceRegistry system (blue boxes), an authorization system (red boxes), and an orchestration system (green boxes). As indicated, higher-level cloud coordination is supported by intercloud communication and management.

architecture [12] and [13], networked control design [14], event-triggered control [15], suitable technologies [16], real-time networking [17]–[19], security [20], migration from legacy systems [21], and engineering for cloud-based automation [22] and [23].

A parallel discussion is ongoing regarding the MES and ERP levels. Some important publications on cloud approaches for MESs and ERP systems include [12] and [24].

This is currently a changing landscape, with a growing number of SOA and cloud proposals for IoT and SoS automation. An analysis of different IoT cloud approaches can be found in [25]. In most of the approaches proposed and studied to date, the key technologies for realizing interoperability and system integration are SOA [26] and various forms of cloud frameworks, such as fog and edge computing.

The use of SOA and various cloud technologies does not preclude the need to satisfy key automation requirements. Thus, the following considerations can be stated:

- The need for real-time performance in TCP/IP-based networks imposes requirements regarding the choice of the physical and transport layers together with network boundary protection from noncontrolled communication to real-time-critical nodes.
- The need for security imposes requirements regarding boundary protection, system authentication and authorization of service consumption, and encryption of payload data.
- The need for scalability imposes requirements regarding various functionalities, e.g., discovery, orchestration, authorization, and service exchanges across protective boundaries.
- The need for engineering simplicity imposes requirements in terms of reducing the number and complexity of dependencies and interactions to be considered.

Some of these requirements appear contradictory; e.g., real-time performance and scalability require networks in which the latency performance can be guaranteed regardless of the number of connected devices in the network.

Current state-of-the-art automation systems are regarded as mission critical for the third generation of production systems.

To overcome these contradictions, the concept of local automation clouds has been introduced [27] and [28]. The local automation cloud concept takes the view that one or a few physically local automation actions are to be supported within a local cloud. Each local cloud is self-contained and does not require any external support to perform the needed functionalities. Thus, a local cloud can be viewed as a protection boundary for automation operations, protecting the local automation operations from any external communication that might impair their real-time performance, safety, or security. This is in contrast to the Internet cloud concept [29].

Service exchanges between local clouds can be made, although without any latency guarantees, thereby enabling the engineering and management of a large automation system consisting of multiple local clouds. To enable the concept of local clouds and their capability for internal and external service exchanges, a local cloud must possess certain properties. The Arrowhead Framework implementation of local clouds specifies the following primary properties of a local automation cloud:

1) self-containment—no external resources are required for operations in the local cloud
    - device, system, and service discovery
    - service orchestration—SoS run-time configuration
    - device, software (SW) system, and service authentication and authorization.
2) security
    - security fence for protection from external networks
    - secure bootstrapping and SW updates
    - support for secure administration and data exchange with external resources.
3) interoperability of automation technology
    - support for protocol, encoding, and semantics transparency.
4) scalability
    - provisions for service discovery between local clouds
    - provisions for service orchestration between local clouds
    - provisions for authentication and authorization between local clouds.
5) automation support
    - support for automation system design, configuration, deployment, operation, and maintenance
    - support for event- and polling-based automation
    - support for service exchange quality of service (QoS)
    - support for service exchange audits
    - support for device, system, and service metadata.

Higher-system-level needs for communication through the protective boundary of a local cloud should be controlled based on engineering and security management decisions. Furthermore, the internal cloud QoS manager should be able to provide feedback before such decisions are deployed, thereby supporting the protection of critical cloud-internal real-time operations.

IoT and SoS Technologies

When discussing suitable technologies for SoS automation systems based on IoT devices, the following basic automation requirements are important:

- interoperability between devices and SW systems
- scalability
- real-time performance
- security
- engineering simplicity.

The relevant technologies are often related to low-level technologies, e.g., protocols (such as CoAP and 6LowPAN) [30] and [31], or various IoT cloud concepts, e.g., Cumulosity, ThingWorx, Xively, Azure, and Websphere [25]. In addition, various concepts, such as local clouds, edge computing, and fog computing, have been discussed (e.g., see [27] and [32]–[35]) with regard to

In most cases, the integration of many IoT concepts into SoSs and automation systems of any scale has currently not been addressed at a level that meets industrial automation system engineering standards.

their possible use in the IoT context. In most cases, the integration of many IoT concepts into SoSs and automation systems of any scale has currently not been addressed at a level that meets industrial automation system engineering standards.

There has been some interesting work on SoS management since the 1970s [36] as well as theoretical foundations for the discussion of SoS technology [37] and [38], SoS engineering [39], and SoS automation [40]. However, in the literature, the basic automation requirements are considered only partially or not at all. Therefore, it is of interest to look at some of these concepts in light of these basic automation requirements.

Local Clouds, Fog and Edge Computing, and Cloud Computing in Automation

Interesting views on cloud computing and cloud manufacturing have been presented by [41]–[43]. The following concepts are considered here:

- local clouds
- fog and edge computing
- cloud computing

The principal differences between these concepts are visualized in Figure 5. The main emphasis in cloud computing is on providing scalable and remote (central) computing and storage resources [29]. Fog and edge computing provide the same but with the resources distributed among edge devices, such as routers, switches, or cell heads [44]–[47]. In contrast to this, the main focus of a local cloud is on protecting its own internal functionalities from external disturbances and thus providing performance guarantees regarding real-time operation, security, safety, and so forth.

Each of these concepts possesses certain features and properties. Evaluations of these concepts with regard to the key requirements for automation yield the results in the following sections.

Cloud Computing

Local IoT devices connect directly to the cloud, where necessary automation-related computing is performed.

- *Real-time performance*: For the transfer of data to and from the cloud, the quality of Internet access and the route to the cloud determine the achieved real-time performance. A certain latency cannot, in general, be guaranteed. Regarding the computation time, the cloud provider can most likely satisfy the requirements for real-time performance, but other applications being executed on the same cloud may influence this capability.
- *Security*: All automation data are transmitted over open ports. Because such transport is bidirectional, this will create a hole in any firewall involved. Thus, each involved IoT device risks various security attacks, and resource-constrained IoT devices will be particularly at risk.
- *Interoperability*: An application executed in the cloud must be able to understand the different protocols, encoding schemes, and semantics used by each IoT device involved. This will require some type of translation functionality because standardization to a single set of protocols, encoding schemes, and semantics is not likely. Such translation is expected to impair real-time performance due to increased data transfer delays.
- *Engineering simplicity*: The engineering of cloud-based SoS solutions, apart from the engineering of automation solutions, requires consideration of the complexities of the different protocols, encoding schemes, and semantics used. Further precautions concerning communication over open Internet channels and cloud computation operations must be considered.

Fog and Edge Computing

Similar to cloud computing, local IoT devices connect directly to the fog or edge to use computational resources. The difference from the cloud scenario is that these computational resources are located physically closer to the automation functionalities to be performed.

- *Real-time performance*: The situation is similar to that for cloud computing. It is, however, easier to limit exposure to the open Internet and to prescribe the properties of the lower layers involved in communication. Some early contributions referring to fog and real time are [48] and [49].
- *Security*: This situation is also similar to that for cloud computing. Again, however, it is easier to limit exposure to the open Internet; see, e.g., [50] and [51].
- *Interoperability*: An application executed in the fog or at the edge must be able to understand the different protocols, encoding schemes, and semantics used by each IoT device involved. Again, this is not likely to be possible without some translation functionality, which is expected to impair real-time performance.
- *Engineering simplicity*: The engineering of cloud-based SoS solutions, apart from the engineering of automation solutions, requires consideration of the complexities of the different protocols, encoding schemes, and semantics used. Further precautions for communication over open Internet channels and fog and edge computing resource operations must be considered.

Local Cloud Automation

Here, each local automation functionality is encapsulated by a self-contained local cloud, which can act as a protective boundary against any external communication.

- *Real-time performance*: This scenario provides full protection from the open Internet, and through local ownership, the properties of

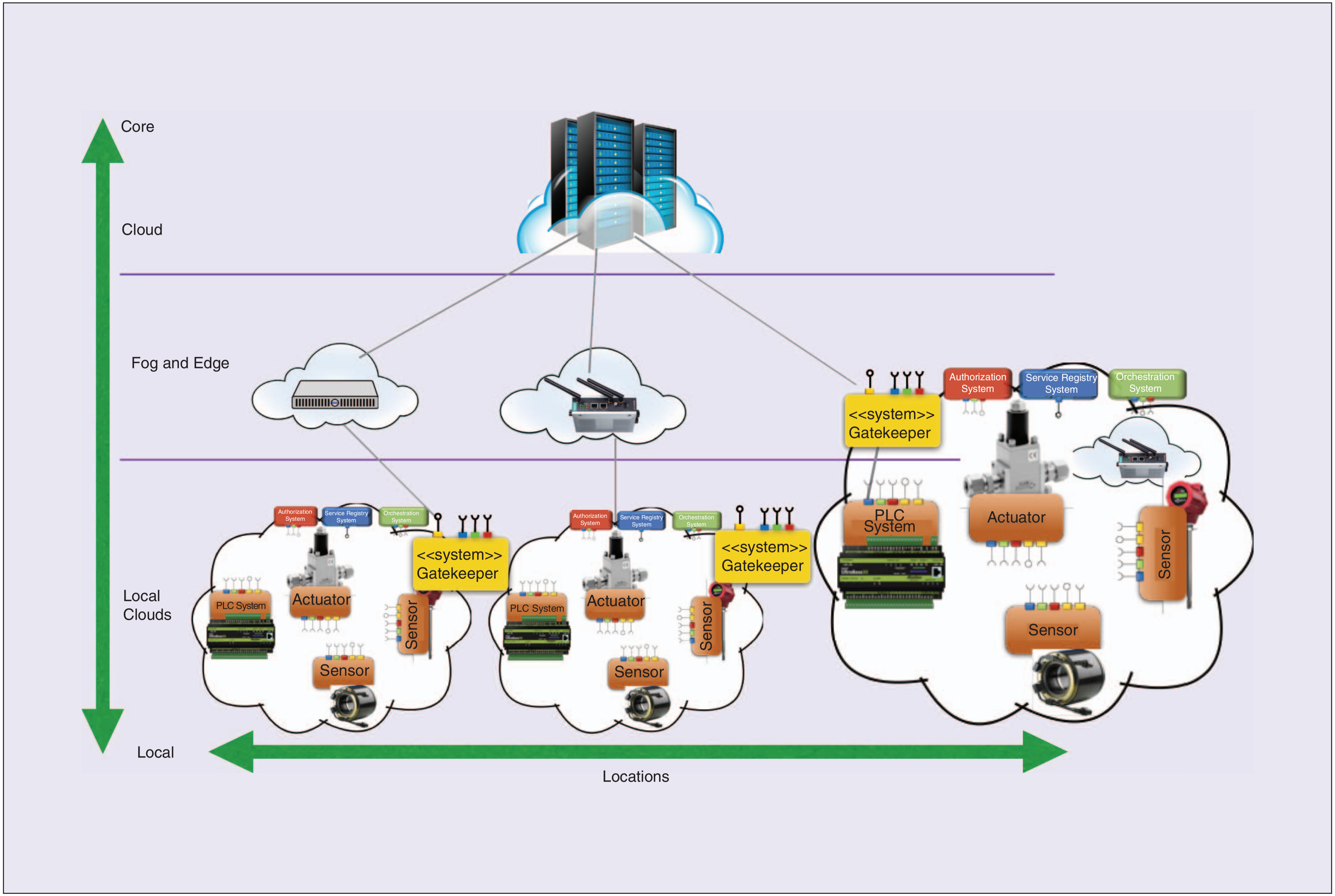


FIGURE 5 – The principal conceptual differences between cloud computing, fog and edge computing, and local automation clouds. Cloud and fog and edge computing provide scalable computing and storage resources in either a central or distributed manner, whereas local automation clouds focus on protecting their devices/systems from external communication to ensure real-time and safety performance.

Based on the basic automation requirements, it is argued that the cloud boundary acts as a protective fence for the communications and computations that are necessary to fulfill the desired automation tasks.

the lower layers involved in communication can be prescribed, thereby enabling real-time performance equivalent to that of current SCADA, DCS, and programmable logic controller (PLC) systems.

- *Security*: Full protection is provided from the open Internet, with the possibility for internal cloud authorization and authentication of the involved HW and SW systems and services consumed.
- *Interoperability*: Automation functionalities are created through the orchestration of service exchanges between the involved IoT devices. For critical automation functionality, it is feasible to standardize to one set of protocols, encoding schemes, and semantics within a specific local cloud, thereby ensuring interoperability with minimal real-time impact. For less time-critical functionalities within or between local clouds, the use of translation functionalities for protocols, encoding schemes, and semantics is feasible.
- *Engineering simplicity*: No consideration for communication over open Internet channels is needed. Automation applications can be created based on legacy automation design standards, e.g., CAEX and IEC 81346, from which the configuration of IoT services and the orchestration of service exchanges can be extracted for implementation in the local cloud. This creates an integrated system functionality as designed at the plant level. Most incompatibilities related to protocols, encoding, compression, and semantics are standardized within the specific local cloud or hidden from the engineering design through the automatic invocation of translation services.

From these key requirements, it can be concluded that local automation clouds seem to be superior to cloud and fog/edge computing for the execution of mission-critical automation. Thus, local clouds are attractive for the implementation of highly distributed and autonomous automation systems. In this article, the Arrowhead Framework is used as the implementation example to demonstrate how a distributed local cloud automation solution might look.

Local Clouds

From the preceding discussion, it is clear that the purpose of a local cloud is to provide a communication and computation environment suitable for automation. Based on the basic automation requirements, it is argued that the cloud boundary acts as a protective fence for the communications and computations that are necessary to fulfill the desired automation tasks, thereby protecting the automation functionalities, particularly their time-critical communications and computations, from external influences. Thus, a basic local automation cloud is a protected network with no inbound or outbound communication, in direct contrast to current mainstream definitions of clouds, which are a metaphor for the Internet.

For a protected local cloud to be implemented, it must possess certain key properties, which include the following:

1) self-containment—no external resources are needed to establish the local cloud
2) a security fence for protection from external networks
3) interoperability between systems at the service level
4) security in relation to device and SW system deployment, local cloud bootstrapping, SW updates, and service exchanges
5) intercloud service exchanges
6) automation support, at both design and run time.

Implementation and Technologies: Arrowhead Framework

The properties of local automation clouds described previously have been implemented in the Arrowhead Framework [52] using, as much as possible, existing and well-established Internet standard technologies. The Arrowhead Framework is the culmination of several European projects, including SOCRADES, IMC-AESOP, Arrowhead, Mantis, EMC2, FAR-EDGE, and Productive 4.0. Implementation solutions to each of points 1–6 above are presented in the sections that follow.

Self-Containment

A self-contained Arrowhead Framework local cloud has three basic functionalities: discovery, orchestration, and authentication/authorization. The discovery of services available within the local cloud is implemented through the ServiceRegistry system [28] and [53]. Here, a service-producing SW system registers each of its services, specifying the interface, data types, and associated metadata using Domain Name System Service Discovery [54] as the underlying technology.

A service must be consumed by an authorized unit whose identity is known and acknowledged. The Arrowhead Framework supports two levels of this security mechanism through its authorization system. The system uses either an X.509 certificate/ssh [55] solution or, for resource-constrained devices, a radius ticket solution [56] and [57].

To allow for the autonomous operation of service exchanges, the orchestration of service consumption is necessary. The Arrowhead Framework defines and implements an orchestration system that imposes orchestration rules/algorithms on application systems to allow them to autonomously perform automation tasks [28].

The ServiceRegistry, authorization, and orchestration systems are regarded as mandatory systems within the Arrowhead Framework. These systems enable the implementation of a minimal and self-contained local automation cloud. These mandatory systems and their produced and consumed services, with indications of their mutual interactions as well as interactions with other core systems, are depicted in Figure 6.

The Arrowhead Framework has recently been applied in more than 20 industrial demonstrations.

A Security Fence for Protection from External Networks

A straightforward method of creating a local cloud security fence is to establish a completely detached network. However, if wireless technology is involved, this might be difficult to make foolproof. Here, authentication and authorization become necessary to prevent unwanted communications within the local cloud. In this way, the unauthorized spread of data can also be hindered.

To address disturbances to critical intracloud communications, the Arrowhead Framework defines and implements a QoSMonitor and a QoSManager [58]. These systems can monitor service exchange latency and impose changes on the intracloud network parameters to, e.g., reduce interference from undesired wireless communications/disturbances.

Interoperability Within the Local Cloud

Interoperability among various devices is established at the service level, with a clearly documented structure for how services can be accessed. Services can be implemented using various SOA protocols, e.g., REST, CoAP, XMPP, MQTT, and OPC-UA, together with various encoding and compression schemes and data semantics. For devices using different protocols, encoding schemes, and so forth, dynamic translation is provided by a translation system [59]. A CoAP- and JSON-based service producer can thus provide its services to a service consumer with REST and XML capabilities.

Security in Relation to Device and Software System Deployment, Local Cloud Bootstrapping, SW Updates, and Service Exchanges

From a security perspective, it is important to know what trusted hardware (HW), SW, and associated services are deployed in the local cloud. For engineering, management, and security purposes, a DeviceRegistry and a SystemRegistry (SW) are also provided. These registries enable the discovery of HW devices and their associated SW. For high-security applications, the HW must possess advanced security controllers to enable the two-way identification of the HW and associated SW [60].

Intercloud Interoperability and Service Exchanges

Service exchanges between applications in different local clouds must be supported. For this purpose, service discovery, authentication and authorization, and orchestration must be supported between local clouds. The Arrowhead Framework core system gatekeeper is used for this purpose [53] and [61]. The gatekeeper acts as an advanced service gateway to external wide area networks. Thus, the same mechanisms that are applied within a local cloud are also used to organize a service exchange between two devices/applications. This service exchange can be executed after its initiation in accordance with the stated orchestration rules/algorithms.

Because these service exchanges will be partially conducted over the open Internet, payload encryption becomes important, together with mechanisms for preventing holes in the gatekeeper firewall. The Arrowhead Framework provides two mechanisms to address this problem [28]. One mechanism is based on the use of the MQTT protocol, in which service access is protected by the MQTT service broker, and payload data are protected with TLS/DTLS [62]. The other mechanism uses DMZ [63] and a double historian concept, in which the requested external service is tunneled through a one-way tunnel between two historian services. This to protect the local cloud services from direct external access.

Automation Support, at Both Design and Run Time

To support the design, engineering, and operation of local cloud automation, the following support systems

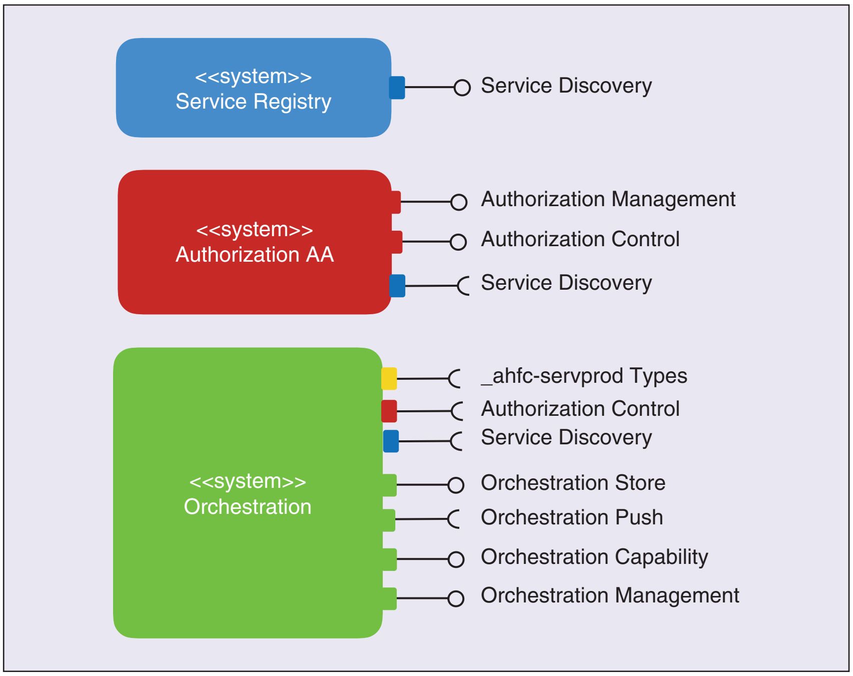


FIGURE 6 – The mandatory core systems of an Arrowhead Framework local cloud, i.e., the ServiceRegistry and authorization and orchestration systems, with their produced and consumed services (see the lollipops). The service color coding indicates the interactions required between the mandatory systems and other systems (yellow lollipop).

The business success demonstrates the industrial rollout capability of SoS automation solutions based on IoT devices and services.

are defined within the Arrowhead Framework:

- *PlantDescription*: Based on the CAEX, IEC 62424 automation system engineering standards, the PlantDescription system enables the capture of plant HW as well as the associated SW and capabilities regarding the production and consumption of services (data). Based on engineering data on plant functionality, orchestration rules are created and pushed to the orchestration system [64].
- *Configuration*: The configuration system captures engineering data for the configuration of plant devices and their associated SW [65].
- *Historian*: The historian system enables service exchange audits and the storage of payload data [28].

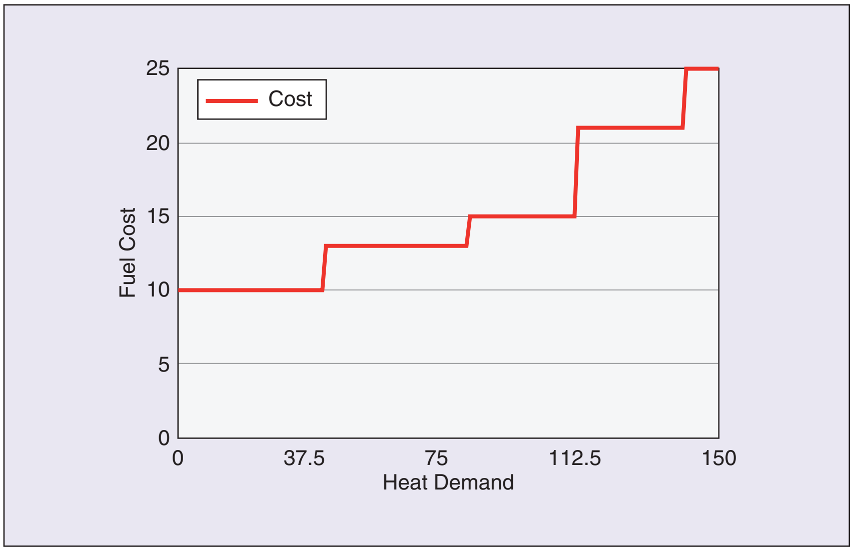


FIGURE 7 – The fuel cost will increase in a stepwise manner when heat demands increase, based on the available heat production units and the fuel price.

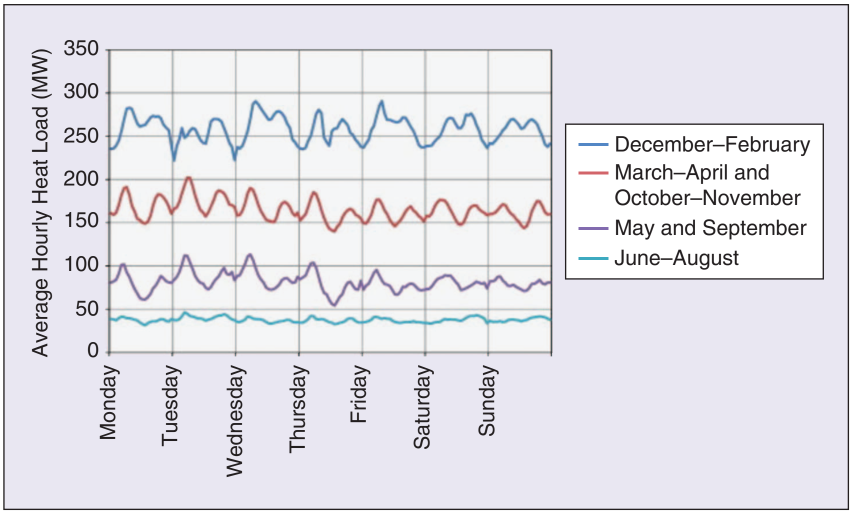


FIGURE 8 – The average weekly heat demand pattern for multidwelling buildings. (Figure courtesy of *Applied Energy*, Elsevier [66].)

Industrial Application: Building Energy Automation

The Arrowhead Framework has recently been applied in more than 20 industrial demonstrations, one of which will be discussed here in some detail. This demonstration addresses energy optimization for district heating networks. In district heating system operation, more expensive primary fuels are used as heat demand increases (see Figure 7). Peak heat demand times are most often fairly short (hours) and related to user behavior in the morning and evening (see Figure 8). Most buildings heated by district heating have a certain heat capacity (heat is stored in the walls). If this heat capacity could be utilized to bridge peak heat demand, several benefits could be achieved:

- the possibility of avoiding the use of expensive primary fuels for peak heat generation
- overall reduced primary fuel costs
- the ability to connect more customers to the existing distribution network.

For this purpose, the district heating substation control systems of each building will reduce peak energy consumption using a control strategy seeking to reduce peak consumption during morning and evening peak hours.

In one of the demonstrations performed within the Arrowhead project, the companies Abelko and Noda formed a joint market proposal regarding peak demand for heat in district heating systems.

Abelko's offerings are as follows:

- an Arrowhead-Framework-compliant PLC,
- IMSE UltraBase30/UltraCom
- an Arrowhead-Framework-compliant SCADA system
- IMSE Comprobo.

Noda provides the following:

- an Arrowhead Framework smart heat grid service
- Noda Smart Heat Grid
- an Arrowhead Framework smart heat building service
- Noda Smart Heat Building.

These four offerings were combined into a joint business offering using the Arrowhead Framework local cloud approach (see Figure 9).

The engineering efforts to integrate the combined solution using both a legacy approach and the Arrowhead Framework approach were measured. Data for this industrial use case were provided by the companies. The companies also provided data on engineering savings for two other commercial use cases; all of these data are presented in Table 1 [27]. The joint market offering has been a market success. The solution is now being rolled out to several larger energy companies and real estate companies. The first-year revenue prediction is €5–10 million.

An analysis of the demonstration yields the following interesting observations:

- The technical integration was made possible by the Arrowhead Framework and achieved a very competitive cost.
- The technological solution enabled a joint business agreement.
- The technological solution enabled an attractive market proposal.
- The business success demonstrates the industrial rollout capability of SoS automation solutions based on IoT devices and services.

Industrialization Status

Considering the case of building energy automation business, it is interesting to analyze the industrial maturity of distributed IoT automation technologies. In Table 2, nine technological areas important to IoT automation are analyzed for market maturity based on product and service availability. This analysis is based on discussions with more than 100 vendors and complementary web searches. Based on the data obtained, a technology readiness level (TRL) range is provided to indicate the market maturity for each of the nine areas. From this analysis, it can be concluded that engineering tools for design, deployment, management, operation, security, and migration are currently the greatest bottleneck hindering the ability to bring IoT and SoS automation solutions to the market. Obviously, the use of existing legacy engineering tools for building automation functionality should be continued, and their outputs must be integrated with IoT-based automation systems and the

Engineering tools for design, deployment, management, operation, security, and migration are currently the greatest bottleneck hindering the ability to bring IoT and SoS automation solutions to the market.

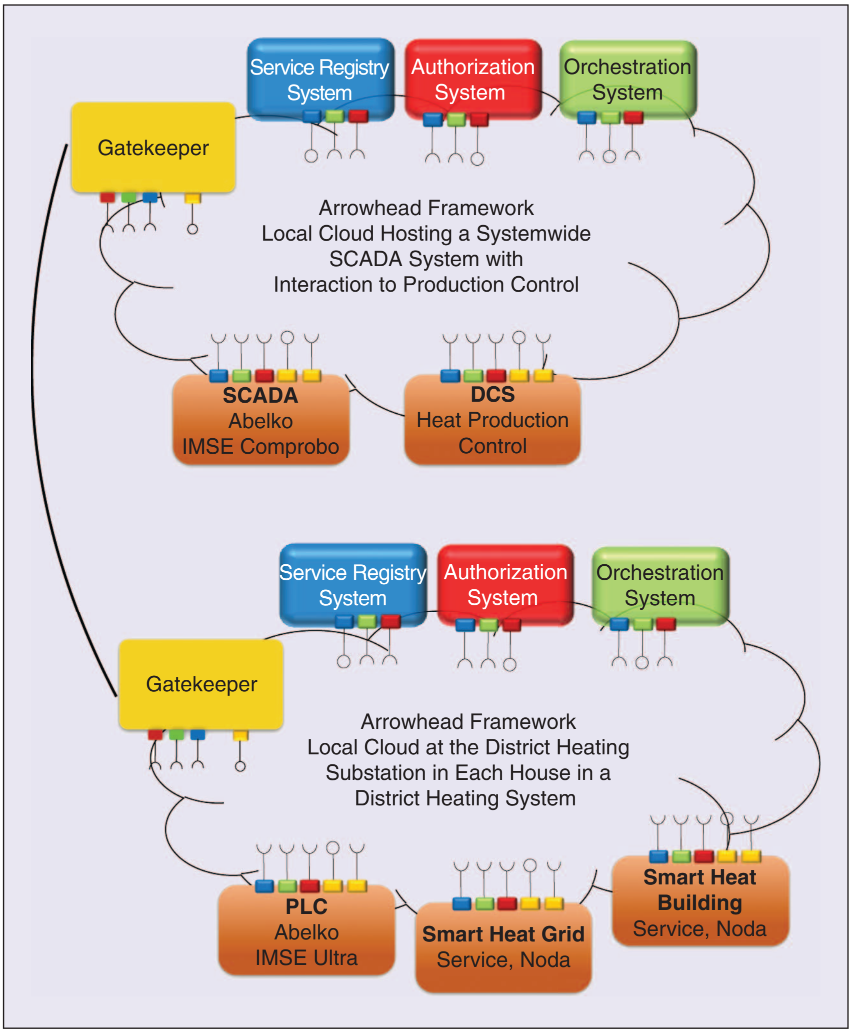


FIGURE 9 – The local cloud configuration for the peak heat demand control system integrating the offerings from the two companies. There is one local cloud for each building, each with a district heating substation. These local clouds provide data to the SCADA-level local cloud where production system control set points are provided to the heat production DCS.

TABLE 1 – ENGINEERING TIME FOR THE IMPLEMENTATION OF AUTOMATION APPLICATIONS VIA LEGACY AUTOMATION AND LOCAL CLOUD AUTOMATION.

APPLICATION	LOCAL CLOUD [H]	LEGACY [H]	GAIN
Building energy automation	6–8	40–48	1:5
Airport information automation	40	160–200	1:4.5
Recycling logistics	80	240–300	1:3.5

The transition from legacy ISA-95-based automation systems to distributed IoT-based automation systems has begun.

new tools needed for these systems. The migration of brownfield production to IoT automation systems is expected to become an interesting market, but again, engineering tools and skills are yet to be developed for this purpose.

Conclusions

The transition from legacy ISA-95-based automation systems to distributed IoT-based automation systems has begun. To satisfy the requirements for automation regarding real-time performance, interoperability, engineering simplicity, security, and so forth, classical Internet and cloud technologies are not satisfactory. However, concepts like local automation clouds based on distributed IoT and SoS solutions have properties that enable implementations that do satisfy the above automation requirements.

Implementation platforms like the Arrowhead Framework offer technologies for enabling the real-world implementation of IoT- and SoS-based automation solutions. This is exemplified here by a real-world, technically and commercially successful example from the energy sector. In this example, a substantial reduction in engineering costs by a factor of three to five compared with legacy implementations is achieved.

Based on the presented technology maturity analysis, it can be concluded that several technological areas necessary for the successful design, implementation, and operation of SoS and IoT automation systems have already started to reach TRLs 7–9 and are approaching wider market introduction. The exception is engineering tools, for which further improvements and innovations are still needed.

Further work toward industrially feasible distributed IoT automation systems will need to address various topics related to ease of engineering, such as 1) the dynamic generation of new services based on changes to device and SW configurations, 2) the autonomous generation of service consumer functionality based on service metadata, 3) dynamic security feature management and engineering, 4) smart service contracts, and 5) anomaly detection (both functional and security related). This list could be much longer but, in its current form, provides an idea of what is currently being worked on as part of European projects.

TABLE 2 – IoT AUTOMATION TECHNOLOGY MATURITY REPRESENTED BY QUALITATIVE JUDGMENTS REGARDING PRODUCTS ON THE MARKET AND QUALITATIVE JUDGMENTS OF VENDOR TRL STATUS AS AN INDICATOR OF MARKET MATURITY.

IoT AUTOMATION TECHNOLOGY	QUALITATIVE JUDGMENT OF PRODUCTS ON THE MARKET	VENDOR TRL STATUS
Robust and real-time communication	Products on the market	8–9
IoT sensors, actuators, PLCs, and so forth	Some products on the market	8–9
DCS and SCADA functionality based on SOA	First products on the market	6–9
MES and ERP functionality based on SOA	First product on the market	6–9
Cloud integration technology based on SOA	Some products on the market	6–9
Engineering tools for cloud-based IoT automation systems	Demonstrated in industrial environments	4–6
Test tools and simulators for cloud-based IoT automation systems	First products on the market	4–9
Migration to cloud-based IoT automation systems	Demonstrated in industrial environments	4–6
Suitable security for cloud-based IoT automation systems	Some products on the market	4–9

Acknowledgments

I would like to thank European Union ECSEL-JU for providing funding for the Arrowhead project under contract 332987, for the Productive 4.0 project under contract 737459, and for the FAR-EDGE project under contract 723094.

Biography

Jerker Delsing (jerker.delsing@ltu.se) received his Ph.D. degree in electrical measurements from Lund University, Sweden, in 1988. Since 1995, he has been a chaired professor in industrial electronics at Lulea University of Technology, Sweden. He is a steering board member of ARTEMIS and ProcessIT Europe. His EISLAB group has been a partner of many European Union projects in the field of production automation, e.g., SOCRADES, IMC-AESOP, Arrowhead (for which he was the coordinator), FAR-EDGE, and Productive 4.0, which has built the expertise for this article.

References

[1] ARC. (2017). Collaborative manufacturing management. *ARC Advisory Group.* [Online]. Available: https://www.arcweb.com/industry-concepts/collaborative-management-model-cmm#

[2] R. Mick and C. Polsonetti, "Collaborative automation: The platform for operational excellence," ARC Advisory Group, Boston, MA, 2003.

[3] P. Leitao, A. W. Colombo, and F. J. Restivo, "Adacor: A collaborative production automation and control architecture," *IEEE Intell. Syst.*, vol. 20, no. 1, pp. 58–66, Jan. 2005.

[4] P. Adolps and U. Apple, "Status report: Rami4.0," VDI/VDE-Gesellshact Mess- und Automatisierungstechnik, Düsseldorf, Germany, 2015.

[5] S.-W. Lin, M. Crawford, and S. Mellor. (2016). The industrial Internet of Things volume g1: Reference architecture. *Industrial Internet Consortium.* [Online]. Available: https://www.iiconsortium.org/IIC_PUB_G1_V1.80_2017-01-31.pdf

[6] B. Scholten, *The Road to Integration: A Guide to Applying the ISA-95 Standard in Manufacturing.* Research Triange Park, NC: International Society of Automation, 2007.

[7] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for Industry 4.0-based manufacturing systems," *Manuf. Lett.*, vol. 3, pp. 18–23, Jan. 2015.

[8] EFFRA. (2013). Factories of the future: Multi-annual roadmap for the contractual PPP under horizon 2020. European Commission. Brussels, Germany. [Online]. Available: http://www.effra.eu/sites/default/files/factories_of_the_future_2020_roadmap.pdf

[9] ProcessIT.EU. (2013). European roadmap for industrial process automation. European Regional Development Fund, European Union.

[Online]. Available: http://www.processit.eu/Content/Files/Roadmap%20for%20IPA_130613.pdf

[10] SOCRADES. (2016). Welcome to SOCRADES 2006–2009. [Online]. Available: http://www.socrades.net

[11] A. W. Colombo, T. Bangemann, S. Karnouskos, J. Delsing, P. Stluka, R. Harrison, F. Jammes, and J. L. Lastra, Eds., *Industrial Cloud-Based Cyber-Physical Systems: The IMC-AESOP Approach*. New York: Springer-Verlag, 2014.

[12] S. Karnouskos and A. W. Colombo, "Architecting the next generation of service-based SCADA/DCS system of systems," in *Proc. Annu. Conf. IEEE Industrial Electronics Society 2011*, Melbourne, Australia, p. 6.

[13] F. Jammes and H. Smit, "Service-oriented paradigms in industrial automation," *IEEE Trans. Ind. Inform.*, vol. 1, no. 1, pp. 62–70, Feb. 2005.

[14] J. Wu and T. Chen, "Design of networked control systems with packet dropouts," *IEEE Trans. Automat. Control*, vol. 52, no. 7, pp. 1314–1319, 2007.

[15] D. Yue, E. Tian, and Q.-L. Han, "A delay system method for designing event-triggered controllers of networked control systems," *IEEE Trans. Automat. Control*, vol. 58, no. 2, pp. 475–481, 2013.

[16] F. Jammes, B. Bony, P. Nappey, A. W. Colombo, J. Delsing, J. Eliasson, R. Kyusakov, S. Karnouskos, P. Stluka, and M. Till, "Technologies for SOA-based distributed large scale process monitoring and control systems," in *Proc. 38th Annu. Conf. IEEE Industrial Electronics Society,* 2012, pp. 5799–5804.

[17] H. Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications*. New York: Springer-Verlag, 2011.

[18] G. Candido, A. W. Colombo, J. Barata, and F. Jammes, "Service-oriented infrastructure to support the deployment of evolvable production systems," *IEEE Trans. Ind. Inform.*, vol. 7, no. 4, pp. 759–767, Nov. 2011.

[19] R. Kyusakov, P. P. Pereira, J. Eliasson, and J. Delsing, "Exip: A framework for embedded web development," *ACM Trans. Web*, vol. 8, no. 4, p. 23, 2014.

[20] K. Fischer and J. Geßner, "Security architecture elements for IoT enabled automation networks," in *Proc. 2012 IEEE 17th Conf. Emerging Technology Factory Automation (ETFA),* pp. 1–8.

[21] O. Carlsson, J. Delsing, F. Arrigucci, A. W. Colombo, T. Bangemann, and P. Nappey, "Migration of industrial process control systems into service-oriented architectures," *Int. J. Comput. Integr. Manuf.*, to be published.

[22] A. Jain, D. Vera, and R. Harrison, "Virtual commissioning of modular automation systems," *Intell. Manuf. Syst.*, vol. 10, no. 1, pp. 72–77, 2010.

[23] N. Kaur, C. S. McLeod, A. Jain, R. Harrison, B. Ahmad, A. W. Colombo, and J. Delsing, "Design and simulation of a SOA-based system of systems for automation in the residential sector," in *Proc. IEEE Annu. Int. Conf. Industrial Technology,* 2013, pp. 1976–1981.

[24] S. Karnouskos, A. W. Colombo, F. Jammes, J. Delsing, and T. Bangemann, "Towards an architecture for service-oriented process monitoring and control," in *Proc. 36th Annu. Conf. IEEE Industrial Electronics Society,* 2010, pp. 1385–1391.

[25] H. Derhamy, J. Eliasson, J. Delsing, and P. Priller, "A survey of commercial frameworks for the Internet of Things," in *Proc. Emerging Technology Factory Automation (ETFA), 2015*, pp. 1–8.

[26] T. Erl, *SOA Principles of Service Design*. Englewood Cliffs, NJ: Prentice Hall, 2007.

[27] J. Delsing, J. Eliasson, J. van Deventer, H. Derhamy, and P. Varga, "Enabling IoT automation using local clouds," in *Proc. World Forum—IoT 2016*, pp. 502–507.

[28] J. Delsing, Ed., *IoT Automation—Arrowhead Framework*. Boca Raton, FL: CRC, 2017.

[29] T. Erl, R. Puttini, and Z. Mahmood, *Cloud Computing: Concepts, Technology & Architecture*. Englewood Cliffs, NJ: Prentice Hall, 2013.

[30] Z. Shelby and C. Bormann, *6LoWPAN: The Wireless Embedded Internet*. Hoboken, NJ: Wiley, 2009.

[31] Z. Shelby, K. Hartke, and C. Bormann. (2014, June). The constrained application protocol (CoAP). Internet Engineering Task Force. Fremont, CA. [Online]. Available: https://tools.ietf.org/html/rfc7252

[32] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. ACM First Edition of the MCC Workshop on Mobile Cloud Computing,* 2012, pp. 13–16.

[33] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, Sept. 2015.

[34] H. Yang and M. Tate, "A descriptive literature review and classification of cloud computing research," *Commun. Assoc. Inform. Syst.*, vol. 31, no. 2, pp. 35–60, 2012.

[35] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc. MobiData 2015*, Hangzhou, China, pp. 37–42.

[36] R. L. Ackoff, "Towards a system of systems concepts," *Manage. Sci.*, vol. 17, no. 11, pp. 661–671, 1971.

[37] M. W. Maier, "Architecting principles for systems-of-systems," in *Proc. INCOSE Int. Symp.*, vol. 6, no. 1, 1996, pp. 565–573.

[38] J. Boardman and B. Sauser, "System of systems—The meaning of 'of,'" in *Proc. 2006 IEEE/SMC Int. Conf. System of Systems Engineering,* Apr. 2006, p. 6.

[39] C. Keating, R. Rogers, R. Unal, D. Dryer, A. Sousa-Poza, R. Safford, W. Peterson, and G. Rabadi, "System of systems engineering," *Eng. Manage. J.*, vol. 15, no. 3, pp. 36–45, 2003.

[40] A. W. Colombo, T. Bangemann, and S. Karnouskos, "A system of systems view on collaborative ind. automation," in *Proc. Annu. Int. Conf. Industrial Technology,* pp. 1968–1975.

[41] X. Xu. (2012). From cloud computing to cloud manufacturing. *Robot. Comput.-Integrated Manuf.* [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0736584511000949

[42] F. Tao, Y. Cheng, L. D. Xu, L. Zhang, and B. H. Li, "CCIoT-CMfg: Cloud computing and Internet of Things-based cloud manufacturing service system," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1435–1442, May 2014.

[43] Z. Bi, L. D. Xu, and C. Wang, "Internet of Things for enterprise systems of modern manufacturing," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1537–1546, May 2014.

[44] A. M. Haubenwaller and K. Vandikas. (2015). Computations on the edge in the Internet of Things. *Procedia Comput. Sci.* [Online]. Available: http://www.sciencedirect.com/science/article/pii/S187705091500811X

[45] F. Jalali, S. Khodadustan, C. Gray, K. Hinton, and F. Suits, "Greening IoT with fog: A survey," in *Proc. 2017 IEEE Int. Conf. Edge Computing (EDGE)*, pp. 25–31.

[46] H. Mueller, S. V. Gogouvitis, A. Seitz, and B. Bruegge, "Seamless computing for industrial systems spanning cloud and edge," in *Proc. 2017 Int. Conf. High Performance Computing and Simulation (HPCS)*, pp. 209–216.

[47] Architecture Working Group, www.OpenFog Consortium.org. (2017, Feb.). OpenFog reference architecture for fog computing. [Online]. Available: https://www.openfogconsortium.org/wp-content/uploads/OpenFog_Reference_Architecture_2_09_17-FINAL.pdf

[48] V. Kochar and A. Sarkar, "Real time resource allocation on a dynamic two level symbiotic fog architecture," in *Proc. 2016 6th Int. Symp. Embedded Computer and System Design (ISED)*, 2016, pp. 49–55.

[49] H. Kopetz and S. Poledna, "In-vehicle real-time fog computing," in *Proc. 2016 46th Annu. IEEE/IFIP Int. Conf. Dependable Systems and Networks Workshop (DSN-W)*, pp. 162–167.

[50] M. Mukherjee, R. Matam, L. Shu, L. Maglaras, M. A. Ferrag, N. Choudhury, and V. Kumar, "Security and privacy in fog computing: Challenges," *IEEE Access*, to be published.

[51] J. Wu, M. Dong, K. Ota, J. Li, and Z. Guan, "FCSS: Fog computing based content-aware filtering for security services in information centric social networks," *IEEE Trans. Emerg. Topics Comput.*, to be published.

[52] J. Delsing. (2016) Arrowhead framework wiki. [Online]. Available: https://forge.soa4d.org/plugins/mediawiki/wiki/arrowhead-f/index.php/Arrowhead_Framework_Wiki

[53] P. Varga, F. Blomstedt, L. L. Ferreira, J. Eliasson, M. Johansson, J. Delsing, and I. M. de Soria, "Making system of systems interoperable—The core components of the Arrowhead Framework," *J. Network Comput. Appl.*, vol. 81, pp. 85–95, Mar. 2016.

[54] S. Cheshire and M. Krochmal. (2013, Feb.). DNS-based service discovery. Internet Engineering Task Force. Fremont, CA. Tech. Rep. RFC 6763. [Online]. Available: https://datatracker.ietf.org/doc/rfc6763/

[55] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. (1999). X. 509 Internet public key infrastructure online certificate status protocol—OCSP. The Internet Society. Reston, VA. Tech. Rep. RFC 2560. [Online]. Available: https://www.ietf.org/rfc/rfc2560.txt

[56] A. DeKok and A. Lior. (2013, Apr.). Remote authentication dial-in user service (RADIUS) protocol extensions. Internet Engineering Task Force. Fremont, CA. Tech. Rep. RFC 6929. [Online]. Available: https://tools.ietf.org/html/rfc6929

[57] P. P. Pereira, J. Eliasson, and J. Delsing, "An authentication and access control framework for CoAP-based Internet of Things," in *Proc. 40th Annu. Conf. IEEE Industrial Electronics Society,* 2014, pp. 5293–293.

[58] M. Albano, P. Barbosa, J. Silva, R. Duarte, L. Lino Ferreira, and J. Delsing, "Quality of service on the Arrowhead Framework," in *Proc. 13th IEEE Int. Workshop Factory Communication Systems (WFCS 2017),* pp. 1–8.

[59] H. Derhamy, J. Eliasson, and J. Delsing, "IoT interoperability—On-demand and low latency transparent multi-protocol translator," *IEEE Internet Things J.*, to be published.

[60] M. Martisch, C. Lesjak, and A. Aldrian, "Enabling smart maintenance services: Broker-based equipment status data acquisition and backend workflows," in *Proc. 2016 IEEE 14th Int. Conf. Ind. Informatics (INDIN)*, pp. 699–705.

[61] C. Hegedűs, D. Kozma, G. Soós, and P. Varga, "Enhancements of the Arrowhead Framework to refine inter-cloud service interactions," in *Proc. 42nd Annu. Conf. IEEE Industrial Electronics Society,* 2016, pp. 5259–5264.

[62] E. Rescorla and N. Modadugu. (2006, Apr.). Datagram transport layer security RFC—RFC 4347. Internet Engineering Task Force. Fremont, CA. [Online]. Available: https://tools.ietf.org/html/rfc4347

[63] S. Jacobs, *Engineering Information Security: The Application of Systems Engineering Concepts to Achieve Information Assurance*. Hoboken, NJ: Wiley, 2015.

[64] O. Carlsson, C. Hegedűs, J. Delsing, and P. Varga, "Organizing IoT systems-of-systems from standardized engineering data," in *Proc. Annu. Conf. IEEE Industrial Electronics Society,* 2016, pp. 5277–5282.

[65] O. Carlsson, P. Punal Pereira, J. Eliasson, J. Delsing, B. Ahmad, R. Harrison, and O. Jansson, "Configuration service in cloud based automation systems," in *Proc. Annu. Conf. IEEE Industrial Electronics Society,* 2016, pp. 5238–5245.

[66] H. Gadd and S. Werner. (2013). Daily heat load variations in Swedish district heating systems. *Appl. Energy.* [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261913000391