Surveying the Features of Industrial SOAs

Ahmed Ismail, Wolfgang Kastner
Institute of Computer Aided Automation - Technische Universität Wien
Vienna, Austria
Email: {aismail, k}@auto.tuwien.ac.at

***Abstract*—Over the previous years, many service-oriented (SO) solutions have been proposed by European research projects for the technological advancement of industrial systems. These projects typically include a software reference architecture (RA) based on the concepts of service-oriented architectures (SOA) and, often-times, an accompanying technology stack to guarantee system-wide interoperability. In this paper, we survey and outline the specific approaches, standards, and specifications applied by five such projects in their pursuit of SO solutions to specific manufacturing and production problems.**

I. Introduction

Industrial enterprises are well-known for the heterogeneity of their technological landscapes. Integration in such environments typically carries large engineering costs. The SO paradigm attempts to reduce the needed efforts by employing design features and patterns specifically geared towards the development of flexible, agile, and manageable systems of interoperable and reusable components. For example, service-orientation grounds itself in the concepts of functional decomposition, where a large problem is broken down into a number of smaller ones. These units may then be addressed using equally small solutions instead of a single monolithic application. The encapsulated modules of logic are designed in a functionally agnostic manner with clearly defined interfaces and boundaries. This ensures that the resulting modules may be reused and/or composed into new services without the need for complex binding measures. The same features also afford the technological independence of modules, whereby the internal implementations of each module are not the concern of the system. As such, the design of integration solutions in a SO manner may reduce the future costs involved in the integration of new technologies and in system evolution [1, 2, 3].

According to [4], projects have been researching the application of SO architecture (SOA) based solutions to industrial problems since 2003. However, we limit our discourse to the more recent and completed projects, limiting our timeline to the period of 2010 to 2017. The purpose of this review is to provide an overview of recent trends in the characteristics and technology choices of research-based SO reference architectures for the industrial domain. Although we refrain from including ongoing research projects, as published information is subject to change, we make a single exception for the Arrowhead framework as it has 77 partners and a budget of 69 million Euros, making it one of the largest European research projects in the field of automation [4]. In total, five projects are surveyed, the Internet of Things at Work (IoT@Work), Production Logistics and Sustainability Cockpit (PLANTCockpit), ArchitecturE for Service-Oriented Process - Monitoring and Control (IMC-AESOP), Embedded systems Service-based Control for Open manufacturing and Process automation (eScop), and Arrowhead framework projects.

Our analysis of said five projects is done from two opposing perspectives to determine the ease with which they may be translated into concrete architectures and the technologies that may be used in such realisations. Section II embarks on establishing the first of these perspectives by applying a software reference architectures analysis framework developed in [5]. Section III then proceeds to provide the second, more technical and detailed perspective, by extracting the specific protocols, standards and specifications used to define the various parts of the architectures' communication stacks. Together, a succinct overview should be able to provide a baseline understanding of the features of the respective architectures.

II. Comparison of Service-Based Projects

We begin by summarising the analysis framework developed in [5]. This framework is made up of two components. The first defines a classification method for categorising architectures based on context, goals, and design. These factors are addressed using a set of interrogatives for which there are a select number of possible values, some of which are mutually exclusive. The factors, interrogatives, and architectural attributes are summarized in Table I [5].

The second element of the framework is in fact a set of five architectural templates and two sub-types that define preset compositions of values from the elements of Table I. An architecture that shares the same attributes as a category is said to belong to its type. The type most relevant to our forthcoming analysis is type 5. This is due to it being the only one for the development of RAs for facilitation based on preliminary technologies through collaborations between research centres and industrial partners, thereby making it the most applicable to the reference architectures reviewed in the coming pages. The combination of values for the type 5 category and the degree to which the five reference architectures match is shown in Table II.

As may be noted from Table II, all of the architectures reviewed are facilitation RAs, include preliminary technologies, and are designed by partnerships between research, industrial software design, and user organisations for application in multiple organisations. In all of the remaining sub-dimensions, however, nearly all of the RAs have divergent properties.

To begin with, we note certain RAs defining more than necessary or missing one or more of the required elements in the D1 dimension. The IoT@Work, eScop, and Arrowhead fit the former description, while IMC-AESOP[1] and PLANTCockpit are of the latter type.

As for the D2 dimension, this is notably one of the more difficult dimensions for classification, with the framework's authors themselves noting that the classification technique applied being subjective and therefore inherently imprecise. However, the deviations noted for IMC-AESOP and PLANTCockpit are irrefutable as certain elements needed by the D2 dimension are not defined at all by the respective RAs. As for the remainder of the RAs, all have detailed specifications for their components, and all but eScop detail their algorithms and protocols as well.

For the D3 dimension, IMC-AESOP defines its architecture in a completely abstract manner, while the remaining four all specify concrete elements in their architectures [6].

Finally, in terms of representation, IMC-AESOP, eScop and Arrowhead all use semi-formal techniques; the first uses the Fundamental Modelling Concepts (FMC) graphical notation, the second the Unified Modelling Language (UML), and the last the Systems Modelling Language (SysML) [7, 8]. As for IoT@Work and PLANTCockpit, neither architecture explicitly declares its use of any specification for representation.

III. Communication Stacks for Interoperability

To ensure the interoperability of the architectural components, the specification of the communication stack is a necessity. This includes the exact definition of the service discovery, service description, data representation, information and message encoding, message exchange, network, media and data link layer, and security layers of the communication stack. These aspects are addressed separately within the remainder of this paper, with the specifications and approaches used by the various architectures stated for each layer. Since not all of the projects specified a stack as part of their architectures, we supplement the information provided with data extracted from the prototypical implementations, pilot demonstrators, and publications. Note that, to conserve space, the network, media and data link layer are presented predominantly in tabular form as Table III.

A. Service Discovery

In three of the five architectures, the web service dynamic discovery (WS-Discovery) is used, excepting eScop and Arrowhead. However, eScop appears to base its design of a discovery protocol on WS-Discovery, while the Arrowhead framework follows a completely different approach by predominantly using the Domain Name System (DNS).

The implementation of the WS-Discovery protocol in IMC-AESOP appears in three different forms. The first involves its use for the discovery of Service Bus instances. The remaining two use WS-Discovery to augment the discovery mechanisms of the OPC Unified Architecture (OPC UA). One approach uses WS-Discovery to allow OPC UA clients and servers to dynamically resolve the address of the OPC UA discovery server. The second has the OPC UA discovery service completely replaced by the WS-Discovery protocol, allowing OPC UA clients and servers to directly discover each other. For resource discovery in constrained environments, however, IMC-AESOP relies on the mechanisms of the Constrained Application Protocol (CoAP), CoAP multicast, and the Constrained RESTful Environments (CoRE) Resource Directory (RD) [9, 10, 11, 12].

In the case of PLANTCockpit, the architecture uses adapters to interface with systems. The discovery protocols in use are therefore dependent on the implemented adapters. An example provides a DPWS adapter, which should include the WS-Discovery protocol, to interface with DPWS devices. As for discovery between the internal components of the PLANTCockpit system, since the system is said to be implemented using the OSGi framework, it may be based on the OSGi service registry; however, this is not explicitly stated, and no other alternative is presented in publicly available materials [13, 14].

The IoT@Work project, like IMC-AESOP, also used the discovery mechanisms of OPC UA and WS-Discovery; however, it did so within the context of auto-configuration. For service discovery, IoT@Work relied on a RESTful Directory Service with interactions taking place using HTTP requests [15].

As for eScop, the project appears to base its discovery mechanisms on WS-Discovery as it uses multicast Hello, Bye and Probe messages [16]. However, closer inspection of published code[1] shows divergences from the WS-Discovery specification, including the use of multicast IP addresses and ports different than those for WS-Discovery.

Finally, the Arrowhead project uses DNS and DNS Service Discovery (DNS SD) for the implementation of service registry functions. For constrained devices, this system is extended to include multicast DNS (mDNS). However, Arrowhead also supports the use of XML-over-HTTP and JSON-over-HTTP for the discovery of RESTful services. The XML-over-HTTP implementation uses an ARROWHEAD-specific DNS protocol. Details on the JSON-over-HTTP protocol are not yet present as an implementation is still to be made public [17, 18, 19, 20].

B. Service Description

Here, a number of projects again use web services specifications, applying them for the definition of service descriptions and service contracts. The Web Services Description Language (WSDL) is used by both IMC-AESOP and PLANTCockpit. eScop, having RESTful interfaces, opts to use the Swagger specification for the creation of human and machine-readable service descriptions. IoT@Work, as previously mentioned, explored the use of WS-Discovery and OPC UA for auto-configuration. It is therefore able to use WS-Discovery and

[1]Please note that we consider the IMC-AESOP RA to be limited to Chapter 3 of [6].

[1]http://www.escop-project.eu/tools/

TABLE I
SUMMARY OF THE APPLIED ANALYSIS FRAMEWORK [5]

Factor	Dimension	Value	Explanation
Goal	G1: Why [2]	Standardisation	Interoperability-focused concrete architectures.
		Facilitation	Guidelines for the design of concrete architectures.
Context	C1: Where [2]	Single Organisation	The architecture is developed to standardise or facilitate the development of software for a single enterprise.
		Multiple Organisations	The architecture is to be used by several organisations that have a common property (industrial domain, technological constraints etc.)
	C2: Who	Software organisations	Apply the reference architecture; requirements providers that may also be involved in the design of the reference architecture.
		User organisations	Users of the resulting software; requirements providers.
		Independent organisations	Refers to research, standardisation, non-governmental and other organisations that do not implement or use the resulting software solutions.
	C3: When [2]	Preliminary	The technologies required for the application of the reference architecture exist only as research experiments, proof-of-concepts, or are partially or completely absent at the time that the architecture is defined.
		Classical	The technologies required for the application of the reference architecture are mature at the time that the architecture is designed.
Design	D1: What	(See explanation)	The element types defined. Possible options are "components and connectors, interfaces, protocols, algorithms, and policies and guidelines" [5]
	How	D2: Detail	Detailed: Many elements belonging to three or more aggregation levels
			Aggregated: A single aggregation level containing a small number of elements.
			Semi-detailed: Between detailed and aggregated.
		D3: Concreteness	Abstract: General definitions of architectural elements.
			Semi-concrete: Specific definition of a selection of options for every component of the architecture.
			Concrete: Defines the option to be used from the selection for every element in the architecture.
		D4: Representation	Informal: The architecture is defined using ambiguous graphical notation and/or through natural language.
			Semi-formal: Uses a non-ambiguous graphical notation that is not based on a mathematical foundation.
			Formal: Uses clearly-defined formal specifications based on mathematical techniques.

Footnotes: [1]Possible values for these dimensions are mutually exclusive.

TABLE II
ATTRIBUTES OF A TYPE 5 REFERENCE ARCHITECTURE [5]
AND DEGREE OF MATCH OF THE ANALYSED REFERENCE ARCHITECTURES

Category	Type 5	IoT@Work	PLANTCockpit	IMC-AESOP	eScop	Arrowhead
G1:	Facilitation	X	X	X	X	X
C1:	Multiple organisations	X	X	X	X	X
C2:	Research Centers (D) Software design organizations (D, R) User Organisations (R)	X	X	X	X	X
C3:	Preliminary	X	X	X	X	X
D1:	Components, algorithms, protocols	≈	≈	≈	≈	≈
D2:	Detailed/Semi-detailed: components, algorithms. Aggregated/semi-detailed: protocols.	≈	≈	≈	≈	≈
D3:	Abstract	-	-	X	-	-
D4:	Formal/Semi-formal	-	-	X	X	X

Notation: X is a match, ≈ means deviations exist, and - means it does not match [5].

WS-MetadataExchange to communicate addressing information, policies, WSDL definitions and other details using WS-Transfer. For OPC UA, IoT@Work relies on the standard's GetEndpoints service to acquire information relevant to the creation of secure communication channels between nodes. Finally, in the case of Arrowhead, the DNS system structures information using the DNS-SD specification, while the XML system uses the Web Application Description Language (WADL) [21, 7, 22, 23, 24, 18, 19]

C. Data Representation and Access

For information representation, only the IMC-AESOP and Arrowhead projects use mature standards for their models. The former uses OPC UA for the representation of data in the upper layers of the enterprise and an augmented form of SenML, modified for increased granularity, for the lower layers [7].

Like IMC-AESOP, Arrowhead highlights the OPC UA and SenML standards as possible solutions for data representation. However, Arrowhead also discusses the use of the Home Performance XML standard and the Constrained RESTful Environments (CoRE) Link Format. The divergence of pilot implementations from the frameworks' specification are noted in the energy production demonstrator pilot where the Thing Markup Language (ThingML) is used instead of the above specifications [25, 26].

The eScop and IoT@Work projects favour an ontology-driven approach. eScop uses the proprietary Manufacturing System Ontology (MSO); an evolved form of the Politecnico di Milano Production Systems Ontology (P-PSO) taxonomy that was originally designed for discrete manufacturing systems [27]. The latter, IoT@Work, develops an ontology inspired by the uCode Relation Model for information representation in its Directory Service. It is important to note, however, that IoT@Work also supports the OPC UA address space and information model and the SNMP MIB data model. The former is used for IoT@Work-compliant devices while the latter allows the system to interface with SNMP devices [15, 28].

Finally, PLANTCockpit creates a meta-model for a database schema for the storage of various data types. It also has an XML schema for the persistence of configurations needed for the visualisation of system data [29].

D. Information Encoding

For message encoding, all architectures support the XML format and all but PLANTCockpit support the JSON specification. For compactness, the Efficient XML Interchange (EXI) specification is used by the IMC-AESOP and ARROWHEAD projects, XML-binary Optimized Packaging and Message Transmission Optimization Mechanism (XOP/MTOM) by IMC-AESOP, and OPC UA Binary by IMC-AESOP, IoT@Work, and, possibly, PLANTCockpit as well as part of its OPC UA adapter. However, none of the projects employ a binary format for JSON, with Arrowhead's historian being the only project describing the inclusion of CBOR as a long-term goal [30, 25, 7, 31, 15, 32, 33, 34, 35, 36].

Other encodings employed include HTML, Security Assertion Markup Language (SAML), eXtensible Access Control Markup Language (XACML), and YAML. The first, HTML, is for the structuring of web content and is explicitly stated as parts of the IoT@Work and PLANTCockpit implementations. SAML and XACML, on the other hand, are used by IoT@Work for authentication, authorisation, and the definition of security policies. YAML, finally, is used by eScop for the definition of Swagger files [15, 33, 37].

E. Message Exchange

Message exchange in the various architectures is done using web-based specifications, message-oriented middleware (MOM), and portions of the OPC UA communication stacks. Aside from DPWS, which has already been covered in the previous chapters, another Internet specification, HTTP is used by IoT@Work for its directory service, and HTTP/HTTPS is used in IMC-AESOP and ARROWHEAD for the transport of SOAP messages. For constrained systems, IMC-AESOP and ARROWHEAD use the CoAP protocol. As for the employed MOMs, we note the use of the Java Message Service (JMS) by PLANTCockpit, AMQP by IoT@Work, and, depending on the demonstrator, XMPP and/or MQTT in Arrowhead. Finally, to address the OPC UA portions of the stack, both IMC-AESOP and IoT@Work implement the native UA Binary profile, which is a combination of UA Binary, UA-SecureConversation, and UA-TCP. IMC-AESOP also implements the UA web services profile which uses UA XML, SOAP and HTTPS. The Arrowhead project also applies OPC UA, as it uses the OPC UA SDK from Unified Automation in a condition monitoring proof of concept. In the case of PLANTCockpit, although it describes OPC UA as a core technology, unfortunately, implementation details are not present in the public deliverables [25, 15, 7, 21, 36, 30, 28, 38, 39, 24, 10, 40, 41].

F. Networking, Data Link and Media

As previously mentioned, the majority of the networking, media, and data link standards and supporting application layer services are listed in Table III. It may be noted that the eScop and PLANTCockpit projects are not included in this table. This is because neither one explicitly defines these layers in available materials. We refrain from making inferences as the underlying assumptions may prove to be inaccurate.

G. Security

For security, the IoT@Work project introduces a capability-based access control mechanism. In this context, capabilities are XML documents composed of elements from the SAML, XACML, Digital Signature and XML encryption specifications. These capabilities can be created and revoked to allow or disallow a process from interacting with a specific element. IoT@Work also uses IEEE 802.1AR secure identifiers and follows the IEEE 802.1X standard for network access control [47, 15, 48].

In the case of PLANTCockpit, an LDAP service is used as a single-sign on (SSO) solution both for access control

TABLE III
THE NETWORKING, MEDIA & DATA LINK STANDARDS AND SUPPORTING APPLICATION SERVICES [25, 42, 43, 44, 45, 7, 15, 46, 28]

Layer	IMC-AESOP	IoT@Work	Arrowhead Framework
Application Services	NTP, IEEE 1588 PTP	NTP, IEEE 1588 PTP, IEEE 802.1Q, SNMP, DHCP, DNS, LLDP, STP	NTP
Networking	IPv4, IPv6	IPv4, IPv6	IPv4, IPv6
Media & Data Link	6LoWPAN, IEEE 802.15.4, IEEE 802.11, RS-485 Modbus, Profibus	NFC, QR, Profinet	6LoWPAN, IEEE 802.15.4, IEEE 802.11p, NFC, UWB, GSM, GPRS, UMTS, RS-485 CAN

and user rights management. As for message encryption in PLANTCockpit's JMS, this is claimed to be carried out by a central component. PLANTCockpit also states that JMS security, in general, is handled by what it terms 'interceptors'. Unfortunately, more information is not available as the majority of its security aspects are described in a private deliverable (D3.2) [49, 50, 30].

The Arrowhead framework, on the other hand, details its features for security in discovery, authorisation control, legacy systems mediation, and communication. In discovery, DNS updates and queries are secured using DNS TSIG keys and DNS Security Extensions (DNSSEC). Authorisation Control and secure communication is carried out using X.509 certificates, TLS/DTLS for TCP/UDP respectively, and a Public Key Infrastructure (PKI). For mediation in legacy systems, the project pursues the development of a secure NFC system. This may be the 'ESTADO' system; a modular device used to supplement legacy devices with a contemporary and secure platform. Further security features are noted in an implementation of 3D swarming systems, where communications use Message Passing Interface (MPI) with Secure Shell (SSH) [18, 51, 25, 52, 53, 54, 55].

As for IMC-AESOP and eScop, security is largely neglected. IMC-AESOP limits its security capabilities to HTTP basic authentication and Role-Based Access Control (RBAC) for service calls. eScop on the other hand, only appears to include input sanitisation measures and to require testing to ensure data persistence and system misuse avoidance [8, 56].

IV. CONCLUSION

In this paper, we address several current research projects for SOAs in the industrial domain. These projects are surveyed from two perspectives to provide the reader with an understanding of the features of the various RAs and the technology stacks that were envisioned by the creators as feasible candidates for their implementation. What is apparent from the first perspective is that the reviewed projects either over or under-specify their RAs, and at times miss critical features required for the creation of concrete implementations. The authors of [5] note that a lack of congruence between a RA and its category makes the RA vulnerable to low adoption rates and criticisms by stakeholders. Due to the lack of reporting on adoption rates we refrain from making the same conclusion. However, the authors of [5] also note that the presence of ambiguities in RAs, such as the informal representation of components, leads to a need for additional documents to clarify the architecture. We provide those interested with the second perspective, the technology stacks, extracted from numerous publications, deliverables, and other materials in the hopes that it may assist in achieving such clarity. In certain cases, where the implementations did not address or make available certain layers of their technology stacks, extra effort may still be necessary in the carrying out of concrete implementations. With these points in mind, we hope that this paper may serve as a guide in the design of future SO RAs in industrial applications.

ACKNOWLEDGMENT

This paper is supported by TU Wien research funds.

REFERENCES

[1] M. Valipour et al. "A brief survey of software architecture concepts and service oriented architecture". In: *IEEE ICCSIT*. 2009.

[2] T. Erl. *SOA design patterns*. 1st ed. Prentice Hall service-oriented computing series from Thomas Erl. Upper Saddle River, NJ: Prentice Hall, 2009.

[3] T. Erl et al. *Next generation SOA: a concise introduction to service technology & service-orientation*. Pearson Education, 2014.

[4] K. Nagorny et al. "A survey of service-based systems-of-systems manufacturing systems related to product life-cycle support and energy efficiency". In: *IEEE INDIN*. 2014.

[5] S. Angelov et al. "A framework for analysis and design of software reference architectures". In: *Information and Software Technology* 54.4 (2012), pp. 417–431.

[6] PLANTCockpit Consortium. *D3.1 Initial Architectural Components of PLANTCockpit*. 2011.

[7] A. Colombo et al., eds. *Industrial Cloud-Based Cyber-Physical Systems*. Cham: Springer International Publishing, 2014.

[8] eScop Consortium. *D2.4 General specification and design of eScop reference architecture*. 2014.

[9] P. Nappey et al. "Migration of a legacy plant lubrication system to SOA". In: *IEEE IECON*. 2013.

[10] B. Bony et al. "Convergence of OPC UA and DPWS with a cross-domain data model". In: *IEEE INDIN*. 2011.

[11] J. Eliasson et al. "A SOA-based framework for integration of intelligent rock bolts with internet of things". In: *IEEE ICIT*. 2013.

[12] R. Kyusakov et al. "EXIP: a framework for embedded Web development". In: *ACM TWEB* 8.4 (2014).

[13] S. Iarovyi et al. "An approach for OSGi and DPWS interoperability: Bridging enterprise application with shop-floor". In: *IEEE INDIN*. 2013.

[14] A. Dennert et al. "Advanced Concepts for Flexible Data Integration in Heterogeneous Production Environments". In: *IFAC Proceedings Volumes* 46.7 (2013).

[15] D. Rotondi et al. *D1.3 - Final Framework Architecture Specification*. 2013.

[16] S. Iarovyi et al. "Cyber–Physical Systems for Open-Knowledge-Driven Manufacturing Execution Systems". In: *IEEE JPROC*. Vol. 104. 5. 2016.

[17] J. Delsing. "Building Automation Systems from Internet of Things". In: IEEE ETFA Keynote Presentation, Luxembourg, 2015. URL: www.etfa2015.org/uploads/docs/ETFA_2015_keynote_JD.pdf (visited on 08/21/2016).

[18] F. Blomstedt et al. *Service Discovery DNS-SD-TSIG-SPDNS Version 1.3*. 2016.

[19] F. Blomstedt. *Service Discovery REST_WS-XML-SPSDTR Version 1.1*. 2016.

[20] F. Blomstedt et al. *Service Discovery REST_WS-JSON-SPSDTR Version 1.1*. 2016.

[21] PLANTCockpit Consortium. *Technical Report - External Interface Specification, First Draft*. 2012.

[22] J. Faist et al. "Webservice-Ready Configurable Devices for Intelligent Manufacturing Systems". In: *IFIP APMS 2015*. Cham: Springer International Publishing, 2015.

[23] L. Dürkop et al. "Service-oriented architecture for the autoconfiguration of real-time Ethernet systems". In: *KommA*. 2012.

[24] W. Mahnke et al. *OPC Unified Architecture*. Springer Berlin Heidelberg, 2009.

[25] P. Varga et al. *Deliverable D7.3 of WP 7*. 2014.

[26] *WP 4 —T4.1 Demonstrator (1st Generation): The Safe Home*. 2014.

[27] E. Negri et al. "Ontology for Service-Based Control of Production Systems". In: *Advances in Production Management Systems: Innovative Production Management Towards Sustainable Growth*. Vol. 460. Cham: Springer International Publishing, 2015, pp. 484–492.

[28] J. Imtiaz et al. *D. 2.5 - Integrated Secure Plug&Work Framework*. 2013.

[29] PLANTCockpit Consortium. *Technical Report - Data and Process Model, First Draft*. 2012.

[30] PLANTCockpit Consortium. *Technical Report - Generic Alarms and Events Data Format*. 2012.

[31] J. Imtiaz et al. "Scalability of OPC-UA down to the chip level enables "Internet of Things"". In: IEEE INDIN, 2013.

[32] E. Negri et al. "Requirements and languages for the semantic representation of manufacturing systems". In: *Computers in Industry* 81 (2016), pp. 55–66.

[33] PLANTCockpit Consortium. *PLANTCockpit White Paper*. 2012.

[34] D. Rotondi et al. *D1.1 - State of the Art and Functional Requirements in Manufacturing and Automation*. 2010.

[35] eScop Consortium. *Orchestration Layer Training Material*.

[36] J. Eliasson. "Arrowhead Historian System". In: Presented at the Arrowhead Budapest Meeting, 2015.

[37] S. Iarovyi et al. "Representation of manufacturing equipment and services for OKD-MES: from service descriptions to ontology". In: *IEEE INDIN*.

[38] A. Skou et al. *Deliverable D5.3 of WP 5*. 2014.

[39] H. Derhamy. "Arrowhead Transparency - Protocol Translation". In: Presented at the Arrowhead Budapest Meeting, 2015.

[40] C. Le Pape et al. *Deliverable D1.3 of WP1*. 2014.

[41] D. Hästbacka et al. "Device status information service architecture for condition monitoring using OPC UA". In: *IEEE ETFA*. 2014, pp. 1–7.

[42] P. Bellavista et al. *Arrowhead D3.3 Appendix 3.3b PO 3.3-002 and PO 3.3-004 of Task 3.3 Details about First Generation Pilot Results*. 2014.

[43] Arrowhead Consortium. *WP3.1.2 PO 002 - Requirements definition - Communication and services*. 2014.

[44] S. Bocchio et al. *Arrowhead D3.3 Appendix 3.1.1.c PO 002 of Task 3.1.1 Task and concepts*. 2014.

[45] D. Kleyko. *System-of-Systems Design (SoSDD) LTU Core Framework Description*. 2016.

[46] A. Houyou et al. *D2.2- Bootstrapping Architecture*. 2011.

[47] S. Gusmeroli et al. "A capability-based security approach to manage access control in the Internet of Things". In: *Mathematical and Computer Modelling* 58.5-6 (2013), pp. 1189–1205.

[48] K. Fischer et al. "Security architecture elements for IoT enabled automation networks". In: *IEEE ETFA*. 2012.

[49] PLANTCockpit Consortium. *Technical Report - Persistency and Synchronization Model*. 2011.

[50] PLANTCockpit Consortium. *Project Final Report*. 2014.

[51] F. Blomstedt et al. *Orchestration System Version 1.1*. 2016.

[52] C. Chrysoulas et al. *Arrowhead System Description (SysD) - Translation System Version 0.4*. 2016.

[53] J. Krimm et al. *DELIVERABLE D8.2 of Work Package 8: Common Service Framework - Generation 1 Version 1.1*. 2014.

[54] C. Lesjak et al. "ESTADO—Enabling smart services for industrial equipment through a secured, transparent and ad-hoc data transmission online". In: *IEEE ICITST*. 2014.

[55] G. Sadrollah et al. "A distributed framework for supporting 3D swarming applications". In: *IEEE ICCOINS*. 2014.

[56] P. Simone et al. *D6.1 Test strategy*. 2014.