Modeling an Industrial Revolution: How to Manage Large-Scale, Complex IoT Ecosystems?

Géza Kulcsár
IncQuery Labs
Budapest, Hungary
geza.kulcsar@incquerylabs.com

Pál Varga
University of Technology and Economics
Budapest, Hungary
pvarga@tmit.bme.hu

Marek S. Tatara
Electronics, Telecommunications and Informatics
Gdańsk University of Technology, Poland
martatar@pg.edu.pl

Federico Montori
Department of Computer Science and Engineering
University of Bologna, Italy
federico.montori2@unibo.it

Michel A. Iñigo
Innovation and Technology Department
MONDRAGON Corporation, Spain
minigo@mondragoncorporation.com

Gianvito Urgese
Urban Studies and Planning
Politecnico di Torino, Italy
gianvito.urgese@polito.it

Paolo Azzoni
Eurotech Group
Udine, Italy
paolo.azzoni@eurotech.com

***Abstract*—Advancements around the modern digital industry gave birth to a number of closely interrelated concepts: in the age of the Internet of Things (IoT), System of Systems (SoS), Cyber-Physical Systems (CPS), Digital Twins and the fourth industrial revolution, everything revolves around the issue of designing well-understood, sound and secure complex systems while providing maximum flexibility, autonomy and dynamics.**

The aim of the paper is to present a concise overview of a comprehensive conceptual framework for integrated modeling and management of industrial IoT architectures, supported by actual evidence from the Arrowhead Tools project; in particular, we adopt a three-dimensional projection of our complex engineering space, from modeling the engineering process to SoS design and deployment.

In particular, we start from modeling principles of the the engineering process itself. Then, we present a design-time SoS representation along with a toolchain concept aiding SoS design and deployment. This brings us to reasoning about what potential workflows are thinkable for specifying comprehensive toolchains along with their data exchange interfaces. We also discuss the potential of aligning our vision with RAMI4.0, as well as the utilization perspectives for real-life engineering use-cases.

***Index Terms*—digital twin modeling, industrial IoT design, system-of-systems modeling**

I. Introduction

Advancements around the modern digital industry gave birth to a number of closely interrelated concepts: in the age of the *Internet of Things (IoT), System of Systems (SoS), Cyber-Physical Systems (CPS), Digital Twins* and the fourth industrial revolution, everything revolves around the issue of designing well-understood, sound and secure complex systems while providing a maximum (ideally, an infinite level) of flexibility, autonomy and dynamics [15].

The grand challenge for Industry 4.0 was set by RAMI4.0 [1], which already suggested the generic reference architecture as well. This made the idea of digital twins widely required in the industrial domains [14], which then get shaped further for design and production engineering [13]. Using digital twins in model-based systems engineering is suggested by [11], although this and similar approaches merely focus on a part of the engineering process—whereas our current paper introduces a broader modeling concept that is based on current standards and engineering best practices.

The present paper revolves around a central observation that while the concept of digital twins became central to advancing the fourth industrial revolution, *its realization depends upon an adequate combination and integration of IoT modeling and management approaches*: while a digital twin is essentially a model of some IoT network elements, this model has to be brought into connection with the actual network (and SoS) operation and management aspects. We can shortly refer to this complex field of study as IoT network design.

In the present paper, we formulate an expert position on some (arguably central) aspects of the grand challenge of *design in industrial IoT*. Thereby, we shortly review some major facets of industrial IoT engineering, with a comprehensive, holistic mindset, yet also based on our concrete experience in the *Arrowhead* ecosystem, a large-scale industrial IoT endeavor of the European Union [2].[1] In such a huge and relevant research complex, analyzing and capturing the best (i.e., most useful) abstractions is unavoidable—but poses a challenge on its own. This paper represents both a (potential, non-exclusive) answer to that challenge from a certain perspective, and also serves as a preliminary reference for a comprehensive, integrated SoS engineering framework, integrating modeling

[1] https://www.arrowhead.eu/

and management aspects.

As a gluing concept of such a vast industrial landscape, *interoperability* has recently emerged as a central abstract notion of conceptual approaches to Industry 4.0 and industrial IoT architecture and modeling. In this context, the *Arrowhead* initiative focuses on fostering interoperability via service and interface descriptions between systems and components in modern SoS scenarios [4], along the principles of *Service-Oriented Architectures* (SOA). Indeed, the Arrowhead community originates from a joint European effort of more than 80 industrial and academical partners with the goal of bridging the interoperability gaps for applications and tools in IoT-based automated industrial scenarios.

The aim of this paper is to present a concise overview of a comprehensive conceptual framework for SoS modeling and management, supported by actual evidence from the *Arrowhead Tools* project; in particular, we focus on three dimensions of our complex engineering space, starting from modeling the engineering process itself (Sect. II), we present a design-time SoS representation (Sect. III) along with a *toolchain* concept aiding SoS design and deployment (Sect. IV). Finally, we reason about the potential of aligning our vision with RAMI4.0 (Sect. V).

The main contribution of the present paper is the unified conceptual framework emerging as an integration of the particular aspects of the Arrowhead ecosystem; moreover, we demonstrate multiple direct connections from the concept space to the actual technical realization of industrial IoT installments built upon Eclipse Arrowhead.[2]

II. The Arrowhead Engineering Process

In the context of the life-cycle management of IoT ecosystems, the possibility to use virtual copies of sensors, actuators, more complex devices and even entire systems is a great challenge that can potentially revolutionise the approaches to product design, development, manufacturing and operations, through the adoption of a digital mirror of the IoT infrastructure that extends also to the full engineering process. The Arrowhead Engineering Process model (AHT-EP) [16], shown in *Fig. 1*, is an emerging flexible solution to support the definition/modelling of engineering processes adopted in different industrial multi-stakeholder use cases. The AHT-EP is composed of eight phases, that cover the full life-cycle of an IoT ecosystem, and that could be fully virtualised to take advantage of the digital-twin concept:

1) During the **Requirements** elicitation phase, the stakeholders cooperate to identify the requirements of the components (HW and SW) composing the IoT ecosystem.
2) In the **Functional design** phase the stakeholders develop the cyber-physical models of the components and their functionalities to be subsequently simulated and validated. During the life-cycle of the product these models will be continuously used to represent and simulate behaviours of the components in the digital twin of the engineering process phases.
3) In the **Procurement & Engineering** phase, the stakeholders select and acquire from external sources the goods and services required to engineer the product and manufacture it. During the selection, it is important that each component of the product and each part of the EP assigned to an external service have a digital model, to ensure the completeness of AHT-EP digital twin. In this phase, the engineering teams design, develop, and test the product, generating a prototype and, after some iterations with refinements, bugs corrections, updates, etc. the final version of the product. Moreover, the engineering teams set up the simulation framework that will continuously support the simulation of the digital twin of the engineering process.
4) In the **Deployment & Commissioning** phase, the stakeholders will install and integrate the product in the final operative environment. Once installed, a product identifier (e.g. a serial number) is associated to a owner/user id and stored in a database that will be accessed during the monitoring and simulation of the system. In the commissioning phase a stakeholder will be responsible for assuring that the product is designed, installed, tested, operated, and maintained according to the operational requirements of the final client.
5) In the **Operations & Management** phase, one of the stakeholders will monitor the data streams coming from the operating product and will be able to explore the status and future behaviours of the product, introducing real data in its digital twin. In case the simulated model presents deviations from the normal behaviour the exploration continues, without any impact on the real product, until a normal behaviour is identified. With this approach, the real system continues to operate according to the operational specification. Moreover, in case the digital version of the product predicts warnings or errors, a session of predictive maintenance can be planned in advance.
6) In the **Maintenance, Decommissioning & Recycling** phase, the stakeholders will perform ordinary and predictive maintenance to achieve, restore, and maintain operational capability of the system. Maintenance intervention are reported in details, all the modifications done on the real product are applied also on the digital version, in order to ensure the high fidelity of the digital twin of the product. In this phase, we also consider the decommissioning of the product at end-of-life and the recycling procedure required to reduce the impact on the environment.
7) In the **Evolution** phase, all the information collected in the "operation & management" and "maintenance" phases are analysed to identify solutions to faults/bugs, define the necessary updates and identify improvements that could bring to new product releases. The digital twin is fundamental to simulate and explore the effects of these updates and new releases. This will ensure the continuous evolution of the product, always respecting the user requirements in an efficient, reliable and flexible way.
8) In the **Training & Education** phase, all the education and professional training activities required by the engineering process, across the entire product life-cycle, will be defined.

[2]https://projects.eclipse.org/projects/iot.arrowhead

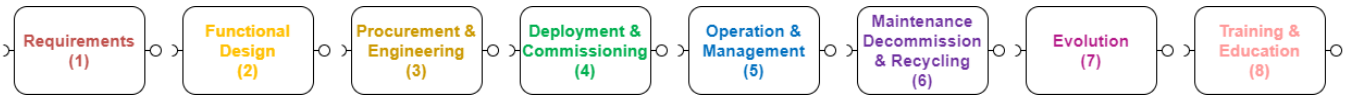

Fig. 1. Arrowhead Engineering Process: *AHT-EP*

Source code documentation, how-to, installation manuals and training courses, together with demonstrators and development kits that use the power of the digital twin, will be provided to the stakeholders involved in the AHT-EP.

The AHT-EP model supports also other phases linked to the product life-cycle, such as Production, Marketing or Sales, that are not directly related to the EP but that can be represented as black boxes, connected and interacting with the AHT-EP. E.g. linking and including the production phase in the EP (see Lu et al. [10]) enables factory operations to be transformed into data-driven evidence-based practices, offering the capabilities of tracing product fault sources, analyzing production efficient bottlenecks and predicting future resource requirements. Within the Arrowhead Tools project, we will support the adoption of the AHT-EP by using Eclipse Arrowhead as a service oriented solution to manage and automate the phases of the EP, as proposed by Kozma et al. [7].

III. System-of-Systems Modeling

Another important track of the Arrowhead research initiative, summarized as *System-of-Systems modeling*, conceives of digital twins as abstract design-time representations of concrete IoT setups to be deployed in the future. Here, the main challenge consists in identifying the right concepts to frame those design-time representations. Within Arrowhead, we are explicitly building upon established techniques of *model-based systems engineering* (MBSE) [12], as we feel that MBSE provides us with an ideal compromise between domain-specific expectations and rigorous design on the one hand, and a flexible, accessible modeling approach on the other hand. In addition, SysML is an excellent base for formulating and validating well-formedness of complex systems.

In particular, we find it important and also convenient to rely on the *de facto* standard language for systems engineering: SysML [5]. SysML allows for a customizable, domain-specific, yet unified model-based representation of actual (to-be-deployed) Arrowhead SoS installments along with a particular configuration the Arrowhead software core called Eclipse Arrowhead and the underlying deployment (device) platform.

In particular, we consider a *high-level SoS modeling profile* (i.e., a canonical SysML language extension) for devising domain-specific, abstract Arrowhead SoS instances [8]. The simplicity of the profile makes it accessible for a broad audience and a variety of different stakeholders. This profile allows for defining a *validation suite* as well as a*custom-tailored textual exchange format* and, thus, a *bidirectional integration and synchronization* between such models and their *corresponding run-time deployments*. In that later context, our profile-based concept modeling methodology is also employed to represent a configurable software (IoT orchestration) baseline, Eclipse Arrowhead [4] along with a device platform digital twin, where the current prototype is based on Eclipse Vorto.[3]

IV. Toolchain Modeling

The next major topic of interest is the capturing of the central conceptual artifacts of the ongoing *Arrowhead Tools* project: *toolchains*, i.e., purposeful, functionally meaningful combinations of multiple tools reflecting the Arrowhead engineering process. But what is a tool in this matter? It is important to say that, by referring to tools, we are talking about Arrowhead Tools, which adhere to a specific definition within the project, with a potential of being extrapolated to a general definition). They are software artifacts (or have pieces of software) that support one or more phases of the engineering process of a product [9]. It is rather essential to distinguish tools from non-tools by stating that an *already established industrial baseline* can exist without tools, however, the presence of one or more tools becomes very important, as they improve the mentioned baseline by either reducing the overall engineering costs, integrating legacy technology, providing interoperability and automation or providing training material. In many cases, tools are used when substantial manual work could be replaced by some kind of automation. Arrowhead Tools is also founded on top of the concept of toolchains, which are defined as collection of tools that are interconnected through interfaces that make tools capable of autonomously manage the flow of information. These connections also need to support loose coupling, late binding and lookup, as phases of the engineering process are not necessarily executed in a waterfall fashion, instead, as stated in Section II, they can be rearranged and reiterated over time. These three properties are extensively supported by Service-Oriented Architectures (SOA), which would treat tools as service providers and/or consumers. In turn, building toolchains with Arrowhead-compatible tools results in declaring some Orchestration rules.

The Arrowhead platform endorses specific, yet general definitions to forge a joint understanding of what tool and a toolchain is. The differentiation between tools and non-tools is crucial from the viewpoint of automating the execution of toolchains, which might be achieved through an adequately designed system for workflow supervisory control (see, e.g., [6]). Contrary to a classical SOA interconnection approach, services are not the source of triggers for data processing, but they are rather executed in a given sequence, which is in turn managed by the previously-mentioned workflow controller. By including the model-based description of particular tools

[3]https://www.eclipse.org/vorto/

supported by the AHT-EP, it is possible to reach a higher level of abstraction for the toolchains execution. This, in turn, opens the possibility for designing a complex, nonlinear and multistakeholder toolchain blending together both design- and run-time tools.

To support the above considerations some exemplary toolchains will be discussed.

A. The Onboarding Toolchain

The onboarding toolchain is developed at DAC.digital[4]. The idea is to automate and simplify onboarding of new sensor nodes to an IoT network. The toolchain consists of the following tools:

- **AHT provider node** - a microcontroller with appropriate firmware compatible with Eclipse Arrowhead. The purpose of the node is to collect measurements from sensors that are attached to it through one of the exposed (hardware) interfaces. It supports the designed onboarding process by e.g. exposing its credentials (public key) through NFC, and supports communication over WiFi.
- **Onboarding application** - this application is a proxy between the provider node and cloud management, allowing to configure nodes that are being onboarded.
- **Cloud management tool** - web application with the user interface being a management platform that passes configurations of onboarded nodes to field gateways, shows statuses of attached local clouds and allows for reconfiguration of the provider nodes.
- **Field gateway** - a device that aggregates measurements of attached provider nodes through the Eclipse Arrowhead it hosts, and manages configuration of nodes in its local cloud.

The above tools are separate artifacts, however, they are connected through specially designed onboarding toolchain, automatizing the information flow and execution order of particular tools. The procedure of toolchain execution can be described in the following steps:

1) A new node is to be onboarded. Use onboarding application to scan (over NFC) its public key
2) Configure the node (setting, e.g., type of sensor attached, local cloud to which it should be attached) and pass the configuration (over the Internet) to the could management tool.
3) Cloud management tool passes the configuration data to the desired field gateway through a secure communication channel.
4) The desired field gateway now enters automatic discovery mode, where it looks for an incoming connection from nodes.
5) Once the onboarded node is powered on, it enters automatic discovery mode and pairs with field gateway. Immediately after connecting, it fetches its configuration.
6) The information about the successful attachment of the node is propagated to the cloud management tool, and the node is available (for monitoring and reconfiguration) in the user interface.

The discussed procedure is a sequential list of steps that have to be executed in order to fulfill the holistic task assigned to this toolchain. Note that this toolchain consists of both purely software artifacts and hardware with appropriate software inside.

The above-mentioned toolchain modeling approach opens a possibility for further research with clear practical consequences. There is ongoing work (on both research and implementation levels) on automating the exchange of information and supervise the execution of tools in toolchain, where on the basis of a 'recipe', Eclipse Arrowhead will serve not only as the interoperability enabler, but also as a coordinator and supervisor of toolchains execution.

Note that such automated and coordinated execution of toolchain is also an enabler for digital twin implementation and integration - as the number of unknown factors affecting the system is reduced since every action is coordinated, and each tool leaves some kind of digital trace. Even the human factor is eventually reflected in the output of particular tools, so it can be also easily injected to the digital twin.

B. The Arrowhead Design Suite

This general-purpose toolchain focuses on how to foster interoperability even before deployment, i.e., during design time. Interoperability is a core concept of Arrowhead, addressing not only (actually, typically not) the communication between IoT devices within an SoS, but rather the ways of data being exchanged between different tools within the scope of Arrowhead Tools.

Technically, the Arrowhead Design Suite (ADS) is a toolchain based on systems modeling, relying on the standard, most established systems engineering language, SysML (cf. Sect. III). The core of ADS is a plugin collection created for MagicDraw (also known as Cameo Systems Modeler) from CATIA No Magic, arguably the most established industry-scale systems engineering tool. (Thus, this MagicDraw Plugin is not an Arrowhead *system*, as it does not communicate with Arrowhead directly, but it is an Arrowhead *tool* aiding the engineering process.) However, ADS provides much more than that: it makes the relevant parts of the created design models alive by communicating them in the right format to the Arrowhead Management Tool, the standard operation interface of the Arrowhead platform (and, thus, not only a tool, but an Arrowhead system *eo ipso*).

ADS unifies different abstraction levels. The highermost abstraction, manifested as the Toolchain Design Tool, is there for specifying the structure of the toolchains themselves.

Thus, Figure 2 shows an example of particular importance: it demonstrates the use of the Toolchain Design tool to capture an excerpt from *ADS itself*. This toolchain representation approach also constitutes a central contribution of the present paper, as it combines the different modeling and management aspects described above, such as:

[4]https://dac.digital

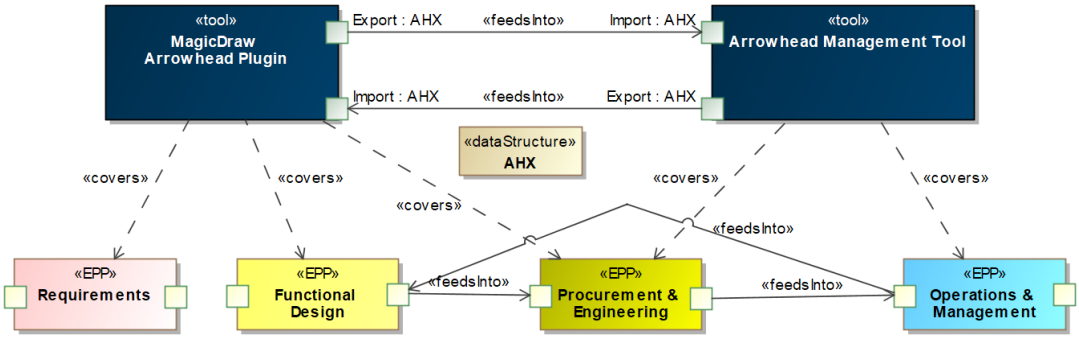

Fig. 2. Excerpt of the Arrowhead Design Toolchain

- a selection of the relevant EPPs (Engineering Process Phases) in the bottom row of the figure, i.e, an engineering process configuration as described in Sect. II;
- a representation of the actual tool setup, building a toolchain as described in the beginning of this section (in particular, we show a very important combination of modeling and management); and
- at the *interoperability interface* of those tools, we also see a reflection of actual *System-of-Systems modeling* (Sect. III): in particular, AHX (the ArrowHead eXchange format) is defined using `SoSysML`, our Arrowhead-specific SysML dialect. Figure 3 shows some details of the SysML specification of AHX; the main purpose of showing these details is to highlight how `SoSysML` is tied back into the technical solutions of Eclipse Arrowhead. Essentially, AHX consists of three arrays of *system definitions*, *service definitions* and *authentication rules* (as in the upper block), respectively (the bottom blocks being specifications for single array entries). These elements are, in turn, the main artifacts which Eclipse Arrowhead utilizes to operate actual Systems of Systems.

We only remark here that the Arrowhead Design Suite encompasses further interoperability formats, for example, to exchange deployment (platform) models or event models between systems and tools, whose elaborate details are out of scope for the present paper.

V. Discussion: Integrating Arrowhead and RAMI4.0 Architectures

Eclipse Arrowhead empowers Industry 4.0 with late binding. This reduces greatly engineering time both in development and maintenance while promoting cybersecurity. Considering that challenge for Industry 4.0 and with the aim of providing a common base around the Industry 4.0, Smart Manufacturing Reference architectural Models (SMRM) has been developed by SDOs and Policy Initiatives on Digitizing Industry around the world which RAMI 4.0 is one of the main reference. The RAMI 4.0 describes the fundamental aspects of I4.0 and aims to achieve a common understanding of what standards and use cases are required for I4.0. In general, the RAMI 4.0 provides a "basic reference architecture" for an I4.0 [3].

On the one hand, it should be noted that the RAMI 4.0 model does not provide details of support for implementation or applications procedure of Industry 4.0 use cases. Although it presents details about the concepts, standards, and interactions, it does not bring the details of implementation and application procedures, as well as suggestion of how to organize and find services and data to support the machine discovery and selection process to perform the operations required by the products. The model combines the production lifecycle (IEC 62890) with the control hierarchy and the product value streams (IEC 62264, IEC 61512).

On the other hand, the Arrowhead framework has developed administrative shells over each asset to promote asset capabilities. The combined asset and administrative shell are referred to as a system. The framework stipulates that the Arrowhead compliant administrative shell must register the capabilities of the asset as services at runtime with a so-called Service Registry system. This exemplifies a service oriented architecture (SOA). Any other system, which needs any such services, can ask the Orchestrator system the address of such desired service, with an additional Authorization mechanism if needed.

Finally, RAMI4.0 brings the possibility to fulfil the potential of production flexibility in Industry 4.0 but it needs to cover real use-cases of automation and digitalization implementation. The incipient Asset Administration Shell initiative started with that approach but it would need to cover the whole Engineering Process of Eclipse Arrowhead. It uses the SysML standard to help RAMI4.0 for covering standardized tools and languages for requirements engineering, MQTT and OPC-UA as interfaces and communications protocols as well as the IEC 61499 Function Block Standard.

VI. Conclusion and Future Work

In this paper, we have presented a comprehensive concept space for *integrating modeling and management in complex, large-scale industrial IoT*.

In particular, in the context of the Arrowhead industrial IoT ecosystem and the ongoing Arrowhead Tools project (the largest digitalization endeavor of the EU), we have outlined a three-dimensional concept space: (i) a configurable *engi-*

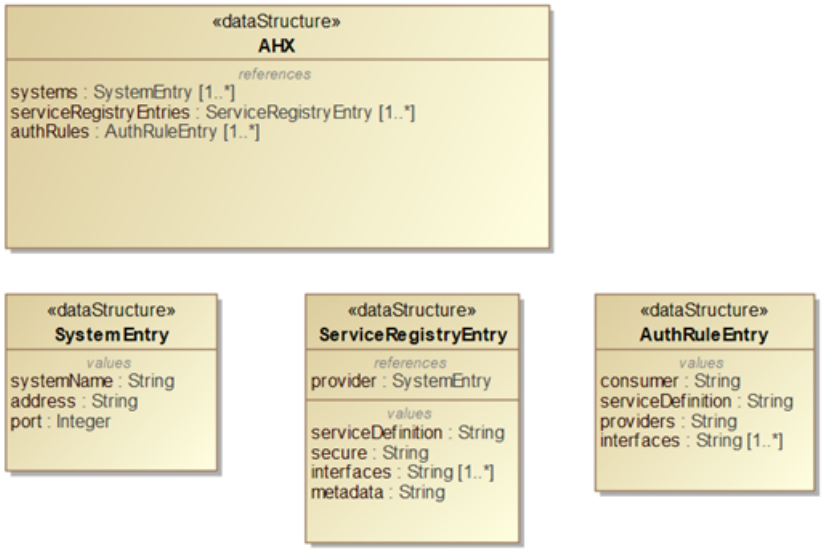

Fig. 3. *AHX*: Example for an Arrowhead Toolchain Interface Definition in SysML

neering process model, representing a highly abstract and process-centric perspective on the engineering problem which has to be tackled; (ii) a novel *toolchain* concept, meaning an architectural and technological setup designed in alignment with the engineering process, but also the technical context of the engineering use-case and the tools and communication channels required to be used there; and (iii) a modeling domain for *systems of systems* (SoS), relying on established industrial practice and standards such as the SysML systems modeling language, but, at the same time, providing an adequate set of modeling concepts for the needs of Arrowhead in particular and large-sclae industrial IoT installments in general.

While we identified the theoretical directions for both a RAMI4.0 alignment and a configurable workflow for utilizing toolchain specifications, we plan to materialize these in some adequate form as future work. Building on existing prototypes, we aim at creating an easy-to-use, comprehensive toolchain design and realization framework, constituting a meta-design tool for industrial IoT in general.

Acknowledgements

This research is funded by ECSEL, the *Electronic Components and Systems for European Leadership Joint Undertaking* under grant agreement No 826452 (Arrowhead Tools), supported by the European Union Horizon 2020 research and innovation programme and by the member states.

Project no. 2019-2.1.3-NEMZ_ECSEL-2019-00003 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the 2019-2.1.3-NEMZ_ECSEL funding scheme.

References

[1] Adolphs, P., Bedenbender, H., Dirzus, D., Ehlich, M., Epple, U., Hankel, M.: Referenzarchitekturmodell industrie 4.0 (rami 4.0) (2015)

[2] Azzoni, P.: From internet of things to system of systems: Market analysis, achievements, positioning and future vision of the ecs community on iot and sos. ARTEMIS-IA Whitepaper **81**, 176 (2020)

[3] Bangemann, T., Bauer, C., Bedenbender, H., Diesner, M., Epple, U., Elmas, F., Friedrich, J., Goldschmidt, T., Göbe, F., Grüner, S.: Industrie 4.0-technical assets: Basic terminology concepts life cycles and administration models. VDI/VDE and ZVEI (2016)

[4] Delsing, J., Varga, P., Ferreira, L., Albano, M., Pereira, P.P., Eliasson, J., Carlsson, O., Derhamy, H.: The arrowhead framework architecture. In: IoT Automation. CRC Press (2017)

[5] Friedenthal, S., Moore, A., Steiner, R.: A practical guide to SysML: the systems modeling language. Morgan Kaufmann (2014)

[6] Kozma, D., Varga, P., Larrinaga, F.: Dynamic multilevel workflow management concept for industrial iot systems. IEEE Transactions on Automation Science and Engineering **PP**, 1–13 (07 2020). https://doi.org/10.1109/TASE.2020.3004313

[7] Kozma, D., Varga, P., Soós, G.: Supporting digital production, product lifecycle and supply chain management in industry 4.0 by the arrowhead framework-a survey. In: IEEE 17th INDIN. vol. 7 (2019)

[8] Kulcsár, G., Kadosa, K., Szvetlin, T., Péceli, B., Horváth, A., Micskei, Z., Varga, P.: From models to management and back: Towards a system-of-systems engineering toolchain. In: Proc. of IEEE NOMS Workshop on Management for Industry 4.0 (2020)

[9] Kulcsár, G., Tatara, M.S., Montori, F.: Toolchain modeling: Comprehensive engineering plans for industry 4.0. In: IECON 2020-46th Annual Conference of the IEEE Industrial Electronics Society. IEEE (2020)

[10] Lu, Y., Liu, C., Kevin, I., Wang, K., Huang, H., Xu, X.: Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues. Robotics and Computer-Integrated Manufacturing **61**, 101837 (2020)

[11] Madni, A.M., Madni, C.C., Lucero, S.D.: Leveraging digital twin technology in model-based systems engineering. Systems **7** (2019)

[12] Micouin, P.: Model Based Systems Engineering: Fundamentals and Methods. John Wiley & Sons (2014)

[13] Schleich, B., Anwer, N., Mathieu, L., Wartzack, S.: Shaping the digital twin for design and production engineering. CIRP Annals **66**(1), 141 – 144 (2017)

[14] Tao, F., Qi, Q.: Make more digital twins. Nature **573**, 490–491 (2019)

[15] Tao, F., Qi, Q., Wang, L., Nee, A.: Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: correlation and comparison. Engineering **5**(4), 653–661 (2019)

[16] Urgese, G., Azzoni, P., van Deventer, J., Delsing, J., Macii, E.: An engineering process model for managing a digitalised life-cycle of products in the Industry 4.0. In: Proc. of IEEE NOMS Workshop on Management for Industry 4.0 (2020)