

Article

Manufacturing Execution System Integration through the Standardization of a Common Service Model for Cyber-Physical Production Systems

Richárd Beregi , Gianfranco Pedone , Borbála Háý and József Váncza 

Centre of Excellence in Production Informatics and Control, Institute for Computer Science and Control, Kende u. 13-17, 1111 Budapest, Hungary; gianfranco.pedone@sztaki.hu (G.P.); borbala.hay@sztaki.hu (B.H.); jozsef.vancza@sztaki.hu (J.V.)

* Correspondence: richard.beregi@sztaki.hu

Abstract: Digital transformation and artificial intelligence are creating an opportunity for innovation across all levels of industry and are transforming the world of work by enabling factories to embrace cutting edge Information Technologies (ITs) into their manufacturing processes. Manufacturing Execution Systems (MESs) are abandoning their traditional role of legacy executing middle-ware for embracing the much wider vision of functional interoperability enablers among autonomous, distributed, and collaborative Cyber-Physical Production System (CPPS). In this paper, we propose a basic methodology for universally modeling, digitalizing, and integrating services offered by a variety of isolated workcells into a single, standardized, and augmented production system. The result is a reliable, reconfigurable, and interoperable manufacturing architecture, which privileges Open Platform Communications Unified Architecture (OPC UA) and its rich possibilities for information modeling at a higher level of the common service interoperability, along with Message Queuing Telemetry Transport (MQTT) lightweight protocols at lower levels of data exchange. The proposed MES architecture has been demonstrated and validated in several use-cases at a research manufacturing laboratory of excellence for industrial testbeds.

Keywords: interoperability; industry 4.0; manufacturing execution system; cyber-physical production system; OPC UA



Citation: Beregi, R.; Pedone, G.; Háý, B.; Váncza, J. Manufacturing Execution System Integration through the Standardization of a Common Service Model for Cyber-Physical Production Systems. *Appl. Sci.* **2021**, *11*, 7581. <https://doi.org/10.3390/app11167581>

Academic Editor: Emanuele Carpanzano

Received: 16 July 2021

Accepted: 16 August 2021

Published: 18 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

There is beauty in how complex the human body is, yet seemingly how naturally and effortlessly it operates day-to-day. For the untrained eyes, a manufacturing plant can seem such a complicated, yet well-functioning entity. Both have a system which connects faculties of sensing, decision-making, and acting, which is essential for their proper functioning [1]. For the human body, this is obviously the nervous system. For a production facility, this is the *Manufacturing Execution System*, or MES in short.

Many definitions and approaches of MES appeared, yet somehow all of them tried to modularize and incorporate every aspect of manufacturing and to combine them into one monolithic system [2]. Thus far, there has not been a minimalist, core concept of MES. One which only has the most basic, *core functionalities* to execute a manufacturing plan. The aim was to take out every intelligent and decision-making aspects of MES to create a *nervous system for manufacturing*, which connects both low-level resources and their operations together, in addition to linking them to the high-level orchestrators, like a scheduler or an Enterprise Resource Planner (ERP). This approach operates with the expectation that both low- and high-level components have built-in intelligence to enable smart production through the digitalization of their environment. The basis for this expectation is the raising phenomena of Cyber-Physical Systems (CPSs) in the manufacturing shop-floor and the

spread of generally accepted distributed decision-making supporting the *Cyber-Physical Production System* (CPPS) concept [3,4].

Manufacturing digitalization prevails more and more, and industrialized countries now have their own initiatives to support it [5]. Industry 4.0 (I4.0) is all about integration and digitalization [6,7], and, given the crucial role of data and Industrial Internet of Things (IIoT) in its technological scenarios, there is a common view that integrated MES are still crucial, although moving toward the broader Manufacturing Operations Management (MOM) and service-oriented approach [8]. Next to this, robotics is expanding in magnitude around the developed world, also in executing manufacturing processes, and is expected to rise from a 15 billion US dollar sector now to 67 billion by 2025 [9].

Overall, I4.0 is demanding a gradual but continuous evolution towards interoperable standards and systems, distributed and self-learning MES services [10]. In this view, modular (not monolithic) MES, with functions becoming services, acquires a central role. Customers in manufacturing require support to multi-plant enterprise orchestration, easy scalability, optimal interplay with modern technologies, and process innovation support through the integration with additive manufacturing, robotics, and Plant Lifecycle Management (PLM) systems. In order to digitally transform industrial automation, significant efforts have to be dedicated to the isolated systems [11].

As I4.0 grows, MES need to grow alongside it [12]. Global networking of production resources and processes, as well as the use of globally interconnected applications, crucially require the adoption of unified standards. *Standardization* in manufacturing architectures evolved from ISA95 to Manufacturing Enterprise Solutions Association International (MESA) [13] in the past decades and, more recently, to service-oriented industrial reference architectures, such as RAMI 4.0 (Reference Architecture Model for Industry 4.0) [14] and IIRA (Industrial Internet Reference Architecture) [15]. All these have proved their applicability also to MES and CPPS, with the intention to organize, define, and integrate automated functions (services) that can be connected to the *core*. These two major standardization proposals are *de-facto* considered reference guidelines for conceiving future business organizations, even though a final consensus or agreement in the scientific and industrial communities has not been reached yet. An introduction to the standardization efforts is reported in Section 2.4, whereas a more comprehensive, cloud- and service-oriented presentation, and comparison of both reference architectures can be found in [16].

This evolutionary process is not straightforward, and an analysis reported also in [10] investigated the evolution of *distributed intelligence architectures* and their expected path of development, what many today identify with Multi-Agent Systems (MAS) [17]. While IIoT is a growth factor for MES in some industries, the availability of newer technologies and approaches can hinder MES deployments. Companies are trying to reduce their future risks and understand future directions in the plethora of I4.0 standards [18]. They are willing to adopt a technology which will be part of the new standards, which is easy to integrate, use, and expand, as well as which can flexibly accommodate new technologies and applications as they are developed. Research work outlined in this paper aimed ultimately at identifying the minimum set of service concepts and technologies that every CPPS should commonly understand and leverage in their standardized service provision.

In line with the argumentation reported in [19], I4.0 is requiring also a new style of MES for several reasons. Firstly, modern consumers are demanding customization. I4.0 offers an unprecedented opportunity for this, but companies need a plant floor software that is ready to adapt to the underlying technologies. Secondly, I4.0 is a concept revolving around CPSs, which is about the merging of virtual and real worlds and the creation of links between information and operations technologies. Thirdly, reference architectures are not always easily nor efficiently applicable in small to mid sized development projects due to their complexity and design requirements.

However, all the above issues cannot be resolved overnight and MES needs to be upgraded alongside the challenges and opportunities of I4.0. In particular, the shop-floor has to be optimized through decentralized management, where autonomous agents may be

seen as a marketplace with demand (the products being built) and supply (the equipment). Future MES needs to accommodate connectivity, mobile, cloud, and advanced service-oriented computing, as well as human collaboration in order to face the challenges of a new industrialization era [19]. Supporting Human–Robot Collaboration (HRC) in a multi-agent setting is a current issue of special relevance [20]. Furthermore, MES needs context-resolution capabilities in order to enable components of a CPPS to act autonomously and release their own intelligence even in a decentralized setting. In this sense, an MES is an orchestrator of complex production processes, typically over multiple layers of abstraction. Finally, IIoT sensory activity and CPPS communications create new data flows and their vertical integration is crucial to ensure an effective response in the enterprise.

1.1. Main Contributions

In the face of new requirements, typical MES functions still remain necessary in the I4.0 era as well, even though in a renewed architecture. Just as the traditional automation pyramid is evolving, the MES should also be changed accordingly, towards providing services. Figure 1 shows how the development process of the MESA automation pyramid is extended with the current proposal in this paper.

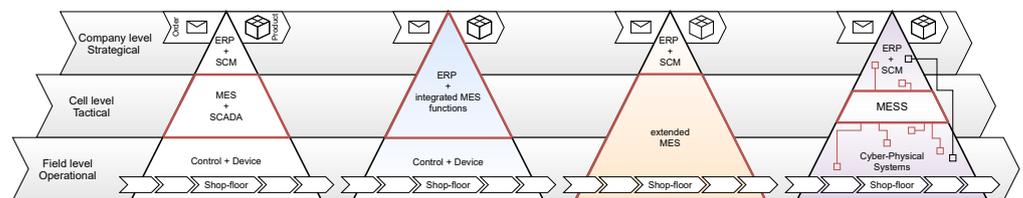


Figure 1. Different functional positioning of Manufacturing Execution System within a company.

Following the above considerations, research presented in this paper targeted the design and realization of a novel concept of *Manufacturing Execution System as a Service* (MESS in the following). This is a new attempt to model, integrate, and orchestrate essential CPS services that make up a modern digitized manufacturing facility. The architecture of MESS facilitates a novel, generic, and simple way for the collaboration of distributed, autonomous manufacturing entities. The focus of the paper is on the core functionality of MESS and the standardized integration of CPS services.

Major points of this research work can be summarized as follows:

- (i) Proposition of a simplified CPS conceptual and information model, whose elements represent the basis for the definition of a common vocabulary in service integration and interoperability semantics (see Section 3.2);
- (ii) Realization of such service integration model as the core of a new, minimalistic concept of MES, which is based on as few assumptions as possible. It privileges standardized industrial technology service and data publishing via Open Platform Communications Unified Architecture (OPC UA) and Message Queuing Telemetry Transport (MQTT) (see Section 3.3);
- (iii) Exploitation of such universal service integration model to a variety of heterogeneous manufacturing cells through by means of a so-called Production Administration Shell (PAS) (see Section 4.3);
- (iv) Demonstration of the presented approach in a research lab for manufacturing. This is a learning factory and an open ecosystem which offers students, researchers, as well as industrial partners the opportunity to perform research on CPPS related topics. An ideal place to study the challenges and to understand the benefits of elevating the CPS to a mature level of interoperability in a production context (see Section 5).

1.2. Paper Outline

After an introduction of I4.0 industrial requirements and the importance to migrate from traditional MES to CPPS-based interoperable environments, motivation and contributions are

presented in short. Section 2 gives a short overview of related works and aims at specifying the goals and requirements of the system targeted in this research. In Section 3, the basic concepts underpinning the methodology are enumerated, and the methodological approach is presented in detail, highlighting the fundamentals of the service integration model, its OPC UA counterpart, as well as the overall MESS system architecture. Section 4 is centered around the realization of the MESS; meanwhile, Section 5 details the application and demonstration of this work in a pilot case study. Discussion and conclusions close the paper.

2. Review of Literature and Standards

The aim of this review is to find the core manufacturing and IT functionalities of a MES that can meet and serve the requirements of CPPS. An overview of recent MES solutions has been performed with a special emphasis on those which could support distributed smart manufacturing. Standardization efforts are reviewed both in regard to expected manufacturing functionalities and applied IT.

2.1. Cyber-Physical Production Systems

An overview of MES's specific role in I4.0 is presented in [21], where it is seen as the digital twin of future factories with an ability to connect real(-time) manufacturing processes with their digital images. I4.0 is the turning point which marks the end of conventional centralized applications: these new developments are scanned in [22], along with the analysis of so-called enabling technologies. Authors in [23] provide an example of basic resource virtualization for the development of CPPS, as a viable process for companies to create digital twins by specifying the digital twin hierarchy, the information to be modeled, and the modeling methods. Anyway, the referred work is not oriented to a possible generalization of the CPPS service interoperability. In [24], a framework for the classification of CPPS developments provided by the scientific society is introduced. As evidenced also in [25], connectivity represents a prior challenge in establishing CPPS on the basis of heterogeneous IT-software environments. An attempt to realize an OPC-UA based service-oriented communication model for CPPS can be found in [26], whose core is centered around the concept of a service bus. In [27], the authors focus on the automated creation of the CPPS digital twin in a body-in-white production system. The outcome of [28] elaborates on the advantages of migrating legacy IT systems for manufacturing operations to a microservice architecture, as an important step towards platform-based ecosystems. Authors in [29] present an Open CPPS automation architecture with the intention to enable vertical integration to become a reality through a CPPS architecture and IEC-61499 standard. Research work in [30] proposes an open architecture for collaborative edge and cloud processing mode, which promises fast operation and upgrading of cloud manufacturing systems through smart monitoring-analysis-planning-execution in a closed loop. An example of architecture that implements CPPS in the cloud with distributed control and the promise to provide fault tolerance are proposed in [31], but authors primarily focus on load balancing and clustering. Research work presented in [32] refers to a CPPS integrated architecture with OPC UA server for 5G-based Smart Manufacturing, but the approach is centralized. Authors in [33] propose a conceptual model of structural and behavioral elements in CPPS, but the attempt to address the gaps of semantic description tools to be included in behavioral elements and in conceptual description remains more at a theoretical stage. A common ontology model in web-based digital twin modeling and remote control of CPPS is proposed in [34]. Finally, an attempt of dynamic reconfiguration of service-oriented resources in CPPS via process-independent approach and multiple criteria for resources and operations can be found in [35].

2.2. Distributed Intelligence Architectures in Manufacturing

Distributed artificial intelligence (DAI) has emerged as a branch of Artificial Intelligence (AI) over thirty years ago as a powerful paradigm for representing and solving complex problems. The growth of the field has been spurred by the advances in distributed computing environ-

ments and the widespread connectivity. Representing a confluence of ideas from several disciplines, DAI became an independent research discipline with two main sub-fields: distributed problem solving and multi-agent systems.

Distributed problem solving deals with the issues related to solving a problem by dividing it among a number of cooperative problem solvers which share the computational burden and knowledge of their partial solutions. *Multi-agent systems*, on the other hand, are concerned with the behavior of loosely coupled autonomous problem solvers, or agents that work together to solve a problem beyond their individual capabilities. Since the DAI field is closely associated with the notion of agents, it is often referred to as multi-agent system research in literature. An important characteristic of DAI is that it combines computational resources of a group of agents such that the group intelligence is more than the sum of the individual agents' capabilities.

Agents made a real break-through two decades ago or so when the emphasis in mainstream AI research shifted from goal-seeking, logic-centered to utility-oriented, rational behavior—from single to multiple, collaborative problem-solving entities. It only mattered that an agent did the “right thing”, even with bounded computational resources [36]. Accordingly, after making observations, it changed its environment by its actions in a way which served best its own interest. The impacts of actions were qualified, and not the mechanisms that generated them. This pragmatic, albeit theoretically founded concept of AI intensified further research in machine learning (when improving the performance of an agent-based on its accumulated experience), in multi-agent systems (when the environment is populated by other agents, too), and also in robotics (when integrating collaborative human and machine agents) [20].

The take-up of agent technologies was accelerated by the development of powerful multi-agent design methodologies, multi-agent simulation as well as programming environments. The agent-based paradigm of computing was early welcome in manufacturing systems control as well [37], with the promise of such much-sought properties like autonomy, cooperation, responsiveness, redundancy, distributedness, and openness [17]. Indeed, it provided an alternative to the traditional, by design centralized, hierarchical, and rigid planning and control schemes. Its current status, achieved advancements, applicability and future outlook are continuously evolving, due to the fact that today's manufacturing systems are becoming increasingly complex, dynamic, and connected [38].

However, even though MAS as a paradigm seems to be suitable to address many of the current requirements imposed by cyber-physical production systems, it still has a long and difficult path for a wider acceptance in industry [39]. Manufacturing services must comply with strong requirements in terms of reliability, robustness, and latency, and solution providers are expected to ensure that agents will operate within certain boundaries of the production, and mitigate unattended behaviors during the execution of manufacturing activities. The so-called Holonic Manufacturing Systems (HMS) addressed these issues in some way, by introducing autonomous, collaborative entities (named holons) to represent the main aspects of manufacturing in terms of product, resource, order, and staff holons. The Product–Resource–Order–Staff Reference Architecture for HMS (PROSA) gained its name from this concept [40]. Holons, as wrappers, hid unnecessary details of the main entities and could realize a spectrum of control schemes, from strictly hierarchical via heterarchical to fully distributed ones [41]. Later, the so-called Activity Resource Type Instance Reference Architecture (ARTI) was born, which combined the strength of the holonic concept and the ideas of typing and instantiating from the information and computer sciences.

On the basis of the PROSA reference architecture, a holonic manufacturing execution system was implemented to support networked production [42]. Here, smart objects representing products found their way of realization in a continuous interplay with intelligent resources. This basically self-organized design opened new research frontiers for investigating issues of distributed intelligence, collaboration, and trust in the context of manufacturing, but in terms of scalability, standardization, and performance guaran-

tees it could not match industrial expectations. In the context of networked production, recently, Ref. [43] suggested a mutualistic resource sharing approach, where the matching between resource offers and requests were made possible by an intermediate *platform*. It is important to note that all decisions were made locally, by the participating autonomous entities. The platform provided only means for information exchange, logical matching of bids demand and supply, and for evaluating and keeping track of the trustworthiness of the partners. Agent-based simulation was used to compare different information exchange mechanisms with respect to utilization rate, service level, and communication load. The findings were applied in the design of an industry-motivated crowdsourced manufacturing platform as well [44].

As an alternative response to the concerns of distributed control in manufacturing, recently a Manufacturing Agent Accountability Framework has been proposed, which is a dynamic authorization framework that defines and enforces boundaries in which agents are freely permitted to exploit their intelligence to reach individual and collective objectives [45].

2.3. MES from the Manufacturing Viewpoint

A MES can be defined by its functionalities according to editor from MPDV Mikrolab (DE) in [2]. Many different MES views and definitions are present in parallel; however, there are some very prominent international or nationally de-facto well-accepted industrial foundations and standardization associations, whose role is outstanding in the MES scene. The biggest three are the Manufacturing Enterprise Solutions Association International (MESA, USA), the National Institute of Standards and Technology (NIST, USA), and the Zentralverband Elektrotechnik- und Elektronikindustrie eV (ZVEI), which are the most well-known in industry.

MESA's MESA-11 model stood the test of time. It was published in 1997, yet, over the years, it obtained many extensions (integrated MES, collaborative MES); however, the eleven *core manufacturing functionalities* stayed the same. They defined MES as an information delivering component from order launch to finished goods, which enables the optimization of production activities and based on the data acquisition functionality it (i.) guides, (ii.) initiates, (iii.) responds to, and (iv.) reports on plant activities as they occur in nearly real time [46]. The numbered activities highlight the crucial functionality expected from a MES.

According to the NIST, MES is a system that uses network computing to automate production control and process automation by downloading routings and schedules, furthermore, uploading production reports. Simply put, the MES bridges the information gap between the business and the shop-floor (control) layer [47]. Here, again, this bridge role reappears and the demand to operate automatically. This is a less specific functionality requirement list; however, NIST released the SIMA (Systems Integration of Manufacturing Applications) Reference Architecture, which contains the accurate manufacturing activities necessary to realize a MES [48].

The ZVEI's VDI (Verein Deutscher Ingenieure) Guideline 5600 categorizes MES in the area of discrete manufacturing as a "process-oriented manufacturing management system" and as the "comprehensive driving force for the organization and execution of the production process" [49]. SEMATECH's (Semiconductor Manufacturing Technology Consortium) CIM (Computer Integrated Manufacturing) Framework defines Manufacturing Information and Execution Systems (MIES), which focus primarily on operations in a factory: resource integration and workpiece management and movement [50].

The ideal MES according to [2], from a functional point of view, consists of guarantees of communication with corporate management applications and with the manufacturing environment, in addition to available function groups that can be activated and used depending on the requirements. These three uncovered function groups are (i.) production, (ii.) quality, and (iii.) personnel, from which the production is relevant in this case and its seven specified sub functionalities. The previously presented MES concepts and their functionalities are listed in Table 1.

Table 1. List of major functional requirements of the Manufacturing Execution System.

Framework, Association/Company (Headquarters, Year of Publication)	Proposed Functionalities
SIMA, NIST (USA, 1987)	production data collection production unit and resource tracking production unit dispatching operation sequence development detailed schedule development data analysis document management
MESA-11, MESA (USA, 1997)	data collection/acquisition operation/detail scheduling product tracking production unit dispatching resource allocation and status process management labor management quality management maintenance management performance analysis document control
MIES, SEMATECH (USA, 1999)	material movement machine control advanced process control factory labor process specification management schedule management factory services factory management and operations
MES, ZVEI (DE, 2007)	data acquisition and processing operating resources management material management personnel management interface management information management detailed planning and detailed scheduling control quality management performance analysis
ideal MES, MPDV Mikrolab (DE, 2007)	production data acquisition machine data collection control station planning table tool and resource management transmission of machine settings material and production logistics

2.4. MES from IT Standpoint

With the rise of the fourth industrial revolution, many attempts to design a general IT modeling and implementation standard have been made. None of the proposed architectures have gained a unique, well-accepted reference role (mainly due to their application-domain specific nature); however, they have become a sort of de-facto expected requirement in the industrial panorama. Hereafter, three of the most recognized and researched technologies nowadays will be briefly presented, which are thus worthy of attention.

Industrial Internet Reference Architecture (*IIRA*) is a standards-oriented industrial internet system open architecture of the Industrial Internet Consortium (*IIC*), which aims to expand industry interoperability and guide the development of technical standards. *IIRA* is a highly abstract general-purpose architecture whose aim is to support a wide range of industrial applications based on use-cases defined by the *IIC*. The Architectural Framework (*IIAF*) is based on the ISO/IEC/IEEE 42010 (2011) standard specifications and codifies the conventions and common practices that are the core of the *IIRA* Internet Information Service (*IIS*). The architectural concepts include concerns (subjects of interest in the system), stakeholders (entities interested in the system), and viewpoints (conventions to construct a description and analysis of specific system concerns).

The core of Reference Architecture Model Industrie 4.0 (*RAMI 4.0*) is the concept of CPS (which is somehow similar to what *IIS* is in *IIRA*), where autonomy is localized and the component systems can make their own decisions. Its reference architecture model is more manufacturing-oriented and represents the integration of multiple stakeholder visions on how to realize *I4.0*-based on existing communication standards and functional descriptions. There are six vertical layers defining the nature of IT components in *I4.0*: business applications, functional aspects, information processing, communication, and integration capabilities, and the ability of assets to implement *I4.0* features. The central concept of *RAMI 4.0* concerns the *I4.0* compliance of components through the deployment of Administration Shell (*AS*), which essentially provides a virtual representation of the entire life-cycle of an object or asset.

The Open Platform Communications Unified Architecture (*OPC UA*) is a product of the *OPC* Foundation ([51]), whose aim is to provide a service-oriented, platform-independent standard whose specification is organized into several documents, ranging from concepts and security model to services, data access, alarms, and so forth. *OPC UA* defines the relations and semantics of services that servers can provide, mapped onto different communication and transport protocols. Interoperability is enabled by mapping concepts between property sets from different domains and/or frameworks. There is an interesting research work presented in [16], which shows a service model similarity evidence and an interoperability affinity between *OPC UA* and *RAMI 4.0* Administration Shell concepts.

2.5. Requirements of the Realized System

The *MESS* will be utilized to execute production activities in a manufacturing facility, as well as to enable service-oriented orchestration and production improvement with other external technologies. The aim of the system is not only to support and improve communication among the *CPSs* inside the facility, but also to improve the communication capability between the production and the other activities in the enterprise (such as process planning, process simulation, resource planning, etc.), to provide updated communication and information analysis to the management, and offer a clear and simple interface to the end-users, in order to monitor and operate the system. Briefly, it is expected of the implemented *MESS* to provide an interoperable, reconfigurable, and reliable information system to meet these requirements.

Several fundamental design considerations have been taken into account in specifying the basic functionality of the *MESS*, as listed hereafter:

- (i) enabling both high- and low-level intelligent solutions in the production control hierarchy;
- (ii) facilitating modern, standardized and easy-to-adopt integration on both end-points;
- (iii) leveraging built-in asynchronous, message-based dispatching systems
- (iv) propagating both historical and real-time data collection;
- (v) obtaining a robust, process-focused, event-based and parallel-running control logic for the realization of a time and product “irrelevant” result;
- (vi) utilizing service-oriented architectural approaches and IT solutions;

(vii) guaranteeing openness to predefined exception handling, as well as to unpredictable execution changing.

A careful selection of the required functionalities has been done for the achievement of an operation-focused and functionally minimalist, but useful MES. The manufacturing execution functions were decomposed according to their specific context and use of interfaces: production process management, data acquisition, and system monitoring. It is expected by the system to initiate, control, and report on production activities as they occur. To support these expectations, a summarized list of major (i.e., considered mandatory) and *core functionalities* for the MESS are catalogued in the following Table 2 based on the previously listed functionality collections.

Table 2. List of major functional requirements of the MESS.

Name	Description	Rationale
Production resource capabilities list	Provide a list of all of the resources available, together with their specific capabilities.	To have an overall view of what can or cannot be done in the system in terms of production capabilities combination.
Resource change management	Provide a preliminary analysis and validation workflow for changes in production resources. It aims at guiding the users along a process of testing, configuration and acceptance of the new production resource and its capabilities.	To identify early problems in production environment configuration.
Production data acquisition and collection	Monitor all resources of the production system, and store the acquired data for later reporting and analysis.	To acquire and collect operational data.
Production tasks dispatching	Address each single production step to the correct CPS and evaluate the overall answer.	To guarantee the link between control and physical execution.
Production process control	Be the system responsible to initiate and execute the list of tasks contained in the production process in the correct order.	To guarantee the correct execution of tasks sequence on the basis of their precedence.
Production process tracking	Monitor the progress of production and provide up-to-the-minute report on the production status.	To provide process supervision and tracking adherent to the real advancement of production process.
Production process planning	Provide an adaptive digital twin of the production process, which makes timely decisions to adjust the schedule and the process plan when unexpected situations occur.	To have a system component capable of managing requests to adapt process execution on the basis of inter-occurred environmental changes.

3. Methodology

The research work presented in this paper embodies a re-elaboration of the Networked Embedded Systems (NES) definition of a CPS [52,53] and its information and service model, with the intention to leverage it for the development and integration of I4.0-compliant CPSs in a distributed MES environment. This was also carried out following the RAMI 4.0 and OPC UA interoperability guidelines. To this end, the necessary industrial equipment and tools have been identified, whereas the CPS common service concepts and its core functionality has been outlined, as detailed in this section.

3.1. Basic Concepts

Manufacturing is fundamentally the process to plan and deploy an optimum way of transforming materials into goods, by means of integrating a variety of resources (people, capital, processes, systems, and enterprises), in order to deliver end-products of value to the market. Production starts with a product design. It always has a Bill of Materials (BoM), which can be translated into a *Process Plan*, which is a sequence of selected manufacturing or logistic related *Capabilities* to be performed by specific production *Resources* on the

components of the BoM at the shop-floor level. A production task can be defined as the reservation of specific Resource based on its required Capability. Execution is the procedure to perform a Process Plan in the production system.

After defining the procedure of production, when a production order arrives, a planner can create a *Routing* by finding the feasible combination of available Resources with the necessary Capabilities to be performed, as well as the required materials and parts, together denominated as *Workpieces*. These one-by-one combinations of Resource, Capability, and Workpiece are called *Operations*, which are the basic units of a Routing. With a scheduler, a Routing's time course (called *Schedule*) can also be determined to finish before the order's deadline. The list of basic concepts leveraged in this work's methodology is reported in Table 3.

Table 3. List of manufacturing concepts utilized.

Name	Description
Resource	Any manufacturing or logistic machine/device or human operator in the production system that has the ability to perform production related activity.
Capability	Abstract description of the functionality that is provided by the Resources towards the system (e.g., gripping, drilling, identification of elements).
Task	A binding between specific Resources and a selected Capability to perform the required production related activity.
Process Plan	A sequence of selected Tasks necessary to perform specific complex production related activities, based on the technological precedence constraints (e.g., assembling, transporting, quality checking).
Workpiece	Every material, part, sub-assembly, assembly and product which is in the production system. (out of scope of this paper)
Operation	A binding between a specific Task and the Workpiece, which is the object of the specified Task. (out of scope of this paper)
Routing/Schedule	The sequence of selected Operations necessary to produce one or more products in the manufacturing system. The Schedule is the time-based dimension of the planned Routing. (out of scope of this paper)

3.2. Cps Service Model and Architecture of the Mess

At the heart of the developed system, there is the concept of CPS, which is in charge to perform production activities. As mentioned, the CPS conceptualization utilized in this manuscript is based on NES definition and comprises the following four basic abilities: sensing, acting, computing, and networking. A CPS can be abstracted to almost anything: a robot, a human worker with network connected device, a camera, a PLC controlled manufacturing unit, a pool of tools, etc. To the MESS, a CPS is like a black box that makes the physical layer seamless, providing a set of production related capabilities and variables. The production capabilities of the CPS can be reached by the production process manager (from now on MESS Core) directly or through a digital counterpart—digital twin—with standardized interface. From the point of view of an external component to the MESS (i.e., user interface, scheduler, and so forth, out of scope in this paper), only these twinned CPSs are visible and discoverable, while physical devices are kept hidden for security and competency reasons. A schematic representation of the system is illustrated in Figure 2.

In order to realize the idea of generally embeddable I4.0-compliant CPSs providing manufacturing services, a "minimalistic" CPS service model has been conceptualized. A CPS is basically expected to embody a set of core service concepts whose selection is necessary to guarantee the following: the core *digital representation* of a CPS; the *service interface* to the MESS collaborative environment; and the *compliance* of the CPS with NES definition and I4.0 components in RAMI 4.0 [16].

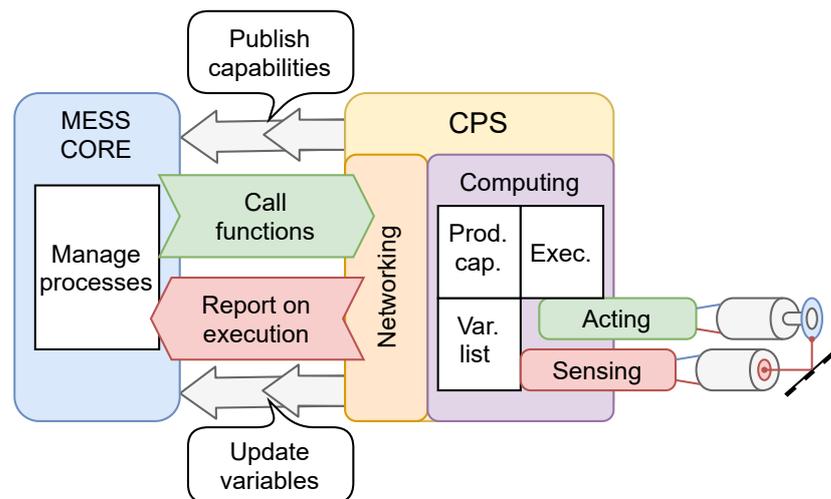


Figure 2. High-level concept of the MESS architecture.

All CPSs publish their capability in terms of (micro or macro) *services*, which can be invoked by means of parameterized *functions*. Invoking a function *call* triggers the instantiation of execution related tasks, which are necessary to track its execution advancement on the basis of even-triggered *reports*. CPSs also have operating parameters called *variables*, which can point to any exposed signal of the specific equipment and whose values can be utilized in the decision-making process of a routing. Functions are organized and linked together in a process plan via precedence edges, which represent the necessary conditions for the specific function to execute (details and an illustrative example in Section 5).

The interface of a CPS identifies the following common functionality:

- (i) enables to connect, disconnect, and refresh requests from the system core;
- (ii) provides information on CPS structure—i.e., device description, capabilities, and variables—and its statuses;
- (iii) enables the execution of a function call;
- (iv) reports on the call's status changes;
- (v) provides the call's historical information;
- (vi) reports changes on subscribed variable; and
- (vii) provides error handling functions (cancel, end, kill process tasks).

Design assumptions:

- (i) one CPS may contain one or more physical devices;
- (ii) on the contrary, one device can only belong to exactly one CPS;
- (iii) variables and functionality of the devices can be reached only through the CPS abstraction model; and
- (iv) the CPS functions can be a logical combination of different physical devices' functions.

Any dedicated, low-level controller is allowed inside the CPS, but each CPS should have a unified interface toward the MESS. The definition of this minimal set of pillar CPS concepts is reported in Table 4.

3.3. Common OPC UA Model of a CPS

Figure 3 illustrates the concept map defined in the OPC UA information space, which is shared by a CPS when connected to the core engine of the MESS. The OPC UA address space is structured hierarchically to foster the interoperability of clients and servers, but only top levels are standardized for all servers (nodes in grey).

All other nodes in the address space follow this initial hierarchy but their specific information model is left to the system designers' freedom to interpret the environmental description. To resolve this initial model uncertainty, a set of common OPC UA nodes is proposed as the basis for every CPS service model. These nodes reflect the concepts

introduced in the previous section and can be either statically or dynamically generated elements of the address space. For instance, objects organizing the overall OPC UA model structure are static (i.e., CPS, Functions, Calls, Variables, and Maintenance Functions), whereas OPC UA nodes related to the actual list of CPS functions, their call statuses, as well as the list of CPS variables are generated according to the specific configuration of the CPS and the execution logic of the process plan (dotted nodes). The core concept of a CPS variable should not be confused with the native OPC UA node of type *variable*, which is proper of the OPC UA specification. CPS and maintenance functions (yellow nodes) are OPC UA *methods*: the former expresses the service granularity of a CPS, while the latter are intended to provide the CPS administrator with a list of utilities regarding the overall process plan, the function calls, and the CPS configuration updates. The capabilities of a CPS follow the naming convention of the OPC UA technology and are specified and extended via a dedicated OPC UA namespace, which is structured hierarchically starting from the main CPS folder. The MESS can handle different reporting mechanisms for tracking the status of a CPS function call execution, but, by default, it requires that each function call generate a proper set of status reporting variables.

Table 4. List of CPS service concepts.

Name	Description
CPS Function	From the point of view of an external production environment, any consumable (micro or macro) service with a well-defined, perceivable utility in the overall process.
CPS Call	The sequence of actions and events after the invocation of a CPS Function on a specific CPS until it is carried out flawlessly.
CPS Report	Any feed-back or status update in the advancement of a CPS Call execution regarding production or environmental changes.
CPS Variable	Any observable physical signal or computed quantity produced by the CPS and published to the external world.

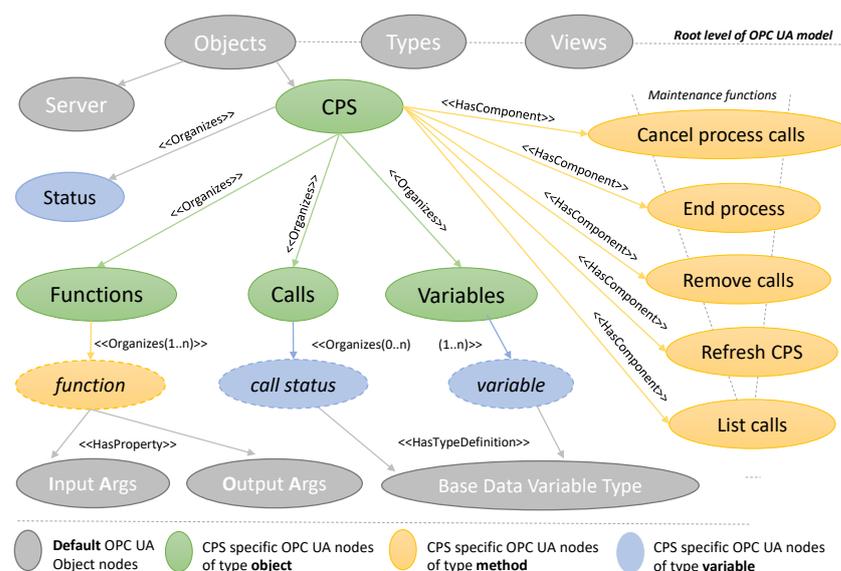


Figure 3. OPC UA information model of CPS core service elements.

4. System Implementation

The MESS is a set of integrated software and hardware components that provide functions for managing production activities, from job order launch to finished products. By the use of current and accurate data, the MESS initiates, guides, responds to, and reports on production activities as they occur (as expected by MESA). It can also interface with other

production information systems, as well as support engineering and business activities (such as scheduling, simulation, process planning, and low-level task dispatching and process control on the physical layer of the CPS; all out of scope in this paper). The major architectural components of the MESS are depicted in Figure 4 and presented in the following sections.

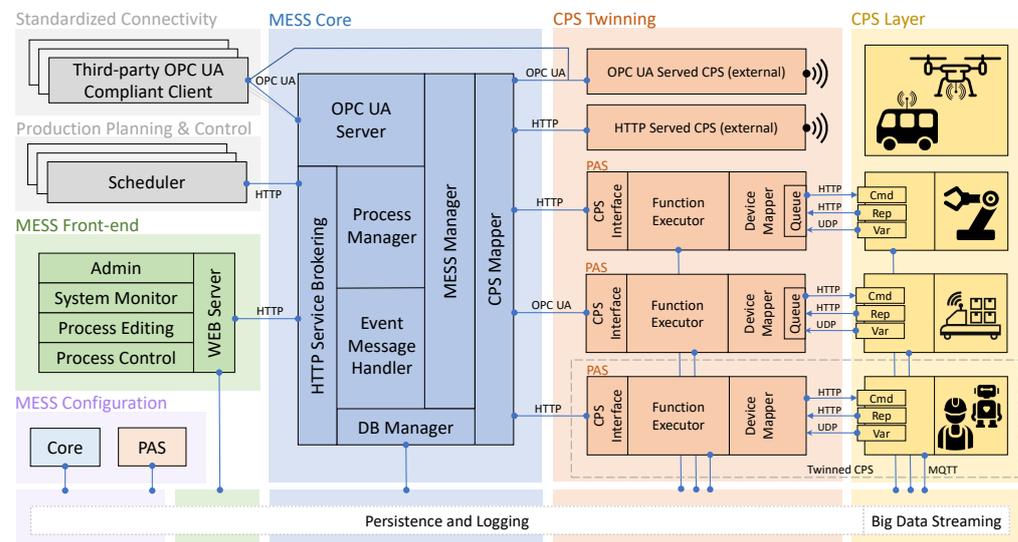


Figure 4. Overall view of the MESS architecture.

4.1. Mess Core

This is the heart of the system: a time-invariant, event-based sequence control of processes, based on the monitored statuses of resources (e.g., management of production processes). The MESS Core controls the synchronization with the CPSs and acknowledges the other system components about the occurring events. As illustrated in Figure 4, its major components can be listed as follows (from left to right):

- *HTTP Service Brokering Manager* for the front-end interface to the client components. It consists of an HTTP server to receive user requests, and an HTTP client to pass event messages to the connected clients;
- *OPC UA Server* for external OPC UA compliant client interface. It embodies and leverages an extended version of the Milo open-source project for complete OPC UA stack management;
- *Process Manager* is in charge for the execution of the process plans. It is one of the most crucial components of the MESS Core component. It manages the parallel execution of different process plans from different sources. It takes care of the proper locking and releasing mechanism of the resources. Nevertheless, it guarantees the precedence constraints between the operations (CPS functions) during the process plan execution, so that technologically irreplaceable operations cannot be interchanged. The sequence of CPS functions is to guarantee an acceptable and feasible production plan;
- *Event Message Handler* implements and manages the event-based logic of the MESS Core. Clients can sign up in order to monitor various kinds of events, and the manager is in charge to notify them both on the status changes of the MESS core and the CPS components, as well as on changes in the physical variables' value. It can also dispatch messages from the CPS;
- *Database Manager* is responsible for the persistent event logging. It records the events in the original form and manages the temporal labeling of the program, process, and operation life-cycles, as well;
- *MESS Manager* controls the communication between the other program layers;

- *CPS Mapper* is finally entitled to provide a unified mechanism to synchronize and consistently manage the connection and the information exchanged between the MESS Manager and the CPS Twinning layer (and its various CPS components).

The overall MESS also embodies a *Process Plan Editor*, which is part of the MESS Graphical User Interface (GUI). It provides a visual tool for the editing of process plan, to be executed by the MESS Process Manager later on. On the front-end side, the function inventory of every registered CPS is accessible and transparent with drag-and-drop feature to help users' interaction. Furthermore, the user interface provides a nearly real-time system overview during the process execution with the presentation of up-to-date production data. Beyond all of this, the admin users have a dedicated and secured interface for the configuration of the production and the MESS system.

4.2. CPS Twinning

As already mentioned, at the core of the designed system, there is the concept of CPS and the exploitation of its *Digital Twin* counterpart in (also virtually augmented) production scenarios. A tool can be any kind of manufacturing unit, an instrument, a device, a robot, a machine, a production line, a transport unit, or even a human worker. These tools are contained in (belong to) a CPS and the MESS Core takes care and manages the connections with all the CPSs. The MESS provides several ways to integrate CPSs and their tools into the system, according to the level of IT expertise and the communication control requested by their designers. Basically, it offers two interface solutions for this connection: by means of an externally served CPS (capable of one between OPC UA—the preferred solution—and HTTP/JSON equivalent service Application Programming Interface, in short API) or by the adoption of a lower-level wrapping Production Administration Shell (PAS) for CPS (part of this research work's contribution and described in detail in the following), respectively. The PAS represents the I4.0 compliant twinning counterpart of a generic production CPS, in which the latter is in control of the physical devices. The great advantage of this CPS Twinning solution is that, regardless of the specific method chosen by the designer for their CPS integration, once connected to the MESS Core, the overall system will seamlessly provide a standardized OPC UA-equivalent transformation of the CPS service and information model to the external world. It is worth recalling how the OPC UA is the technology sitting on top of the MESS (Core and CPS Twinning) standardization and semantic interoperability. It is the technological umbrella that is capable not only of transporting machine data but also semantically describing them in a machine-readable way. An example for the interoperability of a Twinned CPS is shown in Figure 5 through an external, commercial OPC UA client software.

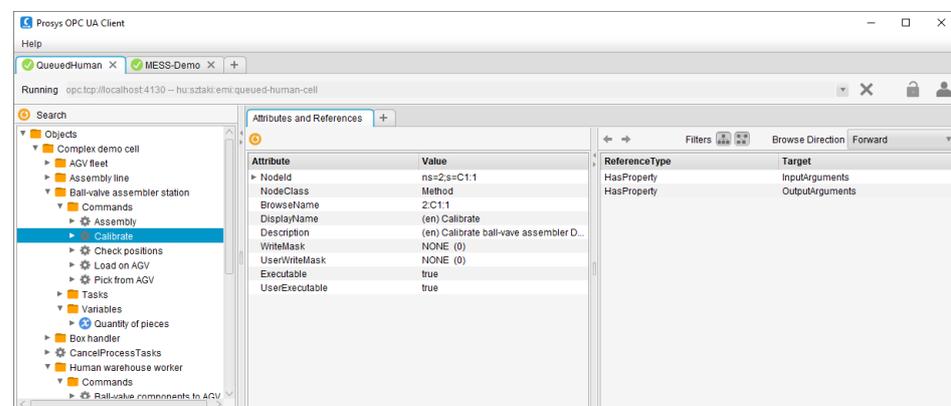


Figure 5. View of Twinned CPS instance via a commercial OPC UA client software.

The externally served CPS solution provides the opportunity to the CPS designers to create their own CPS controller with one of the MESS supported communication methods. The advantage of this solution is that the CPS developer is in charge of the complete

implementation and communication, either via an OPC-UA server or an HTTP client and server pair.

The second solution is one of the peculiar outcomes of this research work and refers to the exploitation of the MESS PAS (details in the next section). Once its configuration is finalized, the PAS enables the construction of complex functionality thanks to the combination of the newly integrated tools' functions. The PAS internally models tools as devices, which can cover any kind of equipment, both individually or in a form of fleet (details in the demo use-case section).

4.3. Production Administration Shell

PAS takes inspiration from the design of the RAMI 4.0 AS (the similarity in representation can also be observed in Figure 6). Simply put, it represents a wrapping component that embodies, organizes, and exposes, in a standardized manner, the CPS functionality towards the overall production execution environment (refer to Figure 4 for architectural details). The goal of the PAS is to provide an easy-to-use tool which helps with transforming typically ad-hoc developed (legacy) manufacturing cells into interoperable, service-based I4.0 CPSs for distributed production environments. PAS embodies a dedicated OPC UA server (extension of the MILO Project, serving both the MESS core and third party OPC UA clients) and an HTTP server/client interface, which is capable of providing the same functionality and services defined for the MESS Core but specified/customized for the local intelligence and complexity of the single CPS.

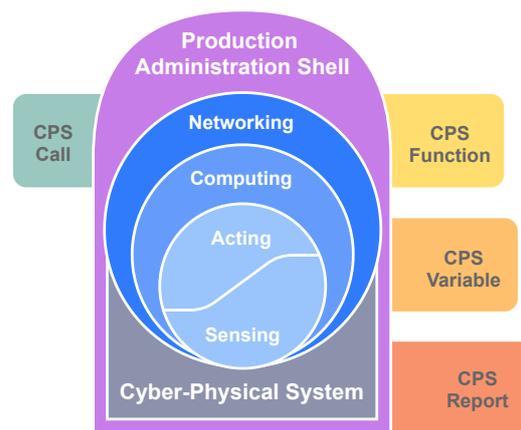


Figure 6. Production Administration Shell for CPS modeling and functionalities.

The PAS embodies a *Device Communication Layer* (DCL) to communicate with the physical devices, which is responsible for sending task execution requests, receiving reports on task execution statuses, sending special (maintenance) service requests to the devices (in order to query or cancel active tasks), as well as collecting the device variables' current values. Sending executable and maintenance tasks, as well as receiving inherent reports and variable updates from the CPS physical layer, can occur through different channels (HTTP or TCP), whereas intensive (Big Data) physical signal variables can be sent over UDP or MQTT, both to internal components or to external, cloud-based services for analytical purposes. As previously mentioned, the CPS Twinning layer of the MESS is able to generate a standardized OPC UA-equivalent transformation of the CPS information model. This is the core feature of the PAS too (details on the interfaces between the MESS Core, the CPS Twinning/PAS, and the low-level CPS communications can be found in Figure 7).

OPC UA was not chosen (although capable) for the intensive, low-level communications between the PAS and the CPS physical layer, due to the following reasons: first of all, the OPC UA is a complex information modeling technology to be deployed, and not just a communication and transport protocol; this can possibly result in communication stack overload and so response time issues at low-level, especially in resource-limited devices; in

other words, OPC UA is not (primarily) intended for IoT bottom-up big data publishing (as also reported in [54]). To this end, the MQTT lightweight publish/subscribe messaging transport technology was selected for bottom-up data interoperability, even considering the possibility of pairing this technology with OPC UA in a publisher–subscriber model, eventually. Alternatively, the PAS has also built-in UDP data exchange interface.

Task execution requests can occur in two different modes, according to the specific device capabilities. By default, this layer handles task queues for the devices and dispatches tasks one-by-one. If a device is natively able to handle execution queues or to process more than one task at a time (what can be called *flooded* task execution), this setting can be omitted. The PAS also contains a built-in planner and a device controller, which provide the same realization of the OPC UA and HTTP interfaces proper of the MESS core but with the main goals to transform the high-level command tasks into low level, device-specific ones, and to control the execution of this new task graph.

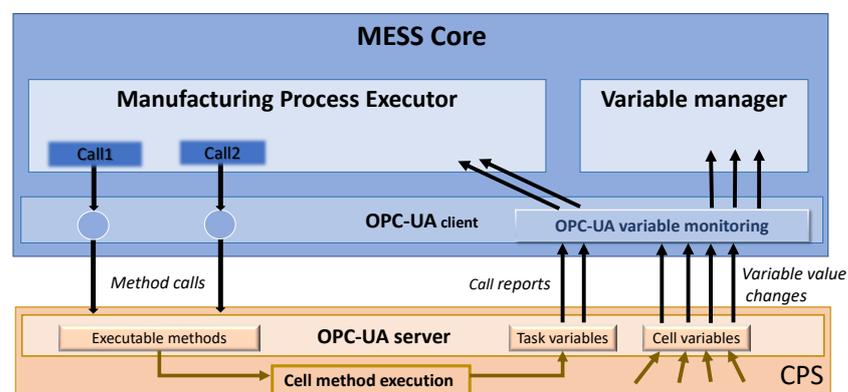


Figure 7. Details on the OPC UA interface between CPS and MESS Core.

The types of CPS utilized in this work comprise a variety of elements: single robot arms (1), production assembly line (2), Automated Guided Vehicle (AGV) fleet, with and without robot arms (3, 5), collaborative robots (4) and human operated components, such as the warehouse and a digital work assistance system (6). There is also a built-in *utility* CPS, whose *internal* capabilities are related to timing, monitoring, and changing conditions in the execution of process plans. A numbered correspondence of CPSs both for the planned and physically realized layout is reported in the following Figures 8 and 9.



Figure 8. Physical components of the CPSs in the demonstrated use case.

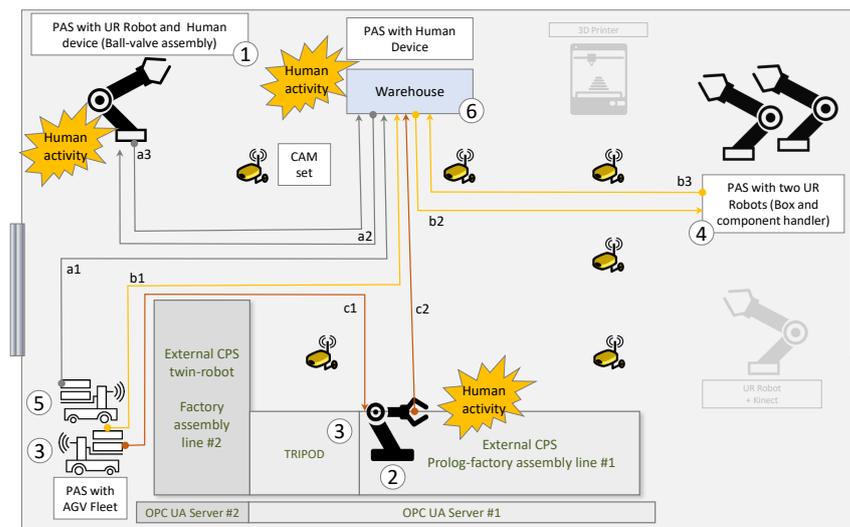


Figure 9. Realized MESS layout: involved CPSs, human activity, and implemented logistics.

5. Demonstration Use-Case

The MESS system was deployed in a pilot cyber-physical production and logistics system which was dedicated to research, education, training, and demonstrations in the academic, industrial, and public sectors [55]. In this *learning factory*, the goal was to verify and validate the MESS system and demonstrate its main services and facilities. The scenarios in the demonstration use-case were rather straightforward by design, in order to clearly exhibit the following features of the MESS: (1) interlinking of heterogeneous resources both (2) of production and of internal logistics; (3) inclusion of physical resources and their full-fledged digital twins; (4) integration of both legacy and state-of-the-art I4.0 ready, as well as (5) of fully automated and human-assisted system components; (6) and demonstration of their easy interoperability. Here, the emphasis is on the fact that some of the integrated elements—such as an assembly line [56], human-robot collaborative workcells [57], or a small fleet of AGVs [58]—were complete, complex CPSs themselves, accompanied by their specific, custom-tailored digital twins.

5.1. The Experimental Scenario

The use-case scenario presented here addresses the production (assembly and delivery) of ball-valves and thermometers in one of the MESS premises. The physical components of the pilot system participating in the use-case scenario are shown in Figure 8. The layout of the realized MESS, the sequence of HRC dynamics, and the involved logistics for the material handling (noted with $a1..a3$, $b1..b2$, $c1..c3$ routes) are depicted in Figure 9.

The main steps of the use-case scenario can be summarized as follows:

- Step 1: Request of an AGV at the warehouse, where the human operator loads the pieces needed at the box handling cell;
- Step 2: In parallel, another AGV is dispatched to the warehouse for the necessary ball-valve assembly pieces;
- Step 3: AGVs deliver the materials to the docking positions indicated;
- Step 4: The box-handling robot picks up the pieces, orders them, and releases the box by positioning it onto the AGV;
- Step 5: At the ball-valve station, the human operator picks up the pieces, places them on the fixture, checks position and tools, and acknowledges back. Calibration and assembly operations then follow. When completed, the human operator loads the assembled ball-valve onto the AGV, which delivers it to the warehouse;
- Step 6: The human operator at the warehouse unloads the manufactured pieces and stores them. The AGV is then released;

- Step 7: At a certain time, another AGV is requested to the factory line #1, which is operating on the thermometers;
- Step 8: When completed, the robot from factory line #1 will load the fixture containing thermometers onto the AGV, which will finally deliver them to the warehouse;
- Step 9: As in point 6;
- Step 10: AGVs are finally released and sent to the docking station for charging.

The list of targets and relative services implemented via OPC UA for each CPS in the demonstrated use-case scenario is reported in Table 5, whilst the graph in Figure 10 illustrates the concatenation of CPS services in the form of a schematic operation graph (routing), with their execution preconditions (precedences between nodes).

Table 5. List of published services for each CPS.

CPS	Integration Type	CPS Controllers	Production Services
Ball-valve assembly station	PAS with HTTP	Robot Controller Human-directed device	M1—Calibrate M2—Assembly ball-valve H1—Pick pieces from AGV, place them on fixture and check position H2—Check tools and robot add-ons correct positioning H3—Load assembled ball-valve onto AGV
Warehouse	PAS with OPC-UA	Human-directed device	H4—Check correct quantity of pieces in containers H5—Put ball-valve components on AGV H6—Pick and stock assembled ball-valve from AGV H7—Put components box on AGV H8—Pick and stock ordered box from AGV H9—Load fixture on AGV H10—Pick, stock pieces from AGV and return fixture to factory docking station
Assembly line	External OPC-UA cell	PLC controller Human operator	F1—Execute assembly order F2—Robot block (activates the secure handling zone of manufactured pieces by the human operator) F3—Robot unblock (de-activates the secure zone) H11—Control correct flow of assembled pieces
Box handler	PAS with HTTP	Box controller	C1—Pick up components from AGV (without robotic arm) C2—Manufacture pieces C3—Load final pieces onto AGV
AGV Fleet	PAS with HTTP	Fleet controller	AGV1—Go and wait to a predefined location AGV2—Transport components to a destination, through an intermediate cell AGV3—Continue to destination AGV4—Finish (indicates that transport is completed)

The previously mentioned *Process Plan Editor* provided by the MESS GUI is where, similarly to the schematic graph, the graph of a process plan can be created by selecting the available operations and connecting them with arrows to form precedence constraints. The process plan for this use-case scenario is shown in Figure 11. Additional information is available for every node derived from the common service model by clicking the info button, as shown in the blue box.

The created plan can be executed and its status can be tracked continuously on the *Process Manager* tab of the MESS GUI. The status of the execution can be visualized by the coloring of each individual task based on their own statuses. This is a quick and easy to understand approach to report on the overall status of the process. The coloring is the following: (i.) grey displays the task is waiting for start signal, (ii.) yellow means the task is under execution; meanwhile, (iii.) the green indicates that the task is finished and (iv.) the red color represents the error status. Except for the error status, the other three can be observed in Figure 12. The communication which reports on the shown status change is visible inside the box in the middle of the figure. The message contains the most necessary data to update the information model of the system: type of the message, process identifier,

timestamp, and the type specific parameters. This lightweight communication between the CPS and the MESS Core allows a reliable operation. The detailed use-case scenario has been executed flawlessly in an automatic and trackable manner.

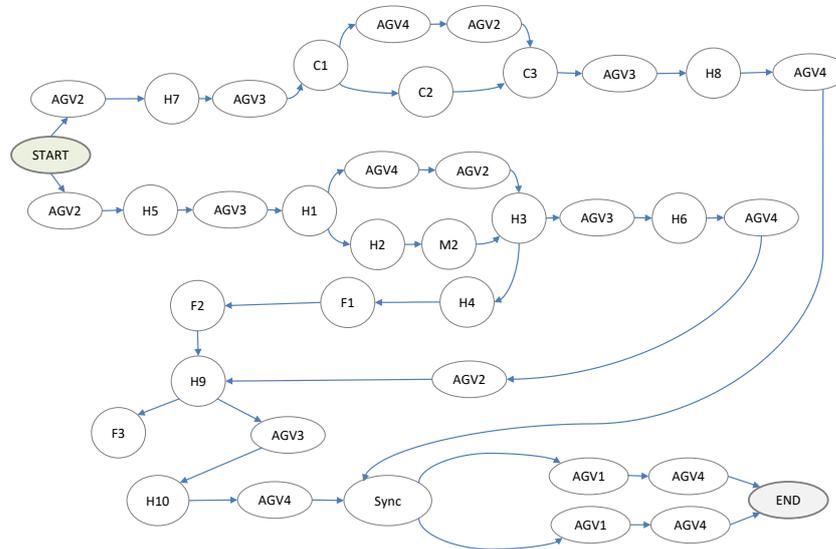


Figure 10. Schematic graph of the demonstrated use-case scenario.

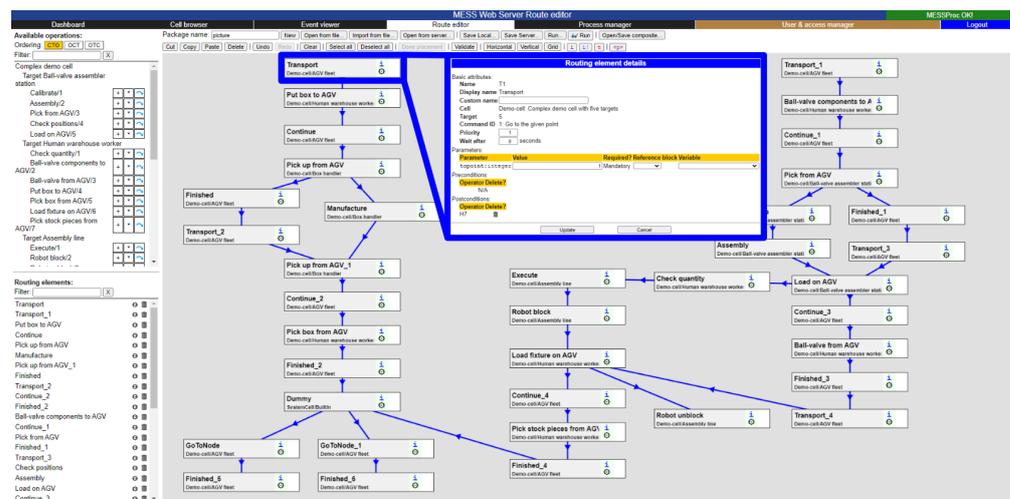


Figure 11. Process plan of the demonstrated use-case scenario.

5.2. Discussion and Lessons Learned

The above experimental setup and scenarios of the use-case proved to be appropriate to demonstrate all the above six features of the MESS. Hence, system components could be (de-)activated in a plug-and-play manner. Production and internal logistics could speak the same language, just like legacy robotic systems and brand new HRC workcells. Production and logistics processes could be run both in the virtual and physical worlds, even in parallel. This opens new opportunities both for right-first-time system design and configuration, as well as anticipating and avoiding failure situations. In settings where humans are closely involved in production, such a service is essential, whereas its lack could be detrimental [20,55].

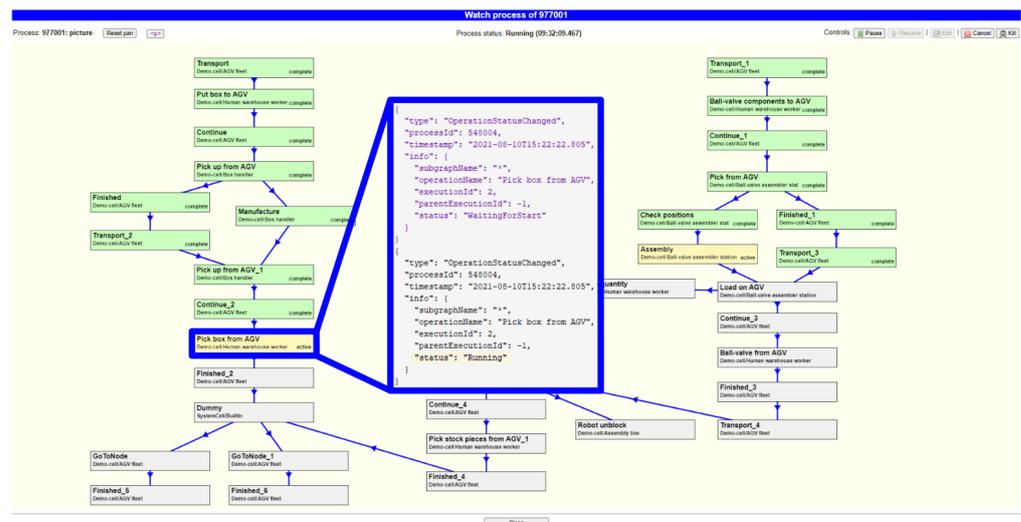


Figure 12. Process execution example with communication.

The implemented MESS and related (PAS-based) CPS resulted in a valid solution for an easier (re)configuration, as well as extensibility and management of the production and logistics processes, thanks to the conceptualized CPS common service model. This led to the conversion of isolated, legacy industrial equipment into I4.0 ready CPSs, which are capable of exposing their functionalities in terms of services. On the other hand, some CPSs evidenced specific issues. Factory line #1, for instance (no. 2 on Figure 8), could natively handle only a predefined number of *tasks*, and this number could not be changed dynamically during the operation. When parameterizing variables, moreover, they could only have read-only properties, due to system security and to avoid unwanted intervention. Nevertheless, by the protocol for writable variables, server-side and client-side writing were also allowed, which might cause unattended security risks. To overcome this, it was necessary to create cohesiveness between the field programmer level and the control programmer level. The establishing of the standardized synchronization of these programming levels requires further development and research work. It was also necessary, for safety reasons, to configure the network permissions in a way to restrict and route communication traffic only from accredited CPSs. After this initial setup, a variety of process plans were successfully carried out by the various CPSs, in a way that single production cells could be incrementally linked to much wider production processes.

6. Conclusions

Industry 4.0, or in general next generation of smart factory developments, will require an unprecedented level of interoperability and standardization, with the intent to facilitate the collaboration of connected cyber-physical entities. The latter, however, are all characterized by a high degree of flexibility, adaptability, and autonomy. In this paper, we suggested a generalized common service model and architecture of CPS-based manufacturing execution systems. The core model is minimalist as far as its underlying assumptions are concerned. Hence, it does not constrain the decision autonomy of collaborating cyber-physical entities, and “only” provides channels for transferring and synchronizing the information which ensue from their decisions. The proposed approach identified the elementary concepts, such as *functions*, *calls*, *variables*, and *reports* as the basis for modeling and providing I4.0 compliant, CPS-based services in a manufacturing environment. They have been developed via standardized technologies enabling semantic interoperability and openness (OPC UA, MQTT).

The universal CPS-based service integration mechanism has been validated in an experimental pilot production and logistics system which included a variety of heterogeneous and autonomous resources, such as manufacturing cells, AGVs, robots, and human–robot

collaborative cells. These CPS components were connected and controlled via the plug and collaborate mechanism of our MESS system in a number of complex scenarios.

Next investigations look forward to the extension of the MESS in particular with including the semantics of the OPC UA common service model to support the adaptation and embodiment of new equipment, like 3D printers, palletizers, and taggers for internal positioning and logistics. The MESS system was prepared also with a distributed intelligent control in mind. Our future research will focus on realizing such a control scheme on the basis of the MESS architecture presented here.

Author Contributions: Conceptualization, R.B. and G.P.; methodology, R.B. and G.P.; software and system realization, B.H. and G.P.; validation, R.B., G.P. and B.H.; formal analysis, R.B. and J.V.; resources, R.B. and J.V.; data curation, G.P. and B.H.; writing—original draft preparation, R.B. and G.P.; writing—review and editing, B.H. and J.V.; visualization, R.B., G.P. and B.H.; supervision, J.V.; project administration, G.P. and J.V.; funding acquisition, J.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been supported by the GINOP-2.3.2-15-2016-00002 grant on an “I4.0 research and innovation center of excellence” and by the ED_18-2-2018-0006 grant on a “Research on prime exploitation of the potential provided by the industrial digitalization”.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AGV	Automated Guided Vehicle
AI	Artificial Intelligence
ANSI	American National Standards Institute
ANSI/ISA-95	International Standard for Enterprise-Control System Integration
API	Application Programming Interface
ARTI	Activity Resource Type Instance Reference Architecture
AS	Administration Shell
BoM	Bill of Materials
CIM	Computer Integrated Manufacturing
Cmd	Command
CPS	Cyber-Physical System
CPPS	Cyber-Physical Production System
DAI	Distributed Artificial Intelligence
DCL	Device Communication Layer
ERP	Enterprise Resource Planning
GUI	Graphical User Interface
HMS	Holonic Manufacturing System
HRC	Human–Robot Collaboration
HTTP	HyperText Transfer Protocol
I4.0	Industry 4.0
IIAF	Industrial Internet Architectural Framework
IIC	Industrial Internet Consortium
IIoT	Industrial Internet of Things
IIRA	Industrial Internet Reference Architecture

IIS	Industrial Internet System
IoT	Internet of Things
ISA	International Society of Automation
IT	Information Technology
JSON	JavaScript Object Notation
MAS	Multi-Agent System
MES	Manufacturing Execution System
MESA	Manufacturing Enterprise Solutions Association International
MESS	Manufacturing Execution System as a Service
MIES	Manufacturing Information and Execution System
MOM	Manufacturing Operations Management
MQTT	Message Queuing Telemetry Transport
OPC UA	Open Platform Communications Unified Architecture
NIST	National Institute of Standards and Technology
NES	Networked Embedded System
PAS	Production Administration Shell
PLM	Plant Lifecycle Management
PROSA	Product-Resource-Order-Staff (Reference) Architecture for HMS
RAMI 4.0	Reference Architecture Model for Industry 4.0
Rep	Report
SEMATECH	Semiconductor Manufacturing Technology Consortium
SIMA	Systems Integration of Manufacturing Applications Reference Architecture
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
Var	Variable
VDI	Verein Deutscher Ingenieure (translation: Association of German Engineers)
ZVEI	Zentralverband Elektrotechnik- und Elektronikindustri eV (translation: Electrical and Electronic Manufacturers' Association)

References

1. Monostori, L.; Váncza, J. Towards living manufacturing systems. *Procedia CIRP* **2020**, *93*, 323–328. [CrossRef]
2. Kletti, J. (Ed.) *Manufacturing Execution System—MES*; Springer: Berlin/Heidelberg, Germany, 2007. [CrossRef]
3. Monostori, L. Cyber-physical Production Systems: Roots, Expectations and R&D Challenges. *Procedia CIRP* **2014**, *17*, 9–13. [CrossRef]
4. Monostori, L.; Kádár, B.; Bauernhansl, T.; Kondoh, S.; Kumara, S.; Reinhart, G.; Sauer, O.; Schuh, G.; Sihn, W.; Ueda, K. Cyber-physical systems in manufacturing. *CIRP Ann.* **2016**, *65*, 621–641. [CrossRef]
5. Ezell, S. Why Manufacturing Digitalization Matters and How Countries Are Supporting It? 2018. Available online: <http://www2.itif.org/2018-manufacturing-digitalization.pdf> (accessed on 30 March 2020).
6. Kilimis, P.; Zou, W.; Lehmann, M.; Berger, U. A Survey on Digitalization for SMEs in Brandenburg, Germany. *IFAC PapersOnLine* **2019**, *52*, 2140–2145. [CrossRef]
7. Ignat, V. Digitalization and the global technology trends. In Proceedings of the 32nd International Conference on Computer Applications in Industry and Engineering, Sibiu, Romania, 14–17 June 2017; Volume 227, p. 012062. [CrossRef]
8. Muhamad Ibrahim, N.; Hassan, M.F. A Comprehensive Comparative Study of MOM for Adaptive Interoperability Communications in Service Oriented Architecture. *Int. J. Trend Sci. Res. Dev.* **2019**, *3*, 23–30. [CrossRef]
9. West, D.M. What Happens If Robots Take the Jobs? The Impact of Emerging Technologies on Employment and Public Policy. 2015. Available online: <https://www.brookings.edu/wp-content/uploads/2016/06/robotwork.pdf> (accessed on 20 April 2020).
10. i-SCOOP. Manufacturing Execution Systems (MES)—Evolution and Software Solutions. Available online: <https://www.i-scoop.eu/industry-4-0/manufacturing-execution-systems-mes-evolutions-software-solutions/> (accessed on 24 March 2020).
11. Stark, R.; Damerau, T. Digital Twin. In *CIRP Encyclopedia of Production Engineering*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 1–8. [CrossRef]
12. Report, M.S. The Multi-Directional Evolution of MES Software. 2018. Available online: <https://www.sme.org/technologies/articles/2018/june/the-multi-directional-evolution-of-mes-software/> (accessed on 20 August 2020).
13. MESA International. MESA Model V2.1. 2019. Available online: <http://www.mesa.org/en/modelstrategicinitiatives/MESAModel.asp> (accessed on 30 June 2020).

14. VDI; VDE; ZVEI. GMA Status Report: Reference Architecture Model Industrie 4.0 (RAMI 4.0). 2015. Available online: https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2016/januar/GMA_Status_Report_Reference_Architecture_Model_Industrie_4.0_RAMI_4.0_/GMA-Status-Report-RAMI-40-July-2015.pdf (accessed on 30 June 2020).
15. Industrial Internet Consortium. Industrial Internet Reference Architecture. 2015. Available online: <https://www.iiconsortium.org/IIRA.htm> (accessed on 23 April 2018).
16. Pedone, G.; Mezgár, I. Model similarity evidence and interoperability affinity in cloud-ready Industry 4.0 technologies. *Comput. Ind.* **2018**, *100*, 278–286. [[CrossRef](#)]
17. Monostori, L.; Váncza, J.; Kumara, S.R. Agent-based systems for manufacturing. *CIRP Ann.* **2006**, *55*, 697–720. [[CrossRef](#)]
18. Enterprise Information Systems. INDUSTRY 4.0 STANDARDS. 2017. Available online: <http://i40.semantic-interoperability.org/> (accessed on 30 June 2020).
19. Almada-Lobo, F. The Industry 4.0 revolution and the future of Manufacturing Execution Systems (MES). *J. Innov. Manag.* **2016**, *3*, 16–21. [[CrossRef](#)]
20. Kemény, Z.; Váncza, J.; Wang, L.; Wang, X.V. Human–Robot Collaboration in Manufacturing: A Multi-agent View. In *Advanced Human-Robot Collaboration in Manufacturing*; Wang, L., Wang, X.V., Váncza, J., Kemény, Z., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 3–41.
21. Mantravadi, S.; Møller, C. An Overview of Next-generation Manufacturing Execution Systems: How important is MES for Industry 4.0? *Procedia Manuf.* **2019**, *30*, 588–595. [[CrossRef](#)]
22. Alcácer, V.; Cruz-Machado, V. Scanning the Industry 4.0: A Literature Review on Technologies for Manufacturing Systems. *Eng. Sci. Technol. Int. J.* **2019**, *22*, 899–919. [[CrossRef](#)]
23. Lu, Y.; Xu, X. Resource virtualization: A core technology for developing cyber-physical production systems. *J. Manuf. Syst.* **2018**, *47*, 128–140. [[CrossRef](#)]
24. Cardin, O. Classification of cyber-physical production systems applications: Proposition of an analysis framework. *Comput. Ind.* **2019**, *104*, 11–21. [[CrossRef](#)]
25. Trabesinger, S.; Pichler, R.; Schall, D.; Gfrerer, R. Connectivity as a prior challenge in establishing CPPS on basis of heterogeneous IT-software environments. *Procedia Manuf.* **2019**, *31*, 370–376. [[CrossRef](#)]
26. Faller, C.; Höftmann, M. Service-oriented communication model for cyber-physical-production-systems. *Procedia CIRP* **2018**, *67*, 156–161. [[CrossRef](#)]
27. Biesinger, F.; Meike, D.; Kraß, B.; Weyrich, M. A digital twin for production planning based on cyber-physical systems: A Case Study for a Cyber-Physical System-Based Creation of a Digital Twin. *Procedia CIRP* **2019**, *79*, 355–360. [[CrossRef](#)]
28. Wunck, C. Towards a Microservice Architecture for the Manufacturing Operations Layer. In Proceedings of the 32nd International Conference on Computer Applications in Industry and Engineering, San Diego, CA, USA, 30 September–2 October 2019; Volume 63, pp. 241–250. [[CrossRef](#)]
29. García, M.V.; Irisarri, E.; Pérez, F.; Estévez, E.; Marcos, M. An Open CPPS Automation Architecture based on IEC-61499 over OPC-UA for flexible manufacturing in Oil&Gas Industry. *IFAC-PapersOnLine* **2017**, *50*, 1231–1238. [[CrossRef](#)]
30. Yang, C.; Lan, S.; Wang, L.; Shen, W.; Huang, G. Big data driven edge-cloud collaboration architecture for cloud manufacturing: A software defined perspective. *IEEE Access* **2020**, *8*, 45938–45950. [[CrossRef](#)]
31. Jo, G.; Jang, S.H.; Jeong, J. Design and Implementation of CPPS and Edge Computing Architecture based on OPC UA Server. *Procedia Comput. Sci.* **2019**, *155*, 97–104. [[CrossRef](#)]
32. Kim, J.; Jo, G.; Jeong, J. A Novel CPPS Architecture Integrated with Centralized OPC UA server for 5G-based Smart Manufacturing. *Procedia Comput. Sci.* **2019**, *155*, 113–120. [[CrossRef](#)]
33. Regal, T.; Pereira, C.E. Towards a conceptual model of structural and behavioral elements in cyber-physical production systems. *IFAC-PapersOnLine* **2019**, *52*, 863–868. [[CrossRef](#)]
34. Liu, C.; Jiang, P.; Jiang, W. Web-based digital twin modeling and remote control of cyber-physical production systems. *Robot. Comput. Integr. Manuf.* **2020**, *64*, 101956. [[CrossRef](#)]
35. Engelsberger, M.; Greiner, T. Dynamic reconfiguration of service-oriented resources in cyber—Physical production systems by a process-independent approach with multiple criteria and multiple resource management operations. *Future Gener. Comput. Syst.* **2018**, *88*, 424–441. [[CrossRef](#)]
36. Russell, S.; Wefald, E. *Do the Right Thing—Studies in Limited Rationality*; MIT Press: Cambridge, MA, USA, 1991.
37. Shih, W.; Srihari, K. Distributed Artificial Intelligence in manufacturing systems control. *Comput. Ind. Eng.* **1995**, *29*, 199–203. [[CrossRef](#)]
38. Arinez, J.F.; Chang, Q.; Gao, R.X.; Xu, C.; Zhang, J. Artificial Intelligence in Advanced Manufacturing: Current Status and Future Outlook. *J. Manuf. Sci. Eng.* **2020**, *142*, 110804. [[CrossRef](#)]
39. Leitao, P. Multi-agent Systems in Industry: Current Trends & Future Challenges. In *Beyond Artificial Intelligence. Topics in Intelligent Engineering and Informatics*; Kelemen, J., Romportl, J., Z.E., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 4. [[CrossRef](#)]
40. Van Brussel, H.; Wyns, J.; Valckenaers, P.; Bongaerts, L.; Peeters, P. Reference architecture for holonic manufacturing systems: PROSA. *Comput. Ind.* **1998**, *37*, 255–274. [[CrossRef](#)]

41. Foit, K.; Banas, W.; Gwiazda, A.; Hryniewicz, P. The comparison of the use of holonic and agent-based methods in modeling of manufacturing systems. *IOP Conf. Ser. Mater. Sci. Eng.* **2017**, *227*, 012046. [CrossRef]
42. Saint Germain, B.; Valckenaers, P.; Van Brussel, H.; Van Belle, J. Networked manufacturing control: An industrial case. *Cirp J. Manuf. Sci. Technol.* **2011**, *4*, 324–326. [CrossRef]
43. Ádám S.; Kádár, B. Platform and direct exchange-based mechanisms for resource sharing in distributed manufacturing: A comparison. *CIRP Ann.* **2021**, *70*, 407–410. [CrossRef]
44. Ádám, S.; Pedone, G.; Egri, P.; Ádám, S.; Nick, G. A mutualistic framework for sustainable capacity sharing in manufacturing. *Procedia CIRP* **2020**, *93*, 938–943. [CrossRef]
45. Beregi, R.; Pedone, G.; Preuveneers, D. Towards trustworthy Cyber-physical Production Systems: A dynamic agent accountability approach. *J. Ambient. Intell. Smart Environ.* **2021**, *13*, 157–180. [CrossRef]
46. MESA International—White Paper #6. In *MES Explained: A High Level Vision*; Technical Report; MESA: Pittsburgh, PA, USA, 1997.
47. Computer Security Resource Center, Information Technology Laboratory, NIST. Glossary. 2021. Available online: https://csrc.nist.gov/glossary/term/Manufacturing_Execution_System (accessed on 31 May 2021).
48. Barkmeyer, E.; Christopher, N.; Feng, S.; Fowler, J.; Frechette, S.; Jones, A.; Jurrens, K.; McLean, C.; Pratt, M.; Scott, H.; et al. *SIMA Reference Architecture Part I: Activity Models*; NIST: Gaithersburg, MD, USA, 1987. [CrossRef]
49. VDI 5600 Blatt 1. *Fertigungsmanagementsysteme (Manufacturing Execution Systems—MES)*; Technical Report; VDI: Düsseldorf, Germany, 2016.
50. Hawker, J.S. CIM Framework architecture and application models. In *Information Infrastructure Systems for Manufacturing II: IFIP TC5 WG5.3/5.7, Proceedings of the Third International Working Conference on the Design of Information Infrastructure Systems for Manufacturing (DIISM'98), Fort Worth, TX, USA, 18–20 May 1998*; Springer US: Boston, MA, USA, 1999; pp. 201–214. [CrossRef]
51. OPC Foundation. OPC Unified Architecture. 2015. Available online: <https://opcfoundation.org/developer-tools/specifications-unified-architecture> (accessed on 19 May 2021).
52. Edward A.L.; Sanjit, A.S. *Introduction to Embedded Systems, A Cyber-Physical Systems Approach*, 2nd ed.; MIT Press: Cambridge, MA, USA, 2017.
53. Mosterman, P.J.; Zander, J. Cyber-physical systems challenges: A needs analysis for collaborating embedded software systems. *Softw. Syst. Model.* **2016**, *15*, 5–16. [CrossRef]
54. OPC Foundation. OPC Unified Architecture. Interoperability for Industrie 4.0 and the Internet of Things. 2019. Available online: https://www.festo.com/rep/en-gb_gb/assets/pdf/GB-OPC-UA-Interoperability-For-Industrie4-and-IoT-EN-v5-June-2017.pdf (accessed on 30 June 2020).
55. Kemény, Z.; Beregi, R.; Tipary, B.; Abai, K.; Nacsa, J. Recent advances in learning content and infrastructure development for layout and process planning courses at the SZTAKI learning factories. *Procedia Manuf.* **2020**, *45*, 319–324. [CrossRef]
56. Szántó, N.; Pedone, G.; Monek, G.; Háý, B.; Jós-vai, J. Transformation of traditional assembly lines into interoperable CPPS for MES: An OPC UA enabled scenario. *Procedia Manuf.* **2021**, *54*, 118–123. [CrossRef]
57. Kemény, Z.; Beregi, R.; Nacsa, J.; Kardos, C.; Horváth, D. Human–robot collaboration in the MTA SZTAKI learning factory facility at Győr. *Procedia Manuf.* **2018**, *23*, 105–110. [CrossRef]
58. Kis, K.B.; Csempesz, J.; Csáji, B.C. A simultaneous localization and mapping algorithm for sensors with low sampling rate and its application to autonomous mobile robots. *Procedia Manuf.* **2021**, *54*, 154–159. [CrossRef]